Feature Fusion for Fingerprint Liveness Detection: a Comparative Study

(Article begins on next page)

22 September 2024

Feature Fusion for Fingerprint Liveness Detection: a Comparative Study

AMIRHOSEIN TOOSI[1], ANDREA BOTTINO[1], (Member, IEEE),
SANDRO CUMANI[1], (Member, IEEE), PABLO NEGRI[2], (Member, IEEE),
AND PIETRO LUCA SOTTILE[1]

[1]Department of Control and Computer Engineering, Politecnico di Torino, 10129 Turin, Italy
[2]National Scientific and Technical Research Council, University of Buenos Aires, Buenos Aires 1053, Argentina

Corresponding author: Andrea Bottino (andrea.bottino@polito.it)

ABSTRACT Spoofing attacks carried out using artificial replicas are a severe threat for fingerprint-based biometric systems and, thus, require the development of effective countermeasures. One possible protection method is to implement software modules that analyze fingerprint images to tell live from fake samples. Most of the static software-based approaches in the literature are based on various image features, each with its own strengths, weaknesses, and discriminative power. Such features can be seen as different and often complementary views of the object in analysis, and their fusion is likely to improve the classification accuracy. This paper aims at assessing the potential of these feature fusion approaches in the area of fingerprint liveness detection by analyzing different features and different methods for their aggregation. Experiments on publicly available benchmarks show the effectiveness of feature fusion methods, which improve the accuracy of those based on individual features and are competitive with respect to the alternative methods, such as the ones based on convolutional neural networks.

INDEX TERMS Fingerprint liveness detection, biometric counterspoofing methods, local image features, feature fusion.

I. INTRODUCTION

Fingerprints are frequently used as an authentication system in a plethora of applications ranging from security to surveillance and forensic analysis [1]. Today fingerprint recognition systems are cost–effective solutions that guarantee high recognition accuracy on large datasets of millions of images. Thanks to these characteristics, they are starting to be deployed in novel scenarios, such as granting access to schools, health or leisure facilities, identifying patients in hospitals, developing pay–with–fingerprint systems and unlocking consumer devices like notebooks or smartphones.

However, these systems are vulnerable to more or less sophisticated forms of malicious attacks. *Direct* attacks operate on the acquisition sensor by using fake biometric traits, while *indirect* attacks target some of the inner modules of the system. Clearly, direct attacks are easier to implement for an intruder, since they only require access to the biometric sensor without any knowledge of the system operation.

The most common direct spoofing method consists in creating a mold of a latent or real fingerprint and then filling it with materials such as gelatine, silicone or Play–Doh [2]–[4].

These attacks can succeed in more than 70% of the cases [5], highlighting the need for specific protection methods that are capable of identifying live samples and rejecting fake ones.

Fingerprint *liveness* detection systems can be either hardware or software based. The former integrate additional sensors to extract the typical characteristics of a vital fingerprint, such as temperature [6], skin electrical conductivity [7], pulse oxymetry [8] and skin resistance [6]. While these methods are effective in practice, several authors have mentioned that, even if extra hardware is added to the system, it can be still spoofed in various ways [8]–[10]. A possible alternative is to use software based methods, which rely on a software module to detect the liveness of a fingerprint image. These solutions are less invasive and more flexible, since they can potentially tackle novel types of attack by updating the software.

Software based approaches can be further divided into dynamic and static ones. Dynamic methods analyze certain phenomena like skin deformation [11] and perspiration [12] on a sequence of images. However, the acquisition of multiple images increases the computational time. Static methods,

on the contrary, focus their analysis on a single fingerprint image, which makes them suitable for general use.

This work focuses on software static methods. Several approaches have been proposed in the literature, most of which are based on the analysis of individual image features. Holistic methods process the image as a whole to derive some global characteristics, such as the texture coarseness [13] and several first and second order statistics like mean, energy, entropy, variance, skewness [14] or Gray–Level Co–Occurrence Matrices [15]. However, as shown in [16], the discriminative power of holistic features is rather low. Better performances are provided by local methods, which rely on mathematical descriptors capable of capturing different texture features on small regions surrounding an image point.

The common characteristic of all these features is that their engineering was based on expert knowledge of the problem under analysis. Thus, they can also be seen as different observations of the same data from different viewpoints, each of which focuses on specific characteristics of the samples. Given their differences, these features often highlight complementary properties of the analyzed objects. For this reason, developing approaches that combine multiple features, capable of mutually exploiting their strengths and, at the same time, softening their weaknesses, could be a valuable solution to improve both the accuracy and the generalization properties of the classification system.

Feature fusion approaches, also referred to as multi–view learning, have been applied in different computer vision tasks, such as object classification [17] and human activity recognition [18], face [19] and facial expression recognition [20], content–based image retrieval [21] and hyperspectral image classification [22]. These works show that multi–view learning is effective and promising in practice. In contrast, this approach has been relatively overlooked in the context of fingerprint liveness detection.

