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

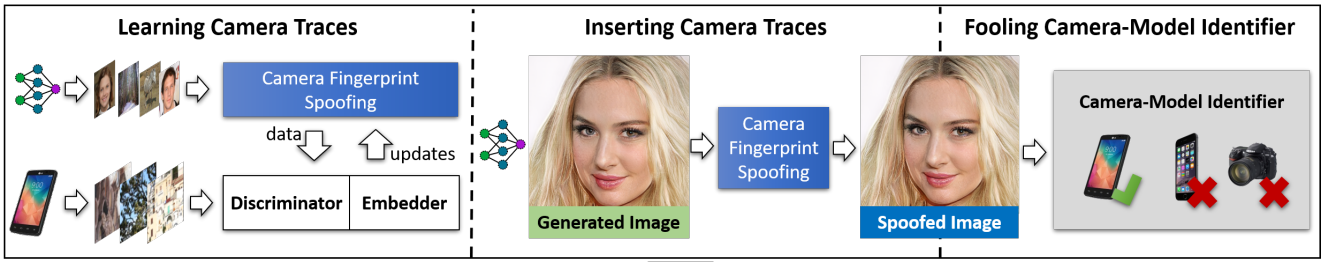

Figure 1: *SpoC* learns to spoof camera fingerprints. It can be used to insert camera traces to a generated image. Experiments show that we can fool state-of-the-art camera-model identifiers which were not seen during training.

Abstract

Thanks to the fast progress in synthetic media generation, creating realistic false images has become very easy. Such images can be used to wrap rich fake news with enhanced credibility, spawning a new wave of high-impact, high-risk misinformation campaigns. Therefore, there is a fast-growing interest in reliable detectors of manipulated media. The most powerful detectors, to date, rely on the subtle traces left by any device on all images acquired by it. In particular, due to proprietary in-camera processes, like demosaicing or compression, each camera model leaves trademark traces that can be exploited for forensic analyses. The absence or distortion of such traces in the target image is a strong hint of manipulation. In this paper, we challenge such detectors to gain better insight into their vulnerabilities. This is an important study in order to build better forgery detectors able to face malicious attacks. Our proposal consists of a GAN-based approach that injects camera traces into synthetic images. Given a GAN-generated image, we insert the traces of a specific camera model into it and deceive state-of-the-art detectors into believing the image was acquired by that model. Likewise, we deceive independent detectors of synthetic GAN images into believing the image is real. Experiments prove the effectiveness of the proposed method in a wide array of conditions. Moreover, no prior information on the attacked detectors is needed, but only sample images from the target camera.

1. Introduction

There have been astonishing advances in synthetic media generation in the last few years, thanks to deep learning, and in particular to Generative Adversarial Networks (GANs). This technology enabled significant improvement in the level of realism of generated data, increasing both resolution and quality [53]. Nowadays, powerful methods exist for generating an image from scratch [28, 7], and for changing its style [60, 29] or only some specific attributes [10]. These methods are very effective, especially on faces, and allow one to easily change the expression of a person [51, 46] or to modify its identity through face swapping [43, 44]. This manipulated visual content can be used to build more effective fake news. In fact, it has been estimated that the average number of reposts for news containing an image is 11 times larger than for those without images [26]. This raises serious concerns about the trustworthiness of digital content, as testified by the growing attention to the deepfake phenomenon.

The research community has responded to this threat by developing a number of forensic detectors. Some of them exploit high-level artifacts, like asymmetries in the color of the eyes, or anomalies arising from an imprecise estimation of the underlying geometry [39, 57]. However, technology improves so fast that these visual artifacts will soon disappear. Other approaches rely on the fact that any acquisition device leaves distinctive traces on each captured image [9], because of its hardware, or its signal processing suite. They allow associating a media with its acquisition device at various levels, from the type of source (camera, scanner, etc.),

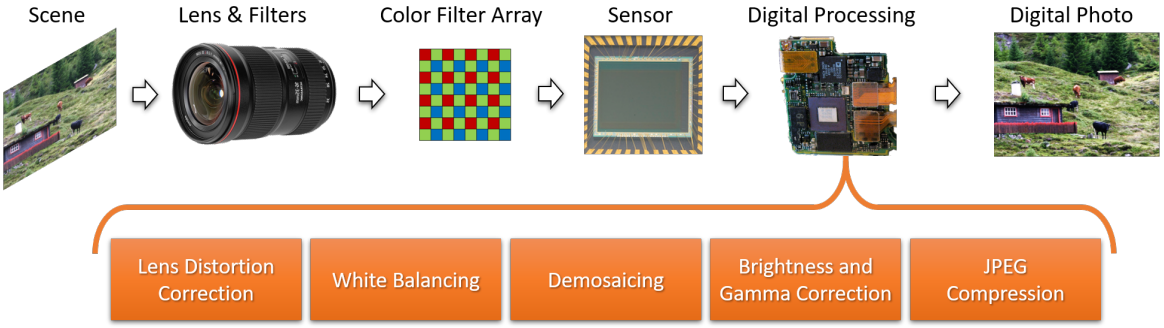

Figure 2: A digital image of a scene contains camera-related traces of the image formation process that could act as a fingerprint of a camera model. The used lenses and filters, the sensor and the manufacturer-specific digital processing pipelines result in unique patterns. These patterns can be used to identify camera models.

to its brand/model (e.g. iPhone6 vs iPhone7), to the individual device [31]. A major impulse to this field has been given by the seminal work of Lukàs et al. [33], where it has been shown that reliable device identification is possible based on the camera photo-response non-uniformity (PRNU) pattern. This pattern is due to tiny imperfections in the silicon wafer used to manufacture the imaging sensor and can be considered as a type of device fingerprint.

Beyond extracting fingerprints that contain device-related traces, it is also possible to recover camera model fingerprints [12]. These are related to the internal digital acquisition pipeline, including operations like demosaicing, color balancing, and compression, whose details differ according to the brand and specific model of the camera [31] (See Fig.2). Such differences help attribute images to their source camera, but can also be used to better highlight anomalies caused by image manipulations [17, 1]. In fact, the absence of such traces, or their modification, is a strong clue that the image is synthetic or has been manipulated in some way [34, 12]. Detection algorithms, however, must confront with the capacity of an adversary to fool them. This applies to any type of classifier, and is also very well known in forensics, where many counter-forensics methods have been proposed in the literature [4]. Indeed, forensics and counter-forensics go hand in hand, a competition that contributes to improving the level of digital integrity over time.

