Vulnerability of Person Re-Identification Models to Metric Adversarial Attacks

Quentin Bouniot, Romaric Audigier, Angélique Loesch
CEA, LIST, Vision and Learning Lab for Scene Analysis
PC 184, F-91191 Gif-sur-Yvette, France
{quentin.bouniot, romaric.audigier, angelique.loesch}@cea.fr

Abstract

Person re-identification (re-ID) is a key problem in smart supervision of camera networks. Over the past years, models using deep learning have become state of the art. However, it has been shown that deep neural networks are flawed with adversarial examples, i.e. *human-imperceptible perturbations. Extensively studied for the task of image closed-set classification, this problem can also appear in the case of open-set retrieval tasks. Indeed, recent work has shown that we can also generate adversarial examples for metric learning systems such as re-ID ones. These models remain vulnerable: when faced with adversarial examples, they fail to correctly recognize a person, which represents a security breach. These attacks are all the more dangerous as they are impossible to detect for a human operator. Attacking a metric consists in altering the distances between the feature of an attacked image and those of reference images, i.e. guides. In this article, we investigate different possible attacks depending on the number and type of guides available. From this metric attack family, two particularly effective attacks stand out. The first one, called Self Metric Attack, is a strong attack that does not need any image apart from the attacked image. The second one, called Furthest-Negative Attack, makes full use of a set of images. Attacks are evaluated on commonly used datasets: Market1501 and DukeMTMC. Finally, we propose an efficient extension of adversarial training protocol adapted to metric learning as a defense that increases the robustness of re-ID models.*[1]

1. Introduction

Person re-identification (re-ID) is an increasingly popular field of research due to its application in video surveillance. In general, re-ID is viewed as a retrieval task. The objective is to rank a gallery of images by order of similarity to a query image. With the advent of deep learning, a lot of new approaches using deep convolutional neural networks (CNNs) have been proposed to achieve this task [29][10][26].

Szegedy *et al*. [22] have shown that deep neural networks can be easily fooled by *adversarial examples*, *i.e*. human-imperceptible perturbations added to an image. These examples have been extensively studied [7][12][25][15][6][27][20] for closed-set classification task (where same classes are used at training and testing). To the best of our knowledge, very few approaches have been proposed to tackle attacks and defenses of retrieval models used for re-ID, an open-set task where identities at training are different from those at testing.

Previous classification attacks are not exploitable in this case, as class information is not used in re-ID. In order to fool a re-ID model, the attacker has to perturb images so as to modify distances between their features. In this way, final ranking based on distance (or conversely, similarity) can become wrong. Metric attacks depend on a reference feature (a *guide*) to distort the distance between the attacked image and other similar images. It can be a *pulling guide* belonging to another identity, which draws the feature away from its initial identity cluster. Otherwise, it can be a *pushing guide* with same identity, that repels the feature away from its initial identity cluster.

In this paper, we study the effect of using pushing or pulling guides, or both to generate metric attacks. We note that the implementation of metric attacks in practice depends on the availability of guides. While the availability of pulling guides (different identity) does not seem a priori to be a problem, the availability of pushing guides (same identity) may be more difficult or even impossible to achieve. Therefore, we propose different attacks that can be applied depending on the availability of these auxiliary images. Two effective attacks can be distinguished in the two extremes, either fully available or not.

We also show how the online adversarial training protocol, effective for robustifying classifiers, can be adapted to defend re-ID models against proposed metric attacks.

Our contributions can be summarized as follows:

[1]The code for the attacks and defenses is available on Github at https://github.com/qbouniot/adv-reid.

- We show that metric attacks can be based on pushing guides, pulling guides or both.
- We show that attack feasibility depends on availability of such guides. So, to cope with two extreme situations of no availability and full availability of guides, we propose **two novel attacks** for metric embedding problems: the **Self Metric Attack** and the **Furthest-Negative Attack**.
- We present a defense to improve the robustness of the models against proposed metric attacks.

After an overview of previous work on this subject in Sec. 2, notations are introduced in Sec. 3. We present the overall framework for metric attacks based on pushing and pulling guides and propose our new metric attacks in Sec. 4. They are evaluated and compared with the state of the art in Sec. 5. Finally, we present, in Sec. 6, a training protocol based on adversarial training to defend re-ID models against the proposed attacks.

2. Related Work

In this section we review the previous work on adversarial attacks and defenses for classification as well as metric learning problems.

2.1. Adversarial Attacks

Classification Attacks Szegedy *et al*. [22] have been the first to show the flaws of deep neural networks on a classification task. Then, a lot of new approaches to generate adversarial examples have emerged. They can be grouped into three families:
(i) *Unbounded attacks* [22][5][27][20] solve a constrained optimization problem to find the adversarial examples with smallest perturbations.
(ii) *Bounded attacks* [7][11][15][6], less time-consuming, perform several steps of projected gradient descent in order to keep perturbation smaller than a threshold.
(iii) *Gradient Reconstitution attacks* [17][16][3] search the input space by following an approximation of the Jacobian or by a random walk. They can bypass possible defenses.