Based on this observation, the rationale of the current work is to analyze the effectiveness of feature fusion approaches as anti–spoofing methods and to compare them with the state of the art. This raises several research questions, such as which features can be combined and how. Since an exhaustive assessment of all the available features and feature fusion methods would have been clearly unfeasible, our approach has been to (i) select a subset of promising features, based on the literature, and (ii) compare methods capable of dealing, from a number of different perspectives, with the various issues involved (e.g. when to fuse, how to cope with the curse of dimensionality, how to provide a shared representation of the different features, and so on).

Our work extends the analysis presented in previous papers tackling the liveness detection problem with multi–view learning approaches, and our experimental results show that feature fusion approaches are (i) effective and able to generalize well, (ii) capable of improving the accuracy of single–view methods, and (iii) competitive with deep learning approaches. Another contribution of our work is *Spidernet*,

a novel two-stage deep neural architecture capable of effectively combining different general image descriptors.

A preliminary version of this work appeared in [23]. The following extensions are introduced: (i) we analyze a larger number of different features; (ii) we compare four different multi–view learning methods (two in [23]); and (iii) we assess our approach on the eleven public benchmarks made publicly available for the LivDet 2009 [2], LivDet 2011 [3] and LivDet 2013 [4] competitions (four in [23]) and thoroughly compare it with other methods in the literature.

In the remainder of the paper, after an analysis of the state of the art, we introduce the features and the multi–view learning approaches selected for our research. Finally, we present and discuss the results of our experiments before drawing the conclusions.

II. RELATED WORKS

As we stated in the introduction, our work is focused on software based static methods, which leverage on the analysis of individual images to tell a live from a fake sample. There is a large literature related to this problem and the recent availability of public benchmarks (such as [2]–[4]) allows comparing the different approaches. Results on these datasets highlight the complexity of the problem, as witnessed by the fact that, usually, the same technique achieves large accuracy differences among different sensors.

As a general remark, static approaches differ mainly in the type of features extracted and if and how the integration of different features is considered. As another option, recent works proposed the use of Deep Convolutional Networks to automatically learn the optimal features for the liveness detection problem. In the following, we separately analyze these three general approaches.

A. APPROACHES BASED ON INDIVIDUAL FEATURES

Image features can be roughly divided into *holistic features*, computed analyzing the image as a whole, and *local features*, extracted in small image patches centered on each image pixel and eventually summarized into a unique feature vector, usually by means of a histogram.

Some of the holistic approaches are based on the observation that, when captured by the same sensor, fake samples produce image with lower quality than live samples. Thus, trying to capture these quality differences could result in highly discriminative features. In [13] the texture coarseness is used to highlight the blemishes present in fakes. In [24] it is observed that live images exhibit higher frequencies than fake ones and, consequently, that the modulus of the Fourier Transform can be a valid liveness detector. A more detailed characterization of the quality differences is attempted in [25], where 25 different quality measures are extracted.

Other holistic approaches are based on textural features extracted from the images. In [26] curvelet energy signatures and corresponding co–occurrences are used to represent fingerprints. Relevant features are selected by means of Sequential Forward Floating Selection. Then, a majority voting

approach combines three different classifiers (Adaboost, Support Vector Machine (SVM), k–NN) to produce the final decision. A different method [14] gathers several first and second order texture statistics and intensity based features into a unique characteristic vector. Feature selection and different classifiers are then combined to get the final results. Gray–Level Co–Occurrence Matrix (GLCM) and wavelet energy signature are analyzed in [15] by first applying feature selection and then fusing the decision of different classifiers.

Despite the efforts, comparisons on public benchmarks show that the discriminative power of holistic features is rather low, and better performances can be obtained by local image descriptors [16], [27].

Local descriptors can be roughly divided into *micro–textural descriptors*, which represent an input image by building statistics on the local micro–pattern variations, and *rich local descriptors*, which provide a much stronger characterization of local patches [27]. Most of these descriptors have been initially designed for coping with different problems in computer vision and quickly found effective applications for fingerprint liveness detection. Examples are various micro–textural descriptors, like Local Binary Pattern (LBP), Weber Local Descriptor (WLD), Binary Statistical Image Features (BSIF) and Local Phase Quantization (LPQ), and the whole class of rich local descriptors, such as Scale-Invariant Feature Transform (SIFT), DAISY and the Scale-Invariant Descriptor (SID).

Recently, some novel micro–textural descriptors expressly designed for fingerprint liveness detection have been proposed. The Histogram of Invariant gradients (HIG) [28] adds to the rotation and translation invariance of Histograms of Oriented Gradient the invariance to curvature and deformations, which characterize fingerprint images. Local Contrast Phase Descriptor (LCPD) [16] is a joint distribution of WLD and LPQ. Convolutional Comparison Pattern (CCP) [29] is a rotation invariant descriptor based on the preliminary segmentation of the fingerprint and on its orientation into a reference direction. DCT based per–pixel binary codes are then computed at multiple scales and summarized into an histogram.

B. FEATURE FUSION APPROACHES

Since the various image features convey different and usually complementary information on the analyzed data, an interesting perspective could be to improve accuracies by integrating multiple features. However, few results based on these feature fusion approaches have been reported.

