Poisoning Attacks to Compromise Face Templates

Battista Biggio, Luca Didaci, Giorgio Fumera, and Fabio Roli
Department of Electrical and Electronic Engineering, University of Cagliari
Piazza d'Armi, 09123 Cagliari, Italy
{battista.biggio,didaci,fumera,roli}@diee.unica.it

Abstract

Adaptive biometric systems update clients' templates during operation to account for natural changes over time (e.g., aging of biometric templates). Recently, it has been shown that this update can be exploited by an attacker to compromise the clients' templates: by presenting a proper sequence of fake biometric traits to the sensor, the attacker may eventually impersonate the targeted clients without any fake trait, and even force the system to deny access to them. This attack has however been shown only for PCA-based face verification, with one template per client, under worst-case assumptions about the attacker's knowledge of the system. In this paper, we show that it can be successful even in the case of multiple templates per client, for different matchers, and under more realistic scenarios, and validate it by experiments to highlight its practical relevance.

1. Introduction

Performance of biometric systems may degrade over time due to factors like temporal variations of the biometrics (e.g., aging), or changes in the environmental or acquisition conditions. Further, the high intra-class variability of each client's data distribution (e.g., different facial expressions) is not usually captured completely during enrollment. To overcome these limitations, *adaptive* biometric systems were proposed, based on the idea of updating the clients' templates during operation (see, e.g., [8, 9, 4]).

Recently, it has been shown that an attacker may exploit the adaptation phase to *compromise* the template gallery of a targeted client, through a *poisoning* attack [1].[1] This attack consists of submitting a carefully designed sequence of fake biometric traits to the sensor, in order to *gradually update* the client's template until it is fully compromised, i.e., replaced with a different one. This may allow the attacker to impersonate the targeted client without using any fake trait, and also force the system to deny access to the targeted client. However, the feasibility of poisoning was only proved for PCA-based face verification, with a single template per client, and under the worst-case assumption that the attacker has *perfect knowledge* of the system, including the victim's template.

In this work, we show that the feasibility of such an attack is much more general. To this end, we exploit a formal model of the attacker that we are developing in the context of *adversarial* pattern recognition problems, described at the beginning of Sect. 3. This allows us to show that poisoning attacks are feasible: (i) under more realistic scenarios characterized by a limited attacker's knowledge, in particular, when only an estimate of the victim's template is available, e.g., a face image collected from a personal web page (Sect. 3.1); (ii) for different matching algorithms than PCA-based ones, in particular, even when biometric samples are not represented in an *explicit* feature space, including the case when matching is performed in an encrypted domain (Sect. 3.2); and (iii) when multiple templates per client are used (Sect. 3.3). We report some experiments to validate the effectiveness of poisoning attacks, and highlight that their success is highly dependent on the particular attacker-victim pair (Sect. 4). To characterize this behavior, we also propose a different definition of wolves and lambs of the Doddington's Zoo. Finally, although we focus on face verification in this work, it is worth noting that, in principle, poisoning attacks can target also systems based on other biometric traits (e.g., fingerprints). This issue is discussed in Sect. 5, together with other future research lines.

2. Background

In this section we briefly summarize the work in [1], where the authors considered a PCA-based face verification system with one template per client. This template was computed by averaging a set of n enrolled images of the same client in the feature space $\mathcal{X}$ (i.e., the space spanned by the PCA eigenvectors), and it was thus referred to as *centroid*. During verification, the centroid $\mathbf{x}_c$ of the claimed identity was updated if the matching score between the submitted trait $\mathbf{x}$ and $\mathbf{x}_c$ was equal to or greater than a predefined threshold θ_c, according to the so-called *template self-*

[1] Poisoning was originally investigated in the field of adversarial machine learning to mislead online learning algorithms [7].

978-1-4799-0310-8/13/$31.00 ©2013 IEEE

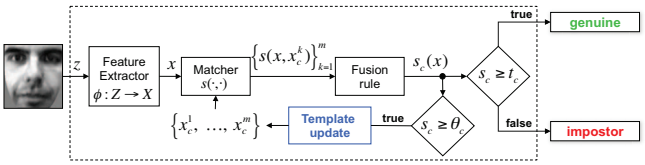

Figure 1. Components of an adaptive biometric verification system that exploits self-update.

update technique [8, 9]. In particular, two policies were considered to update $\mathbf{x}_c$ to $\mathbf{x}'_c$, based on the following expression:

$$\mathbf{x}'_c = \left(1 - \frac{1}{n}\right)\mathbf{x}_c + \frac{1}{n}\mathbf{x}. \quad (1)$$

The first, namely, the *infinite window* policy, updates the centroid without discarding any of the past n samples; accordingly, n has to be increased by 1 *before* each update, and the impact of each update thus reduces as n grows. The second, called the *finite window (average-out)* policy, is more adaptive, since discards the current centroid at each iteration, while keeping n fixed to its initial value (see [7, 1] for further details). It was then shown how an attacker who has *perfect knowledge* of the system, including the victim's template $\mathbf{x}_c$, can forge a sequence of fake traits that allows him to compromise $\mathbf{x}_c$ by replacing it with a desired template $\mathbf{x}_a$ (e.g., an image of his own face). In particular, the *optimal* attack was derived as the *shortest* sequence of such fake traits, as discussed in detail in Sect. 3.1.

