

Detection of Face Recognition Adversarial Attacks

Fabio Valerio Massoli^{a,*}, Fabio Carrara^a, Giuseppe Amato^a, Fabrizio Falchi^a

^a*ISTI-CNR, via G. Moruzzi 1, 56124 Pisa, Italy*

Abstract

Deep Learning methods have become state-of-the-art for solving tasks such as Face Recognition (FR). Unfortunately, despite their success, it has been pointed out that these learning models are exposed to *adversarial* inputs — images to which an imperceptible amount of noise for humans is added to maliciously fool a neural network — thus limiting their adoption in sensitive real-world applications. While it is true that an enormous effort has been spent in order to train robust models against this type of threat, adversarial detection techniques have recently started to draw attention within the scientific community. A detection approach has the advantage that it does not require to re-train any model, thus it can be added on top of any system. In this context, we present our work on adversarial samples detection in forensics mainly focused on detecting attacks against FR systems in which the learning model is typically used only as a features extractor. Thus, in these cases, train a more robust classifier might not be enough to defence a FR system.

In this frame, the contribution of our work is four-fold: i) we tested our recently proposed adversarial detection approach against classifier attacks, i.e. adversarial samples crafted to fool a FR neural network acting as a classifier; ii) using a k-Nearest Neighbor (kNN) algorithm as a guidance, we generated deep features attacks against a FR system based on a DL model acting as features extractor, followed by a kNN which gives back the query identity based on features similarity; iii) we used the deep features attacks to fool a FR system

*Corresponding author

Email address: fabio.massoli@isti.cnr.it (Fabio Valerio Massoli)

on the 1:1 Face Verification task and we showed their superior effectiveness with respect to classifier attacks in fooling such type of system; iv) we used the detectors trained on classifier attacks to detect deep features attacks, thus showing that such approach is generalizable to different types of offensives.

Keywords: Deep Learning, Face Recognition, Adversarial Attacks, Adversarial Detection, Adversarial Biometrics

1. Introduction

Deep Learning (DL) quickly occupied a central role in recent AI-related technological breakthroughs covering multiple fields and applications: vision (e.g., image classification [1], object detection [2]), natural language processing [3] and the combination of them (e.g., multi-modal [4], sentiment analysis [5]). Despite achieving state-of-the-art performance in many scenarios, deep learning models still suffer from deficiencies that strongly limit their adoption in sensitive applications. Among others, the vulnerability of DL models in adversarial settings still poses challenges: it is relatively easy for an attacker to manipulate the output of a model by tampering its input often in an imperceptible way. The existence of these perturbed inputs — known as *adversarial examples* [6, 7] — constitutes one of the major roadblocks in security-related applications such as DL-based biometrics systems for surveillance and access control that, despite performing brilliantly in natural settings [8], can be easily evaded by knowledgeable adversaries. Face Recognition enabled by Deep Neural Networks (DNN) is a case in point. Several successful applications of deep models to FR have been proposed in the literature [9–11]. Indeed, this kind of technology enables AI surveillance programs in multiple countries [12] and has already found its way into consumers products [8]. However, researchers already showed how this kind of systems can be jeopardized by adversarial attacks both in the digital [13, 14] and physical domain [15, 16].

In order to counteract adversarial vulnerability, a considerable research effort provided a multitude of defensive approaches for adversarial attacks that can be

roughly categorized in two methodologies, that is *rectification* and *adversarial input detection*. In rectification methods, the goal is to recover the intended output of the model by increasing the robustness of the system, e.g. by trying to remove adversarial perturbation from the input [17, 18] or by increasing the robustness of the model itself [19, 20]. On the other hand, adversarial detection aims at detecting an occurred attack by analyzing the behavior of the model (without changing it) and signaling anomalous events [21–24]. Notwithstanding, many of the proposed adversarial detection methods fall prey to strong adversaries too [25], recent techniques exploiting the training data manifold to ground the predictions of a model [26, 27] exhibit good trade-offs between detection performance and resilience to attacks [28] (as well as tackling a more general problem, that is obtaining good confidence measurements for predictions of deep models [29]).

While most of the adversarial detection schemes are tested on small or low-resolution benchmarks (such as MNIST and CIFAR datasets), this work aims at evaluating one of the aforementioned training-manifold-based adversarial detection methodologies, specifically [30], in a realistic security-related application that is facial recognition.

Facial recognition systems usually do not usually implement recognition based on deep-learning classifiers but rather follow a similarity-based approach: deep models are used to extract features from visual facial data, and decisions rely on similarity measurements among those features. Indeed, standard benchmarks for facial recognition, such as IJB-B [31] and IJB-C [32], define two evaluation protocols, that is 1:1 Face Verification and 1:N Face Identification. The former requires to investigate if a person’s identity is known or not by comparing its features vector against a database of known identities, while the latter requires to match two images to assess if they belong to the same person or not.

Sticking to those protocols, we provide an analysis of adversarial attacks and further detection in facial recognition systems that implement face identification and verification relying on state-of-the-art deep learning models. In particular,

our contributions are the following: **i)** we tested our recently proposed detection technique [30] against classifier attacks, i.e. adversarial samples crafted to fool a state-of-the-art FR neural network acting as a classifier; **ii)** we generated deep features attacks, using a kNN algorithm as a guidance, to attack a FR system that fulfills the Face Identification task by means of a DL model, acting as a backbone features extractor, followed by a kNN which gives back the query identity based on features similarity; **iii)** we used deep features attacks to fool a FR system on the Face Verification task, and we showed their superior effectiveness with respect to classifier attacks in fooling such type of system; **iv)** we used the detectors trained on classifier attacks to detect deep features attacks, thus showing that such approach is generalizable to different types of attacks. The rest of the paper is organized as follows. In Section 2, we briefly reviewed some related works. In Section 3, we described the algorithms used to craft adversarial examples, while in Section 4, we described the adversarial detection technique used in our study. In Section 5, we presented the experimental campaigns that we conducted, and finally, in Section 6, we reported the conclusions of our work.

2. Related Work

2.1. Adversarial Attacks

After the seminal work of [7] in which adversarial examples were first studied in DNN, in the last years an exploding growth in studies of adversarial attacks and defenses has been witnessed. Since the early works, the abundant presence of adversarial examples for standard deep neural networks was confirmed by researchers who proposed multiple crafting algorithms to efficiently find them. Among the most relevant attacking algorithms available in the literature, there are the box-constrained L-BFGS [7], FGSM and its variants [16, 33, 34], and CW [35]. We dedicated Section 3 for a more detailed review of these algorithms, as we adopted them in this work to generate adversarial examples.

2.2. Face Recognition Adversarial Attacks

Face Recognition is among the most important topics in computer vision. This field has drawn the attention of the scientific community since the early 90s, when [36] proposed the Eigenfaces approach. DL models, especially leveraging on the properties of Deep Convolutional Neural Network, started to dominate this field since 2012 reaching performances up to 99.80% [37], thus overcoming human performance on this task. Despite the effort in training very robust DL models, such systems still show some weaknesses. For example, it has been shown that state-of-the-art face classifiers experience a performance drop when tested against low resolution images [38].

Moreover, they are vulnerable to adversarial attacks considering both the black-box [13] and white-box [14, 15] settings.

Concerning the attacks to face recognition systems, Sharif et al. [15] demonstrated the feasibility and effectiveness of physical attacks by dodging recognition and impersonating other identities using eyeglass frames with a malicious texture. Dong et al. [13] successfully performed black-box attacks on face recognition models and demonstrated their effectiveness in a real-world deployed system. Modern attacks on facial recognition systems either exploit generative models obtaining a more natural perturbation [14] or find natural adversarial examples by modifying identity-independent attributes [39, 40], such as hair color, makeup, or the presence of glasses.

Pautov et al. [41] focused on physical world attacks to the LResNet100E-IR FR system. Specifically, they realized adversarial patches that when attached to the area of the face of a person, such as eyes, nose or forehead, or when projected on wearable accessories, allowed the attacker to fool the FR system by leading it to recognize him or her with a different identity.

2.3. Adversarial Defenses

Obtaining a system that is robust to adversarial examples turned out to be a challenging and still open task. The robustness of a model can be increased via adversarial training [33, 42] or model distillation [43]. In general, techniques

that try to smooth, change, or hide the gradient surface of the model seen by an attacker called gradient-masking defenses, are able to increase the attack effort needed to find an adversarial example, but the enhanced model is still vulnerable to stronger attacks.