In [30] various combinations of image features (LPQ, LBP, curvelet GLCM and valley wavelets) were used to train a linear SVM. Good results were obtained aggregating LPQ and LBP, but the accuracy was saturating adding more features, thus highlighting the importance of carefully selecting features according to both their performance and complementarity.

Another approach [31] combines various image filters, statistic measures and quality indexes. After feature selection,

Multilayer Perceptron with one hidden layer and SVM were compared, showing higher accuracies for SVM.

Finally, the study in [32] compared the integration of LBP+LPQ and LPQ+WLD. Individual features were chained and fed to a linear kernel SVM. The results clearly highlight that (i) any combination of multiple features provide better accuracy than that of individual features, and (ii) the integration of WLD and LPQ is the optimal one.

C. OTHER APPROACHES

Recently, very good results have been obtained using Convolutional Neural Networks (CNN). In [33] CNN–based feature extraction using random weights was compared to LBPs. In both cases, PCA–compressed features were classified using SVMs, showing the higher performances of CNN features.

In another interesting approach [34], authors describe *spoofnet*, a deep CNN architecture able to greatly improve the results of other state–of–the–art approaches. Spoofnet has been created by optimizing both the architecture hyperparameters and the filter weights via back–propagation algorithm. It is worth noting that the proposed approach relies on data augmentation and dataset specific image cropping.

III. IMAGE FEATURES

Given the demonstrated relevance of micro–textural and rich local features to the fingerprint liveness detection problem ([27]), we decided to focus our research on them. Speaking in general, these features are characterized by different invariant properties (e.g. to illumination, scale, rotation, translation, blur and so on). When individual features are considered, it is clear that the more invariance they express the better it is. However, when multiple features are combined, this issue is less pressing since we expect that the mutual contribution of different features helps overcoming their individual limitations.

Thus, our selection process was mainly based on the discriminative power of the features, which we inferred from both the results available in the literature and those of our preliminary tests. Another selection criterion was the possibility to apply a descriptor to all the experimental benchmarks. For instance, we ruled out CCP [29] since it requires fingerprint segmentation and, thus, it cannot be applied to one of our benchmarks (Swipe) due to the lack of robust segmentation algorithms capable of coping with the peculiar characteristics of its images (details are given in Section V-A).

In the following, we shortly describe the selected features for each of the two main categories.

A. MICRO-TEXTURAL LOCAL DESCRIPTORS

These descriptors capture the statistical behavior of small image patches, generally highlighting, as a pre–processing step, the high frequency components of the signal. Such descriptors are usually represented as binary codes and then encoded into an histogram that describes the whole image. Common parameters for these features are, therefore,

the size of the local patch and the number of code bits used.

Several results in the literature ([35]–[38]) show that micro–textural features computed in a multi–scale fashion (i.e. using different size of the local patches) usually provide better accuracies than the same features computed at a single resolution. Hence, when practicable, we also considered this option (see Section V-C.1).

1) CO–OCCURRENCE OF ADJACENT LBPS (COALBP)

In the context of local textural features, one of the most famous descriptor is the LBP. Briefly, an LBP is a binary pattern representing the intensity relations between a pixel and its neighbors. One of the main drawbacks of the original LBP formulation is that it lacks structural information. To overcome this issue, in [36] the authors introduced a new variant where the co–occurrence among multiple adjacent LBPs is measured. A single–scale CoALBP histogram has size 1,024.

2) ROTATION–INVARIANT CO–OCCURRENCE OF ADJACENT LBPS (RICLBP)

The CoALBP features can vary significantly depending on the orientation of the target object. In order to cope with this problem, a recent extension [37] introduces the concept of rotation equivalence class of CoALBPs. This is achieved by attaching a rotation invariant label to each LBP pair, so that all CoALBPs corresponding to different rotations of the same LBPs have the same value. Thus, the size of the final histogram is reduced to 136.

3) WEBER LOCAL DESCRIPTOR (WLD)

WLD [35] is built on two components computed on each pixel: orientation and differential excitation. The orientation is simply the angle of the local gradient, while the differential excitation is the ratio between the sum of neighboring pixel intensity and the intensity of the pixel itself. Typically the orientation is quantized into 8 directions and the differential excitation into 120 levels, both encoded into a histogram of 960 elements.

4) LOCAL PHASE QUANTIZATION (LPQ)

This operator [39] is invariant to contrast, illumination and centrally symmetric blur, such as the one caused by linear motion and out of focus. LPQ exploits the blur invariance property of the phase spectrum and encodes phase information in a way similar to the coding mechanism of LBPs. LPQ codes are obtained by locally computing, for each image pixel, the phase of the 2D Short Term Fourier Transform and quantizing only some selected frequency components. LPQ codes are represented with 8 bits and the final descriptor has size 256.

Using this descriptor requires optimizing the trade–off between the discrimination power of the LPQ descriptor and its blur–tolerance. Decreasing the local patch size helps capturing more detailed local information, but at the same time it reduces the descriptor tolerance to blur.

