Sparse Support Faces

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

Abstract

Many modern face verification algorithms use a small set of reference templates to save memory and computational resources. However, both the reference templates and the combination of the corresponding matching scores are heuristically chosen. In this paper, we propose a well-principled approach, named sparse support faces, that can outperform state-of-the-art methods both in terms of recognition accuracy and number of required face templates, by jointly learning an optimal combination of matching scores and the corresponding subset of face templates. For each client, our method learns a support vector machine using the given matching algorithm as the kernel function, and determines a set of reference templates, that we call support faces, corresponding to its support vectors. It then drastically reduces the number of templates, without affecting recognition accuracy, by learning a set of virtual faces as well-principled transformations of the initial support faces. The use of a very small set of support face templates makes the decisions of our approach also easily interpretable for designers and end users of the face verification system.

1. Introduction

Face verification can be considered as the client-specific two-class problem of deciding whether the submitted face image belongs to the claimed identity, *i.e.*, if the claim comes from either a *genuine* or an *impostor* user. Early work has implemented this approach by mapping the pixel values of face images onto reduced vector spaces [24, 3, 12], and then exploiting classification algorithms like neural networks and support vector machines (SVMs) [14, 9, 10]. However, simple features like pixel values can be effective only if a large set of face images is collected during enrollment (and used for classifier training), representative of all the conditions that can be incurred during verification, *e.g.*, pose and illumination changes [26, 1, 22]. Since collecting a large training set is often difficult, only a small number of face templates is available in practice for each client. Thus, many face verification methods use template matching algorithms that compensate for potential variations not captured by the training data, without even mapping faces onto a vector space, but matching directly structured representations like graphs [4, 19]. Under this setting, often studied in the extreme case as the *one sample per person* problem [22, 19, 15], the main issue is that feature extraction and template matching can be computationally demanding. As a consequence, the use of methods that exploit a limited number of template matchings (*e.g.*, mean, maximum, and cohort-based fusion) has become a de facto standard [2, 23, 16, 17] (Sect. 2). However, these methods select both the reference templates to be matched against the submitted face image during verification (including the set of cohorts, *i.e.*, face images that do not belong to the claimed identity), and the technique to combine the corresponding matching scores according to heuristic criteria.

To overcome these limitations, we propose a well-principled approach that jointly learns an optimal combination of matching scores and a small set of face templates, with the goal of outperforming state-of-the-art methods both in terms of recognition accuracy and number of reference templates (Sect. 3). We first learn an SVM for each client, using the given *matching algorithm* as the *kernel function*. This enables verifying a claimed identity through a linear combination of matching scores computed between the submitted face and the subset of face templates *automatically* selected by the SVM learning algorithm from the training data, corresponding to its support vectors. To reduce the number of required templates, we then approximate the SVM decision function by replacing the selected face templates with a much smaller set of *virtual* support faces, optimized together with their coefficients to preserve recognition accuracy. We name our approach *sparse support faces*, as it uses a very small set of face templates corresponding to well-principled transformations of the initial support faces. We empirically validate it on two well-known benchmark datasets (Sect. 4). We finally discuss related work (Sect. 5), conclusions and future work (Sect. 6).

2. Template-based Face Verification

Let us assume that face images are objects in an abstract space $\mathcal{X}$, $\mathcal{D} = \{\boldsymbol{x}_i, c_i\}_{i=1}^n$ is a training set of n faces with the corresponding identities $c_i \in \mathcal{C}$, and $s : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ is the matching algorithm, returning a similarity score between two face images. Given a claimed identity c, we denote the set of reference templates belonging to c ("genuine" templates) and all the other ones ("impostors") respectively with $\mathcal{G}_c : \{\boldsymbol{x}_i \in \mathcal{D} \mid c_i = c\}$ and $\mathcal{I}_c : \{\boldsymbol{x}_i \in \mathcal{D} \mid c_i \neq c\}$. Given a submitted face $\boldsymbol{x}$ and a claimed identity c, the scores $s(\boldsymbol{x}, \boldsymbol{x}_i)$ are referred to as *match scores* if $\boldsymbol{x}_i \in \mathcal{G}_c$, and as *non-match* or *cohort scores* if $\boldsymbol{x}_i \in \mathcal{I}_c$. Let $\{y_i\}_{i=1}^n \in \{-1, +1\}^n$ be a set of client-specific labels denoting whether the corresponding $\boldsymbol{x}_i \in \mathcal{D}$ belongs to the claimed identity $c_i = c$ ($y_i = +1$) or not ($y_i = -1$). Face verification can now be formalized as the problem of learning a function $f_c : \mathcal{X} \mapsto \mathcal{Y} = \{-1, +1\}$ for each identity $c \in \mathcal{C}$, that returns $+1$ if the identity claim c for a given face image $\boldsymbol{x}$ is genuine, and -1 otherwise. Without loss of generality, we assume that f_c is obtained by thresholding a discriminant function $g_c : \mathcal{X} \mapsto \mathbb{R}$ with a client-specific decision threshold $\theta_c \in \mathbb{R}$, such that $f_c(\boldsymbol{x}) = +1$ if $g_c(\boldsymbol{x}) \geq \theta_c$, and -1 otherwise. In template-based verification, the functions g_c are defined as combinations of matching scores computed between $\boldsymbol{x}$ and a subset of face templates either in $\mathcal{G}_c$ (genuine template matching), or both in $\mathcal{G}_c$ and $\mathcal{I}_c$ (cohort-based template matching). In the following we summarize these two kinds of techniques.