In this work, we propose a method to synthesize traces of cameras using a generative approach that is agnostic to the detector (i.e., not just targeted adversarial noise). We achieve this by training a conditional generator to jointly fool an adversarial discriminator network as well as a camera embedding network. To this end, the proposed method injects the distinctive traces of a target camera model in synthetic images, while reducing the original generation traces themselves, leading all tested classifiers to attribute such images to the target camera ('targeted attack').

To the best of our knowledge, this is the first work that inserts real camera signatures into synthetic imagery to fool unknown camera identifier. Indeed, previous work on fooling camera model identification, based on adversarial noise [22, 37] or pattern injection [30, 8], always considered only real images. In addition, when generating the attack, assume advance knowledge of the attacked CNN-based classifier or, at least, all camera models used to train them [22, 37, 8]. In contrast, we work in an open set scenario, and require exclusively a suitable number of images coming from the target camera model[1].

Hence, our main contributions are the following:

- we devise a GAN-based approach that inserts real camera traces in synthetic images;
- we show that GAN generated images lose their specific traces under this attack;
- we carry out targeted attacks against CNN-based camera model detectors and show that they are easily fooled by our approach;
- we compare results with several state-of-the-art methods based on adding adversarial noise and show that they are ineffective on such a forensic application.

2. Related work

GAN attribution. Synthetic images do not contain camera-related traces. However, they do contain traces related to the pipeline used to generate them. With reference to GAN-generated images some interesting works have been published recently. In [38], using a procedure similar to PRNU-based extraction, evidence is given that

1. In principle, we do not even need to know the camera model itself.

each GAN image is characterized by a sort of artificial fingerprints related to the specific GAN architecture. A similar analysis is carried out in [58], where it is shown that GAN fingerprints also depend on training data and random initialization seeds. In [2], instead, the problem of GAN attribution is solved using generator inversion, with the additional advantage of re-creating the generation process and improving the interpretability of the attribution decision. Finally, in [59], it is recognized that GAN artifacts are caused by the up-sampling components included in the common GAN pipeline, which is identified using a classifier working on spectrum data rather than image pixels.

All these works point out that GAN synthetic images possess peculiar intrinsic characteristics which can be exploited to tell them apart from real ones. Such analyses also justify the good performance of supervised CNN-based methods when training and test include GAN-based images generated using the same architectures [36, 42].

Camera model identification. Aside from explicitly using the device or camera fingerprints, camera model identification can be also carried out using deep learning-based strategies [52, 5, 40]. These approaches are very effective when the training set includes models that are also present in the test set, as shown from the results of the IEEE Forensic Camera Model Identification Challenge hosted on the Kaggle platform[2] in 2018. In general, most successful solutions are based on very deep networks [50, 24, 11], that perform well even when data are subjected to common post-processing operations, like the re-compression routinely performed when uploading images on social networks.

Counter-forensics on source identification. There is a large literature in the forensics community on forensic-aware attacks, which exploit knowledge about detection methods to fool them [4]. Counter-forensics methods allow researchers to better understand how current methods can be attacked by a malicious user in order to improve their performance in an adversarial setting. In [18] a PRNU fingerprint is added to the image in order to make it appear as it was taken by a different device. However, this procedure is not easy to carry out, and a suitable countermeasure, the triangle test can be used to discover such copy-fingerprint attacks [19]. More sophisticated procedures are proposed in [14] where an approach based on seam-carving is used for anonymization, and [16], which uses a patch replacement attack to desynchronize sensor noise fingerprints, while preserving an acceptable visual quality.

Adversarial attacks to camera identification. Adversarial attacks are conceived to fool a classifier by adding imperceptible perturbations [20]. [22] and [37] investigate on the vulnerability of deep learning based camera model classifiers in a white box setting. They show that the attack is effective only when the image is uncompressed, while in realistic forensic applications the perturbation noise is hidden by the compression artifacts. The analyses also highlight the difficulty of transferring such attacks in the context of camera model identification. This is part of a more general behavior. Contrary to what happens in many computer vision applications [20, 32], in forensics applications, where detectors rely on tiny variations of the data, transferability for non-targeted attacks is not achieved easily [21, 3].

Adversarial attacks using generative networks. In the literature, several papers have already addressed the problem of using generative networks to create adversarial examples. In [49] and [54] the adversarial examples are generated from scratch, with no further constraint. Other papers instead are more relevant to our scenario and use a generative approach to slightly modify an existing image [23, 45, 56, 8, 13]. In [45] the classifier under attack is supposed to be perfectly known (white-box scenario) and its gradient is used to train the generator of adversarial samples. In [23], instead, the attacker is only allowed to query the classifier and observe the predicted labels (black-box scenario). With this information, a substitute classifier is trained and its gradient is used to train the generator. The strategies of [45] for white-box scenarios and of [23] for black-box scenarios are adapted by Chen et al. [8] for the specific problem to fool a camera model classifier. In [56], instead, these approaches are further extended by including a discriminator network to distinguish between original and attacked images, pushing the generator to improve the fidelity of the attacked image to the original. To adapt to a face recognition scenario, where the number of classes is not fixed in advance, in [13] the target classifier is replaced with a face matcher based on the cosine similarity in the embedding space.

Unlike previous works, our proposal does not require knowledge of the classifier under attack [45, 56, 8], and not even of the labels output by the network in response to selected queries [23, 56, 8]. Moreover, although the use of an embedder was already proposed in [13], we use a less restrictive loss in the embedding space. In fact, while in [13] the distance of the generated example is minimized with respect to a random sample of the target class, we minimize distances to a representative anchor vector, and focus mainly on critical outliers with respect to this anchor. In addition, our discriminator does not aim to preserve perceptual quality [56, 13], but to improve the generators ability to introduce traces of the target camera model, and to remove peculiar traces of synthetic images. The objective is

2. https://www.kaggle.com/c/sp-society-camera-model-identification/

thus similar to [8]; however, in our work we inject camera model traces in synthetic images and not real ones. More importantly, our strategy does not need to include the camera model classifier in the generation process as done in [8], but requires only images coming from the target distribution.