Another strategical direction consists of detecting adversarial examples, that is creating robust systems composed by a vulnerable model and a detection system that signals occurring attacks. Detection subsystems are often implemented as binary detectors that discern authentic and adversarial inputs. Gong et al. [21] proposed to train an additional binary classifier that decides whether an input image is pristine or tampered. Grosse et al. [22] adopted statistical tests in the pixel space to demonstrate the discernibility of adversarial images and proposed to introduce the "adversarial" class in the original classifier which is contextually trained with the model. Similarly, Metzen et al. [24] proposed a detection subnetwork that relies on intermediate representations constructed by the model at inference time. However, many detection schemes have been proven to be bypassable [25].

Novel detection methods rely on the training data manifold for grounding the model prediction and detect anomalies. Carrara et al. [30] and Papernot and McDaniel [27] showed that a kNN scheme based on intermediate representations of the training set can be used to define a score that measures the confidence of the classification produced by a deep model: such score can then be used to filter out adversarial examples but also authentic errors occurring. To cope with the computational cost incurred by a kNN scheme on huge training sets, Carrara et al. [26] proposed a method that embeds multiple representations in the training space via a distance-based transformation and then performs detection in this space.

To our knowledge, the most relevant work that copes with detecting tampered facial recognition is Goswami et al. [44], in which the authors attacked facial recognition systems in a classification setting and devised a detection approach to decide whether to recover the original input. In the detection part, they proposed to compare intermediate network activations to their average val-

ues defined over a training set, and used layer-wise distances as features in a two-class SVM adversarial detector. However, their analysis only included the recognition-by-classification setting, while we covered additional real-world settings, such as attacks on kNN identification and verification systems.

3. Adversarial Attacks

In this section, we described some of the most famous algorithms used for adversarial samples generation.

3.1. L-BFGS

Szegedy et al. [7] formalised the adversarial attack as an optimization problem that is solved by means of the L-BFGS algorithm. Specifically, it can be expressed as

$$\begin{aligned} \min_r \quad & c \cdot \|r\|_2 + \mathcal{L}(x+r, t) \\ \text{subject to} \quad & L^m \leq x+r \leq U^m, \end{aligned} \quad (1)$$

where $[L, U]^m$ represents the range of validity for pixel values, and the value of $c > 0$ is found by line-search. The goal of the optimizer is then to find the minimum adversarial perturbation r to the input image x which causes the model to classify $x_{adv} = x+r$ as belonging to the target class t .

3.2. FGSM

The Fast Sign Gradient Method [33] (FGSM) is a one-step method in which the optimal max-norm constrained perturbation is found by following the direction of the gradient $\nabla_x J(\theta, x, y)$ of the objective function used to train the DL model with respect to the input image $x \in \mathbb{R}^m$. The adversarial example is then given by

$$x_{adv} = x + \epsilon \cdot \text{sign}(\nabla_x J(\theta, x, y_{true})), \quad (2)$$

where θ are the model parameters, x is the input image, y_{true} is its label, and ϵ is the maximum distortion allowed on the input such that $\|x - x_{adv}\|_\infty < \epsilon$.

3.3. BIM

The Basic Iterative Method [16] (BIM) applies the FGSM [33] attack multiple times with small step size. It is given by

$$\begin{aligned} x_0^{adv} &= x, \\ x_{N+1}^{adv} &= \text{Clip}_{x,\epsilon} \{ x_N^{adv} + \alpha \cdot \text{sign}(\nabla_x J(\theta, x_N^{adv}, y_{true})) \}, \end{aligned} \quad (3)$$

where the $\text{Clip}(\cdot)$ function clips the values of the pixels at each iteration step to the allowed pixel range, and α is the used step size.

3.4. MI-FGSM

The MI-FGSM method [34] is an iterative procedure that can be generalized to other types of attacks by substituting the current gradient with the accumulated ones from all the previous steps. The velocity vector in the gradient direction is given by

$$g_{N+1} = \mu \cdot g_N + \frac{J(x_N^{adv}, y)}{\|\nabla_x J(x_N^{adv}, y)\|_1}, \quad (4)$$

where $x_0^{adv} = x$, $g_0 = 0$, μ is the decay factor of the running average, and y is the ground truth label. Subsequently, the adversarial example in the ϵ -vicinity measured by L_2 distance is given by

$$x_{N+1}^{adv} = x_N^{adv} + \alpha \cdot \frac{g_{N+1}}{\|g_{N+1}\|_2}, \quad (5)$$

where $\alpha = \epsilon/T$ with T being the total number of iterations.

3.5. Carlini-Wagner Attacks

Carlini and Wagner [35] (CW) proposed three gradient-based attacks each based on a different distance metric, namely L_0 , L_2 and L_∞ attacks.

Given an input x and a target class t , different from the original class of the sample, the L_2 attack is given by

$$\begin{aligned} \min \quad & \left\| \frac{1}{2}(\tanh(w) + 1) - x \right\|_2^2 + c \cdot f \left(\frac{1}{2}(\tanh(w) + 1) \right) \\ \text{with} \quad & \\ f(x^{adv}) &= \max(\max\{Z(x^{adv})_i : i \neq t\} - Z(x^{adv})_t, -k), \end{aligned} \quad (6)$$

where f is the objective function, $Z(\cdot)$ are the logits before the softmax layer, w is the variable that represent the adversarial noise in the $\tanh(\cdot)$ space, and k is a parameter that allows to control the confidence with which the misclassification occurs.

Concerning the L_∞ attack, it is not fully differentiable, and the standard gradient descent does not perform well for it. Equation 7 shows the L_∞ version of the attack:

$$\text{minimize } c \cdot f(x + \delta) + \sum_i [(\delta_i - \tau)^+]. \quad (7)$$

where τ is a threshold value for the adversarial perturbation. Finally, the L_0 attack is based on the idea of iteratively use L_2 to find a minimal set of pixels to be modified to generate an adversarial sample.

3.6. Deep Features Attack

All the previous attacks were based on the goal of generating noise which fools the DL model to output a wrong class label for the specific input. Sabour et al. [45] proposed an approach in which the guiding principle was to create a perturbation of the input image in such a way that its internal representation was similar to the one of a target image. Starting from a source image I_s and a guide image I_g , the goal was to perturb I_s thus generating a new image I_α such that its internal representation, at a layer k in the model, $\phi_k(I_\alpha)$, generated by the DL model under attack, had an Euclidean distance from $\phi_k(I_g)$ as small as possible, while I_α remained close to the source I_s . Specifically, I_α was defined to be the solution to the constrained optimization problem

$$\begin{aligned} I_\alpha &= \arg \min_I \| \phi_k(I) - \phi_k(I_g) \|_2^2, \\ &\text{subject to } \| I - I_s \|_\infty < \delta, \end{aligned} \quad (8)$$

where δ was the maximum allowed perturbation on each pixel of the source image.

4. Adversarial Detection Method

The ultimate goal of our work was to detect adversarial samples. In order to accomplish that, we started from the approach we exploited in [30]. During the forward step of the threatened model, we collected the deep features, at the output of specific layers, to which we subsequently applied an average pooling operation, thus obtaining a single features vector at each selected layer. Then, we computed the distance among each vector and the class representatives, centroids or medoids, of each class, at each layer, obtaining an embedding which represented the trajectory of the input image in the features space. Such a trajectory was then fed to a binary classifier that is used as adversarial detector.

In our experiments, we used the test set of the VGGFace2 [9] dataset, which comprises 500 identities, and the state-of-the-art Se-ResNet-50 from [9]. Specifically, we extracted the deep features at the end of each of the 16 bottleneck blocks of the model. As the adversarial detector, we tested two different architectures: a Multi Layer Perceptron (MLP) and a Long-Short Term Memory (LSTM) network. The former was made by a hidden layer of 100 units followed by the ReLU non-linear function and a Dropout layer. The latter had a hidden state size of 100. In both cases, the output of the detector was fed into a Fully Connected (FC) layer followed by a sigmoid activation function. A schematic view of the entire system is shown in Figure 1.

Considering the 16 bottlenecks of the model from which we collected the deep features and the 500 different identities of the dataset, each embedding was then represented by a 8000-dimensional vector. In this vector, each i -th dimension represented the distance between the i -th internal representation and a class representative in a specific layer.

5. Experimental Results

In this section, we reported the experimental results we have obtained so far. First, we focused on the detection of the attacks against a state-of-the-

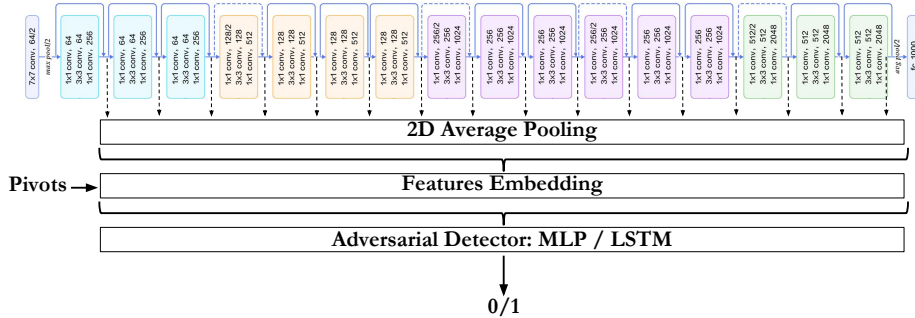


Figure 1: Schematic view of the detection algorithm. The deep features are extracted at the end of each of the bottlenecks of the Se-ResNet-50 model.

art FR model acting as a classifier. We crafted adversarial inputs by means of known algorithms, and then we trained and tested our detector against them. Afterwards, we generated deep features (DF) attacks (Subsection 3.6) using a kNN algorithm as guidance for the optimization procedure.