These attacks can be *White-box* or *Black-box* depending if they, respectively, have access to the model or not. If the image is modified to be classified as a *targeted* class, the attack is said to be *targeted*. Otherwise, if the image is modified to be classified as any other class different from the original class, the attack is said to be *non-targeted*.

All these methods try to keep class predictions as far away as possible from their proper class by attacking models at the logit level. These types of approaches are therefore *only valid in classification tasks*. They can't be used in the same way on an open-set ranking task such as re-ID [2][30].

Metric Attacks For ranking or retrieval tasks, the distance between features of images is used for evaluation. In this case, *metric attacks* are based on *pushing* the features of an input image away from their identity cluster (*non-targeted attack*) or *pulling* to another cluster (*targeted attack*) in the embedding space using a reference feature, *i.e.* a guide. For person re-ID, an open-set ranking task, the Opposite-Direction Feature Attack (ODFA) [30] *pulls* the feature of the attacked image in the opposite direction with an artificial guide. Bai *et al*. [2] extend classification attacks, namely Fast-Gradient Sign Method (FGSM) [7], Iterative FGSM (IFGSM) a.k.a. Basic Iterative Method (BIM) [12] or Projected Gradient Descent (PGD) [15] and Momentum IFGSM (MIFGSM) [6], to *push* away the feature with a guide instead of misleading the logits.

For other image retrieval tasks than re-ID, Tolias *et al*. [23] propose to attack an image retrieval system, so that images of a wrong *targeted* class are retrieved for an attacked query image. Nevertheless, this *pulling guide attack* has little effect on the overall closed-set ranking, as shown by the small drop in mAP performance. Liu *et al*. [14] propose to generate attacks with a given number of iterations. Yet, no size constraint is considered for the adversarial noise, so that perturbations can be noticeable.

For any of these metric attacks, the use of guides involves other images. But once additional images are used, why not draw more information from them? Otherwise, in the opposite extreme case, what could the attacker do if he does not have access to additional images but only to the images to be attacked?

2.2. Adversarial Defenses

Confronted with these attacks, several approaches have been presented in the state of the art to make the models more robust.

Classification Defenses A first type of defense tries to prevent against misclassification coming from adversarial examples by *obfuscating the gradients* [18][13][8]. However, these defenses are still vulnerable to more specific adversarial attacks [1] [4] [24].

Currently, *adversarial training* is the most robust way to defend against adversarial attacks: computing adversarial versions of each images during training. The model can thus be trained on the adversarial versions [15] or on a mix of original and adversarial images [7]. The adversarial images can also be generated with several models [25].

The adversarial training protocol cannot be used directly with any metric attacks. Some need a guide of the same class as the attacked image, which is not always the case in a random batch of training images.

Metric Defenses To our knowledge, only one defense protocol [2] has been proposed against metric attacks. This defense is based on a generation of an adversarial version of the training set obtained with a frozen version of the trained model. The defended model is then trained on both original and adversarial training sets. As a frozen model is used to generate attacks, this protocol will be referred to as *offline adversarial training*. Indeed, in classification tasks, the adversarial training generates adversarial examples *online*, *i.e.*, while the defended model evolves. Besides, this defense was evaluated in a black-box setting only. An evaluation in white-box setting is necessary to confirm its effectiveness.

We propose an extension of online adversarial training protocol for metric attacks, *Guide-sampling Online Adversarial Training*, as a defense for re-ID models.

3. Re-Identification: an Open-Set Ranking Problem

In this paper, we consider the task of person re-ID which goal is to rank a *gallery* set of images for each image of a *query* (or probe) set by order of similarity. This similarity is derived from the distance between the features relative to two images of person. Re-ID task protocol uses a training set of person images $\mathbf{X} = \{x_i\}_{i \in [\![1,N_x]\!]}$ and a testing set subdivided in a query set of images $\mathbf{Q} = \{q_i\}_{i \in [\![1,N_q]\!]}$ and a gallery set of images $\mathbf{G} = \{g_i\}_{i \in [\![1,N_g]\!]}$. N_x, N_q, N_g denote respectively the numbers of person images within the training, query and gallery sets.

Each image x_i (resp. q_i, g_i) in these sets is associated to a person with identity $\ell(x_i)$ (resp. $\ell(q_i)$, $\ell(g_i)$). For each identity l, we define $\mathbf{I}_l$ the subset of images (from the training, query or gallery sets) with identity l. Note that identities in training and testing sets are *different*, whereas query and gallery sets share identities.

Due to this open-set problem constraint, re-ID task is generally coped with as a metric embedding learning problem. In other words, a mapping $f : \mathbf{E} \rightarrow \mathbf{F}$ is learned in such a way that images of same identity in the space $\mathbf{E}$ of images correspond to close feature vectors in the embedding space $\mathbf{F}$, according to a given/learned metric. Conversely, images with different identities correspond to distant features. Thereafter, $f(x)$ is denoted by f_x.