3. Reference Scenario

In this section, we describe in more detail our reference scenario. We want to show that inconsistencies in camera traces in forensics cannot be reliable even when the attacker has very little knowledge about the classifier.

Targeted attack: Our goal is to make a synthetic image appear as if it was taken by a specific real camera by inserting peculiar traces of the latter. Hence, our attack is targeted, and our aim is to fool CNN-based camera model identification detectors that will recognize the generated image as if it was taken by the target camera model.

Available knowledge: We assume that the attacker has no knowledge about the specific classifier and cannot make queries to it. We only suppose that the training images are drawn from the same distribution, that is, generated by the targeted camera model. However, no knowledge is given on the other camera models on which the classifier is trained.

Visual imperceptibility: We require that the attacked images look realistic and do not present visible artifacts. The generation of realistic synthetic images is out of the scope of this work, and we simply assume they are available. However, we require that the attack does not change the image content and introduces a perceptually acceptable distortion.

Processing pipeline: Images are JPEG compressed after being modified to simulate a realistic scenario. This is very important since fake content is especially harmful when it is uploaded on the internet and shared with a malicious goal, such as creating fake news, and propagating false information.

4. Proposed Method

Let $\mathcal{R}_M$ be the set of real images generated by the target camera model M, and $\mathcal{S}$ a set of synthetic images generated by some software tools. We aim to process the images in $\mathcal{S}$ so as to make them forensically indistinguishable[3] from images in $\mathcal{R}_M$. That is, we want to find a transformation $T_M(\cdot)$ such that, with high probability,

$$F_i(T_M(x)) = M, \qquad x \in \mathcal{S}, \;\; i = 1, \dots, N$$

with F_i's the available camera model classifiers. Since state-of-the-art forensic classifiers focus on the traces left on all acquired images by the model-specific processing suite, such traces must be injected into the original image. We pursue this goal by training a convolutional neural network on images drawn from $\mathcal{R}_M$, such that eventually, for each $x \in \mathcal{S}$, $y = T_M(x)$ looks to classifiers as belonging to $\mathcal{R}_M$. Meanwhile, to remain undetected, the attack is required not to modify the semantic content of the image, and hence to minimize some suitable measure of distortion $d(x, y)$ between original and modified images.

4.1. Architecture

Our goal is to be agnostic to any signature-based camera model identifier; i.e., we want more than inserting adversarial noise to fool a specific identifier. To this end, we train a CNN through a reformulated GAN schema. Specifically, our proposal involves the use of three networks: a generator, a discriminator and an embedder. These are described in the following, and depicted in Fig.3.

Generator: The Generator, $G(\cdot) = T_M(\cdot)$, has the goal of introducing traces of a specific camera model M into the input image, while preserving the semantic content. We adopt an architecture formed by 7 convolutional layers. The output of the last layer is summed to the input image and then a hyperbolic tangent is applied to enforce valid color values (all images are in $[-1, 1]$). Note that, before entering the generator, the input image is Gaussian filtered with $\sigma = 0.4$ to remove possible high-frequency imperfections, such as checkerboard artifacts.

Discriminator: In conventional GANs, the discriminator, $D(\cdot)$, is required to distinguish real from generated images, with the aim of pushing the generator to improve over time. Accordingly, we could ask the discriminator to separate the sets $\mathcal{R}_M$, real images of the target camera model, and $G(\mathcal{S})$, modified synthetic images. Instead, we train it to tell apart $\mathcal{R}_M$ from both $G(\mathcal{S})$ and $\mathcal{S}$, the set of original synthetic images. By doing so, the generator is encouraged not only to introduce traces of the target camera model, but also to remove traces peculiar of synthetic images. We use a patch-based discriminator [25] realized as a fully-convolutional network with a fixed first layer. The fixed layer takes as input the RGB image and returns 9 channels, which are the original RGB components and their third-order horizontal and vertical derivatives. In practice, this first layer extracts the so-called image residuals, which highlight discriminating camera traces, speeding-up the network convergence [52].

3. Of course, the synthetic images should also be visually realistic, but we do not address this problem, here, and assume the generation tool produces already visually plausible images.

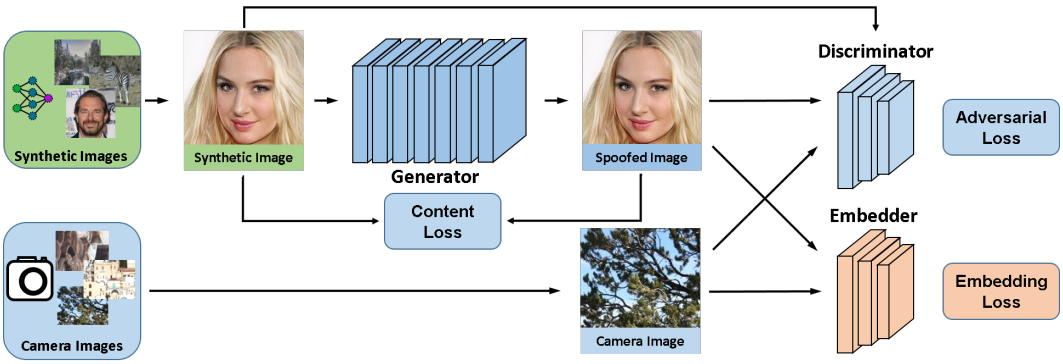

Figure 3: Our architecture is based on three components: a classical GAN setup, using a generator as well as a discriminator, and an embedding network. By applying a content loss between synthetic images as well as spoofed images, we ensure that our generator keeps the image content intact. The pre-trained embedder checks if we only generate specific camera traces. In comparison to regular GAN methods, we use all three involved images for our discriminator, i.e., synthetic, spoofed and camera typical images, and adjust the loss correspondingly.

Embedder: In addition to a generator and discriminator, we use an embedder, $E(\cdot)$. This network is pre-trained offline, using images drawn from many camera models, to extract a compact 512-dimensional feature vector. It is trained using a triplet-loss [48] with the goal of obtaining the same feature vector for all images acquired by the same camera model. Therefore, it provides a compact, model-specific, representation of the image, independent of the current data. For the embedder, we use the same fixed first layer used for the discriminator.