3. Compromising adaptive face verification

Although assuming that the attacker has perfect knowledge of the system may be realistic in some cases, to understand whether poisoning is a relevant threat for adaptive biometric systems, alternative *attack scenarios* should also be investigated. To this end, we exploit a general framework we are currently developing for *adversarial* pattern recognition problems, for which biometrics is a well-defined application example. This framework, summarized below, allows one to *formally* define an attack scenario and the corresponding *optimal* attack strategy, based on specific assumptions on the attacker's *goal*, *knowledge*, and *capability*.

Goal. According to a recent taxonomy of potential attacks against machine learning algorithms [5], the attacker's goal can be defined by the desired *security violation*, which may be an integrity, availability, or privacy violation. In the considered application, integrity is violated if the attacker is able to impersonate a specific client (intrusion); availability is violated if the targeted client is denied access (denial-of-service); and privacy is violated if the attacker is able to steal confidential information from the system, e.g., the victim's template(s).

Knowledge. The components of a biometric verification system that may be known to the attacker are (see Fig. 1): (i) the targeted client's template gallery $\{\mathbf{x}_c^k\}_{k=1}^m$; (ii) the feature extraction algorithm $\phi : \mathcal{Z} \mapsto \mathcal{X}$, that maps the input face images $z \in \mathcal{Z}$ to the feature space $\mathcal{X}$; (iii) the matching algorithm $s(\cdot,\cdot)$; (iv) the fusion rule $s_c(\cdot)$, which aggregates the matching scores between the submitted trait and the claimed identity's templates; (v) the template update algorithm; (vi) the verification threshold t_c; and (vii) the self-update threshold θ_c. Note that for centroid-based PCA, the template gallery of each client only includes the centroid $\mathbf{x}_c$, and that the fusion rule is simply $s_c(\mathbf{x}) = s(\mathbf{x}, \mathbf{x}_c)$.

Capability. The attacker may control some test data, by submitting samples during operation. If the system performs template update, he may also compromise the clients' templates, which amounts to controlling some training data.

As in [1], we assume that the attacker's goal is to violate system integrity by gradually compromising the template(s) of a targeted client, through a sequence of carefully designed fake traits. This eventually allows the attacker to impersonate the targeted client without using any fake trait. Among the different, potential scenarios that our model allows one to investigate, in this paper we assume that the attacker knows the matching algorithm and the update policy (as in [1]), but has only an estimate of the victim's template. This simulates a realistic scenario in which the attacker is able to gather a face image of the victim from a personal web page, or a social network. Note that more restrictive assumptions on the attacker's knowledge would amount to assuming *security-by-obscurity*, i.e., that the system is secure based on some details that have to be kept *secret* from the attacker. Since this is not often guaranteed in practice, according to the paradigm of *security-by-design*, we assumed here that the attacker knows the system components, but not the biometric templates (i.e., the *training* data).

In the following, we extend the results in [1] under the assumed attack model. We first derive the optimal poisoning attack strategy when a *centroid* per client is used, and biometric samples are represented in an explicit feature space. We then show how a poisoning attack can be carried out for other matching algorithms than PCA-based ones, when inverting the mapping of biometric samples to a feature space may be difficult, or no representation in an explicit feature space is available (e.g., matching of hashed or encrypted templates [6]). We finally extend the above results to the case of multiple templates per client.

3.1. Poisoning with limited knowledge

We derive now the optimal attack strategy in the case of limited knowledge, as described in the previous section. The goal is to replace the victim's centroid $\mathbf{x}_c$ with an attacker's sample $\mathbf{x}_a$, while minimizing the number of attack iterations. To this end, as shown in [7, 1], at each iteration the attack sample $\mathbf{x}$ can be obtained by minimizing the distance between $\mathbf{x}_a$ and the centroid updated by $\mathbf{x}$. We focus here on the *finite window (average-out)* update policy de-