Genuine template matching. Simple statistics are often used to combine the match scores between $\boldsymbol{x}$ and the genuine templates in $\mathcal{G}_c$, like in the **mean** and **max** rule:

$$g_c(\boldsymbol{x}) = \frac{1}{|\mathcal{G}_c|} \sum_{\boldsymbol{x}_i \in \mathcal{G}_c} s(\boldsymbol{x}, \boldsymbol{x}_i) \enspace , \enspace g_c(\boldsymbol{x}) = \max_{\boldsymbol{x}_i \in \mathcal{G}_c} s(\boldsymbol{x}, \boldsymbol{x}_i) \enspace .$$

Cohort-based template matching. Information coming from the *non-match scores*, disregarded by the above techniques, is additionally exploited by cohort-based techniques, to improve recognition accuracy at the expense of a slightly higher number of matchings [2, 23, 16, 17]. Existing cohort-based techniques rely on heuristic criteria to select a small set of cohort templates $\mathcal{N}_c \subseteq \mathcal{I}_c$ a priori. Typically, the closest templates to the ones in $\mathcal{G}_c$ are selected to this end. Two popular schemes for combining the corresponding matching scores are Aggarwal-max and T-norm.[1]
Aggarwal-max [2] approximates the likelihood ratio rule [18] by combining match and non-match scores as:

$$g_c(\boldsymbol{x}) = \max_{\boldsymbol{x}_i \in \mathcal{G}_c} s(\boldsymbol{x}, \boldsymbol{x}_i) \, / \max_{\boldsymbol{x}_j \in \mathcal{N}_c} s(\boldsymbol{x}, \boldsymbol{x}_j) \enspace .$$

T-norm [23] normalizes data to have zero mean and unit variance, similarly to Z-norm [13], but estimates such parameters during verification, based on the available $\mathcal{N}_c$:

$$g_c(\boldsymbol{x}) = \frac{1}{\sigma_c(\boldsymbol{x})} \left(\frac{1}{|\mathcal{G}_c|} \textstyle\sum_{\boldsymbol{x}_i \in \mathcal{G}_c} s(\boldsymbol{x}, \boldsymbol{x}_i) - \mu_c(\boldsymbol{x}) \right) \enspace ,$$

where $\mu_c(\boldsymbol{x})$ and $\sigma_c(\boldsymbol{x})$ are the mean and standard deviation of the cohort scores $\{s(\boldsymbol{x}, \boldsymbol{x}_j) : \boldsymbol{x}_j \in \mathcal{N}_c\}$.

As all the discussed methods are based on heuristics, it is an open issue to analyze if recognition accuracy can be improved by combining the matching scores in a more principled manner, and using a smaller set of templates.

3. Sparse Support Faces

To overcome the limitations of the aforementioned template matching schemes, we propose a theoretically-sound verification approach that jointly learns a principled combination of matching scores, along with an optimal subset of reference face templates. To this end, we exploit the well-known SVM learning framework [8]. We first learn an SVM for each client, using the matching score as the kernel function.[2] The resulting decision function is $f_c(\boldsymbol{x}) = \operatorname{sign} g_c(\boldsymbol{x})$, where g_c is given as:

$$g_c(\boldsymbol{x}) = \textstyle\sum_{i=1}^n \alpha_i y_i s(\boldsymbol{x}, \boldsymbol{x}_i) + b \enspace , \tag{1}$$

and the parameters $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$ and b are set by the SVM learning algorithm. Besides learning an optimal linear combination of matching scores, according to a principled criterion, this approach provides a *sparse* solution, *i.e.*, only few elements in $\boldsymbol{\alpha}$ are not null, requiring one to compute matching scores during verification only against the corresponding subset of training samples, *i.e.*, the *support vectors* (SVs). This amounts to *automatically* selecting a subset of genuine and impostor face templates, corresponding to the SVs in $\mathcal{G}_c$ (if $y_i = +1$) and in $\mathcal{I}_c$ (if $y_i = -1$), together with the combination coefficients $\boldsymbol{\alpha}$, without requiring any heuristic, a-priori selection. Since this approach does not even require face images to be explicitly represented in a vector space, we more properly refer to the SVs here as *support faces* (SF), and to the SVM as Support Face Machine (SFM). This approach also exhibits two additional advantages: (i) it does not require selecting a client-specific threshold θ_c using a validation set (*i.e.*, $\theta_c = 0$); and (ii) it can naturally deal with strong class imbalances in the training data (in our case, few match scores against a large number of non-match scores) by setting a different value of the SVM regularization parameter C for each class.