4.2. Loss function of the generator

The goal of our scheme is to train an effective generator. To this end, we define its objective function as the sum of three losses,

$$\mathcal{L}_G = \mathcal{L}_{CNT} + \lambda_E \mathcal{L}_{EMB} + \lambda_A \mathcal{L}_{ADV} \tag{1}$$

each of which drives the generator towards a specific goal: preserving the scene content ($\mathcal{L}_{CNT}$), fooling the embedder ($\mathcal{L}_{EMB}$) and, fooling the discriminator ($\mathcal{L}_{ADV}$). These losses are detailed in the following.

Scene content preserving loss: To achieve the first goal, we use a combination of an objective distortion measure as well as a perceptual loss between input and the output of the generator:

$$\mathcal{L}_{CNT} = \mathcal{L}_{REC} + \lambda_p \mathcal{L}_{PER} \tag{2}$$

As distortion measure we use an L1 distance between the two images, $\mathcal{L}_{REC} = \mathbb{E}_{x\sim\mathcal{S}}\left[\|x - G(x)\|_1\right]$. Following [27], we define the perceptual loss $\mathcal{L}_{PER}$ as the sum of the L1 distances between the feature maps extracted by the VGG-19 network trained on ImageNet.

Embedder Loss: The second goal of the generator is to fool the embedder, namely, to ensure that feature vectors extracted from the transformed images are indistinguishable from those of real images of the target model. To this end, by averaging the feature vectors of real images, we first compute an anchor vector, $e_M = \mathbb{E}_{z\sim\mathcal{R}_M}\left[E(z)\right]$, representing the camera model in the embedding space. Then we should aim at minimizing the distance, $d(x) = \|E(G(x)) - e_M\|_1$, between feature vectors extracted from transformed images and this anchor vector. However, from the literature on triplet loss, it is well known that better results are obtained by comparing distances [48]. Therefore, we first define a reference distance $d_{ref} = \mathbb{E}_{z\sim\mathcal{R}_M}\left[\|E(z) - e_M)\|_1\right]$, and then define the loss to minimize as

$$\mathcal{L}_{DST} = \mathbb{E}_{x\sim\mathcal{S}}\left[\,|d(x) - d_{ref} + m|_+\right] \tag{3}$$

where $|x|_+ = x$ for $x > 0$ and 0 otherwise, and m is the margin of the triplet-loss fixed to 0.01 in our experiments. To further help the generator to fool the embedder, we include also a feature matching loss, $\mathcal{L}_{FM}$, proposed in literature [53] to stabilize the training of GANs, which we compute based on the feature maps extracted by the embedder of the real and the transformed images. The final embedding loss is then

$$\mathcal{L}_{EMB} = \mathcal{L}_{DST} + \lambda_f \mathcal{L}_{FM} \tag{4}$$

Adversarial Loss: The discriminator, trained in parallel with the generator, should output values close to 1 for real images, $x \in \mathcal{R}_M$, and close to zero for synthetic images, $x \in S$, both before and after being modified. Accordingly, it relies on a modified binary cross-entropy loss:

$$\begin{aligned}\mathcal{L}_D = &-\mathbb{E}_{x\sim\mathcal{R}_M}\left[\log D(x)\right] + \\ &-\tfrac{1}{2}\mathbb{E}_{x\sim\mathcal{S}}\left[\log(1 - D(G(x))) + \log(1 - D(x))\right]\end{aligned} \tag{5}$$

which pools original and modified synthetic images.

A major goal of our generator is to modify the synthetic images to fool the discriminator, i.e., to make the discriminator believe they come from the target camera model. Therefore, for the last term of Eq.(1) we adopt the standard adversarial loss based on the binary cross-entropy:

$$\mathcal{L}_{ADV} = -\mathbb{E}_{x\sim\mathcal{S}}\left[\log D(G(x))\right] \tag{6}$$

which is minimized when $D(G(x)) \to 1$, that is, synthetic images are classified as real after being modified. Thus, generator and discriminator concur to modify the synthetic images to be similar to images of the target camera model but also different from the original synthetic images.

5. Results

In this section we present the results of our experiments. Specifically, we evaluate the proposed method in terms of its ability to (a) deceive detectors of GAN-generated images into believing they are dealing with real images (see Sec. 5.1) and, (b) deceive camera model identifiers into recognizing the modified images as acquired by the target camera model (see Sec. 5.2). Special attention will be devoted to testing robustness to off-training GANs. To this end, experiments will be carried out on images generated by GAN architectures never seen in the training phase.

We consider a dataset of real images acquired by different camera models and a dataset of synthetic images generated by various GAN architectures. For the real images, we use the 10 camera models adopted in the IEEE Forensic Camera Model Identification Challenge, 400 images per model are used for training, and 50 for testing. From the test set we sample 25000-patches to test the camera model identifiers. For the synthetic images, six GAN architectures are considered: StarGAN [10], CycleGAN [60], ProGAN [28], StyleGAN [29], RelGAN [55], and bigGAN [6]. For each of the first five architectures, we take 20000 images for training and 2000 for testing. In addition, 2000 bigGAN images are used for testing, but none for training. All experiments are carried out on 256×256-pixel patches. For both real images and high-resolution GAN images (generated by ProGAN and StyleGAN) patches are extracted at random locations from the whole image.

For each of the 10 camera models, a different generator-discriminator couple is trained, using only real images of the selected model. We use ADAM optimizers with a batch size of 10 and 30 patches, respectively. For both networks, the learning rate is set to 10^{-4}, and the ADAM moments to 0.5 and, 0.999. Through preliminary experiments, the loss weights are set to $\lambda_E = 1$, $\lambda_A = 0.1$, $\lambda_p = 0.001$, and $\lambda_f = 0.01$. Training stops after $50K$ iterations.

The embedder is trained in advance using an external dataset of 600 camera models and a total of 20394 images publicly available on dpreviewer.com. We also adopt ADAM for the embedder, with a batch of 80 patches, a learning rate of 10^{-4} and default moments.

	GAN detectors	
	Xception	Spec
before attack	99.98%	99.91%
after our attack	20.85%	4.83%

Table 1: True Positive Rate for the GAN detectors before and after the proposed attack using GAN architectures within the training-set.

	GAN detectors	
	Xception	Spec
before attack	90.50%	20.25%
after our attack	2.50%	11.64%

Table 2: True Positive Rate for the GAN detectors before and after the proposed attack using a GAN architecture outside the training-set.