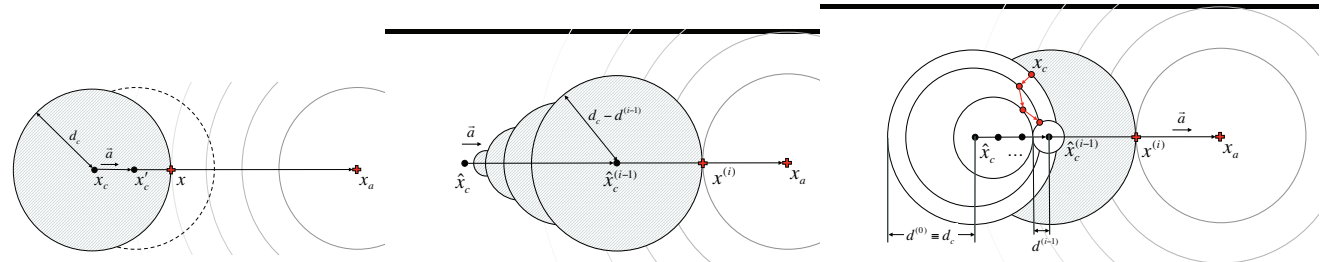

Figure 2. *Left*: poisoning attack with perfect knowledge. The gray-filled circle highlights the feasible domain $||\mathbf{x} - \mathbf{x}_c|| \leq d_c$; the circles centered on $\mathbf{x}_a$ represent the objective function $||\mathbf{x} - \mathbf{x}_a||$, minimized by the attack point $\mathbf{x}$ on the feasible domain. The updated centroid $\mathbf{x}'_c$ and the feasible domain for the next attack iteration are also shown. *Middle* and *right*: poisoning attack with limited knowledge. In the middle plot, we highlight that the feasible domain $||\mathbf{x} - \mathbf{x}_c|| \leq d_c - d$ grows after each attack iteration, since the uncertainty d decreases (see the empty circles in the right plot). In the right plot, we also report the trajectory of the 'true' template $\mathbf{x}_c$ during the attack sequence.

scribed in Sect. 2, but our analysis can be easily extended to the infinite window case. It is easy to show that both the update policies described by Eq. (1) drift the victim's template toward $\mathbf{x}$, proportionally to $||\mathbf{x} - \mathbf{x}_c||$. This implies that the optimal attack will be as close as possible to $\mathbf{x}_a$. Accordingly, one may find the optimal attack sample by solving the following optimization problem:

$$\min_{\mathbf{x}} ||\mathbf{x} - \mathbf{x}_a||, \text{ s.t. } ||\mathbf{x} - \mathbf{x}_c|| \leq d_c, \tag{2}$$

where the constraint expresses the update condition $s_c(\mathbf{x}) \geq \theta_c$ in terms of distances in feature space. The solution can be found analytically by solving the Karush-Kuhn-Tucker conditions, which yields $\mathbf{x} = \mathbf{x}_c + d_c\vec{a}$, where $\vec{a} = \frac{\mathbf{x}_a - \mathbf{x}_c}{||\mathbf{x}_a - \mathbf{x}_c||}$ is the so-called *attack direction* [7]. This means that, at each iteration, the attack sample $\mathbf{x}$ will lie at the intersection between the hypersphere which corresponds to the update condition, and the line connecting $\mathbf{x}_a$ and $\mathbf{x}_c$, as shown in Fig. 2 (left). This is the same solution found in [7, 1], under the assumption of perfect knowledge.

Let us now consider the case of limited knowledge, in which the 'true' victim's template $\mathbf{x}_c$ is not known, but an estimate $\hat{\mathbf{x}}_c$ is available to the attacker. In particular, we assume that the true template $\mathbf{x}_c$ lies on a hypersphere of radius d centered on the estimated template $\hat{\mathbf{x}}_c$, that is:

$$\mathbf{x}_c = \hat{\mathbf{x}}_c + d\vec{v}, \tag{3}$$

where $||\vec{v}|| = 1$, and $d > 0$ represents the uncertainty about the true x_c. In this case the update condition becomes $||\mathbf{x} - \hat{\mathbf{x}}_c - d\vec{v}|| \leq d_c$, which has to be satisfied for any $\vec{v}$. Therefore, we can upper bound the left-hand side by $||\mathbf{x} - \hat{\mathbf{x}}_c|| + d$, and rewrite the constraint in Eq. (2) as $||\mathbf{x} - \hat{\mathbf{x}}_c|| \leq d_c - d$, which defines again a spherical domain. The solution can be thus obtained similarly to the previous case, as:

$$\mathbf{x} = \hat{\mathbf{x}}_c + (d_c - d)\vec{a}, \tag{4}$$

where $\vec{a}$ is now defined as $\vec{a} = \frac{\mathbf{x}_a - \hat{\mathbf{x}}_c}{||\mathbf{x}_a - \hat{\mathbf{x}}_c||}$.

At the first attack iteration, it is reasonable to assume the maximum possible uncertainty about the position of $\mathbf{x}_c$ that is compatible with the update condition, i.e., $d = d_c$; accordingly, the first attack will be $\mathbf{x} = \hat{\mathbf{x}}_c$. At the next iterations, the attack samples can be still obtained from Eq. (4), provided that the attacker can compute the *updated* values of $\hat{\mathbf{x}}_c$ and d. To this end, we first substitute $\mathbf{x}_c$ from Eq. (3) into Eq. (1), which yields:

$$\mathbf{x}'_c = \underbrace{\left[\left(1 - \frac{1}{n}\right)\hat{\mathbf{x}}_c + \frac{1}{n}\mathbf{x}\right]}_{\hat{\mathbf{x}}'_c} + \underbrace{\left[\left(1 - \frac{1}{n}\right)d\right]}_{d'}\vec{v}. \tag{5}$$