The goal of this approach was to fool a FR system in which the CNN was only used as a features extractor, while the final identity was assigned according to the output of a similarity measurement among deep features. In our experiments, we considered a system that used a kNN algorithm to assign an identity to the probe image. This is a typical solution for FR systems to fulfill the Face Identification task. Thereafter, we used these deep adversarial features to attack a FR system against the Face Verification task. Finally, we used the detectors, trained on the classifier attacks, to detect deep features attacks thus showing the generalization property of the detection approach.

5.1. Dataset

As we already stated, in our experiments we employed the test set of the VGGFace2 [9] dataset. It comprises 500 identities, with an average of ~ 340 images for each identity.

As a first step, for each of the 500 classes of the dataset, we randomly selected 10 images to be used as “natural” images and 10 to be used for adversarial

synthesis. These selected images were then used to train and test the adversarial detector.

We then split the remaining part of the dataset into train, validation and test sets and used them to train a FC layer on top of the state-of-the-art CNN we used in the study presented in Subsection 5.2.

5.2. Classifier Attacks and Detection

In the first set of experiments, we focused on the classifier attacks. As a first step, we replaced the classifier layer of the state-of-the-art facial recognition model [9] with a 500-ways FC layer, and we trained it. To train the model, we used the SGD optimizer with batch size of 256 and a learning rate of 10^{-3} halved every time the loss plateaus. As a preprocessing step, we resized the images so that the shortest side measured 256 pixels. Afterwards, we randomly cropped a 224x224 region of the image, and we subtracted the average pixel value channel-wise. For model evaluation, we used the same preprocessing with the exception that the random crop was substituted by a central crop.

In order to produce adversarial samples, we used the *foolbox*¹ implementation of the BIM [19], MI-FGSM [34], and CW [35], with L_2 norm, attacks. As far as the first two are concerned, we considered a maximum perturbation $\epsilon \in \{0.03, 0.07, 0.1, 0.3\}$, number of iterations $\in \{30, 50\}$, and for each combination, we considered the targeted and the untargeted versions of the attacks. The ϵ values were considered as fractions with respect to the maximum pixel value, which is 255. Instead, for the CW [35] attack, we considered the implemented default value of the parameters, i.e. 5 binary search steps and a number of max iterations equals to 1000. After the adversarial samples generation, we trained the detectors. The MLP and the LSTM were both trained using the Adam optimizer [46] for 150 epochs with a batch size of 256 and an initial learning rate ranging from 10^{-4} to 10^{-3} which was reduced by a factor 10 every time the loss reached a plateau. Moreover, to balance the sample distribution within

¹<https://foolbox.readthedocs.io/en/stable/>

mini-batches, we employed a weighted random sampler thus avoiding a bias towards attacks with higher multiplicity.

In Figure 2, we showed the Receiving Operating Characteristics (ROC) curves from the adversarial detection considering targeted and untargeted attacks for each architecture, distance metric, and class representative combination. As a summary, in Table 1 and Table 2, we reported the Area Under the Curve (AUC) values relative to each attack considering their targeted and untargeted versions, respectively.

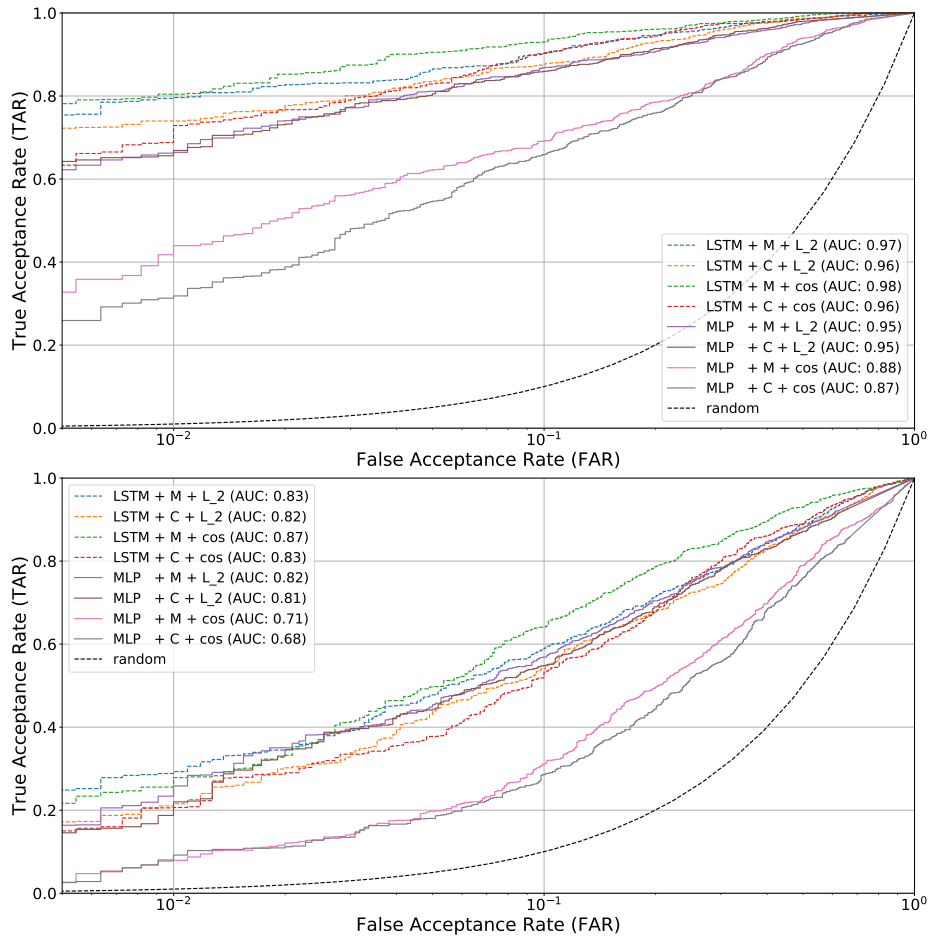


Figure 2: ROCs for each model and distance metric combination. Top: targeted attacks. Bottom: untargeted attacks

Table 1: Area Under the Curve (AUC) values for each configuration of architecture, pivot-selection and embedding function considering each targeted attack independently. The last column is a summary of the single-attacks AUCs.

Configuration	BIM	CW	MI-FGSM	Macro-AUC
LSTM + M + L_2	0.977	0.871	0.986	0.944
LSTM + C + L_2	0.970	0.857	0.982	0.936
LSTM + M + cos	0.986	0.904	0.991	0.960
LSTM + C + cos	0.968	0.895	0.981	0.948
MLP + M + L_2	0.964	0.793	0.979	0.912
MLP + C + L_2	0.962	0.808	0.979	0.916
MLP + M + cos	0.890	0.668	0.940	0.832
MLP + C + cos	0.868	0.720	0.915	0.834

Table 2: Area Under the Curve (AUC) values for each configuration of architecture, pivot-selection and embedding function considering each untargeted attack independently. The last column is a summary of the single-attacks AUCs.

Configuration	BIM	CW	MI-FGSM	Macro-AUC
LSTM + M + L_2	0.878	0.615	0.889	0.794
LSTM + C + L_2	0.863	0.596	0.869	0.776
LSTM + M + cos	0.929	0.599	0.930	0.819
LSTM + C + cos	0.884	0.568	0.886	0.779
MLP + M + L_2	0.885	0.559	0.882	0.775
MLP + C + L_2	0.874	0.557	0.874	0.768
MLP + M + cos	0.763	0.460	0.769	0.664
MLP + C + cos	0.730	0.467	0.739	0.645

As it was made clear from [Figure 2](#), [Table 1](#), and [Table 2](#), the LSTM with the medoids strategy gives the best results. We can also notice how the CW [\[35\]](#)

algorithm generates samples which are more difficult to be detected with respect to other algorithms. Moreover, it is clear how the untargeted attacks are typically more difficult to detect with respect to targeted attacks. A possible explanation is that these attacks typically find the closest adversarial to the input image, thus an embedding method based on the distance between representation may have difficulties in detecting such attacks. More details on this intuition were given in Subsection 5.3.