5.1. Fooling GAN-image detectors

Here, we show that our architecture largely succeeds in removing the peculiar features of GAN images that make them distinguishable from real images. To this end, we challenge two CNN-based GAN-image detectors. The first one is a general-purpose deep network, Xception [11], which proved to be very effective in several forensics applications, especially for detecting GAN images [36] and video deepfakes [47]. In addition, we consider the spectrum-based (Spec) classifier[4] proposed in [59], which detects the frequency-domain peaks caused by the up-sampling steps of common GAN pipelines. Both detectors are trained on the same dataset used for the proposed method. Testing is carried out on 256×256-pixel central crop of 10000 synthetic images.

In Tab. 1, we show the results of this first experiment. Both detectors have a true positive rate (TPR) close to 100%, that is, they detect GAN images with near certainty (the same holds for real images, not shown in the table). However, after modifying the synthetic images with our attack, the TPRs decrease to 20.85% and 4.83%. It is also interesting to observe the effect of our attack on the spectra of GAN images (see Fig. 4). The spurious peaks almost disappear, making detection virtually impossible.

In a second experiment, we work on the off-training images generated by the bigGAN architecture. Results are reported in Tab. 2. While Xception keeps detecting GAN images with high accuracy, the spectrum-based classifier pro-

4. https://github.com/ColumbiaDVMM/AutoGAN

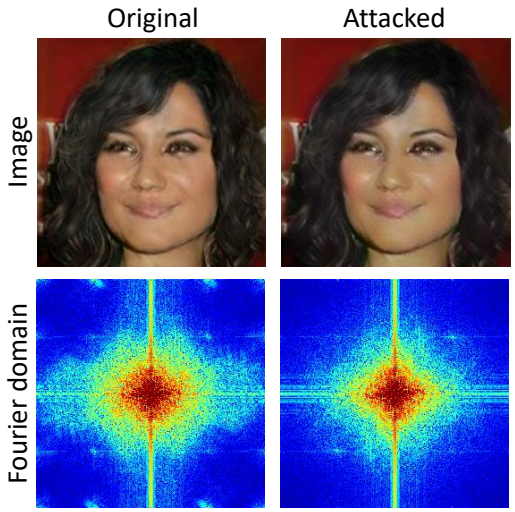

Figure 4: The first row shows the original image and its version after the attack. In the second row, we present the corresponding Fourier transforms. It can be seen that the GAN images present peaks in Fourier domain caused by the up-sampling modules present in the architecture. These artifacts reduce after the image has been attacked by the injection of the camera traces.

vides worse results. This is also confirmed by the findings of Rössler et al. [47] on the robustness of very deep CNNs, while proving the fragility of the spectrum-based method (Spec). Nonetheless, after modifying the images with our generator, Xception's TPR reduces drastically to 2.50%.

5.2. Fooling camera model ID detectors

In this section we analyze the ability of our method to deceive camera model identifiers. To this end, we consider four CNN-based target classifiers. The first two architectures, Tuama2016 [52], and Bondi2017 [5], have been proposed specifically for camera model identification. Moreover, in view of the results of the Kaggle competition on camera model identification, we also consider two deep general-purpose architectures, Xception [11] and InceptionV3 [50], trained to work as camera model classifiers. The performance of all these classifiers on our 10-models test set, in the absence of any attack, is shown in Tab. 3. Although all methods perform well, the very deep networks clearly outperform the other classifiers.

We will compare the results of our proposal with four baseline methods for the generation of adversarial attacks: Projected Gradient Descent (PGD)[5] [35], Translation-Invariant Momentum Iterative Fast Gradient Sign Method (TI-MI-FGSM) [15], Generative Adversarial Perturbation (GAP)[6] [45], and Generative Adversarial Attack against Camera Identification (Adv-Cam-Id) [8]. All these methods perform white-box attacks, i.e., they are trained with reference to a known target classifier. Here, we will consider both Bondi2017 and Xception as target classifiers. However, to assess the transferability of the attack, we will evaluate performance also on all off-training classifiers. To improve transferability, we also perform training an ensemble of classifiers, as proposed in [15]; performance is evaluated on the fourth one. For a fair comparison, we set the parameters of all methods in order to obtain a PSNR of ≈31dB. We refer to the original papers for further detail.

Tuama2016	Bondi2017	Xception	InceptionV3
74.1%	87.1%	97.0%	95.4%

Table 3: Accuracy of camera model identification evaluated on 25000 patches of dimension equal to 256×256 pixels.

Performance is measured in terms of a Successful Attack Rate (SAR), the fraction of modified synthetic images that are classified as acquired by the target model. Results are computed on the central 256×256 crop of 10000 synthetic images, and averaged over 5 target models (Motorola DroidMaxx, Samsung GalaxyS4, Sony Nex7, iPhone 4s and, Motorola X). Note that in the absence of any attack, the average SAR is 10% for our 10-class setting. Results are reported in Tab. 4 (top).

Several considerations are in order. First of all, it is clear that all white-box attacks are very effective when tested on the very same classifier used during training, Bondi2017 in the first block, Xception in the second one. Such results are emphasized with red text. However, there is no reason to expect such a scenario in practice, as the defender is free to use any classifier for camera identification. Therefore, the most interesting results are those in black. On off-training classifiers, only Adv-Cam-Id, among the baselines, provides a reasonably good performance, never exceeding 60% though. Instead, SpoC performs quite well on all classifiers, with SAR going from 53% to 81%. With respect to Adv-Cam-Id, the best baseline, it improves from 10% to 30%. Training on an ensemble of classifiers should ensure higher transferability. However, the results of the third block do not seem especially encouraging, with only minor improvements and an average SAR barely exceeding 10%.