The above equation shows that, for any point $\mathbf{x}$ used to update $\mathbf{x}_c$, the updated template $\mathbf{x}'_c$ still lies on a hypersphere of radius d' centered on the updated estimate $\hat{\mathbf{x}}'_c$, and, in particular, that $d' < d$, i.e., the uncertainty on the true template position decreases. After i template updates, it is easy to show that the value of d will be thus given by:

$$d^{(i)} = d_c\left(1 - \frac{1}{n}\right)^i, \tag{6}$$

where d_c is the initial value of d (as in our case). The above equation highlights that the uncertainty on the position of $\mathbf{x}_c$ with respect to $\hat{\mathbf{x}}_c$ decreases *exponentially* as i increases. If we now substitute the attack sample $\mathbf{x}$ from Eq. (4) into the first term of Eq. (5), we obtain:

$$\hat{\mathbf{x}}'_c = \hat{\mathbf{x}}_c + \frac{1}{n}(d_c - d)\vec{a}, \tag{7}$$

which allows the attacker to update $\hat{\mathbf{x}}_c$, and, thus, to compute all the attack samples according to Eq. (4). The attack progress is also illustrated in Fig. 2 (middle and right plots).

By iteratively expanding the recursive term in Eq. (7) we obtain the i-th centroid estimate as:

$$\hat{\mathbf{x}}_c^{(i)} = \hat{\mathbf{x}}_c + \frac{i}{n}d_c\vec{a} - \frac{1}{n}\sum_{k=0}^{i-1} d^{(k)}\vec{a}, \tag{8}$$

where $\sum_{k=0}^{i-1} d^{(k)}$, according to Eq. (6), is a geometric series with ratio $(1 - 1/n)$, that thus equals $d_c - d^{(i)}$. Substituting $d^{(i)}$ from Eq. (6), and rearranging terms in Eq. (8) to compute $||\hat{\mathbf{x}}_c^{(i)} - \hat{\mathbf{x}}_c||$, it is possible to find the minimum number

of iterations i required to complete the attack. In particular, if the attack ends when $\hat{\mathbf{x}}_c^{(i)} = \mathbf{x}_a$, then the number of attack samples i is given by:

$$i = n \left[\frac{||\mathbf{x}_a - \hat{\mathbf{x}}_c||}{d_c} + 1 - \left(1 - \frac{1}{n}\right)^i \right], \qquad (9)$$

which is upper bounded by $n\left(\frac{||\mathbf{x}_a - \hat{\mathbf{x}}_c||}{d_c} + 1\right)$. This means that, as in the case of perfect knowledge with the same update policy [1, 7], the number of iterations scales *linearly* with respect to $\frac{||\mathbf{x}_a - \hat{\mathbf{x}}_c||}{d_c}$. Further, our result shows that poisoning with limited knowledge requires at most *only* n attack samples more. Finally, it is worth noting that the attack derived in this section is equivalent to that obtained with perfect knowledge, if one assumes the *worst-case* in which $\mathbf{x}_c$ is the farthest from $\mathbf{x}_a$, i.e., $\mathbf{x}_c = \hat{\mathbf{x}}_c - d_c\vec{a}$.

3.2. Building the spoofing attack samples

Poisoning attacks provide a sequence of *feature vectors* of attack samples $\{\mathbf{x}_a^{(i)}\}_{i=1}^p$. However, the attacker is required to submit a sequence of *spoofed* traits to the sensor, namely, images whose features will correspond to the desired $\mathbf{x}_a^{(i)}$. It is thus necessary for the attacker to *invert* the feature mapping, that is, to find the function $\phi^{-1} : \mathcal{X} \mapsto Z$ (if it exists). For a PCA-based matching algorithm, this is simply achieved by computing the pseudo-inverse of the eigenvectors' matrix. More generally, inverting the feature mapping is not trivial, since the function ϕ may be not analytically defined, or reverse-engineering the matching algorithm may be too difficult.[2]

In the following we suggest a heuristic method to overcome this problem. Assuming that the matching score is a monotonically decreasing function of the distance in feature space, we reformulate the attack strategy in Eq. (2) as:

$$\max_{\mathbf{x}} s(\mathbf{x}, \mathbf{x}_a), \text{ s.t. } s_c(\mathbf{x}) \geq \theta_c. \qquad (10)$$

Essentially, we are looking for the closest attack sample to the attacker's face image, that updates the victim's template(s). Since the objective function $s(\cdot,\cdot)$ is not known analytically in this case, we can search for an approximate solution using heuristics. We propose here a simple search heuristic based on a convex combination of $\mathbf{z}_a$ and $\mathbf{z}_c$: $\mathbf{z} = (1-\alpha)\mathbf{z}_c + \alpha\mathbf{z}_a$, where the optimal value of $\alpha \in [0,1]$ can be found efficiently by binary search. This amounts to searching the attack image directly in the space of face images $\mathcal{Z}$ instead of inverting the feature mapping. Clearly, this may not yield the optimal attack sequence, since it is not guaranteed that the sequence of attacks will lie on the line connecting $\mathbf{x}_a$ and $\mathbf{x}_c$ in feature space. Nevertheless, even in this case, we will show by experiments that this technique allows the attacker to perform a very efficient and effective attack against matching algorithms which are difficult to reverse-engineer.