Sparse SFM (SSFM) with Virtual Faces (VF). Although the SFM solution is sparse, the number of SFs grows linearly with the training set size, potentially requiring an unacceptable number of matchings for verification, and hindering the interpretability of the SFM decisions. As the second step of our approach, we thus propose a strategy to significantly reduce the set of SFs without affecting the

[1] Although they were defined assuming $|\mathcal{G}_c| = 1$, we consider here a generalized version to consistently deal with multiple templates per person.

[2] Although the matching score may be not a proper (*i.e.*, positive semidefinite) kernel, recent advancements in similarity-based learning have shown that SVMs can still provide principled solutions when trained on indefinite kernels [20, 7, 11] and even on biometric similarities [5].

Algorithm 1 Sparse Support Face Machine (SSFM)

Input: the training data $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1}^n$; the matching algorithm s; the regularization parameters C and λ; the initial support faces $\{\boldsymbol{z}_j^{(0)}\}_{j=1}^m$; the step size η; a small number ϵ.
Output: The coefficients $\boldsymbol{\beta}$ and the support faces $\{\boldsymbol{z}_j\}_{j=1}^m$.

Learn an SVM on $\mathcal{D}$, with kernel s and parameter C.
Compute $\boldsymbol{g}$ by classifying $\mathcal{D}$ with the trained SVM.
Set the iteration count $q \leftarrow 0$.
Compute $\boldsymbol{\beta}^{(0)}$ for $\boldsymbol{z}_1^{(0)}, \ldots, \boldsymbol{z}_m^{(0)}$ using Eq. (4).
repeat
  Set $j \leftarrow \mathrm{mod}(q, m) + 1$ to index a support face.
  Compute $\frac{\partial \Omega}{\partial \boldsymbol{z}_j}$ using Eq. (5).
  Increase the iteration count $q \leftarrow q + 1$
  Set $\boldsymbol{z}_j^{(q)} \leftarrow \boldsymbol{z}_j^{(q-1)} + \eta \frac{\partial \Omega}{\partial \boldsymbol{z}_j^{(q-1)}}$.
  if $\boldsymbol{z}_j^{(q)} \notin \mathcal{X}$, **then** project $\boldsymbol{z}_j^{(q)}$ onto $\mathcal{X}$.
  Set $\boldsymbol{z}_i^{(q)} = \boldsymbol{z}_i^{(q-1)}, \forall i \neq j$.
  Compute $\boldsymbol{\beta}^{(q)}$ on $\boldsymbol{z}_1^{(q)}, \ldots, \boldsymbol{z}_m^{(q)}$.
until $\left|\Omega\left(\boldsymbol{\beta}^{(q)}, \boldsymbol{z}^{(q)}\right) - \Omega\left(\boldsymbol{\beta}^{(q-1)}, \boldsymbol{z}^{(q-1)}\right)\right| < \epsilon$
return: $\boldsymbol{\beta} = \boldsymbol{\beta}^{(q)}$, and $\boldsymbol{z} = \boldsymbol{z}^{(q)}$.

SFM accuracy. Our idea is to approximate the discriminant function $g_c(\boldsymbol{x})$ (Eq. 1) with a *sparser* linear combination:

$$h_c(\boldsymbol{x}) = \textstyle\sum_{j=1}^m \beta_j s(\boldsymbol{x}, \boldsymbol{z}_j) \ ,$$

where $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_m) \in \mathbb{R}^m$ is the vector of coefficients, and $\boldsymbol{z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m) \in \mathcal{X}^m$ is a set of face templates[3] whose number m is *budgeted*, *i.e.*, it is fixed a priori, depending on application-specific requirements. Clearly, to obtain a *sparser* solution, this number has to be lower than the original number of SFs. Inspired by the work in [21], we propose a novel reduction method by formulating the problem as a regression task in which we aim to minimize the distance on the training points between the target function g_c and h_c, with respect both to $\boldsymbol{\beta}$ *and* to the choice of the face templates $\boldsymbol{z}$. In practice, to attain a better trade-off between accuracy and number of SFs, we do not constrain the SFs $\boldsymbol{z}$ to belong to $\mathcal{D}$, but allow our algorithm to create new, *virtual* faces. The corresponding objective function is:

$$\min_{\boldsymbol{\beta}, \boldsymbol{z}} \Omega = \tfrac{1}{n} \textstyle\sum_{k=1}^n u_k \left(h(\boldsymbol{x}_k) - g(\boldsymbol{x}_k)\right)^2 + \lambda \boldsymbol{\beta}' \boldsymbol{\beta} \ , \quad (2)$$

where the scalars $u_1, \ldots, u_n$ balance the contribution of each point $\boldsymbol{x}_k$ to the empirical loss (if classes are unbalanced, *e.g.*, $|\mathcal{G}_c| \ll |\mathcal{I}_c|$, face templates in $\mathcal{G}_c$ can be assigned higher weights than the ones in $\mathcal{I}_c$), the quadratic regularizer $\boldsymbol{\beta}'\boldsymbol{\beta}$ controls overfitting, and λ is a regularization parameter. Eq. (2) can be rewritten in matrix form:

$$\Omega(\boldsymbol{\beta}, \boldsymbol{z}) = \tfrac{1}{n}\left(\boldsymbol{h}'\mathbf{U}\boldsymbol{h} - 2\boldsymbol{h}'\mathbf{U}\boldsymbol{g} + \boldsymbol{g}'\mathbf{U}\boldsymbol{g}\right) + \lambda \boldsymbol{\beta}'\boldsymbol{\beta} \ , \quad (3)$$

where the vectors $\boldsymbol{g}, \boldsymbol{h} \in \mathbb{R}^n$ contain the values of g_c and h_c for the training points, and $\mathbf{U} \in \mathbb{R}^{n \times n}$ is a diagonal matrix with $\mathrm{diag}(\mathbf{U}) = (u_1, \ldots, u_n)$. Note also that $\boldsymbol{h} = \mathbf{S}_{\boldsymbol{xz}}\boldsymbol{\beta}$, where $\mathbf{S}_{\boldsymbol{xz}} \in \mathbb{R}^{n \times m}$ is the similarity matrix computed between $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ and the set of virtual faces $\boldsymbol{z}$. For each client, Problem (3) can be solved by iteratively modifying $\boldsymbol{\beta}$ and $\boldsymbol{z}$, as detailed below (see also Algorithm 1 and Fig. 1).

$\boldsymbol{\beta}$-step. The optimal coefficients $\boldsymbol{\beta}$ are computed assuming that the set of virtual faces $\boldsymbol{z}$ remains unchanged. This yields a simple ridge regression problem, which can be solved by deriving Eq. (3) with respect to $\boldsymbol{\beta}$, assuming $\boldsymbol{z}$ constant, and then setting the gradient to zero. By denoting with $\mathbb{I} \in \mathbb{R}^{m \times m}$ the identity matrix, one yields:

$$\boldsymbol{\beta} = \underbrace{\left(\tfrac{1}{n}\mathbf{S}'_{\boldsymbol{xz}}\mathbf{U}\mathbf{S}_{\boldsymbol{xz}} + \lambda\mathbb{I}\right)^{-1}}_{\mathbf{M}^{-1}} \underbrace{\left(\tfrac{1}{n}\mathbf{S}'_{\boldsymbol{xz}}\mathbf{U}\right)}_{\mathbf{N}} \boldsymbol{g} \ . \quad (4)$$

$\boldsymbol{z}$-step. To update $\boldsymbol{z}$, we iteratively minimize the objective through gradient descent, as no closed-form solution is available. Deriving with respect to a given $\boldsymbol{z}_j$, one yields:

$$\frac{\partial \Omega}{\partial \boldsymbol{z}_j} = \tfrac{1}{n}(\boldsymbol{h} - \boldsymbol{g})' \, \mathbf{U}\left(\beta_j \frac{\partial \mathbf{S}_{\boldsymbol{xz}_j}}{\partial \boldsymbol{z}_j} + \mathbf{S}_{\boldsymbol{xz}} \frac{\partial \boldsymbol{\beta}}{\partial \boldsymbol{z}_j}\right) + \lambda \boldsymbol{\beta}' \frac{\partial \boldsymbol{\beta}}{\partial \boldsymbol{z}_j} \ , \quad (5)$$

where $\mathbf{S}_{\boldsymbol{xz}_j}$ is the j-th column of $\mathbf{S}_{\boldsymbol{xz}}$, using the numerator-layout convention for matrix derivatives. The term $\frac{\partial \boldsymbol{\beta}}{\partial \boldsymbol{z}_j}$ can be obtained by deriving Eq. (4) with respect to $\boldsymbol{z}_j$, as

$$\frac{\partial \boldsymbol{\beta}}{\partial \boldsymbol{z}_j} = \mathbf{M}_j^{-1}\left[\left(\frac{\partial \mathbf{N}_j}{\partial \boldsymbol{z}_j}' - \frac{\partial \mathbf{M}_j}{\partial \boldsymbol{z}_j}' \mathbf{M}^{-1}\mathbf{N}\right)\boldsymbol{g}\right]' , \quad (6)$$

where $\mathbf{M}_j$ and $\mathbf{N}_j$ denote the j-th column of $\mathbf{M}$ and $\mathbf{N}$.