Different mapping functions can be learned, depending on the dissimilarity metric chosen to train and evaluate re-ID models. As deep neural networks are state-of-the-art, we focus on the two main types of loss minimization commonly employed in re-ID:
(1) the classification model (C) implicitly learns an embedding space through cross-entropy loss minimization [26];
(2) the triplet model (T) explicitly learns a metric embedding through triplet loss minimization [21][10].

At testing time, feature vectors are ranked according to the chosen/learned dissimilarity metric (typically, a distance derived from Cosine similarity or L_2).

4. Proposed Attacks for Metrics

4.1. Metric Attacks

In the re-ID context, adversarial attacks cannot mislead a model to a wrong class, as classes are not known at testing time in an open-set setting. It rather tries to perturb images so as to distort the distance D between feature vectors and reduce overall retrieval performance. To do this, metric attacks use a *guide* g, *i.e.* a reference feature. A guide can induce two kinds of perturbations:
Pulling guide: *decreases* the distance between features of images with *different* identities.
Pushing guide: *increases* the distance between features of images with *same* identities.

Following this terminology, ODFA [30] uses an *artificial* pulling guide (in the opposite direction) whereas the attacks introduced by Bai *et al*. [2] can either be pushing or pulling guide attacks. One question that arises at this stage is whether pushing or pulling guides are equally effective. On the one hand, a *pulling guide* moves the perturbed image closer to the guide. The fact that the image is close to the guide does not necessarily imply that the distances to images of the same identity increase to the extent that these images are relegated to the last rows. In some cases, the pulling guide could affect only very partially the first ranks of the ranking.

On the other hand, with a *pushing guide*, using the *triangular inequality*, we have:

$$D(f_{\hat{x}_i}, f_g) \leq D(f_{\hat{x}_i}, f_{x_j}) + D(f_{x_j}, f_g) \qquad (1)$$

with x_i the given image to attack, $\hat{x}_i$ the corresponding attacked image and x_j and g two images with the same identity as x_i. The pushing guide attack increases $D(f_{\hat{x}_i}, f_g)$. If we assume $D(f_{x_j}, f_g)$ is small (which is the case when $\ell(x_j) = \ell(g)$), then $D(f_{\hat{x}_i}, f_{x_j})$ increases.

This means that in the embedding space, by increasing the distance between the adversarial image and the guide, the adversarial image moves away from all other images similar to the guide. Therefore, the adversarial image ends up far from its original identity cluster. However, the adversarial feature can be pushed in a direction where there is no feature with another identity. Thus, a greater distance is needed to change the ranking.

In addition, the use of a guide generally implies access to additional images that serve as guides. Yet, can we assume additional images are available during the attack?

Consequently, we investigate hereafter novel metric attacks that efficiently leverage information from the set $\mathbf{A}$ of images available during the attack. We also propose to use

Algorithm 1 Self Metric Attack

Input : Model f, input image x, number of iterations N, adversarial bound ϵ, iteration step size α, distance function D, *clip* function to project in L_∞ ball

Output : Adversarial image $\hat{x}$

Generate random noise η with $||\eta||_\infty \leq \epsilon$

$\hat{x} \leftarrow x + \eta$

for $n = 0$ to N **do**

  $\Delta^{SMA} \leftarrow \frac{\partial D(f(\hat{x}), f(x))}{\partial \hat{x}}$

  $\hat{x} \leftarrow \hat{x} + \alpha \cdot \text{sign}(\Delta^{SMA})$

  $\hat{x} \leftarrow clip(\hat{x}, x, \epsilon)$

end for

multiple guides when possible and show that pushing and pulling guides can be combined efficiently.

4.2. Self Metric Attack

When we do not have access to additional image ($\mathbf{A} = \emptyset$) we propose the *Self Metric Attack* (SMA) that uses the *image itself* as a pushing guide. A noisy version of the image under attack is thus used in order to make possible the computation of the attack. In other words, we push the feature of the noisy image away from the original image. In this case, this means replacing f_g by f_{x_i} in inequality 1.

As illustrated in Figure 1, the proposed Self Metric Attack alters the input image with a random noise η (with $||\eta||_\infty \leq \epsilon$) before pushing the noisy image away from the original image in the feature space. The constraint on η ensures the identity is preserved in the noisy image. This generates a perturbation in the image space constrained in norm inside the L_∞-ball centered around the original image.

Formally, our *Self Metric Attack* is defined as the following iterative optimization:

$$\begin{aligned}
\hat{x}^{(0)} &= x + \eta \\
\hat{x}^{(n+1)} &= \Pi_x^\epsilon \left(\hat{x}^{(n)} + \alpha \cdot \text{sign}(\Delta_n^{SMA}) \right) \\
\Delta_n^{SMA} &= \frac{\partial D(f_{\hat{x}^{(n)}}, f_x)}{\partial \hat{x}^{(n)}}
\end{aligned} \tag{2}$$

with x the input image under attack, $\hat{x}^{(n)}$ the resulting adversarial image after the n-th iteration, ϵ the adversarial bound and α the iteration step size. Π_x^ϵ is the clip function, which ensures that $||\hat{x}^{(n+1)} - x||_\infty \leq \epsilon$ and that $\hat{x}$ is a valid image, *i.e.* the pixels are in the range $[0, 1]$. The resulting algorithm is described in Algorithm 1.