5) ROTATION–INVARIANT VERSION OF LPQ (RILPQ)

LPQs can be further improved by adding rotation invariance. As shown in [40], this can be done by first applying a blur insensitive filter that estimates the local texture orientation at each location and, then, orient accordingly the phase estimation step of LPQ.

6) LOCAL CONTRAST PHASE DESCRIPTOR (LCPD)

This descriptor has been expressly proposed in [16] to tackle the fingerprint liveness detection problem. The idea behind LCPD is, basically, to combine the best characteristics of WLD and LPQ. We recall that WLD is characterized by two components related to contrast and orientation. In LCPD, the contrast is first computed with a Laplacian of Gaussian operator, which helps better dealing with the intrinsic noise of fingerprint images, and then quantized on N levels. The orientation is computed with the RILPQ descriptor, which guarantees higher robustness to noise and image rotation compared to the gradient used in WLD. The final LCPD descriptor has size $N \times 256$, where we fixed $N = 8$ for all devices and datasets according to the suggestions in [27].

7) BINARY STATISTICAL IMAGE FEATURES (BSIF)

BSIF [41] are histograms of binary codes obtained by applying to local image patches a set of filters learned from natural images. Such filters are computed by maximizing the statistical independence of their output using Independent Component Analysis. The bits of the binary codes are obtained by simply thresholding the filter responses. The statistical independence of the filter outputs improves the representation capabilities of BSIF when compared with operators that produce dependent output. Furthermore, these filters are not built upon the training set of a specific benchmark and, thus, they do not require to fine–tune their parameters for each application. In order to allow the combination of different scales, we used 10 bits for the binary codes, i.e. an histogram of size 1,024 for each scale.

B. RICH LOCAL DESCRIPTORS

Compared with micro–textural features, these descriptors provide a much stronger characterization of the local image patches. To improve their distinctiveness, they are often coupled with a feature–specific keypoint detector that returns a local measure of the feature uniqueness, usually promoting high–contrast regions of the image, such as object corners.

For tasks like image registration, object tracking and recognition, robust matching algorithms can be applied to pair keypoint descriptors obtained from different images. However, when these descriptors are used for object classification, the common practice shows that better results are obtained using a dense approach, i.e. computing a local descriptor on every image pixel or on a regular grid. A compact representation of this dense set can be obtained using bag–of–words (BoW) models. In our work, as suggested in [27], we used for all features a basis of 600 words, which were

computed with vector quantization approaches from 30 random training images picked from both live and fake sets and different spoofing materials.

1) SCALE INVARIANT FEATURE TRANSFORM (SIFT)

SIFT [42] is invariant to uniform scaling and rotation, and partially invariant to affine distortion and illumination changes. We computed SIFT in both a sparse and dense way. The sparse version, referred in the following as keypoint–SIFT (KSIFT), applies the detector to identify the keypoints where descriptors can be computed. However, since computing robust keypoint correspondences between fingerprint images is virtually impossible, we transformed the extracted descriptors into a normalized histogram using again a BoW approach.

Dense–SIFT (DSIFT) were obtained computing a descriptor for each image pixel. We note that DSIFT does not guarantee scale–invariance, since the descriptor scale is not obtained with the SIFT keypoint detector but it is fixed beforehand.

2) DAISY

This is a descriptor specifically designed to be extracted in a pixel–wise dense way [43]. While providing distinctiveness and robustness properties similar to SIFT, it is much faster to compute. The name DAISY derives from the fact that the descriptor is computed on a neighborhood organized in concentric circles that resembles the flower shape.

3) SCALE–INVARIANT DESCRIPTOR (SID)

Rich local descriptors deals with scale changes by using a keypoint detector to provide a local estimate of their optimal scale. However, this approach often reduces the locations where a reliable estimate can be obtained (e.g. usually ruling out object edges). SID [44] aims at overcoming this issue with a two step approach. First, the image is log–polar sampled around a point of interest, extracting samples at varying scales proportional to the logarithmic distance from the point of interest. This process converts scaling and rotations into translations in log–polar coordinates. Then, the variations related to these translations are removed by computing and normalizing the Fourier Transform modulus of the transformed signal.

We computed SID in a dense way, extracting a descriptor per pixel.

IV. FEATURE FUSION APPROACHES TO FINGERPRINT LIVENESS DETECTION

In this section we describe the different feature fusion approaches we experimented with. Since an exhaustive assessment of all the feature fusion methods available in the literature would have been clearly unfeasible, we rather selected different approaches capable of tackling, under different perspectives, the inherent challenges of feature fusion, such as:

- when combining multiple features, which is the best strategy to follow between *early* fusion (i.e., fusion at feature level, Sections IV-B, IV-C, IV-E) and *late* fusion (i.e., fusion at decision level IV-D)?
- given that different features have different characteristics and belong to different representational spaces, how can we harmonize or normalize them (IV-B, IV-C, IV-E)?
- since combining multiple views increases the number of variables, which data reduction techniques, such as feature selection (IV-B, IV-D) or subspace transformations (IV-C, IV-E), are suitable to soften the curse of dimensionality problem?
- rather than engineering methods to aggregate features on the basis of some expert knowledge, can we exploit Deep Learning approaches to automatically learn such combinations (IV-E)?