Computing the derivative of the j-th column of $\mathbf{S}_{\boldsymbol{x},\boldsymbol{z}_j}$, *i.e.*, of $s(\boldsymbol{x}_1, \boldsymbol{z}_j), \ldots, s(\boldsymbol{x}_n, \boldsymbol{z}_j)$, with respect to the corresponding $\boldsymbol{z}_j$, depends on the given matching algorithm s.

Gradient of $s(\boldsymbol{x}_i, \boldsymbol{z}_j)$. If s has an analytical representation, like in the case of kernels, the derivative can be easily computed; *e.g.*, for the RBF kernel, $s(\boldsymbol{x}_i, \boldsymbol{z}_j) = \exp(-\gamma||\boldsymbol{x}_i -$

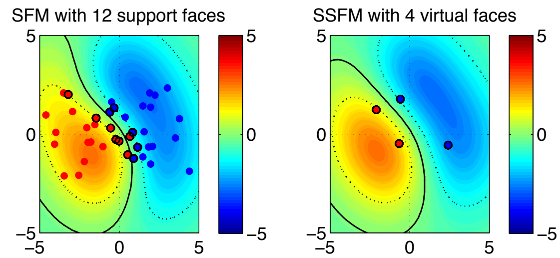

Figure 1. A two-dimensional example showing the values of $g(\boldsymbol{x})$ (in colors) for an SFM (*left*) trained on 40 samples (red and blue points in the left plot), and of $h(\boldsymbol{x})$ for our reduced SSFM (*right*). Our algorithm almost exactly replicates the SFM's decision boundary $g(\boldsymbol{x}) = 0$ (solid black line) using only 4 SFs (highlighted with black circles) instead of 12. The margin bounds corresponding to $g(\boldsymbol{x}) = \pm 1$ are also shown as dotted black lines.

[3] For simplicity we do not consider an explicit bias term, as g_c can be normalized to have zero mean, and θ_c can be adjusted accordingly. However, a (regularized) bias term b can be also considered without any significant modification to our subsequent derivations.

$z_j||^2$), and $\frac{\partial s(\boldsymbol{x}_i,\boldsymbol{z}_j)}{\partial \boldsymbol{z}_j} = 2\gamma \exp(-\gamma||\boldsymbol{x}_i - \boldsymbol{z}_j||^2)(\boldsymbol{x}_i - \boldsymbol{z}_j)$. Otherwise, the gradient can be only approximated numerically, by querying $s(\cdot, \boldsymbol{z}_j)$ in a neighborhood of $\boldsymbol{z}_j$. This is computationally costly, especially if $\boldsymbol{z}_j$ is high dimensional. Since the matching score reasonably increases while shifting $\boldsymbol{z}_j$ towards $\boldsymbol{x}_i$, even if this shift is operated linearly in the image space (*i.e.*, considering the pixel values as features), we approximate the gradient heuristically as

$$\frac{\partial s(\boldsymbol{x}_i,\boldsymbol{z}_j)}{\partial \boldsymbol{z}_j} = s(\boldsymbol{x}_i, \boldsymbol{z}_j)\,(\boldsymbol{x}_i - \boldsymbol{z}_j) \quad . \tag{7}$$

4. Experiments

Our experiments aim to show that our method based on sparse support faces can outperform the existing, heuristic template-based verification methods, both in terms of recognition accuracy and number of template matchings required for verification. We consider the well-known benchmark *AT&T*[4] and *BioID*[5] face datasets, respectively consisting of 40 clients with 10 face images each, and of 1,521 face images belonging to 23 different clients. We consider a problem setting where half of the clients are enrolled into the system, while the remaining identities are used only as impostors during verification. The training set consists of 5 randomly-selected face images per enrolled client (impostors are simulated using face images of enrolled clients that do not belong to the claimed identity). During verification, we simulate impostor attempts using the non-enrolled identities, and genuine claims using all the remaining face images of the claimed identity not used in the training set. This ensures that the set of impostors is different between training and test (verification) sets. Results are averaged over 5 repetitions with different training-test pairs and client splits.