4.3. Furthest-Negative Attack

There might also be many images available, even more for different identities. In this case, we can take advantage of the information given by all the images. Single-Guide metric attacks [2], consider a *single* guide g to find the pushing direction. With a single guide, the direction of the perturbation is highly dependent on the guide chosen. When multiple images of a given identity are available (in a set $\mathbf{A}$), we propose to use them all to have *a better approximation* of the direction.

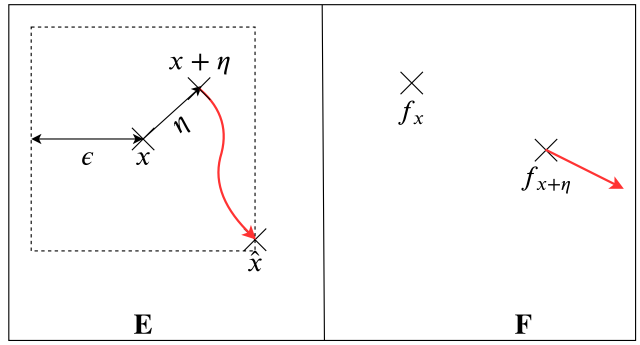

Figure 1: Illustration of SMA. Noise η is bounded in norm to ensure that the noisy image remains in the same identity cluster. The noisy image $x + \eta$ is the starting point of the attack, and the original input image x is the guide for the attack. The noisy image is then pushed away (red straight arrow) from the original image in the feature space **F**. The red curved arrow shows the resulting movement in the image space **E**.

Algorithm 2 Furthest Negative Attack

Input : Model f, input image x under attack, number of iterations N, adversarial bound ϵ, iteration step size α, distance function D, *clip* function to project in L_∞ ball, set $\mathbf{A}$ of available images

Output : Adversarial image $\hat{x}$

$\hat{x} \leftarrow x$

$l_{far} \leftarrow \arg\max_{i \in \ell(\mathbf{A})} D\left(f(x), \sum_{g \in \mathbf{I}_i \cap \mathbf{A}} \frac{f(g)}{|\mathbf{I}_i \cap \mathbf{A}|}\right)$

$\mathbf{Pull} \leftarrow \mathbf{I}_{\ell_{far}} \cap \mathbf{A}$

$\mathbf{Push} \leftarrow \mathbf{I}_{\ell(x)} \cap \mathbf{A}$

for $n = 1$ to N **do**

  $\Delta_n^{FNA} \leftarrow \sum_{g_{push} \in \mathbf{Push}} \frac{\partial D(f(\hat{x}), f(g_{push}))}{\partial \hat{x}} - \sum_{g_{pull} \in \mathbf{Pull}} \frac{\partial D(f(\hat{x}), f(g_{pull}))}{\partial \hat{x}}$

  $\hat{x} \leftarrow \hat{x} + \alpha \cdot \text{sign}(\Delta^{FNA})$

  $\hat{x} \leftarrow clip(\hat{x}, x, \epsilon)$

end for

Furthermore, to be more efficient and induce the biggest change in the ranking, the attack should ensure the attacked feature moves closer to another identity cluster. In addition, the attacked feature should have the highest distance with features from its initial identity cluster. Therefore, our proposal is to move the attacked feature toward the furthest cluster, *i.e.*, the cluster at the highest distance.

Consequently, we propose the *Furthest-Negative Attack*

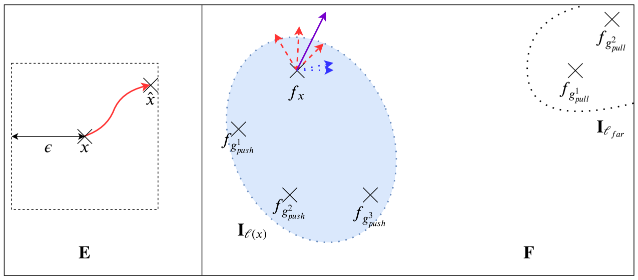

Figure 2: Illustration of FNA. The red curved arrow shows the perturbation generated in the image space $\mathbf{E}$ (on the left). The perturbation is constrained in a L_∞-ball (dotted square) of size ϵ around image x. In the feature space F (on the right), the red dashed arrows show the direction to move away from the identity cluster. The blue dotted arrows show the direction to the features of the furthest cluster $\mathbf{I}_{\ell_{far}}$. The purple solid arrow shows the resulting direction. The direction of the furthest cluster of features is used to find the direction that will most affect the ranking.