Finally, to visually understand the difference among the performances of the best detector on the various attacks, in Figure 3, we showed the relative ROCs.

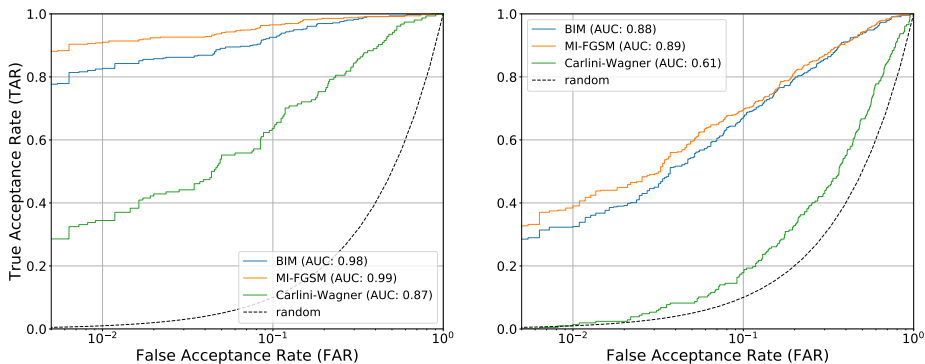


Figure 3: ROCs for each attack considering the best trained detector. Left: targeted attacks. Right: untargeted attacks.

According to Figure 3, it was made even clearer how hard it could be to detect adversarial samples generated by means of the CW [35] attack, especially considering untargeted attacks.

5.3. Deep Features Attacks

As previously described in Subsection 3.6, it is possible to use the distance among deep representations as a guiding principle to craft adversarial samples instead of wrong label assignment. Thus, nurturing this idea, we conducted new experiments in which we synthesized adversarial samples by using the distance among deep features as a guidance [45].

The main idea behind this approach was to emulate a real world application

scenario for a FR system in which a learning model is used as a features extractor whose output is used to fulfill the FR task by means of a similarity measurement.

5.3.1. Face Identification

As a real-world application case, we considered a system relying on a CNN and a kNN algorithm to accomplish the task of Face Identification. Specifically, what typically happens in this case is that the features vector extracted from the probe face image has to be compared against a database of known identities to identify the person. Each identity in the database is commonly represented by a template vector, i.e. a vector of features obtained by averaging several deep representations extracted from different images of the same person. Subsequently, the similarity among the probe vector and the available templates is computed. This is what is required, for example, when testing FR model performances on the IJB-B [31] and IJB-C [32] benchmark datasets. Following this principle, we evaluated the centroids for each of the 500 classes of the dataset. To conduct our experiments, we used the original state-of-the-art model from Cao et al. [9] as a features extractor.

The adversarial generation was formulated as an optimization problem [45] solved by using the L-BFGS-B algorithm. Specifically, the constraint was used to set a threshold, δ , on the maximum perturbation on each pixel of the original image as defined in Equation 8. In order to adapt the adversarial samples generation to our needs, we used a kNN classifier as guidance through the optimization procedure. The optimization was then stopped once the targeted or untargeted attack’s objective were met, that is, the kNN had classified the adversarial as belonging to the guide-image class or it had simply misclassified the face image considering targeted and untargeted attacks respectively. A schematic view of the algorithm is shown in Figure 4. In our experiments we considered the values of $\delta \in \{5.0, 7.0, 10.0\}$. An example of adversarial samples generated for each threshold value is shown in Figure 5. As we can see from Figure 5, the generated images look equal to the original ones, i.e. there is not evident trace of the guide image into the adversarial one. Considering targeted

attacks we obtained a success rate of 95.6%, 96.2% and 96.3% considering a value for $\delta = 5.0, 7.0, 10.0$, respectively. Instead, concerning untargeted attacks we obtained 96.8% success rate for $\delta = 5.0, 7.0, 10.0$, respectively. In all the samples generations we considered a maximum number of iterations equals to 700.

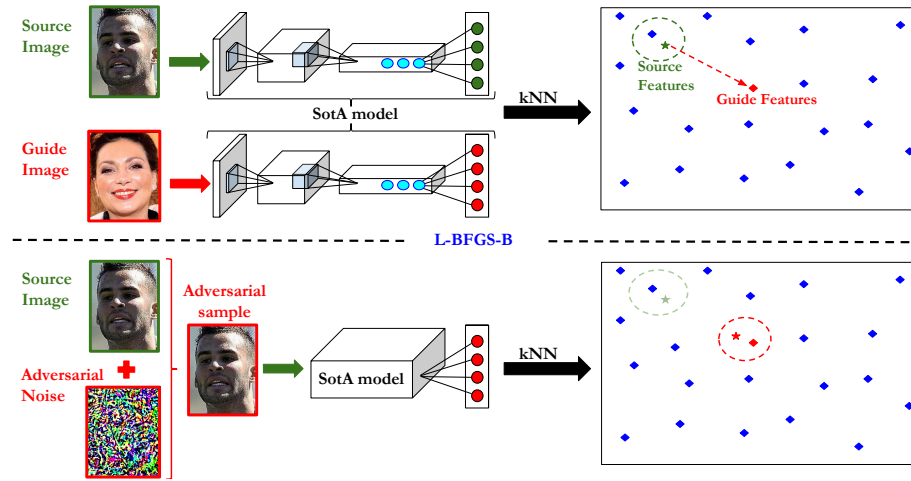


Figure 4: Schematic view of the adversarial generation procedure considering a state-of-the-art (SotA) model as features extractor and a kNN to asses the face identity.

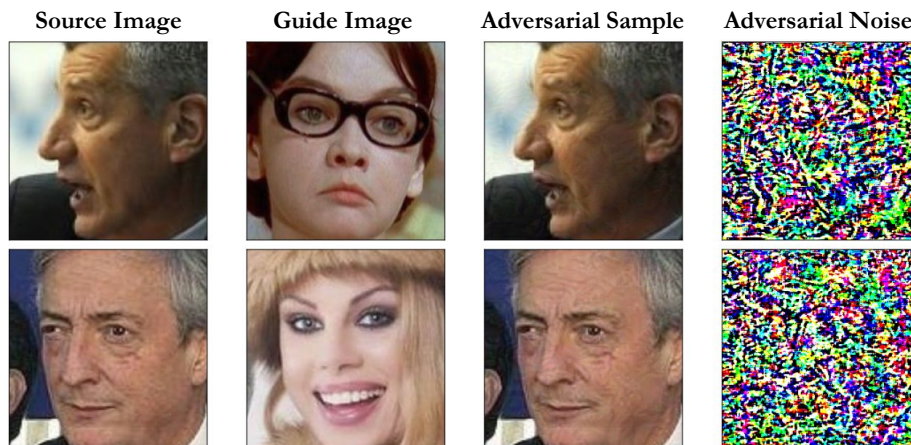


Figure 5: Adversarial samples for three different values of the threshold applied while solving the optimization problem. Top: $\delta = 5.0$. Middle: $\delta = 7.0$. Bottom: $\delta = 10.0$

Differently from what was previously done in Subsection 5.2, we formulated our approach with the purpose of fooling a FR system which was not based on the simple model classification, but rather on similarities among deep representations. Indeed, it is not guaranteed that even if a model misclassifies a face image the adversarial deep representation will be then close enough to the representation of the wrong face class predicted by the model to fool the FR system.

In Figure 6, Figure 7, and Figure 8, we reported some results to justify our intuition. The figures show the distribution of the distance among the adversarial samples and the centroids of their relative classes considering classifier attacks (BIM [16], MI-FGSM [34], and CW [35]) and DF attacks [45], using a kNN as guidance.

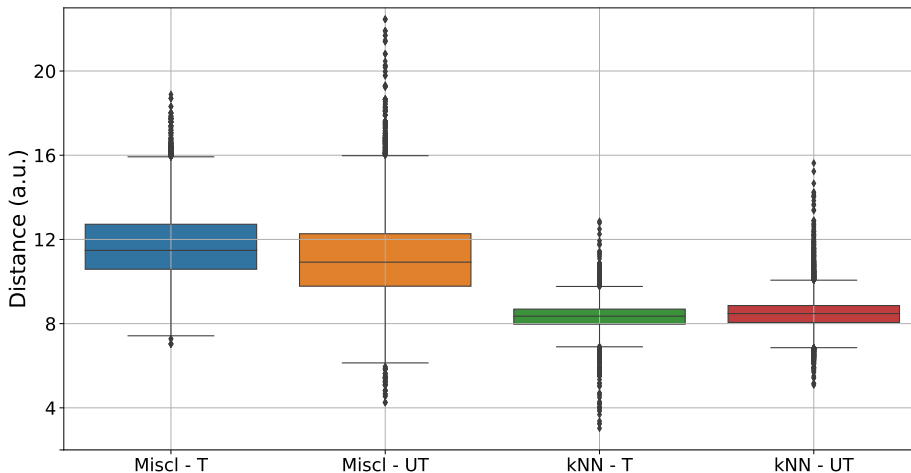


Figure 6: Euclidean distance among deep features of adversarial samples and the assigned class centroid for classifier attacks (blue and orange) and kNN-guided attacks (green and red). “ - T” refers to targeted attacks while “ - UT” refers to untargeted attacks.