Matching algorithms. We consider two algorithms:
1) Eigenface-based RBF Kernel [24, 14]. We first map the pixel values of face images onto a reduced vector space using principal component analysis (PCA). We select their number d, *i.e.*, the dimensionality of the space, as the number of eigenvalues that preserves 95% of the variance of the data. We then use the RBF kernel $s(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\gamma\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2)$ to compute similarities in that vector space. The parameter γ is set as $1/d$, to scale dynamically with the eigenspace dimension. On average, $d = 56$ for the AT&T data, and $d = 41$ for the BioID data.
2) Elastic Bunch Graph Matching (EBGM) [25]. For this algorithm, $s(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is not analytically given, and it is not even a proper (*i.e.*, positive semidefinite) kernel function.

Verification methods. We compare our method (SSFM), with the state-of-the-art template-based methods for face verification described in Sect. 2, denoted with mean, max, aggarwal-max, and t-norm, and with two techniques for reducing the number of support vectors proposed in [21], denoted with SFM-sel and SFM-red (see Sect. 5). We also consider the standard, *unpruned* SFM (*i.e.*, without applying our reduction method) for comparison.

We implemented SVM-based methods using LibSVM [6], and chose the regularization parameter $C \in \{10^{-1}, 10^0, \ldots, 10^3\}$ by maximizing recognition accuracy through a 3-fold cross-validation. Since the classes are highly unbalanced, we used a different C value for each class, multiplying it by the prior probability of the opposite one, estimated from training data. A proper gradient step size η and regularization factor λ have to be selected for our SSFM (see Algorithm 1), to attain fast convergence and to ensure a stable solution (*i.e.*, that the matrix $\mathbf{M}$ in Eq. 4 is well conditioned). Based on preliminary experiments, we set $\eta = 0.5$ for both datasets, $\lambda = 10^{-6}$ for the Eigenface-based RBF Kernel, and $\lambda = 10^{-3}$ and $\lambda = 10^{-5}$ for the EBGM on the *BioID* and on the *AT&T* data, respectively. The gradients of $s(\boldsymbol{x}_i, \boldsymbol{z}_j)$ required by the $\boldsymbol{z}$-step of our algorithm are analytically computable for the RBF Kernel, while for the EBGM we exploit the approximate gradient of Eq. 7, as s is not given analytically (see Sect. 3).

Results. Results are reported in Fig. 2. For each method, we obtain the corresponding curve by varying the client-specific threshold θ_c for each client, thus obtaining the fraction of incorrectly-rejected genuine claims (false rejection rate, FRR) as a function of the fraction of incorrectly-accepted impostors (false acceptance rate, FAR). We then average the corresponding curves over all clients and repetitions. We also report the average number of matchings required by each method for verification. For all methods except SFM, this number is fixed a priori (budgeted): for mean and max, we exploit all the available genuine templates in the training set (5 matchings); for the cohort-based methods aggarwal-max and t-norm, we additionally consider 5 impostor templates (10 matchings); and for SFM-sel, SFM-red, and SSFM we respectively set 10, 2 and 2 matchings, when the RBF Kernel is used as the matching algorithm, and 5 matchings for all of them when the EBGM is used.

First, note that the cohort-based methods aggarwal-max and t-norm may significantly worsen recognition accuracy with respect to mean and max, that only exploit genuine templates, although their budget is higher (see, *e.g.*, results for the EBGM in Fig. 2). Second, the unpruned SFM outperforms all the heuristic methods mean, max, aggarwal-max and t-norm for all datasets and matching algorithms, although requiring a much higher number of matchings. This limitation is overcome by our SSFM, which equals the performance of SFM by using only 2 and 5 matchings (instead of more than 20 and 15), respectively, for the RBF kernel and EBGM matching algorithms. Conversely, the SVM reduction approaches SFM-sel and SFM-red would require a

[4] http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html
[5] https://www.bioid.com/About/BioID-Face-Database