(FNA) which combines multiple pushing guides and pulling guides from the furthest cluster. Formally, we define the FNA as the following iterative optimization:

$$\begin{aligned}
\hat{x}^{(0)} &= x \\
\hat{x}^{(n+1)} &= \Pi_x^\epsilon \left(\hat{x}^{(n)} + \alpha \cdot \text{sign}(\Delta_n^{FNA})\right) \\
\Delta_n^{FNA} &= \sum_{g_{push} \in \mathbf{I}_{\ell(x)} \cap \mathbf{A}} \frac{\partial D(f_{\hat{x}^{(n)}}, f_{g_{push}})}{\partial \hat{x}^{(n)}} \\
&\quad - \sum_{g_{pull} \in \mathbf{I}_{l_{far}} \cap \mathbf{A}} \frac{\partial D(f_{\hat{x}^{(n)}}, f_{g_{pull}})}{\partial \hat{x}^{(n)}}
\end{aligned}$$

with the same notations as Equation 2. In this case, $\mathbf{A}$ is the set of available images that contains multiple images of the same identity as the attacked image and others with different identities.

The identity l_{far} of the furthest cluster is computed for each image x under attack:

$$l_{far} = \arg\max_{i \in \ell(\mathbf{A})} D\left(f_x, \sum_{g \in \mathbf{I}_i \cap \mathbf{A}} \frac{f_g}{|\mathbf{I}_i \cap \mathbf{A}|}\right)$$

The attack is described in Algorithm 2 and illustrated in Figure 2. The furthest cluster of features helps find the direction that will perturb the ranking the most. However, we are still pushing the feature away from its cluster by using guides with the same identity. This allows to head towards the least similar cluster of features while moving away from the other similar features.

The higher the norm of the adversarial noise, the greater the movement in the feature space and the more significant the heading towards the least similar cluster become.

5. Attack Evaluation

In this section, we detail the evaluation protocol used to benchmark our attacks and compare them to the state of the art.

5.1. Experimental Settings

We trained a ResNet-50 [9], following the training procedure by Xiong *et al.* [26] to obtain a *Classification model (C)* and the training procedure by Hermans *et al.* [10] to obtain a *Triplet model (T)* as explained in Section 3. For both models C and T, we use as mapping f the features after the average pooling layer, before the last layer. This gives an embedding size of 2048. We chose the same dimensionality for a fair comparison of the effectiveness of the attacks. We consider two metrics during the evaluation, L_2 or Cosine.

The evaluation is done on the Market-1501 dataset [28] (Market) and DukeMTMC-reID [19] dataset (Duke), two datasets commonly used in re-ID. *Market* contains 1501 identities taken by 6 cameras and spread out in $12,936$ bounding boxes for training (750 identities), $19,732$ for the gallery and 3368 queries (regrouped into 751 identities). *Duke* is composed of $36,411$ images taken by 8 cameras, and representing $1,404$ identities: 702 identities ($16,522$ images) are used for training and the other 702 identities for evaluation, split into $2,228$ query images and $17,661$ gallery images. For all experiments, the pixel values, perturbation strength ϵ and perturbation step α are normalized to [0,1] by dividing by 255.

The overall performance is evaluated with the mean average precision (mAP). The baseline performance (without attack) is given in Table 1 in column *Original*.

We perform the benchmark of the attacks by considering the Query set as our available images ($\mathbf{A} = \mathbf{Q}$). The benchmarks are carried out on a single Titan X GPU.

5.2. FNA: pushing or pulling?

First of all, we want to compare pulling and pushing attacks. To do so, Figure 3 shows the performance of the attacks with varying strength ϵ for model C on Market, with a comparison of the pulling and pushing effects in FNA. We evaluate the performance of FNA without pushing guides (green dotted curve), without pulling guides (red straight curve) and with both (light blue dashed and dotted curve). From the graph, we can see that the pulling effect becomes more effective than the pushing effect only when $\epsilon \geq 5$. However, combining both is always more effective.

To confirm our analysis on the furthest cluster, we also evaluate the pulling effect when using any random negative

Dataset	Model	Metric	mAP (%)				
			Original	ODFA [30]	SG. IFGSM [2]	SMA *(Ours)*	FNA *(Ours)*
Market	T	L_2	67.72	25.65	0.25	0.18	**0.06**
		Cosine	67.22	63.02	**0.05**	**0.05**	**0.05**
	C	L_2	76.02	43.73	3.20	2.34	**0.53**
		Cosine	77.53	75.65	0.21	0.26	**0.07**
Duke	T	L_2	60.83	23.48	0.40	0.17	**0.06**
		Cosine	60.33	55.94	**0.05**	**0.05**	**0.04**
	C	L_2	64.85	39.43	3.66	2.53	**0.29**
		Cosine	67.64	65.89	0.16	0.32	**0.06**
Computing time (s)			80	250	260	250	310
Additional images				×	✓	×	✓

Table 1: Performance (mAP in %) of re-ID models on Market1501 and DukeMTMC under white-box attack of the query for $\epsilon = 5$ (and $\alpha = 1$ with 15 iterations for iterative attacks). Rows are the models under attack, evaluated with L_2 and Cosine metrics, while columns are the attacks performed: ODFA [30], Single-Guide IFGSM (SG. IFGSM) [2], SMA and FNA. Lower is better for the attack. Given a model type and a metric, best attacks are in bold numbers. We report computing time (in seconds) on a single Titan X GPU to attack the whole Query set (3368 images), and the need for additional images.