As we can see from Figure 6, Figure 7, and Figure 8, even though classifier attacks are able to fool a CNN model, the distance among them and the centroids of their classes is larger than the one obtained when considering malicious samples generated by means of the deep representation-based attack. Thus, the latter represents a greater threat, with respect to the former types

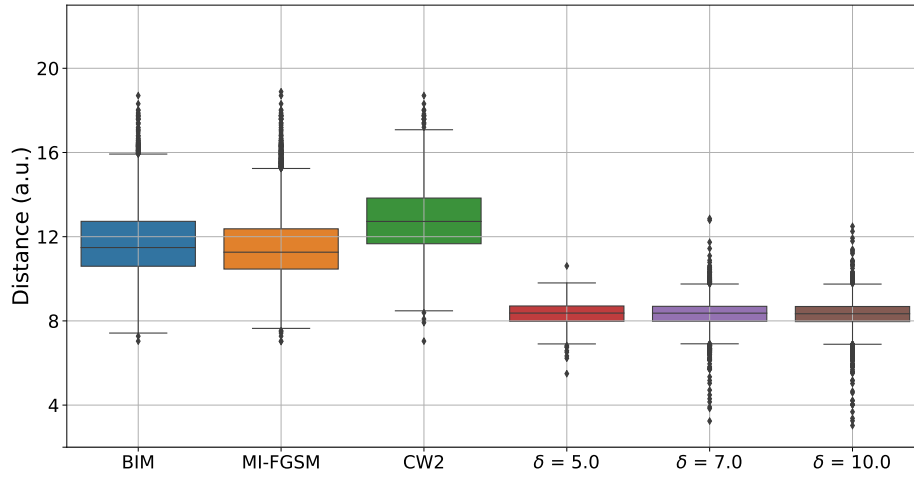


Figure 7: Euclidean distance among deep features of adversarial samples and the assigned class centroid considering each targeted attack singularly. The “ δ ” values correspond to the maximum L_∞ perturbation allowed, for each pixel, for the kNN-guided attacks.

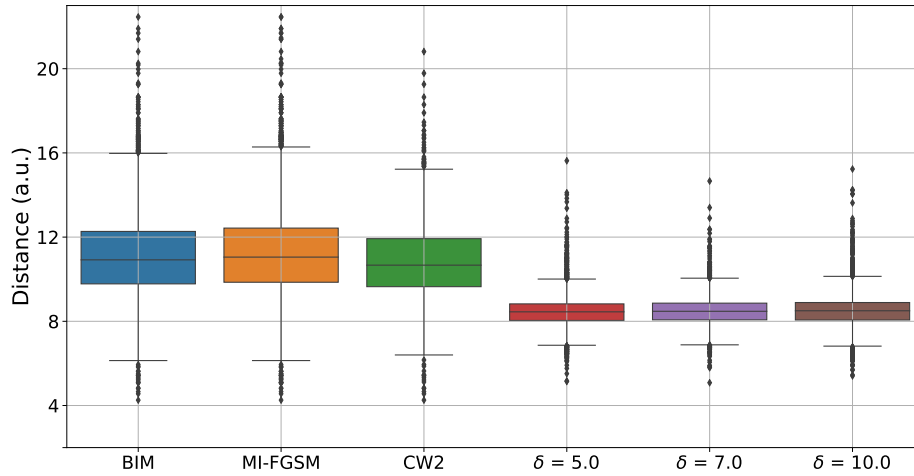


Figure 8: Euclidean distance among deep features of adversarial samples and the assigned class centroid considering each untargeted attack singularly. The “ δ ” values correspond to the maximum L_∞ perturbation allowed, for each pixel, for the kNN-guided attacks.

of attacks, for a FR model. Moreover, considering targeted and untargeted settings for the classifier attacks, the untargeted attacks are, on average, closer to the class centroids when compared to the targeted ones. This result supported

the observation of the lower detection performance when we tested our detectors against untargeted attacks (Subsection 5.2).

Taking into account Figure 7 and Figure 8, we can also notice that the average distance among the adversarial samples from its class centroid is quite stable among the three different values of the threshold we used when considering malicious images generated with the deep features attack. This behaviour is supported by the observation that, independently from the threshold applied, the majority of the pixel perturbations are below, in the sense of an L_∞ distance, a threshold of 5.0. Thus, a higher threshold has the effect that only a small portion of the image is perturbed above that threshold itself. Specifically, considering the values of $\delta \in [5.0, 7.0, 10.0]$, the percentage of pixels whose perturbation is within an L_∞ distance of 5.0 is 88.3%, 85.6%, 84.7% respectively. This behaviour is shown in Figure 9 for targeted attacks. In the case of an untargeted setting, the results were almost identical.

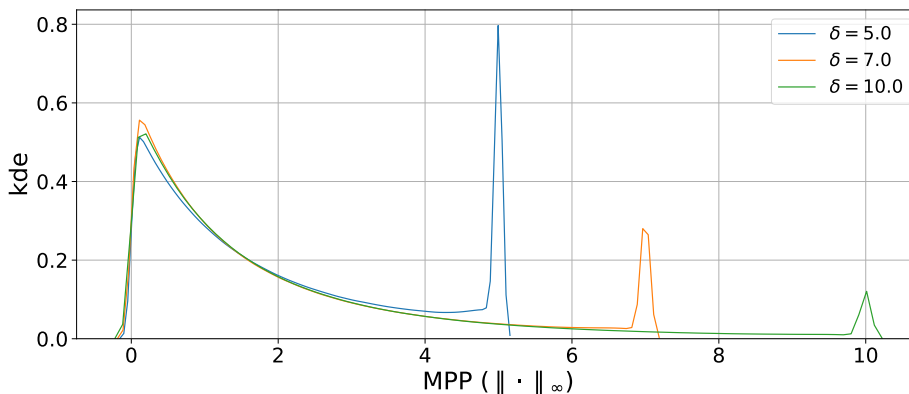


Figure 9: Maximum Pixel Perturbation (MPP) distribution considering targeted deep representations with different thresholds

5.3.2. Face Verification

In this section, we studied the ability of adversarial attacks to fool a FR system tested against the Face Verification protocol in which two face images are compared to claim if they belong to the same identity or not. In the DL context, such decision is typically based upon similarity measurements among

deep features extracted from the input faces. Specifically, once the model has been trained, the ROC curve is typically evaluated and a threshold value is chosen to be used as a reference value. Then, two faces are said to belong to the same identity if their similarity score exceeds the predefined threshold. An example of a real world application scenario of this kind of (such a) protocol is a restricted access area control system. Since in this type of applications the False Positives pose a greater threat than the False Negatives, it is important to evaluate the ROC curve down to very low values of the False Acceptance Rate (FAR). Such a demand translates into the requirement of evaluating the similarity scores among a larger number of negative pairs with respect to the positive ones.

Even in this case, the architecture of a FR system was made by a features extractor and a module which worked out the similarity measurements. As a features extractor we used the state-of-the-art model from Cao et al. [9], while we considered the cosine among features vectors as similarity measurement.

After training the model, we obtained a ROC curve with an AUC value equals to 99.03%. Then, we used the Equal Error Rate (EER) threshold which we found equal to 0.448 as a threshold value for the similarity measurement. At this point, we hypothesized two possible scenarios for the adversarial attacks:

- *Impersonation Attack.* In this case, we wanted to fool the system by leading it to falsely predict that two face images belonged to the same identity. This situation emulated the case in which an intruder intends to enter a restricted area or, in a more general case, when someone is made recognizable as a different person.
- *Evading Attack.* This case is the opposite of the previous one, i.e., we wanted the system not to recognize a person by saying that the two images belonged to different identities. This circumstance imitated the condition of someone whose identity is made unrecognizable.

From the FR system perspective, in the former we needed the two images, which belonged to different people, to be “equal enough”, i.e., their similarity

measurement had to be above the threshold we had previously defined, while in the latter, we needed the two images to be “distant enough”, i.e. below the threshold.