In the discussion section we will try to provide answers to these questions on the basis of our experimental results. For the sake of clarity, we inform that in the following we will use interchangeably the terms features and views, feature fusion approaches and multi–view learning.

A. NOTATION

We introduce the notation that will be used throughout this section. We denote with $y = \{y^1, \ldots, y^K\}$ a test sample described under K views, where each view y^k is defined into its own representation space, a subset of $\mathbb{R}^{m_k}$, and each sample $y \in \mathbb{R}^m$, where $m = \sum_{k=1}^K m_k$.

We also denote our training set as $X = \{X^1, \ldots, X^K\}$. Here, $X^k = \{X_1^k, \ldots, X_J^k\}$ is the training set for view k, J is the number of classes and $X_j^k = \{x_{jki}\}$, $i = 1, \ldots, n_{jk}$, where n_{jk} is the number of train samples for the k–th view of the j–th class (thus, $X_j^k \in \mathbb{R}^{m_k \times n_{jk}}$).

B. FEATURE CHAINING

A simple but effective way of combining multiple representations of the same sample is to concatenate the characteristic vector of each representation. The concatenated samples are then classified by means of a linear SVM, an approach that has shown to provide good results for a wide set of different features [16], [27], [29], [30], [32], [34], [45]. The success of linear kernels can be motivated by the high dimensionality of the chained features. This characteristic guarantees a proper class separation without necessarily requiring their expansion into a higher dimensionality space, as that provided by non–linear kernels (the interested reader can refer to [46] for details). Linear SVMs also provide huge benefits in terms of computational and memory requirements, since (i) the separation hyperplane can be computed offline and (ii) scoring reduces to a simple dot–product in feature space. Finally, the presence of a regularizer imposing a penalty on the classification weights allows the model to implicitly select the most discriminative features, thus making explicit feature selection less relevant. Nevertheless, in our preliminary tests we also investigated the effectiveness of an additional feature selection step based on the Relief algorithm [47].
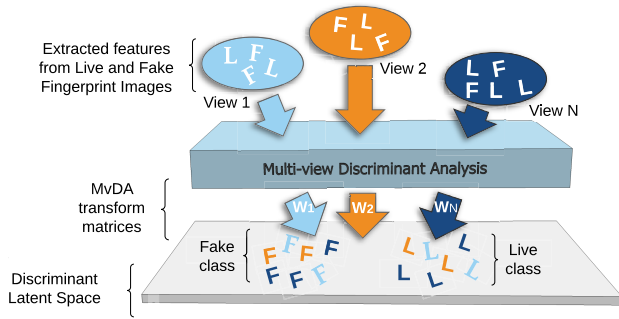
FIGURE 1. Overview of MvDA method. The different views (i.e. features) extracted from fingerprint images are projected into a discriminant common latent space by computing a proper linear transformation for each view. Here, samples from different views are represented with distinct colors and the letters denote the view class (F for fake and L for live).

C. MULTI–VIEW DISCRIMINANT ANALYSIS

Multi-view Discriminant Analysis (MvDA) has been proposed in [48]. It is a subspace learning approach that transforms the different views describing a sample into a common latent space L which is *discriminant* with respect to the classification variable. In other words, MvDA tries to compute a latent space where the between–class variations (both intra–view and inter–view) are maximized and the within–class variations (again, both intra–view and inter–view) are minimized. Thus, MvDA performs a sort of (supervised) optimal feature vector reduction and, at the same time, improves the class separability, thus allowing for simpler classification in the latent space (see Figure 1).

In brief, MvDA computes the K linear transformations $w_1, \ldots, w_K$ that project each of the K views of a sample into the latent space L. We recall that $X_j^k = \{x_{jki}\}$ is the set of training samples for class j and view k. Each sample x_{jki} is projected into L as $l_{jki} = w_k^T * x_{jki}$. Since the common space should maximize the between–class variation S_B^l and minimize the within–class variation S_W^l between all views, the required projection matrices w_k can be obtained by optimizing the following generalized Rayleigh quotient:

$$(w_1, \ldots, w_K) = \arg \max_{w_1,\ldots,w_K} \frac{Tr(S_B^l)}{Tr(S_W^l)} \qquad (1)$$

The two scatter matrices S_B^l and S_W^l are defined as:

$$S_W^l = \sum_{j=1}^{J} \sum_{k=1}^{K} \sum_{i=1}^{n_{jk}} (l_{jki} - \mu_j)(l_{jki} - \mu_j)^T \qquad (2)$$

where $\mu_j = \sum_{k=1}^{K} \sum_{i=1}^{n_{jk}} l_{jki}$ is the mean of all samples from the j–th class and k–th view in the common space, and

$$S_B^l = \sum_{j=1}^{J} n_j (\mu_j - \mu)(\mu_j - \mu)^T \qquad (3)$$

where $n_j = \sum_{k=1}^{K} n_{jk}$ is the total number of samples of the j–th class over all views, $\mu = \frac{1}{n} \sum_{j=1}^{J} \sum_{k=1}^{K} \sum_{i=1}^{n_{jk}} l_{jki}$

is the mean of all projected training samples and $n = \sum_{j=1}^{J} \sum_{k=1}^{K} n_{jk}$ is the number of all training samples.