3.3. Template galleries

Thus far, we have considered poisoning attacks against centroid-based verification systems, where one template per client is available (*centroid*). However, the same attack can also be effective against verification systems based on multiple templates per client depending on the update policy, and the score fusion rule (see Fig. 1).

Let us first consider the *maximum* score fusion rule $s_c(\mathbf{x}) = \max\{s(\mathbf{x}, \mathbf{x}_c^k)\}_{k=1}^m$, and the *nearest-out* update policy [7], which replaces the closest template to the submitted trait $\mathbf{x}$ (i.e., $\text{argmax}_{x_c^k}\{s(\mathbf{x}, \mathbf{x}_c^k)\}_{k=1}^m$), if $s_c(\mathbf{x}) \geq \theta_c$. In this case, the poisoning attack considered in Sect. 3.1 can be applied without modifications; the attack proceeds as described, and the minimum number of iterations is still given by Eq. (9). In particular, since templates are replaced instead of being gradually updated, the update policy can be still described by Eq. (1) with $n = 1$, and the value of $s_c(\mathbf{x})$ at the i-th iteration can be estimated by the attacker as $s(\mathbf{x}^{(i)}, \mathbf{x}^{(i-1)})$, given that the previous attack sample $\mathbf{x}^{(i-1)}$ is now part of the victim's gallery. This attack eventually replaces the *nearest* victim's template to the initial attack sample (i.e., $\hat{\mathbf{x}}_c$).[3] The attack can be then iterated, using the same attack sequence. If the initial attack sample $\hat{x}_c$ is close enough to update another client's template, the latter can be eventually replaced with another arbitrary sample. Therefore, the whole gallery may be potentially compromised.

If we consider the same fusion rule but cache-style update policies [4], compromising the whole gallery may be even easier. In fact, at each attack iteration a different template is replaced, and we can thus obtain a fully compromised gallery if the number of iterations is at least equal to the size of the template gallery m.

In general, if a different score fusion rule is adopted, the value of $s_c(\mathbf{x})$ at each iteration is not exactly known, since the gallery itself is not completely known to the attacker; e.g., this is true for the mean rule $s_c(\mathbf{x}) = 1/m \sum_{k=1}^m s(\mathbf{x}, \mathbf{x}_c^k)$. Thus, generally speaking, we should consider a different poisoning attack strategy. While we left a more detailed investigation to future work, an interesting case can be discussed here. Let us consider the sum rule and the *nearest-out* update policy. As previously discussed, under this policy the attack essentially replaces *one* gallery template during its progress. The constraint in Eq. (10) can be thus rewritten at the i-th attack iteration as $s(\mathbf{x}^{(i)}, \mathbf{x}^{(i-1)}) \geq m\theta_c - \Lambda(\mathbf{x})$, where $\Lambda(\mathbf{x}) =$

[2] Note that this problem is similar to the so-called pre-image problem in kernel methods, where the goal is to find samples in the input space that correspond to given points in the feature space, being unknown the feature mapping [10].

[3] Note that, since $n = 1$ in this case, a significantly smaller number of iterations is required to replace *one* of the victim's templates.

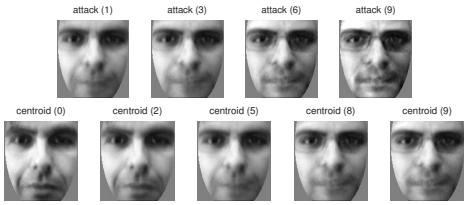

Figure 3. Attack samples (top) and victim's centroids (bottom) for poisoning with perfect knowledge, at different iterations.

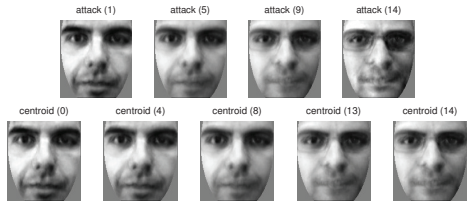

Figure 4. Attack samples (top) and estimated centroids (bottom) for poisoning with limited knowledge, at different iterations.

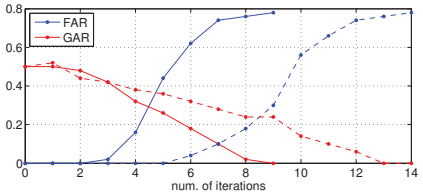

Figure 5. FAR and GAR for poisoning with perfect (solid lines) and limited (dashed lines) knowledge, at different iterations.