To carry out these experiments, we consider the CW [35] attacks and the kNN guided ones in the targeted and untargeted settings. What we expected was the deep representation attacks to be more effective with respect to the classifier attacks. The results for the *Impersonation Attack* scenario are reported in Figure 10. In this case, we considered what follows: first, we randomly selected negative matches, and we kept the second image fix, i.e. we analysed pairs of faces (x, x^-) where x^- was an image from a different class of x . Then, we looked upon an adversarial image, whose adversarial class corresponded to the one of x^- , and used it in place of the first image of the match, i.e. we accounted the pairs (x_{adv}, x^-) where x_{adv} was an adversarial sample, crafted from x , whose adversarial class was the same as the x^- one. Then, we considered two similarity measurements:

- “Original” which represents the value of the cosine among the deep features of x and x^- ;
- “Adversarial” which represents the cosine between the deep features of x_{adv} and x^- .

In Table 3, we reported the percentage of matches which overcame the EER threshold, before and after the attacks, considering the targeted and untargeted settings.

As we can observe from Table 3, the DF attacks [45] (kNN-guided) are much more effective in pushing the similarity between the adversarials and the natural images above the recognition threshold. Such conclusion holds for targeted and untargeted attacks. Instead, the behaviour of the CW [35] was unpredictable in this set up, thus we can conclude that even though the CW [35] algorithm is among the strongest ones concerning classification attacks, although it is not very effective against the Face Verification protocol.

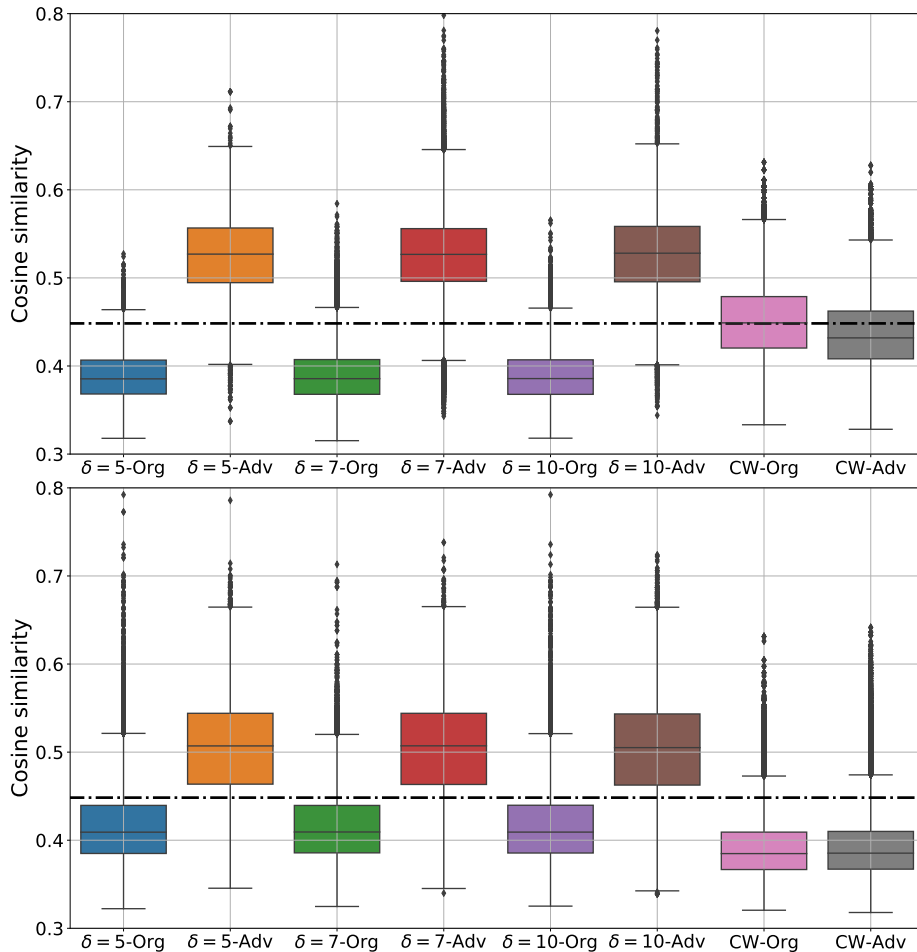


Figure 10: Cosine similarity distribution for kNN-guided and CW attacks in the *Impersonation Attack* scenario. Top: targeted attacks. Bottom: untargeted attacks. “- Org” refers to the cosine among natural images while “- Adv” refers to the cosine between the natural image and the adversarial one. The dash-pointed line represents the EER threshold.

As far as the *Evading Attack* scenario is concerned, the results are reported in Figure 11. Differently from the previous case, we started by collecting positive matches, i.e. pairs of images (x, x^+) in which x and x^+ belonged to the same class, and then we substituted x with one of its adversarial, x_{adv} , whose class was different from the x^+ one. Thus, we obtained the following similarity measurements:

Table 3: Percentage of matches which overcame the EER threshold, before and after the attacks, considering the targeted and untargeted settings.

	Targeted		Untargeted	
	Original	Adversarial	Original	Adversarial
$\delta = 5$	4.0	92.8	19.9	81.5
$\delta = 7$	4.0	93.9	19.9	81.6
$\delta = 10$	4.4	93.6	19.8	81.1
CW [35]	50.5	34.9	8.0	9.3

- “Original” which represents the value of the cosine among the deep features of x and x^+ ;
- “Adversarial” which represents the cosine between the deep features of x_{adv} and x^+ .

As we explained before, in this scenario the purpose of the attack was to push the similarity below the operational level of the FR system.

In Table 4, we reported the percentage of the matches that were below the EER threshold, before and after the attacks, considering the targeted and untargeted settings.

Table 4: Percentage of matches which are below the EER threshold, before and after the attacks, considering the targeted and untargeted settings.

	Targeted		Untargeted	
	Original	Adversarial	Original	Adversarial
$\delta = 5$	5.4	18.9	4.1	9.9
$\delta = 7$	4.3	22.9	4.3	11.4
$\delta = 10$	4.3	26.2	4.1	12.5
CW [35]	4.0	17.1	4.0	6.2

By observing Table 4, it was clear that the DF attacks [45] (kNN-guided)

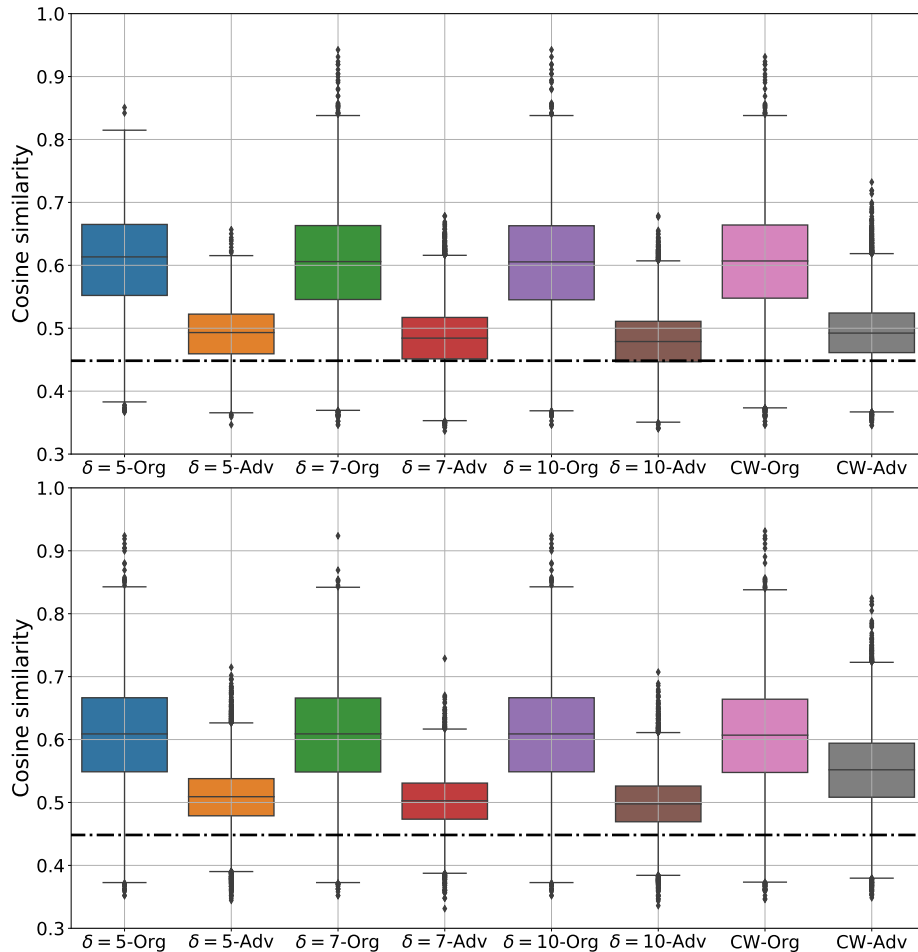


Figure 11: Cosine similarity distribution for kNN-guided and CW attacks in the *Evading Attack* scenario. Top: targeted attacks. Bottom: untargeted attacks. “- Org” refers to the cosine among natural images while “- Adv” refers to the cosine among the natural image and the adversarial one. The dash-pointed line represents the EER threshold.

were more effective than the CW [35] attacks in this case too. We can notice that on average the targeted attacks performed better than the untargeted ones, which was an expected behaviour since an untargeted attack ends as soon as the adversarial is associated with a different identity, therefore it would not have gone any further from the original image.