We also consider a scenario where images are JPEG compressed using the quantization table of the target camera model and we repeat the analysis, see Tab. 4 (bottom). This corresponds to a realistic setting where images are compressed before being spread over the web. In this case, two contrasting phenomena occur. On one side, JPEG compression acts as a defense against adversarial perturbations reducing their strength. On the other side, since the images

5. https://github.com/BorealisAI/advertorch

6. https://github.com/OmidPoursaeed/Generative_Adversarial_Perturbations

Uncompressed Images

Target Arch.	Method	PSNR	Model Classifiers			
			Tuama2016	Bondi2017	Xception	Incept.V3
Bondi2017	PGD	30.98	12.31	96.23	0.49	2.65
	TI-MI-FGSM	31.51	8.05	63.56	9.02	0.59
	GAP	30.85	9.91	74.34	2.46	17.16
	Adv-Cam-Id	30.24	18.38	100.00	50.08	60.70
Xception	PGD	**33.73**	8.80	8.27	96.84	0.30
	TI-MI-FGSM	30.93	12.27	8.68	99.95	1.82
	GAP	31.51	6.69	1.93	98.00	21.29
	Adv-Cam-Id	29.91	23.29	41.35	99.98	42.19
Ensemble (off-target)	PGD	32.88	12.23	11.39	5.06	2.22
	TI-MI-FGSM	31.32	13.24	9.58	13.10	2.80
	GAP	31.04	26.59	4.83	18.22	17.91
	SpoC (ours)	31.71	**53.34**	**67.57**	**81.37**	**71.65**

JPEG Compressed Images

Target Arch.	Method	PSNR	Model Classifiers			
			Tuama2016	Bondi2017	Xception	Incept.V3
Bondi2017	PGD	30.98	1.02	96.17	0.03	0.84
	TI-MI-FGSM	31.38	31.21	31.72	1.40	2.00
	GAP	31.69	20.71	67.41	3.09	22.35
	Adv-Cam-Id	30.18	24.08	89.46	54.51	58.79
Xception	PGD	**33.52**	24.71	6.06	98.21	0.19
	TI-MI-FGSM	30.77	35.05	3.16	87.94	6.51
	GAP	32.09	5.36	0.57	96.42	5.54
	Adv-Cam-Id	29.69	37.59	37.19	97.52	43.28
Ensemble (off-target)	PGD	32.70	20.31	9.21	1.16	0.53
	TI-MI-FGSM	31.07	29.98	3.23	5.24	5.48
	GAP	31.97	25.45	4.25	12.10	6.79
	SpoC (ours)	31.41	**55.57**	**64.73**	**73.43**	**69.25**

Table 4: Averaging results on 50000 images attacked using 5 different camera models. Performance is measured in terms of a Successful Attack Rate (SAR). We compare with white-box attacks on Bondi2017 (first block) and Xception (second block) and hence discard values on such architecture for the analysis (red). We also compare with methods using an ensemble of classifiers (third block). Results on attacked images (top) and on their JPEG compressed version (bottom).

have been compressed using the same quantization matrix of the camera model under attack, some performance improvement can arise thanks to these specific traces. For these reasons, the edge of the proposed method with respect to competitors is slightly reduced. Instead, in terms of model classifiers, the network proposed by Tuama et al. [52] gets significantly worse, as it relies heavily on compression artifacts to distinguish different camera models.

6. Conclusion

In this work, we proposed a GAN-based method to attack forensic camera model identifiers. Our scheme allows to inject model-specific traces into the input image such to deceive the classifier into believing the image was acquired by the desired target camera model. Moreover, the method requires no prior knowledge on the attacked network, and works even on completely synthesized images, removing also traces of the synthesis process.

Experimental results prove the effectiveness of the proposed method and, as a consequence, the weaknesses of current forensic detectors, calling for new approaches, less dependant on critical hypotheses, and more resilient to unforeseen attacks.

Acknowledgement

We gratefully acknowledge the support of this research by the AI Foundation, a TUM-IAS Rudolf Mößbauer Fellowship, and Google Faculty Award. In addition, this material is based on research sponsored by the Air Force Research Laboratory and the Defense Advanced Research Projects Agency under agreement number FA8750-16-2-0204. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory and the Defense Advanced Research Projects Agency or the U.S. Government.

References

[1] Shruti Agarwal and Hany Farid. Photo Forensics from JPEG Dimples. In *IEEE International Workshop on Information Forensics and Security*, 2017. 2

[2] Michael Albright and Scott McCloskey. Source Generator Attribution via Inversion. In *IEEE CVPR Workshops*, 2019. 3

[3] Mauro Barni, Kassem Kallas, Ehsan Nowroozi, and Benedetta Tondi. On the Transferability of Adversarial Examples against CNN-based Image Forensics. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018. 3

[4] Mauro Barni, Matthew Stamm, and Benedetta Tondi. Adversarial multimedia forensics: Overview and challenges ahead. In *European Signal Processing Conference (EUSIPCO)*, pages 962–966, 2018. 2, 3

[5] Luca Bondi, Luca Baroffio, David Güera, Paolo Bestagini, Edward Delp, and Stefano Tubaro. First steps toward camera model identification with convolutional neural networks. *IEEE Signal Processing Letters*, 24(3):259–263, 2017. 3, 7

[6] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis, 2018. 6, 14

[7] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019. 1

[8] 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, 4, 7, 13

[9] Mo Chen, Jessica Fridrich, Miroslav Goljan, and Jan Lukàš. Determining image origin and integrity using sensor noise. *IEEE Trans. Inf. Forensics Security*, 3:74–90, 2008. 1

[10] 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. 1, 6

[11] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3, 6, 7

[12] Davide Cozzolino and Luisa Verdoliva. Noiseprint: a CNN-based camera model fingerprint. *IEEE Trans. Inf. Forensics Security*, (1):14–27, Jan. 2020. 2

[13] Debayan Deb, Jianbang Zhang, and Anil K. Jain. AdvFaces: Adversarial Face Synthesis. *arXiv preprint arXiv:1908.05008v1*, 2019. 3

[14] Ahmet Emir Dirik, Hüsrev Taha Sencar, and Nasir Memon. Analysis of seam-carving based anonymization of images against PRNU noise pattern-based source attribution. *IEEE Trans. Inf. Forensics Security*, 9(12):2277–2290, 2014. 3

[15] 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. 7, 12

[16] John Entrieri and Matthias Kirchner. Patch-Based Desynchronization of Digital Camera Sensor Fingerprints. In *Electronic Imaging*, pages 1–9, 2016. 3

[17] Pasquale Ferrara, Tiziano Bianchi, Alessia De Rosa, and Alessandro Piva. Image forgery localization via fine-grained analysis of CFA artifacts. *IEEE Trans. Inf. Forensics Security*, 7(5):1566–1577, 2012. 2