$\sum_{k \neq k'} s(\mathbf{x}, \mathbf{x}_c^k)$), and k' is the index of the victim's template replaced by the first attack sample $\hat{\mathbf{x}}_c$. The term $\Lambda(\mathbf{x})$ reasonably decreases during the attack progress, since the attack sample is moving away from the victim's templates. Thus, the minimum value of $s(\mathbf{x}^{(i)}, \mathbf{x}^{(i-1)})$ required to perform update increases after each iteration. The mean rule may thus prevent complete poisoning if the attacker tries to compromise one template at a time. This inspires some potential countermeasures to poisoning, similar to data sanitization in intrusion detection systems [2].

4. Experiments

We experimentally evaluate the effectiveness of our approach on the same dataset used in [1]. It consists of 40 clients with 60 images each, acquired under different lighting conditions and facial expressions, in two temporal sessions. We randomly selected 5 images per client for training, i.e., to obtain the PCA eigenvectors and the clients' centroids (or galleries); a further set of 5 images per client was used to tune the acceptance threshold t_c and the update threshold θ_c of each client, respectively at the 1% and 0% FAR operating points; and the remaining 50 images per client were used for testing. Since choosing different data splits did not substantially affect our results, we report results for a single data split.

Centroid-based PCA. We start by considering a PCA-based face verification task in which each user is authenticated by matching the submitted trait against the *centroid* associated to the claimed identity. For each client, the centroid is initially obtained by averaging the 5 available training images, and updated according to the *finite-window (average-out)* policy (Eq. 1). In the case of limited knowledge, the attacker must provide an estimate of the victim's template to start the attack sequence. A rational choice could be to search for a victim's image in frontal pose, acquired under proper lighting conditions. To simulate this behavior, we selected the victim's image available to the attacker as the victim's image in the test data that yields the maximum matching score with the *true* victim's centroid.

In Figs. 3 and 4, we report the sequence of attack images and corresponding centroids at different attack iterations (reported in parentheses) for a given attacker-victim pair, in the case of perfect knowledge (Fig. 3), and limited knowledge (Fig. 4). More precisely, in the case of limited knowledge (Fig. 4), the depicted centroids represent the attacker's estimates of the true victim's centroid. According to the analysis of Sect. 3.1, the first attack sample in Fig. 4 is equal to the initial estimate of the victim's centroid. Further, the centroids at iterations 13 and 14 in Fig. 4 are identical to those depicted in Fig. 3 at iterations 8 and 9. The reason is that the uncertainty in the estimated template (represented by $d^{(i)}$ in Eq. 6) reduces exponentially with the attack progress. Lastly, note that poisoning with limited knowledge requires 5 attack iterations more than poisoning with perfect knowledge (14 attacks against 9) to fully compromise the victim's template, coherently with the analysis in Sect. 3.1 (recall that $n = 5$ in this case).

As pointed out in [1], even if the number of iterations required to fully compromise the victim's template may be too high for the attack to be practical, the False Acceptance Rate (FAR) of the attacker may grow rather quickly through the attack progress; similarly, the Genuine Acceptance Rate (GAR) of the victim may significantly decrease after few attack iterations. The FAR and GAR values as a function of the attack progress for the attacker-victim pair considered before are shown in Fig. 5. It is worth noting how the FAR and GAR values obtained by poisoning with limited knowledge attain essentially the same FAR and GAR values of the perfect knowledge case after 5 iterations. This is also confirmed by repeating the analysis on all the possible attacker-victim pairs, as shown in Fig. 6, where we report the FAR and the GAR for each given victim, averaged over all possible attackers, at different iterations. However, in the case of limited knowledge, the attack does not produce any effect against some victims (e.g., victim 4, 15), since the initial estimate of their templates did not cause any update.

In Fig. 6 we also report the so-called 'zoo plot', for both perfect and limited knowledge cases [11, 3]. The Doddington's Zoo defines clients according to different categories of 'animals', essentially depending on whether they match

well against themselves, and/or poorly against others. In the standard definition, the animals are defined considering an attacker who attempts to impersonate different clients using *his own* biometric traits. In the case of poisoning, attackers and victims (respectively, wolves and lambs) can be redefined depending on whether they can easily compromise a victim's template. For instance, wolves can be defined based on the number of iterations required to attack a certain user (averaged on all possible victims), and lambs can be defined based on the number of iterations required to compromise their template (averaged on all possible attackers). For each client, alternatively considered as an attacker or a victim, these two values are reported on the two-dimensional scatter plot (i.e., the zoo plot) in Fig. 6. First, it can be appreciated that more iterations are required to attackers (wolves) to compromise the victim's template (lambs) in the case of limited knowledge. Second, wolves do not necessarily become lambs when they are targeted by the attack, and vice-versa, since the number of attack iterations depends on the victim's update threshold, which is user-specific, and may thus change when inverting the role of the attacker (wolf) with that of the victim (lamb).