5.3.3. Detection

Finally, we tested our detectors, trained on classifier attacks (Subsection 5.2), on the newly generated adversarial samples. The resulting ROC curves, according to a threshold of $\delta = 5$ and $\delta = 10$ for targeted and untargeted attacks configurations, are shown in Figure 12. We did not report the ROC for the case $\delta = 7$ since the results were almost identical to the case with $\delta = 10$.

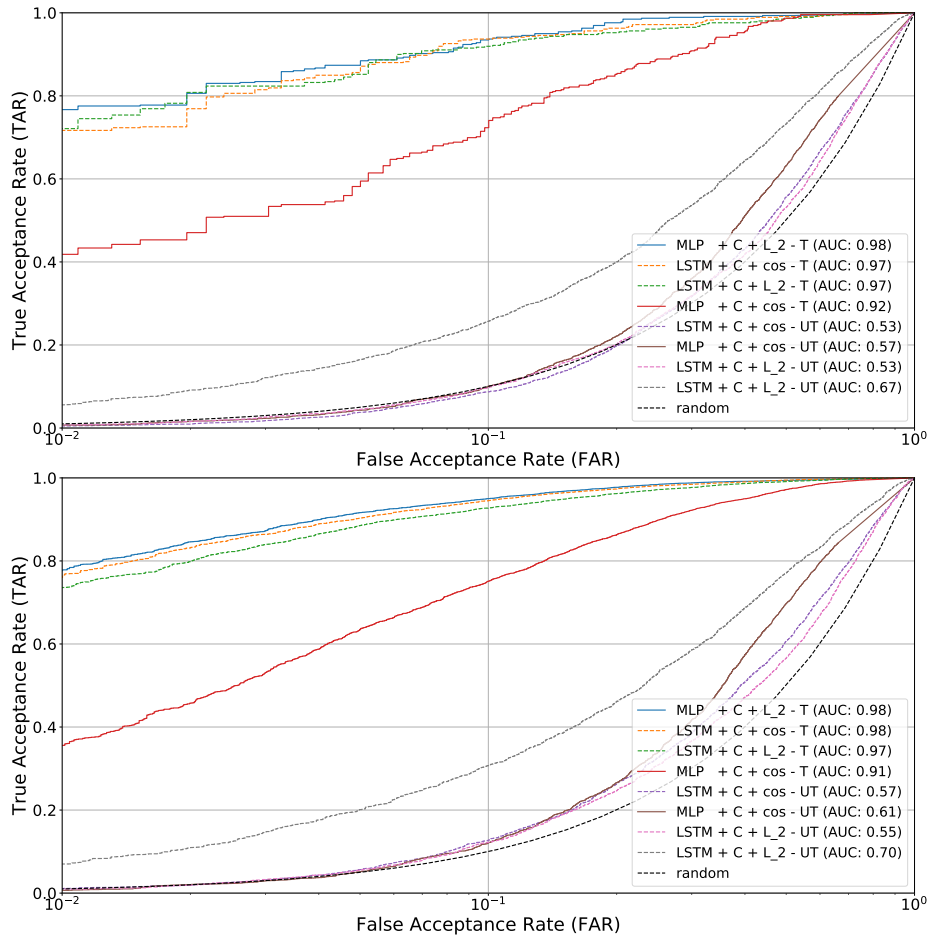


Figure 12: ROCs for the best models considering adversarial attacks generated with $\delta = 5.0$ (top) and $\delta = 10.0$ (bottom). “ - T” refers to targeted attacks while “ - UT” refers to untargeted attacks.

As a summary, the AUC values were reported in Table 5 and Table 6 for

targeted and untargeted attacks, respectively. According to the results shown

Table 5: AUC values for the best performing detectors for each threshold value considered in our experiments in the case of targeted attacks.

Configuration	AUC	δ
MLP + L_2	0.976	5
LSTM + cos	0.972	5
LSTM + L_2	0.969	5
MLP + cos	0.915	5
MLP + L_2	0.977	7
LSTM + cos	0.975	7
LSTM + L_2	0.968	7
MLP + cos	0.908	7
MLP + L_2	0.980	10
LSTM + cos	0.978	10
LSTM + L_2	0.972	10
MLP + cos	0.915	10

in [Figure 12](#), [Table 5](#), and [Table 6](#), we could see that, even though the adversarial detectors were trained on different attacks, they displayed high performances in detecting kNN-guided attacks too. This result has a relevant significance since it means that, despite the different attacks’ objectives, adversarial samples share some common behaviours in the inner layers of a deep model. Moreover, it highlights the generalization capacity of our detection approach. Furthermore, we can acknowledged that while the AUC values were very close for the targeted attacks, in the case of untargeted attacks the LSTM performed considerably better than the MLP.

Table 6: AUC values for the best performing detectors for each threshold value considered in our experiments in the case of untargeted attacks.

Configuration	AUC	δ
LSTM + cos	0.530	5
MLP + L_2	0.492	5
MLP + cos	0.571	5
LSTM + L_2	0.671	5
LSTM + cos	0.541	7
MLP + L_2	0.481	7
MLP + cos	0.596	7
LSTM + L_2	0.688	7
LSTM + cos	0.573	10
MLP + L_2	0.467	10
MLP + cos	0.609	10
LSTM + L_2	0.700	10

6. Conclusions

Adversarial samples represent a serious threat to DL models, especially as they set a serious limitation especially on the use of learning models in sensitive applications. Despite the scientific community’s effort in trying to train robust NN, a knowledgeable attacker usually succeeds in finding ways to attack a model.

Except for the adversarial training, another approach to enhance the robustness of AI-based systems to the adversarial threat is detecting these malicious inputs. In several previous studies the properties of the offensive samples are exploited in order to detect them. Compared to adversarially training a model, the detection of these images has several advantages, e.g. it does not require to re-train any model nor it does not require to specifically design new training strategies to flatten the model loss manifold.

In light of these facts, we proposed our study on the detection of the ad-

versarial samples. Specifically, we exploited the different behaviour of adversarial samples in the inner layers of a DL model with respect to natural images.

We conducted our experiments in the context of Face Recognition, for which we crafted adversarial samples considering wrong-label assignment and deep representation distance as objectives in the targeted and untargeted settings. We first considered the NN acting as a classifier, and then we conducted our attacks against a FR system in which the learning model was employed as features extractor. As far as the classifier attacks are concerned, the best detector reached an AUC value of 99% on the adversarial detection task.

The results obtained from the deep features attacks against a FR system are even more interesting. In this case, we considered a more realistic application scenario for a FR system in which the DL model was used as a features extractor, and the final task was accomplished by means of similarity measurements among the descriptors vectors. Specifically, we observed that i) classifier attacks are much less effective in fooling a FR system; ii) the detectors, trained on the first type of attacks, reached an AUC value of 98% and 70% for the deep representation attacks, which they had never seen before, for targeted and untargeted attacks, respectively. These last results are of great impact considering the idea of an “universal” adversarial detector. Moreover, this also means that, despite the different objectives of the various kind of attacks, they actually share some common properties that can, or perhaps should, be exploited to recognize adversarial attacks and build more robust systems without the need to periodically change the model to increase its robustness.

Acknowledgments

This work was partially supported by the AI4EU project, funded by the EC (H2020 - Contract n. 825619), and by Automatic Data and documents Analysis to enhance human-based processes (ADA) project, CUP CIPE D55F17000290009. We gratefully acknowledge the support of NVIDIA Corporation with the dona-

tion of the Titan V GPU used for this research.

References

- [1] A. Krizhevsky, I. Sutskever, G. E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, in: F. Pereira, C. J. C. Burges, L. Bottou, K. Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 25, Curran Associates, Inc., 1097–1105, URL <http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>, 2012.
- [2] R. Girshick, Fast r-cnn, in: Proceedings of the IEEE international conference on computer vision, 1440–1448, 2015.
- [3] L. Deng, Y. Liu, Deep Learning in Natural Language Processing, Springer, 2018.
- [4] F. Carrara, A. Esuli, T. Fagni, F. Falchi, A. Moreo Fernández, Picture it in your mind: generating high level visual representations from textual descriptions, Information Retrieval Journal 21 (2) (2018) 208–229, ISSN 1573-7659, URL <https://doi.org/10.1007/s10791-017-9318-6>.
- [5] A. Ortis, G. M. Farinella, S. Battiato, An Overview on Image Sentiment Analysis: Methods, Datasets and Current Challenges, in: Proceedings of the 16th International Joint Conference on e-Business and Telecommunications, ICETE 2019 - Volume 1: DCNET, ICE-B, OPTICS, SIGMAP and WINSYS, Prague, Czech Republic, July 26-28, 2019., 296–306, URL <https://doi.org/10.5220/0007909602900300>, 2019.
- [6] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, F. Roli, Evasion attacks against machine learning at test time, in: Joint European conference on machine learning and knowledge discovery in databases, Springer, 387–402, 2013.