[18] Thomas Gloe, Matthias Kirchner, Antje Winkler, and Rainer Böhme. Can we trust digital image forensics? In *ACM International Conference on Multimedia*, pages 78–86, 2007. 3

[19] Miroslav Goljan, Jessica Fridrich, and Mo Chen. Defending against fingerprint-copy attack in sensor-based camera identification. *IEEE Trans. Inf. Forensics Security*, (6):227–236, March 2011. 3

[20] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. 2015. 3

[21] Diego Gragnaniello, Francesco Marra, Giovanni Poggi, and Luisa Verdoliva. Analysis of adversarial attacks against CNN-based image forgery detectors. In *European Signal Processing Conference*, pages 384–389, 2018. 3

[22] David Güera, Yu Wang, Luca Bondi, Paolo Bestagini, Stefano Tubaro, and Edward Delp. A Counter-Forensic Method for CNN-Based Camera Model Identification. In *IEEE CVPR Workshops*, 2017. 2, 3

[23] Weiwei Hu and Ying Tan. Generating Adversarial Malware Examples for Black-Box Attacks Based on GAN. *arXiv preprint arXiv:1702.05983v1*, 2019. 3

[24] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3

[25] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 4

[26] Zhiwei Jin, Juan Cao, Yongdong Zhang, Jianshe Zhou, and Qi Tian. Novel visual and statistical image features for microblogs news verification. *IEEE Trans. on Multimedia*, 19(3):598–608, 2017. 1

[27] 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. 5, 13

[28] 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. 1, 6

[29] 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. 1, 6

[30] Matthias Kirchner and Rainer Böhme. Synthesis of color filter array pattern in digital images. In *Media Forensics and Security*, 2009. 2

[31] Matthias Kirchner and Thomas Gloe. Forensic camera model identification. In T.S. Ho and S. Li, editors, *Handbook of Digital Forensics of Multimedia Data and Devices*, pages 329–374. Wiley-IEEE Press, 2015. 2

[32] Yanpei Liu, Xinyun Chen, Shanghai Jiao Tong, Chang Liu, and Dawn Song. Delving into tranferable adversarial examples and black-box attacks. In *International Conference on Learning Representations*, 2017. 3

[33] Jan Lukàš, Jessica Fridrich, and Miroslav Goljan. Digital camera identification from sensor pattern noise. *IEEE Trans. Inf. Forensics Security*, 1(2):205–214, 2006. 2

[34] Siwei Lyu, Xunyu Pan, and Xing Zhang. Exposing Region Splicing Forgeries with Blind Local Noise Estimation. *International Journal of Computer Vision*, 110(2):202–221, 2014. 2

[35] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv:1706.06083*, 2017. 7, 12

[36] Francesco Marra, Diego Gragnaniello, Giovanni Poggi, and Luisa Verdoliva. Detection of GAN-Generated Fake Images over Social Networks. In *IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 384–389, 2018. 3, 6

[37] Francesco Marra, Diego Gragnaniello, and Luisa Verdoliva. On the vulnerability of deep learning to adversarial attacks for camera model identification. *Signal Processing: Image Communication*, 65, 2018. 2, 3

[38] Francesco Marra, Diego Gragnaniello, Luisa Verdoliva, and Giovanni Poggi. Do GANs leave artificial fingerprints? In *IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 506–511, 2019. 2

[39] Falko Matern, Christian Riess, and Mark Stamminger. Exploiting visual artifacts to expose deepfakes and face manipulations. In *IEEE WACV Workshop on Image and Video Forensics*, 2019. 1

[40] Owen Mayer, Belhassen Bayar, and Matthew C. Stamm. Learning unified deep-features for multiple forensic tasks. In *ACM Workshop on Information Hiding and Multimedia Security*, pages 79–84, 2018. 3

[41] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018. 12

[42] Huaxiao Mo, Bolin Chen, and Weiqi Luo. Fake Faces Identification via Convolutional Neural Network. In *ACM Workshop on Information Hiding and Multimedia Security*, June 2018. 3

[43] Ryota Natsume, Tatsuya Yatagawa, and Shigeo Morishim. RSGAN: Face Swapping and Editing using Face and Hair Representation in Latent Spaces. In *ACM SIGGRAPH*, 2018. 1

[44] Yuval Nirkin, Yosi Keller, and Tal Hassner. FSGAN: Subject Agnostic Face Swapping and Reenactment. In *IEEE International Conference on Computer Vision*, Oct. 2019. 1

[45] 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. 3, 7, 13

[46] Shengju Qian, Kwan-Yee Lin, Wayne Wu, Yangxiaokang Liu, Quan Wang, Fumin Shen, Chen Qian, and Ran He. Make a Face: Towards Arbitrary High Fidelity Face Manipulation. In *IEEE International Conference on Computer Vision*, 2019. 1

[47] Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. FaceForensics++: Learning to Detect Manipulated Facial Images. In *International Conference on Computer Vision (ICCV)*, 2019. 6

[48] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 5

[49] Yang Song, Rui Shu, Nate Kushman, and Stefano Ermon. Constructing Unrestricted Adversarial Examples with Generative Models. In *Conference on Neural Information Processing Systems (NIPS)*, 2018. 3

[50] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 3, 7

[51] Justus Thies, Michael Zollhöfer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2Face: Real-Time Face Capture and Reenactment of RGB Videos. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2387–2395, June 2016. 1

[52] Amel Tuama, Frédéric Comby, and Marc Chaumont. Camera model identification with the use of deep convolutional neural networks. In *IEEE Workshop on Information Forensics and Security (WIFS)*, pages 1–6, 2016. 3, 4, 7, 8

[53] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional GANs. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 1, 5

[54] Xiaosen Wang, Kun He, and John E. Hopcroft. AT-GAN: A Generative Attack Model for Adversarial Transferring on Generative Adversarial Nets. *arXiv preprint arXiv:1904.07793v3*, 2019. 3

[55] 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. 6

[56] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating Adversarial Examples with Adversarial Networks. In *International Joint Conference on Artificial Intelligence*, 2018. 3

[57] Xin Yang, Yuezun Li, Honggang Qi, and Siwei Lyu. Exposing GAN-synthesized Faces using Landmark Locations. In *ACM Workshop on Information Hiding and Multimedia Security*, pages 113–118, 2019. 1