PCA with template galleries. We now consider PCA-based face verification where the submitted face image is matched against a gallery of 5 templates, instead of a single *centroid*. The matching score is computed as the maximum one obtained from the gallery. The update policy is the *nearest-out*, namely, the template which achieves the maximum matching score with the submitted trait is replaced by the latter, if the score is equal to or greater than the update threshold. We consider poisoning with limited knowledge. The initial estimate of the victim's template is obtained as in the previous case. As described in Sect. 3.3, each attack iteration will replace one of the victim's templates with the current attack sample. Therefore, with respect to poisoning with perfect knowledge, only *one* additional iteration is required to replace one of the victim's templates in gallery with the initial estimate available to the attacker, who then gains perfect knowledge of the attacked template.

Results are reported in Fig. 7 (top row). It can be noted that this attack requires a significant lower number of iterations to fully compromise a victim's template than those required in the case of centroid-based PCA, since the victim's template is *replaced* at each iteration by the current attack sample, instead of being *gradually* updated (see Sect. 3.3). The average FAR attains similar values to those shown in Fig. 6 for the centroid-based PCA cases, while the average GAR does not decrease significantly, since only one template (out of five) is compromised. Finally, in this case too, the attack is not successful against some victims, due to a poor initial estimate of their templates.

EBGM with template galleries. We run the same experiment as described for PCA with template galleries, but

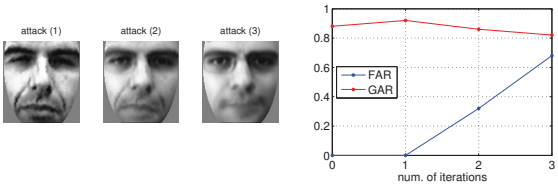

Figure 8. EBGM with template galleries. Attack samples (*left*) and FAR and GAR (*right*) at different iterations.

using the Elastic Bunch Graph Matching (EBGM) algorithm,[4] instead of the PCA. In this case, we obtain the spoofed face images according to the heuristic approach described in Sect. 3.2, namely, without explicitly inverting the feature mapping. Although this approach essentially ignores knowledge of the feature space, it turned out to be very effective. In Fig. 8 we report the results for a given attacker-victim pair; with only three attack iterations, the attacker is able to replace one of the five victim's templates. Although only one template out of five is compromised, the FAR reaches a very high value. For the same reason, the GAR does not decrease. Similar results hold when considering all possible attacker-victim pairs (Fig. 7, bottom row).

5. Conclusions and future work

In this work we have shown that poisoning attacks against adaptive face verification systems may be successful even if the attacker has limited knowledge of the system, at the expense of a slightly higher number of iterations, and provided that the attacker has a good-quality estimate of the victim's template. We proposed a heuristic algorithm that allows one to attack *any* matching algorithm, without even exploiting knowledge of the underlying feature representation, and discussed how poisoning can be performed when a template gallery is available instead of a single centroid per client. We also showed that the success of this attack is clearly user-specific, and redefined wolves and lambs of the Doddington's Zoo accordingly.

As future work, our formulation of poisoning can be extended to account for *genuine* template updates that may occur during the attack sequence, similarly to [7]. Another interesting open issue is to consider poisoning attacks against systems based on different biometrics (e.g., fingerprints), although fabricating the desired sequence of fake traits may be much more difficult in these cases. Last but not least, the design of specific countermeasures to poisoning should be also investigated; e.g., as mentioned in Sect. 3.3, one may adopt fusion rules that produce similar effects to data sanitization [2] (like the *mean* rule), namely, that prevent some of the client's templates to be shifted too far from the rest of the gallery. However, the trade-off that may arise between the system's adaptability and its vulnerability to poisoning should be quantitatively analyzed.

[4] http://www.cs.colostate.edu/evalfacerec/algorithms5.php

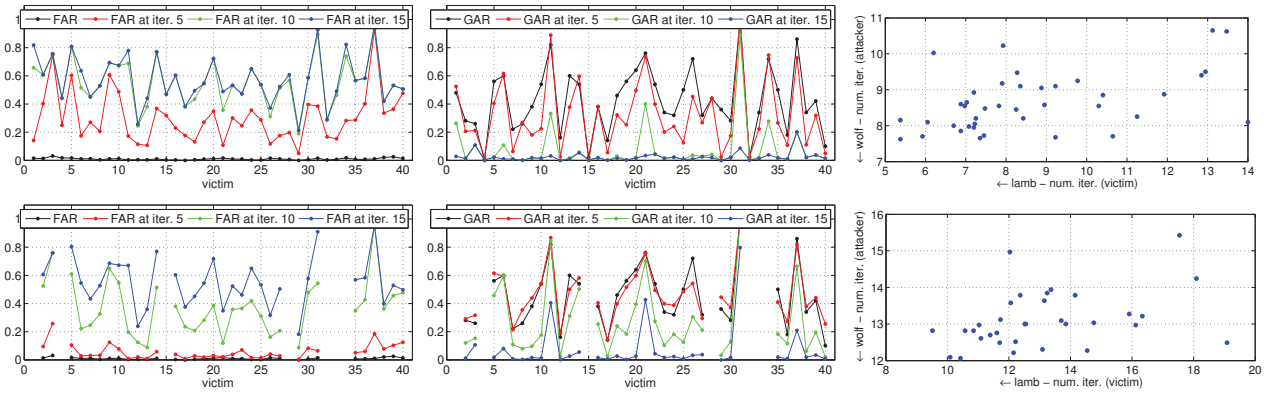

Figure 6. Poisoning centroid-based PCA with perfect (top) and limited (bottom) knowledge. *First* and *second column*: FAR and GAR averaged over all possible attackers, given the victim, at different iterations. *Third column*: zoo plot showing the number of iterations averaged over all victims, given the attacker, against the number of iterations averaged over all attackers, given the victim.