After the projection matrices $w_1, \ldots, w_K$ have been obtained, the train and test samples are first transformed into the latent space, i.e. each sample is transformed into a set of points in L. These points are then concatenated to obtain the characteristic vectors of the samples, which are finally fed to a linear SVM for classification.

Details on the analytic solution of MvDA can be found in [48]. Summarizing, in the described method the within and between class scatter matrices in the common space L are expressed in terms of two matrices D and S in the feature space as follows:

$$S_W^l = W^T D W$$
$$S_B^l = W^T S W$$

where $W = [w_1^T, w_2^T, \ldots, w_K^T]^T$ and D and S are derived from, respectively, (2) and (3). Then the trace ratio problem in (1) is transformed into a more tractable ratio trace, which can be solved through generalized eigenvalue decomposition. The optimal dimension of the latent space L has been estimated by means of cross–validation on the training set.

D. MULTI-VIEW REAL ADABOOST

Real Adaboost is an improvement of the original Adaboost algorithm [49], which outperforms the standard formulation in several practical cases and allows an effective combination of different descriptors [50], [51]. The basic idea of Adaboost is to build a highly accurate classifier by combining several "weak" classifiers. The various Adaboost versions mainly differ on the design of the weak classifiers and whether and how confidence measures of their predictions are considered to improve the overall robustness [52].

The first step of the multi–view Real Adaboost consists in chaining the different features of each sample, i.e. $y = (y^1, \ldots, y^K)$ and $X = (X^1, \ldots, X^K)$. Each training sample x_i in X has an associated label c_i. Since we have a two class problem, we assume, without loss of generality, that $c_i \in \{-1, +1\}$. The decision rule on a test sample y is then implemented as:

$$H(y) = sign \left(\sum_{t=1}^{T} h_t(y) \right) \qquad (4)$$

where T is the total number of weak classifiers h_t composing the *strong classifier H*, and each of the h_t is a real valued function.

In brief, the method is iterative and at each iteration $t = 1, \ldots, T$ it computes the classification function h_t that better discriminates the two classes. This is done by first defining a set of classifiers and their confidence level. Then, the most confident function is selected and the algorithm is iterated. At each round, the misclassified samples are emphasized, so that the classifier built on the next iteration can try to compensate for errors in the previous steps.

In details, each sample $x_i = \left\{ v_i^g \right\}_{g=1,\ldots,m}$ is a real–valued vector composed by m feature variables v_i^g, where m is the sum of the size m_k of each view. Then, we define a distribution $\omega = \{\omega_i\}_{i=1,\ldots,n}$ that assigns a weighting value to each training sample i. The initial weights are $1/n$ for each sample.

For each feature variable g, we also extract the lists $\mathbf{v}_g^+ = \left\{ v_i^g | c_i = 1 \right\}$ and $\mathbf{v}_g^- = \left\{ v_i^g | c_i = -1 \right\}$ of the values that it assumes on, respectively, the positive and negative training samples. Such sets are clearly constant for all the iterations.

Finally, we start constructing iteratively our weak classifiers as follows. For each iteration t and each feature variable g:

1) compute the two conditional probabilities $P_g^+ = P_\omega(\mathbf{v}_g^+)$ and $P_g^- = P_\omega(\mathbf{v}_g^-)$ as the weighted histograms of $\mathbf{v}_g^+$ and $\mathbf{v}_g^-$ computed on a predefined number of bins (16 in our implementation); when applied to sample y, P_g^+ (P_g^-) returns the bin value of the g–th feature variable of y in the positive (negative) distribution;

2) define the following classification function for g:

$$h_g(y) = \frac{1}{2} log \left(\frac{P_g^+(y) + \epsilon}{P_g^-(y) + \epsilon} \right) \qquad (5)$$

where ϵ avoids division by zero, and can be equal to $1/n$; when applied to a test sample y the sign of h_g returns the label assigned to y;

3) compute the confidence of h_g from the Chi square distance between P_g^+ and P_g^- as:

$$Z_g = 1 - \chi^2(P_g^+, P_g^-) \qquad (6)$$

clearly, Z_g is lower when the two distributions are different (i.e., h_g is more discriminant) and higher when they are similar (i.e., h_g is less discriminant);

Once the pool of m classification functions has been created (one for each feature variable), we select the weak classifier h_t at step t as the one with the lowest Z measure. In other words, at each iteration t, the method greedily selects the best view and its best variable to define h_t. As a result, the final strong classifier will (in general) include classification functions from different feature spaces.

Then, given h_t, the sample weights are updated as follows:

$$\omega_i = \omega_i \cdot e^{-c_i \cdot h_t(x_i)} \qquad (7)$$

This update rule aims at increasing the weight of the training samples that are wrongly classified by h_t. Thus, these samples will have, on the next iteration, an higher influence on the probability distributions and Adaboost will focus on trying to find a proper discriminant function for them.