[58] Ning Yu, Larry Davis, and Mario Fritz. Attributing Fake Images to GANs: Learning and Analyzing GAN Fingerprints. In *IEEE International Conference on Computer Vision (ICCV)*, 2019. 3

[59] Xu Zhang, Svebor Karaman, and Shih-Fu Chang. Detecting and Simulating Artifacts in GAN Fake Images. In *IEEE Workshop on Information Forensics and Security (WIFS)*, 2019. 3, 6

[60] 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. 1, 6

Supplemental Material

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 state-of-the-art detectors to believe the image was acquired by that model. In this supplemental document, we report the details of the architectures used for Generator, Discriminator, and Embedder (Sec. A). Moreover, we carry out ablation studies to validate our choices (Sec. B). For reproducibility, we report the parameters of the used comparison methods used in the main paper (see Sec. C). Finally, we analyze the scenario where we attack images generated by a GAN architecture not used in the training set (see Sec. D).

A. Architectures

Generator Our generator is composed of seven convolutional layers with a fixed stride equal to one (see Fig.5). 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 [41]. 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.6.

Embedder As shown in Fig.7, 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 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.

B. Ablation study

To show the importance of the usage of an embedder and a discriminator in the training scheme, we compare two variants. In the first variant, *only-embedder*, we remove the discriminator by setting $\lambda_D = 0$ for the loss of the generator (Equation 1 of the main paper). While in the second variant, *only-discriminator*, we remove the embedder by setting $\lambda_E = 0$ for the loss of the generator. The other hyperparameters are not modified. Tab. 5 shows the performance of the variants to deceive four camera-model classifiers. Performance is measured in terms of Successful Attack Rate (SAR). As can be seen this supports our choices. The *only-embedder* variant obtains the worst results with the maximum SAR of 48.13% for Tuama2016 and a SAR lower than 15% for the other classifiers, while the *only-discriminator* variant performs worse for all the four camera-model classifiers.

C. 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) [35]: 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) [15]: 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.

Figure 5: Generator architecture.

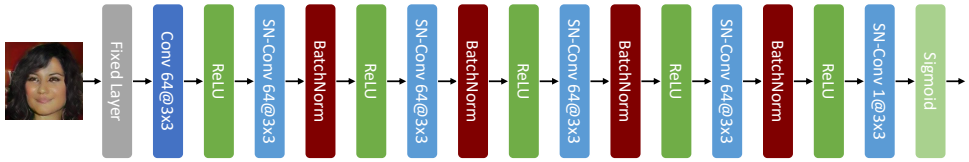

Figure 6: Discriminator architecture.

		Model Classifiers			
Method	PSNR	Tuama2016	Bondi2017	Xception	Incept.V3
Proposal	31.71	**53.34**	**67.57**	**81.37**	**71.65**
only-embedder	49.71	48.13	6.76	10.75	13.18
only-discriminator	31.18	47.91	63.81	66.73	70.36

Table 5: Averaged results on 50000 images attacked using 5 different camera models. Performance is measured in terms of Successful Attack Rate (SAR).

GAP (Generative Adversarial Perturantion) [45]: 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 [27]. In our experiments, the epsilon of the constraint is set equal to 8.

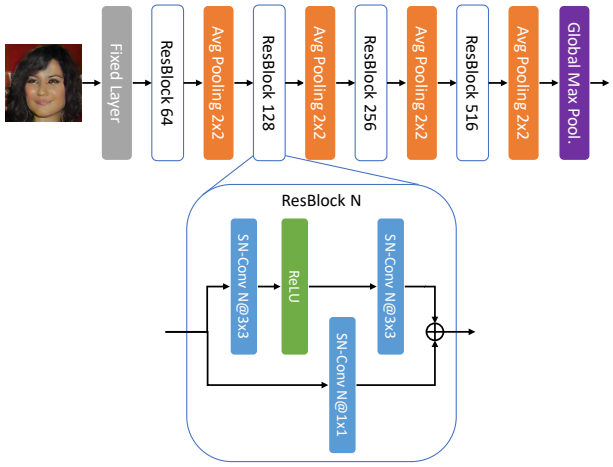

Figure 7: Embedder architecture.

Adv-Cam-Id [8]: The white-box attack proposed in [8] 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.

D. Generalization

Both the proposed technique and the reference techniques based on a generative network (GAP, and Adv-Cam-Id) require a training phase. To test their ability to generalize, we add a further experiment, on images generated by a GAN architecture not used in the training set. In Table 6 we show the results. GAP shows about the same performance, measured in terms of SAR and PSNR, on data generated

			Model Classifiers			
Target Arch.	Method	PSNR	Tuama2016	Bondi2017	Xception	Incept.V3
Bondi2017	PGD	30.67	9.33	82.78	1.24	3.50
	TI-MI-FGSM	31.51	5.88	23.08	1.89	0.32
	GAP	30.82	5.34	46.00	7.16	6.70
	Adv-Cam-Id	25.29	17.95	99.87	37.66	58.88
Xception	PGD	34.04	5.06	1.13	97.60	0.15
	TI-MI-FGSM	30.91	7.83	0.57	99.98	0.57
	GAP	31.27	6.57	0.67	87.97	7.32
	Adv-Cam-Id	25.32	23.20	44.36	99.36	42.34
Ensemble (off-target)	PGD	32.73	12.46	1.90	7.34	0.68
	TI-MI-FGSM	31.29	8.56	0.93	4.04	0.77
	GAP	30.99	24.55	0.48	17.28	8.87
	SpoC (ours)	31.93	**53.95**	**53.05**	**78.60**	**68.00**

Table 6: Averaging results on 10000 images attacked using 5 different camera models using a GAN architecture [6] outside the training set. Performance is measured in terms of a Successful Attack Rate (SAR). We compare with white-box attacks on Bondi2017 (first block) and Xception (second block) and hence discard values on such architecture for the analysis (red). We also compare with methods using an ensemble of classifiers (third block).

by architectures outside and inside the training-set. On the contrary Adv-Cam-Id presents a PSNR loss of about 5dB, while preserving a reasonably good SAR. Finally, comparing the SpoC with respect to the other methods, it continues to have the better results, with SAR going from 53% to 78%, without any reduction in PSNR.