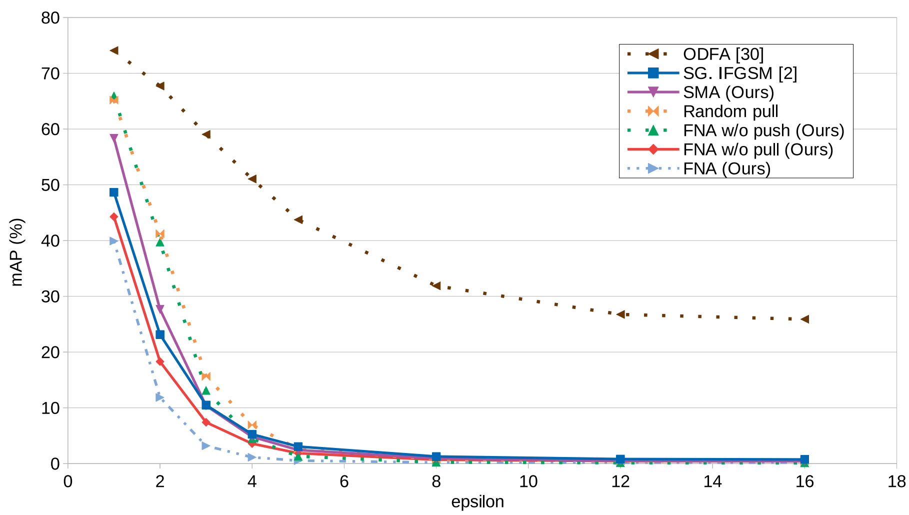

Figure 3: Performance (mAP in %) of model C with L_2 metric, under attack for varying values of ϵ on Market. Straight curves use pushing guides and dotted curves use pulling guides. FNA (dashed curve) uses both. In average, FNA uses 4 pushing guides and/or 4 pulling guides. Lower mAP for stronger attack.

cluster (orange dotted curve) instead of the furthest cluster (green dotted curve). We can see that choosing the furthest cluster is consistently more efficient. Compared with the pushing effect of FNA (red straight curve), pulling with a random cluster is always less effective.

Thus, empirically, pulling guides seem to be effective when they are part of the furthest cluster of features and when the perturbation is sufficiently high. Otherwise, pushing guides seem to be more efficient. A combination of both leads to the best results.

5.3. Comparison with the state of the art

We compare our SMA and FNA to the state of the art: Single-Guide Attacks [2] and ODFA [30]. As shown by Bai *et al.* [2], IFGSM (a.k.a. BIM or PGD) is the strongest variant of Single-Guide (SG.) Attacks. For the sake of clarity, we did not include FGSM and MIFGSM variants. In order to assess efficiency of the attacks, Table 1 compares their computing time and the amount of information they use in input, alongside their effectiveness. These data are reported for $\epsilon = 5$ on two types of re-ID models, using L_2 or Cosine metrics on two different datasets. In particular, we study the impact of the number of additional images used during the attack.

What stands out in both Table 1 and Figure 3 is that ODFA is less efficient than the other attacks. Even though ODFA and SMA both require the smallest amount of information (*i.e*. no additional image) and have same computing time, SMA is much more efficient. As shown in Table 1, it leads to a mAP drop difference of about 40 percentage point (p.p.) for the same computing time.

Of course, using additional images is more efficient. However, when $\epsilon \geq 3$, SMA becomes as efficient as SG. IFGSM for less computing time.

With more available images from the same identity (pushing guides), FNA without pulling is more efficient than both SG. IFGSM and SMA for slightly more computing time. But that superiority tends to fade with larger perturbations ($\epsilon \geq 8$). If even more images of different identities (pulling guides) are available, FNA becomes the strongest option to attack a re-ID model, while being a little bit slower (typically, 310 s instead of 260 s for SG. IFGSM).

Overall, we can see that the efficiency of the attacks depends on the number of available images used. FNA is stronger because it uses both types of guides (pushing and pulling) simultaneously. For instance, for model C with an L_2 similarity on Market, the baseline performance of 76.02% drops to 43.73% after ODFA, to 3.20% after SG. IFGSM, to 2.34% after SMA, to 0.53% after FNA. We achieve similar results on Duke.

For greater ϵ, all attacks but ODFA are effective. Yet, added perturbations become noticeable by a human.

Thus, the two proposed methods are efficient attacks. Choice between these options will depend on the availability of additional images during the attack.

6. Defending Re-ID models

In this section, we detail a protocol for adversarial training with any metric attacks and compare the robustness of our models with the defense proposed by Bai *et al*. [2].

6.1. Guide sampling Online Adversarial Training

As explained in Section 2.2, adversarial training can be done *online* or *offline*, depending on how the adversarial examples are generated. With offline adversarial training, the model trains with a single version of adversarial example. With online adversarial training, the attack uses an updated version of the model to generate more up-to-date adversarial examples. We therefore expect online adversarial training to be more robust than offline.