From this description, it is also clear that Real Adaboost performs an (implicit) feature selection, since only the most promising feature variables from the different views are included into the final classifier.

As for the implementation details, the only parameter of the algorithm is T, the number of weak classifiers. We verified experimentally that the algorithm has a similar behavior for different attribute groups, reaching a plateau after a certain
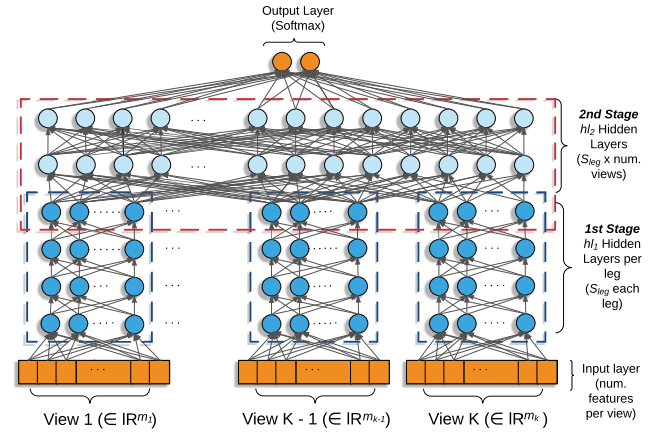

FIGURE 2. *Spidernet* architecture. In the first stage, each view is independently processed by a network "leg" and projected into a common latent space. In the second stage, the transformed views are first combined and then classified.

number of iterations. We also found that the starting value of this plateau is somewhat related to the discriminative power of the different views. This observation allowed us defining a heuristic rule to select T based on the residuals from Principal Component Analysis on the training set (as a note, usual T values range between 600 and 800).

E. SPIDERNET

As a contribution to this comparative study, we propose *Spidernet*, a novel two–stage Deep Neural Network (DNN) architecture that, starting from general image descriptors (fig. 2), is capable of simultaneously learning a suitable transformation of the different features into a common latent space (in the first stage) and carrying out a classification based on the feature fusion in that space (in the second stage).

The input of the network consists of the stacked characteristic vectors of the different views and the last layer consists of two softmax activated units whose outputs can be interpreted as class posterior probabilities. The first stage is composed by hl_1 hidden layers, which are not fully connected. Instead, the characteristic vector of each view is independently propagated through a network "leg", whose hidden layers' size is s_{leg}. This allows for a sensitive reduction of the number of weights with respect to a fully connected architecture and, consequently, of the overfitting issues caused by the small number of training samples. The last layers of each leg are then concatenated and used as inputs for a fully connected architecture with hl_2 hidden layers (of size s_{leg} times the number of views). Thus, intuitively, during training the network jointly learns the feature transformation (in the spider *legs*) and the aggregation and classification rules (in the spider *body*).

The final classifier is built upon two steps: learning the network weights (*weight optimization*) and estimating the optimal architecture hyperparameters (*architecture optimization*).

TABLE 1. Characteristics of the datasets used in the experiments.

Dataset	*LivDet2009*			*LivDet2011*				*LivDet2013*			
Scanner	Biom.	XMatch	Identix	Biom.	Digital	Italdata	Sagem	Biom.	XMatch	Italdata	Swipe
Res.(dpi)	569	500	686	500	500	500	500	569	500	500	96
Image size	312x372	480x640	720x720	312x372	355x391	640x480	352x384	312x372	800x750	480x640	1500x208
Live samples	1993	2000	1500	2000	2004	2000	2009	2000	2500	2000	2500
Fake samples	2000	2000	1500	2000	2000	2000	2037	2000	2000	2000	2000
Total subjects	50	254	160	200	82	92	200	45	64	45	70
Materials	1	3	3	5	5	5	5	5	5	5	5
Co-operative	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes

1) WEIGHT OPTIMIZATION

The activation function of all hidden units is sigmoidal. The network weights are initialized by training a Restricted Boltzmann Machine (RBM) by means of Contrastive Divergence (CD) [53]. Stochastic Gradient Descent (SGD) with a decreasing learning rate and momentum is then used to fine–tune the network parameters [54]. For each dataset, the batch size is one fortieth of the training set size. The objective function for fine–tuning is the negative cross–entropy between network outputs and training labels.

Both dropout [55] and L2 regularization were used to soften the overfitting issues. In our work, we used an effective dropout strategy, proposed in [56], with an initial dropout of 0.5 decreased at each epoch. The coefficient for L2 regularization was set to 0.001.

2) ARCHITECTURE OPTIMIZATION

We optimized the hl_1, hl_2 and s_{leg} hyperparameters in the following way. For each benchmark, we equally divided the training samples into a training and a validation set, taking care into putting all samples of an individual into the same set to enforce robustness to cross–individual variations. Then, we defined a variation interval for each parameter as $hl_1 \in [2, 4]$, $hl_2 \in [1, 3]$ and $s_{leg} \in [30, 200]$ with a step size of ten. Finally, for each combination of the hyperparameters, we computed a functional based on a weighted combination of loss on the train set and accuracy on the validation set. The architecture providing the lowest functional value was finally re–trained on the whole training set and used to classify the test set.