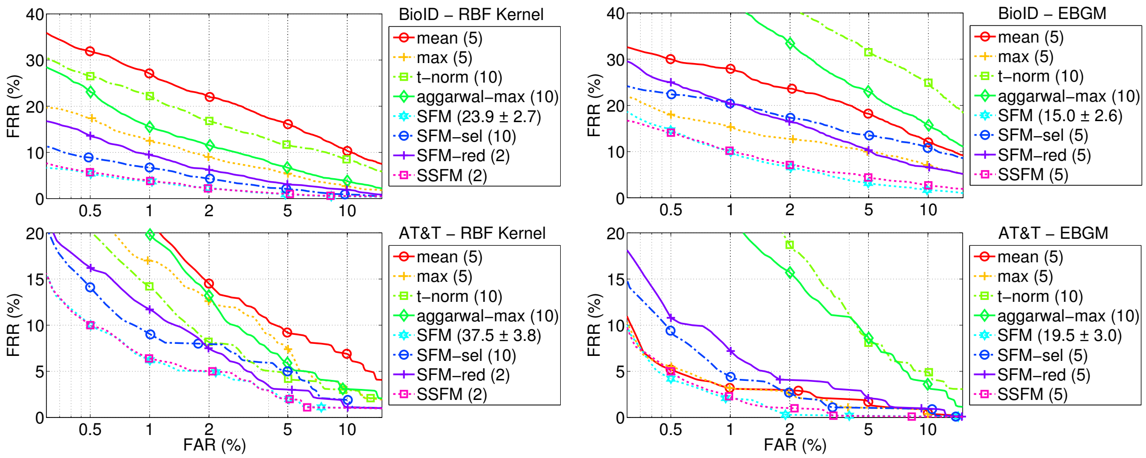

Figure 2. Results in terms of averaged FRR vs FAR values for the *BioID* (*top row*) and *AT&T* (*bottom row*) face datasets, and for the *Eigenface-based RBF Kernel* (*left column*) and *EBGM* (*right column*) matching algorithms. For each method, we also report the average number of matchings required for verification and the standard deviation over all repetitions (in parentheses).

Figure 3. Support faces found by our SSFM method for a client in the *BioID* dataset, using the EBGM as matching algorithm, along with the corresponding $\boldsymbol{\beta}$ values. Positive and negative coefficients respectively denote genuine and impostor templates.

higher number of matchings (*i.e.*, templates) to reach the performance of SSFM. Our results not only show that using a principled approach for combining additional information coming from non-match scores can significantly improve recognition accuracy, but also that this can be done very efficiently using an *extremely* sparse set of support faces, without even knowing the matching algorithm analytically.

Interpretability. We finally report a representative example of the virtual faces created by our SSFM, to show that its decisions are also interpretable. One may appreciate from Fig. 3 that the genuine SFs (corresponding to positive $\boldsymbol{\beta}$ values) are obtained by fusing genuine templates, preserving the aspect of the given client. Impostor SFs (negative $\boldsymbol{\beta}$ values) are instead obtained as a combination of faces of different identities, trying to compactly represent meaningful information from the impostor (cohort) faces. Although they may not represent any real user, they still resemble (valid) face images. This makes SSFM interpretable in the sense that it classifies an image as genuine if it is *sufficiently similar* to the genuine SFs and *different* from impostor ones.

5. Related Work

Our approach builds on [5], where SVMs were trained for each client using the matching algorithm as the kernel function. This showed potential improvements to the recognition accuracy, but the use of all the resulting SVs significantly limits its practical impact. Here we have extended that approach through a novel algorithm that can drastically reduce the number of required matchings by creating a small set of *virtual* SFs, without affecting recognition accuracy. Experiments have shown that existing methods for SV reduction (SFM-sel and SFM-red) do not achieve comparable reduction rates. They make the SVM solution sparser by minimizing the ℓ_2 distance between the hyperplane normal of the given SVM and that of the reduced SVM in kernel space, with respect to the choice of the dual coefficients $\boldsymbol{\alpha}$ and the reduced SVs [21]. While $\boldsymbol{\alpha}$ can be analytically found, as in our method, the choice of the reduced SVs is different: SFM-sel removes one SV at a time from the initial set using a Kernel PCA-based selection; and SFM-red creates a new SV at each iteration by minimizing the given objective. Both approaches are greedy, as they iteratively construct the reduced SVs by respectively removing and adding one SV at a time, up to the given budget. The reason of their unsatisfactory performance in our experiments is twofold: (i) they require the matching algorithm to be a proper kernel to *uniquely* determine the coefficients $\boldsymbol{\alpha}$;[6] and (ii) they neither modify the SVs that are already part of the reduced expansion, nor reconsider the discarded ones. Our approach overcomes such limitations by optimizing a different objective (suited to indefinite kernels too) and by iteratively modifying every SF during the optimization.

6. Conclusions and Future Work

We have proposed a new verification approach based on the concept of *sparse support faces* that, relying on the well-principled SVM learning algorithm, and on a novel reduction algorithm, can jointly learn an optimal combination of matching scores along with a proper subset of support faces, even when the matching algorithm is not a proper kernel and the corresponding function is not analytically given. This improves both recognition accuracy and efficiency, as only

[6] In fact, the notion of a hyperplane in kernel space exploited in the given objective is only consistent if a proper kernel is used.

an *extremely* low number of matchings is required for verification. Further, exploiting few support faces and a simple linear combination scheme makes the resulting decisions also easily interpretable for both designers and end users of the face verification system. Notably, our reduction algorithm can be also exploited as a *general* template reduction technique. In fact, the function g_c (Eq. 2) may be *any* discriminant function, and not necessarily an SVM. This opens interesting research directions, especially in biometric settings where template matching is computationally expensive, and template reduction is mainly driven by heuristics. It would also be interesting to apply our method to other biometrics, and, in particular, to fingerprint recognition. Finally, to foster research in this area, we are releasing the implementation of our approach as an open source library.[7]

Acknowledgments. This work has been partly supported by the projects CRP-18293 and CRP-59872 funded by Regione Autonoma della Sardegna, L.R. 7/2007, Bando 2009.