Algorithm 3 Guide-sampling Online Adversarial Training

Input : Model f, training set $\mathbf{X} = \{x_i\}_{i \in [\![1, N_x]\!]}$, number of epochs T, number of pushing guides N_{push}, number of pulling guides N_{pull}
Output : Defended model f

```
1: for t = 1 to T do
2:     for batch B ∈ X do
3:         for x in B do
4:             Sample g_push^1, ..., g_push^{N_push} pushing guides and g_pull^1, ..., g_pull^{N_pull} pulling guides from X;
5:         end for
6:         Generate adversarial batch B̂ using the guides sampled;
7:         Compute the loss with B̂ and update model f parameters by back propagation;
8:     end for
9: end for
```

Furthermore, as shown in previous work [15] [20], the stronger the attack, the more robust the defense. This prompts us to consider efficient metric attacks. However, as commented in Sec. 2.2, it is not straightforward to perform online adversarial training using *any metric attacks*. SMA can be applied but other metric attacks need additional images from the corresponding identity. Yet, using classical random batches for training gives no guarantee that several images have the same identity.

Therefore, to extend the online adversarial training of Madry *et al*. [15] with any metric attacks, we propose a *Guide sampling Online Adversarial Training* (GOAT) protocol. We sample additional images from the training set *during training* and generate an adversarial example using these images as guides. Thus, for a batch of training images, we sample one batch of N_{push} *pushing* guides and another of N_{pull} *pulling* guides to generate adversarial examples and use them to update the parameters of the model.

The strength of the attack depends on the chosen N_{push} and N_{pull}. Indeed, with $N_{push} = N_{pull} = 0$, there will be no guide. In this case, SMA can be used to generate adversarial examples. Otherwise, FNA can be used, with pushing (if $N_{pull} = 0$), pulling (if $N_{push} = 0$) or both (if $N_{push} \neq 0$ and $N_{pull} \neq 0$). Unlike Bai *et al*. [2], we are generating the adversarial examples during training with the latest iteration of the model. The generated adversarial examples are then used for training. Algorithm 3 illustrates the defense with GOAT.

This protocol supports online adversarial training with any metric attacks and any number of guides.

Dataset	Defended Model	Metric	mAP (%)			
			Original	SG. IFGSM [2]	SMA *(Ours)*	FNA *(Ours)*
Market	C_{off} [2]	L_2	**70.11**	2.57	1.66	0.58
		Cosine	**71.99**	1.33	0.09	0.05
	$C^{0,0}_{GOAT}$ *(Ours)*	L_2	68.08	21.67	25.92	11.77
		Cosine	69.99	15.51	22.91	8.7
	$C^{4,1}_{GOAT}$ *(Ours)*	L_2	69.95	**26.33**	**29.45**	**16.76**
		Cosine	71.55	**20.24**	**26.28**	**13.41**
Duke	C_{off} [2]	L_2	**60.77**	1.32	0.52	0.13
		Cosine	**63.28**	0.09	0.05	0.04
	$C^{0,0}_{GOAT}$ *(Ours)*	L_2	59.44	19.73	23.59	8.16
		Cosine	62.53	11.25	20.12	7.76
	$C^{4,1}_{GOAT}$ *(Ours)*	L_2	57.07	**24.85**	**28.59**	**14.12**
		Cosine	60.47	**19.09**	**26.92**	**14.09**

Table 2: Performance (mAP in %) of the defended re-ID models under white-box attack for $\epsilon = 5$. Rows are the defended models under attack, evaluated with Cosine and L_2 metrics, while columns are the attacks (SG. IFGSM [2], SMA, FNA) used. For GOAT, N_{push}, N_{pull} are written in superscript. Higher mAP is better for the defense. Bold numbers are the best defense, for a given metric and a given model type.

6.2. Comparison of the Defenses

The robustness of the defense depends on the strength of FNA and therefore on N_{pull}, N_{push}. Pulling guides must be sampled in the furthest cluster and pushing guides from the same cluster than the attacked image. We train two defended models with GOAT. For the first one, $C^{0,0}_{GOAT}$, we use SMA with $N_{push} = N_{pull} = 0$. For the second one, $C^{4,1}_{GOAT}$, we choose $N_{push} = 4$ and $N_{pull} = 1$ and FNA. We sample the pulling guide as the furthest image in the batch.

We consider the same evaluation settings as in Section 5. We reproduce the offline adversarial training defense [2] (C_{off}) for a white-box comparison with our defended models.

Table 2 presents the performance of the defended models for $\epsilon = 5$ on Market and Duke. It can clearly be seen that, according to our intuition, both models defended with GOAT are more robust than the respective models defended with offline adversarial training. For instance, performance of model C_{off} drops from 70.11% to 0.58%, $C^{0,0}_{GOAT}$ drops from 68.08% to 11.77% and $C^{4,1}_{GOAT}$ drops from 69.95% to 16.76% when attacked with our FNA and evaluated on Market with a L_2 similarity. In addition, we can see that using more guides during training leads to a more robust defense. When trained with $N_{push} = N_{pull} = 0$, the performance under attack is globally about 5 p.p. lower than with $N_{push} = 4$ and $N_{pull} = 1$.