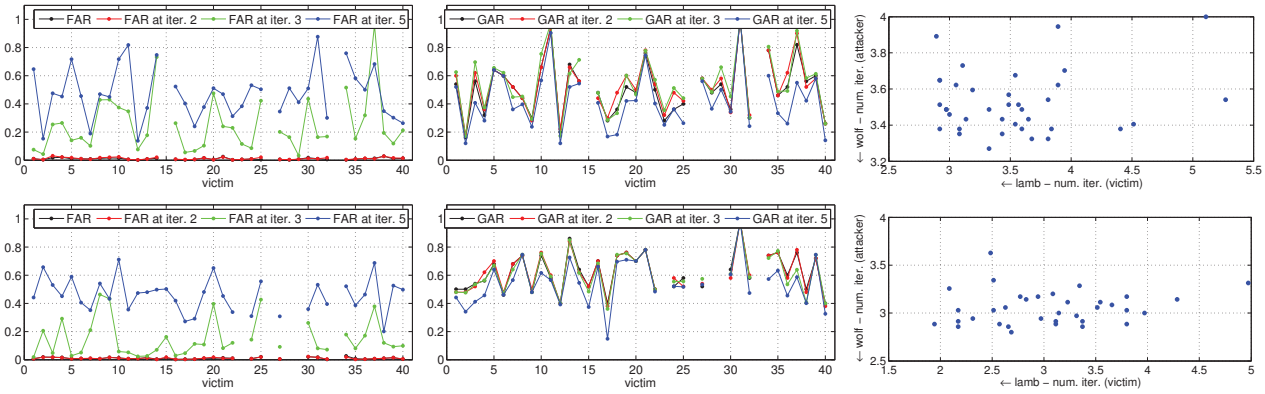

Figure 7. Poisoning PCA (top) and EBGM (bottom) with template galleries. See the caption of Fig. 6 for further details.

Acknowledgements

This work has been partly supported by the TABULA RASA project, 7th Framework Research Programme of the European Union (EU), grant agreement number: 257289; and by the project CRP-18293 funded by Regione Autonoma della Sardegna, L.R. 7/2007, Bando 2009.

References

[1] B. Biggio, G. Fumera, F. Roli, and L. Didaci. Poisoning adaptive biometric systems. SSPR/SPR 2012, pp. 417-425 1, 2, 3, 4, 5

[2] G. F. Cretu, A. Stavrou, M. E. Locasto, S. J. Stolfo, and A. D. Keromytis. Casting out demons: Sanitizing training data for anomaly sensors. IEEE Symp. on Security and Privacy 2008, pp. 81-95 5, 6

[3] G. R. Doddington, W. Liggett, A. F. Martin, M. A. Przybocki, and D. A. Reynolds. Sheep, goats, lambs and wolves: a statistical analysis of speaker performance in the NIST 1998 speaker recognition evaluation. ICSLP 1998 5

[4] B. Freni, G. L. Marcialis, and F. Roli. Replacement algorithms for fingerprint template update. ICIAR 2008, pp. 884-893 1, 4

[5] L. Huang, A. D. Joseph, B. Nelson, B. Rubinstein, and J. D. Tygar. Adversarial machine learning. IEEE Internet Computing 15(5): 4-6, 2011. 2

[6] A. K. Jain, K. Nandakumar, and A. Nagar. Biometric template security. *EURASIP J. Adv. Signal Process*, 2008:1–17, 2008. 2

[7] M. Kloft and P. Laskov. Online anomaly detection under adversarial impact. In *13th Int'l Conf. on Artificial Intell. and Statistics*, pp. 405–412, 2010. 1, 2, 3, 4, 6

[8] A. Rattani, B. Freni, G. Marcialis, and F. Roli. Template update methods in adaptive biometric systems: A critical review. ICB 2009, pp. 847-856. 1, 2

[9] F. Roli and G. Marcialis. Semi-supervised PCA-based face recognition using self-training. SSPR/SPR 2006, pp. 560-568. 1, 2

[10] B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K.-R. Muller, G. Rätsch, and A. J. Smola. Input space versus feature space in kernel-based methods. *IEEE Trans. on Neural Netw.*, 10(5):1000–1017, 1999. 4

[11] N. Yager and T. Dunstone. The biometric menagerie. *IEEE Trans. on Pattern Analysis and Mach. Intell.*, 32(2):220 –230, 2010. 5