References

[1] A. F. Abate, M. Nappi, D. Riccio, and G. Sabatino. 2D and 3D face recognition: A survey. *Patt. Rec. Lett.*, 28(14):1885–1906, 2007. 1

[2] G. Aggarwal, N. Ratha, and R. Bolle. Biometric verification: Looking beyond raw similarity scores. In *CVPR-W*, 2006. 1, 2

[3] P. N. Belhumeur, J. a. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE TPAMI*, 19(7):711–720, 1997. 1

[4] J. Beveridge, D. Bolme, B. Draper, and M. Teixeira. The CSU face identification evaluation system. *Mach. Vision and Applications*, 16(2):128–138, 2005. 1

[5] B. Biggio, G. Fumera, and F. Roli. Learning sparse kernel machines with biometric similarity functions for identity recognition. In *IEEE 5th Int'l Conf. Biom.: Theory, App. and Sys.*, pp. 325–330, 2012. 2, 5

[6] C.-C. Chang and C.-J. Lin. LibSVM: a library for support vector machines, 2001. 4

[7] Y. Chen, E. K. Garcia, M. R. Gupta, A. Rahimi, and L. Cazzanti. Similarity-based classification: Concepts and algorithms. *JMLR*, 10:747–776, 2009. 2

[8] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995. 2

[9] G. Guo, S. Li, and K. L. Chan. Face recognition by support vector machines. In *4th IEEE Int'l Conf. Automatic Face and Gesture Rec.*, pp. 196–201, 2000. 1

[10] G. Guo, S. Z. Li, and K. L. Chan. Support vector machines for face recognition. *Image Vision Comput.*, 19(9-10):631–638, 2001. 1

[11] B. Haasdonk. Feature space interpretation of SVMs with indefinite kernels. *IEEE TPAMI*, 27(4):482 –492, 2005. 2

[12] X. He, S. Yan, Y. Hu, P. Niyogi, and H.-J. Zhang. Face recognition using Laplacianfaces. *IEEE TPAMI*, 27(3):328–340, 2005. 1

[13] A. Jain, K. Nandakumar, and A. Ross. Score normalization in multimodal biometric systems. *Patt. Rec.*, 38(12):2270 – 2285, 2005. 2

[14] K. Jonsson, J. Kittler, Y. Li, and J. Matas. Support vector machines for face authentication. *Image Vision Comput.*, 20(5-6):369–375, 2002. 1, 4

[15] A. M. Martinez. Recognizing imprecisely localized, partially occluded, and expression variant faces from a single sample per class. *IEEE TPAMI*, 24(6):748–763, 2002. 1

[16] A. Merati, N. Poh, and J. Kittler. Extracting discriminative information from cohort models. In *IEEE 4th Int'l Conf. Biom.: Theory, App. and Sys.*, 2010. 1, 2

[17] A. Merati, N. Poh, and J. Kittler. User-specific cohort selection and score normalization for biometric systems. *IEEE TIFS*, 7(4):1270–1277, 2012. 1, 2

[18] K. Nandakumar, Y. Chen, S. C. Dass, and A. Jain. Likelihood ratio-based biometric score fusion. *IEEE TPAMI*, 30:342–347, 2008. 2

[19] B.-G. Park, K.-M. Lee, and S.-U. Lee. Face recognition using face-arg matching. *IEEE TPAMI*, 27(12):1982–1988, 2005. 1

[20] E. Pekalska, P. Paclik, and R. P. Duin. A generalized kernel approach to dissimilarity based classification. *JMLR*, 2:175–211, 2001. 2

[21] 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 TNN*, 10(5):1000–1017, 1999. 3, 4, 5

[22] X. Tan, S. Chen, Z.-H. Zhou, and F. Zhang. Face recognition from a single image per person: A survey. *Patt. Rec.*, 39(9):1725–1745, 2006. 1

[23] S. Tulyakov, Z. Zhang, and V. Govindaraju. Comparison of combination methods utilizing t-normalization and second best score model. In *CVPR-W*, 2008. 1, 2

[24] M. Turk and A. Pentland. Eigenfaces for recognition. *J. Cognitive Neuroscience*, 3(1):71–86, 1991. 1, 4

[25] L. Wiskott, J.-M. Fellous, N. Kuiger, and C. von der Malsburg. Face recognition by elastic bunch graph matching. *IEEE TPAMI*, 19(7):775–779, 1997. 4

[26] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE TPAMI*, 31(2):210–227, 2009. 1

[7] http://pralab.diee.unica.it/en/SparseBiometricRecognition/