Overall, training with strong metric attacks offers a better robustness while keeping competitive re-ID performance on clean data. Indeed, as shown in Table 2 defending model C on Market decreases the baseline from about 77% mAP to about 70%.

7. Conclusion

Person re-ID is a task where security and robustness are critical. Yet, best models for re-ID are based on deep neural networks, thus, they are prone to adversarial attacks.

In this work, we investigated the effect of pushing and pulling guides on metric attacks. We found that the combination of the two gives the best performance. Then, we proposed two metric attacks for person re-ID depending on the available images: The **Self Metric Attack (SMA)** is *self-sufficient* and easy to deploy because it computes the perturbation using only the input image. The **Furthest-Negative Attack (FNA)** combines pushing guides with pulling guides to generate an adversarial image using all the information available.

These attacks outperform the state of the art of metric attacks on person re-ID. Choice between them depends on the availability of images. The more images available the stronger the attack.

Finally, we studied adversarial defense to improve the robustness of re-ID models against metric attacks. To use efficient metric attacks, we proposed an extension of online adversarial training. Guide sampling Online Adversarial Training (GOAT) trains the model by mining pulling and pushing guides from the training set. The performance of the defense depends on the guides chosen.

References

[1] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning (ICML)*, 2018. 2

[2] Song Bai, Yingwei Li, Yuyin Zhou, Qizhu Li, and Philip H. S. Torr. Metric Attack and Defense for Person Re-identification. *arXiv:1901.10650v2*, 2019. 2, 3, 4, 6, 7, 8

[3] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *International Conference on Learning Representations (ICLR)*, 2018. 2

[4] Nicholas Carlini and David Wagner. Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security - AISec '17*, pages 3–14, Dallas, Texas, USA, 2017. ACM Press. 2

[5] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *Symposium on Security and Privacy (S&P)*, 2017. 2

[6] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2

[7] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2014. 1, 2

[8] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens van der Maaten. Countering adversarial images using input transformations. In *International Conference on Learning Representations (ICLR)*, 2018. 2

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5

[10] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv:1703.07737v4*, 2017. 1, 3, 5

[11] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *International Conference on Learning Representations (ICLR), Workshop*, 2017. 2

[12] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial Machine Learning at Scale. In *International Conference on Learning Representations, ICLR*, 2017. 1, 2

[13] Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Xiaolin Hu, and Jun Zhu. Defense against adversarial attacks using high-level representation guided denoiser. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[14] Zhuoran Liu, Zhengyu Zhao, and Martha Larson. Who's afraid of adversarial queries? the impact of image modifications on content-based image retrieval. In *ACM International Conference on Multimedia Retrieval (ICMR)*. ACM, 2019. 2

[15] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018. 1, 2, 7

[16] N. Narodytska and S. Kasiviswanathan. Simple black-box adversarial attacks on deep neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR), Workshop*, 2017. 2

[17] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *European Symposium on Security and Privacy (EuroS&P)*, 2016. 2

[18] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *Symposium on Security and Privacy (S&P)*, 2016. 2

[19] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. Performance Measures and a Data Set for Multi-target, Multi-camera Tracking. In *European Conference on Computer Vision (ECCV), Workshop*. 2016. 5

[20] Jérôme Rony, Luiz G. Hafemann, Luiz S. Oliveira, Ismail Ben Ayed, Robert Sabourin, and Eric Granger. Decoupling direction and norm for efficient gradient-based l2 adversarial attacks and defenses. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2, 7

[21] Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A unified embedding for face recognition and clustering. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 3

[22] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014. 1, 2

[23] Giorgos Tolias, Filip Radenović, and Ondrej Chum. Targeted mismatch adversarial attack: Query with a flower to retrieve the tower. In *International Conference on Computer Vision (ICCV)*, 2019. 2

[24] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. *arXiv:2002.08347*, 2020. 2

[25] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations (ICLR)*, 2017. 1, 2

[26] Fu Xiong, Yang Xiao, Zhiguo Cao, Kaicheng Gong, Zhiwen Fang, and Joey Tianyi Zhou. Good practices on building effective CNN baseline model for person re-identification. In *International Conference on Graphics and Image Processing (ICGIP)*. SPIE, 2018. 1, 3, 5

[27] Zhewei Yao, Amir Gholami, Peng Xu, Kurt Keutzer, and Michael Mahoney. Trust region based adversarial attack on neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2

[28] Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. Scalable person re-identification: A benchmark. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 5

[29] Liang Zheng, Yi Yang, and Alexander G. Hauptmann. Person Re-identification: Past, Present and Future. *arXiv:1610.02984v1*, 2016. 1

[30] Zhedong Zheng, Liang Zheng, Zhilan Hu, and Yi Yang. Open Set Adversarial Examples. *arXiv:1809.02681v1*, 2018. 2, 3, 6