- [7] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, R. Fergus, Intriguing properties of neural networks, arXiv preprint [arXiv:1312.6199](https://arxiv.org/abs/1312.6199) .
- [8] K. Sundararajan, D. L. Woodard, Deep learning for biometrics: a survey, *ACM Computing Surveys (CSUR)* 51 (3) (2018) 65.
- [9] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, A. Zisserman, Vggface2: A dataset for recognising faces across pose and age, in: *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, IEEE, 67–74, 2018.
- [10] G. Amato, F. Carrara, F. Falchi, C. Gennaro, C. Vairo, Facial-based Intrusion Detection System with Deep Learning in Embedded Devices, in: *Proceedings of the 2018 International Conference on Sensors, Signal and Image Processing*, ACM, 64–68, 2018.
- [11] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, L. Song, Sphereface: Deep hypersphere embedding for face recognition, in: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 212–220, 2017.
- [12] S. Feldstein, The Global Expansion of AI Surveillance, Working Paper, Carnegie Endowment for International Peace, 1779 Massachusetts Avenue NW, Washington, DC 20036, URL <https://carnegieendowment.org/files/WP-Feldstein-AISurveillance.final1.pdf>, 2019.
- [13] Y. Dong, H. Su, B. Wu, Z. Li, W. Liu, T. Zhang, J. Zhu, Efficient Decision-based Black-box Adversarial Attacks on Face Recognition, in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7714–7722, 2019.
- [14] Q. Song, Y. Wu, L. Yang, Attacks on State-of-the-Art Face Recognition using Attentional Adversarial Attack Generative Network, arXiv preprint [arXiv:1811.12026](https://arxiv.org/abs/1811.12026) .

- [15] M. Sharif, S. Bhagavatula, L. Bauer, M. K. Reiter, Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition, in: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, ACM, 1528–1540, 2016.
- [16] A. Kurakin, I. Goodfellow, S. Bengio, Adversarial examples in the physical world, arXiv preprint [arXiv:1607.02533](https://arxiv.org/abs/1607.02533) .
- [17] X. Li, F. Li, Adversarial Examples Detection in Deep Networks with Convolutional Filter Statistics., in: ICCV, 5775–5783, 2017.
- [18] F. Liao, M. Liang, Y. Dong, T. Pang, X. Hu, J. Zhu, Defense against adversarial attacks using high-level representation guided denoiser, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1778–1787, 2018.
- [19] A. Kurakin, I. Goodfellow, S. Bengio, Adversarial examples in the physical world, arXiv preprint [arXiv:1607.02533](https://arxiv.org/abs/1607.02533) .
- [20] N. Papernot, P. McDaniel, X. Wu, S. Jha, A. Swami, Distillation as a defense to adversarial perturbations against deep neural networks, in: 2016 IEEE Symposium on Security and Privacy (SP), IEEE, 582–597, 2016.
- [21] Z. Gong, W. Wang, W.-S. Ku, Adversarial and clean data are not twins, arXiv preprint [arXiv:1704.04960](https://arxiv.org/abs/1704.04960) .
- [22] K. Grosse, P. Manoharan, N. Papernot, M. Backes, P. McDaniel, On the (statistical) detection of adversarial examples, arXiv preprint [arXiv:1702.06280](https://arxiv.org/abs/1702.06280) .
- [23] M. Amirian, F. Schwenker, T. Stadelmann, Trace and detect adversarial attacks on CNNs using feature response maps, in: 8th IAPR TC3 Workshop on Artificial Neural Networks in Pattern Recognition (ANNPR), Siena, Italy, September 19–21, 2018, IAPR, 2018.

- [24] J. H. Metzen, T. Genewein, V. Fischer, B. Bischoff, On detecting adversarial perturbations, arXiv preprint [arXiv:1702.04267](https://arxiv.org/abs/1702.04267) .
- [25] N. Carlini, D. Wagner, Adversarial examples are not easily detected: Bypassing ten detection methods, in: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, ACM, 3–14, 2017.
- [26] F. Carrara, F. Falchi, R. Caldelli, G. Amato, R. Becarelli, Adversarial image detection in deep neural networks, *Multimedia Tools and Applications* 78 (3) (2019) 2815–2835.
- [27] N. Papernot, P. McDaniel, Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning, arXiv preprint [arXiv:1803.04765](https://arxiv.org/abs/1803.04765) .
- [28] C. Sitawarin, D. Wagner, On the Robustness of Deep K-Nearest Neighbors, arXiv preprint [arXiv:1903.08333](https://arxiv.org/abs/1903.08333) .
- [29] A. Kendall, Y. Gal, What uncertainties do we need in bayesian deep learning for computer vision?, in: Advances in neural information processing systems, 5574–5584, 2017.
- [30] F. Carrara, R. Becarelli, R. Caldelli, F. Falchi, G. Amato, Adversarial examples detection in features distance spaces, in: Proceedings of the European Conference on Computer Vision (ECCV), 0–0, 2018.
- [31] C. Whitelam, E. Taborsky, A. Blanton, B. Maze, J. Adams, T. Miller, N. Kalka, A. K. Jain, J. A. Duncan, K. Allen, et al., Iarpa janus benchmark-b face dataset, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 90–98, 2017.
- [32] B. Maze, J. Adams, J. A. Duncan, N. Kalka, T. Miller, C. Otto, A. K. Jain, W. T. Niggel, J. Anderson, J. Cheney, et al., IARPA janus benchmark-c: Face dataset and protocol, in: 2018 International Conference on Biometrics (ICB), IEEE, 158–165, 2018.

- [33] I. J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples (2014), arXiv preprint [arXiv:1412.6572](https://arxiv.org/abs/1412.6572) .
- [34] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, J. Li, Boosting adversarial attacks with momentum, arXiv preprint .
- [35] N. Carlini, D. Wagner, Towards evaluating the robustness of neural networks, in: 2017 IEEE Symposium on Security and Privacy (SP), IEEE, 39–57, 2017.
- [36] M. A. Turk, A. P. Pentland, Face recognition using eigenfaces, in: Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE, 586–591, 1991.
- [37] M. Wang, W. Deng, Deep face recognition: A survey, arXiv preprint [arXiv:1804.06655](https://arxiv.org/abs/1804.06655) .
- [38] F. V. Massoli, G. Amato, F. Falchi, C. Gennaro, C. Vairo, Improving Multi-scale Face Recognition Using VGGFace2, in: International Conference on Image Analysis and Processing, Springer, 21–29, 2019.
- [39] H. Qiu, C. Xiao, L. Yang, X. Yan, H. Lee, B. Li, SemanticAdv: Generating Adversarial Examples via Attribute-conditional Image Editing, arXiv preprint [arXiv:1906.07927](https://arxiv.org/abs/1906.07927) .
- [40] K. Kakizaki, K. Yoshida, Adversarial Image Translation: Unrestricted Adversarial Examples in Face Recognition Systems, 2019.
- [41] M. Pautov, G. Melnikov, E. Kaziakhmedov, K. Kireev, A. Petiushko, On adversarial patches: real-world attack on ArcFace-100 face recognition system, arXiv preprint [arXiv:1910.07067](https://arxiv.org/abs/1910.07067) .
- [42] R. Huang, B. Xu, D. Schuurmans, C. Szepesvári, Learning with a strong adversary. arXiv 2015, arXiv preprint [arXiv:1511.03034](https://arxiv.org/abs/1511.03034) .

- [43] N. Papernot, P. McDaniel, X. Wu, S. Jha, A. Swami, Distillation as a defense to adversarial perturbations against deep neural networks, arXiv preprint [arXiv:1511.04508](https://arxiv.org/abs/1511.04508) .
- [44] G. Goswami, A. Agarwal, N. Ratha, R. Singh, M. Vatsa, Detecting and mitigating adversarial perturbations for robust face recognition, *International Journal of Computer Vision* 127 (6-7) (2019) 719–742.
- [45] S. Sabour, Y. Cao, F. Faghri, D. J. Fleet, Adversarial manipulation of deep representations, arXiv preprint [arXiv:1511.05122](https://arxiv.org/abs/1511.05122) .
- [46] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) .