V. EXPERIMENTAL RESULTS

In the following, we describe the results of our experiments. First, we introduce the experimental benchmarks, along with information on the baselines used to assess the results (Section V-A). We then describe the strategy used to identify the optimal feature combinations (Section V-B). Finally, we discuss the experimental results (Section V-C).

A. LIVDET DATASETS

The benchmarks used in this work are those made publicly available for the LivDet 2009 [2], LivDet 2011 [3] and LivDet 2013 [4] competitions. These datasets have been largely used in the literature and enable a comparison with a great variety of methods.

Overall, the benchmarks consist in eleven sets of live and fake fingerprints acquired with different devices (Table 1), all of which are equipped with flatbad scanners, with the exception of Swipe, which has a linear sensor. Its images are obtained by swiping the fingerprint and thus include a temporal dimension as well. Each dataset is divided into separate training and test sets, and is characterized by a different image size and resolution, number of individuals, number of fake and live samples and number and type of materials used for creating the spoof artifacts. Nine out of the eleven fake sets were acquired using a consensual method, where the subject actively cooperated to create a mold of his/her finger, increasing the challenges related to the analysis of these datasets.

1) BASELINES

According to the standard LivDet protocols, the results are reported in terms of the *half total error rate* (HTER), which is the average between the percentage of misclassified live (*ferrlive*) and fake (*ferrfake*) samples, i.e. HTER $= \frac{ferrlive + ferrfake}{2}$.

Different baselines, summarized in Table 2, were used to assess our results. The first one (*Baseline*) includes the best results achieved by the analysis of individual features only and allows appreciating that feature fusion approaches are indeed capable of outperforming single–view learning methods. For each dataset, we provide a reference to the best scoring attribute in the table. The second baseline (SOA) collects the "best of the best results" selected from any approach following the LivDet experimental protocol. This second baseline determines if we are indeed competitive against the state–of–the–art. Alternatively, it can highlight limitations of our approach.

Since some authors (e.g. [27]) discarded the Crossmatch 2013 dataset, due to its generalization problems [4], we considered, for a fair comparison, two different average results for each baseline: *Avg*, which includes all eleven datasets, and Avg_{XM^-}, which rules out Crossmatch 2013.

2) IMAGE PREPROCESSING AND FEATURE EXTRACTION

For each benchmark we extracted all the attributes described in Section III without applying any preliminary image segmentation or pre–processing. This might appear a counterintuitive choice, especially when fingerprint segmentation is not taken into account, since removing the background helps

TABLE 2. Baselines (*Baseline* and SOA) and experimental results. Numbers in **bold** represent an improvement of the corresponding value in SOA, numbers in *italic* of that in *Baseline*. ∗ denotes a statistically significant difference (*p* < 0.05) with respect to *Baseline*, and † a statistically significant difference (*p* < 0.05) with respect to SOA.

Dataset	LivDet2009			LivDet2011				LivDet2013				Avg	Avg_{XM-}
Sensors	Biom.	XMatch	Identix	Biom.	Digital	Italdata	Sagem	Biom.	XMatch	Italdata	Swipe		
						Baselines							
Baseline	1.0	3.3	0.5	4.9	2.0	11.0	2.7	1.1	17.5	1.3	2.8	4.4	3.1
	LCPD [27]	SID [27]	KSIFT [27]	LCPD [16]	SID [27]	LCPD [16]	LCPD [16]	BSIF [27]	CCP [29]	LCPD [27]	DAISY [27]		
SOA	0.3	1.8	0.5	4.9	1.9	5.1	2.7	0.2	1.8	0.1	0.9	1.8	1.8
	WLD+LPQ [32]	Aug CNN [33]	KSIFT [27]	LCPD [16]	CNN [33]	CNN [33]	LCPD [16]	spoofnet [34]	spoofnet [34]	spoofnet [34]	spoofnet [34]		
						View Groups							
G1: SID RICLBP LCPD DSIFT LPQ–3+LPQ–5							$m = 4,424$						
AdaBoost	1.7	2.4	*0.0* †	3.1	3.2	5.8 ∗	2.6	1.0	6.4 ∗	1.5	5.5	3.0 ∗	2.7
Linear SVM	1.7	*1.8* ∗	0.9	*1.8* †	1.4	4.3 ∗	2.2	0.7	3.9 ∗	1.3	2.7	2.1 ∗	1.9 ∗
Spidernet	1.5	2.1 ∗	0.3	2.3 †	1.4	2.1 †	2.0	0.7	3.4 ∗	0.9	1.6	1.7 ∗	1.5 ∗
MvDA	1.3	**0.9** †	*0.0* †	0.7 †	0.7 ∗	4.9 ∗	2.3	0.5	4.4 ∗	0.5	1.3	1.6 ∗	1.3 †
G2: SID RICLBP LCPD DSIFT WLD LPQ–5+LPQ–7							$m = 7,304$						
AdaBoost	2.1	2.0 ∗	*0.0* †	3.0 †	3.3	5.8 ∗	2.2	0.9	5.8 ∗	1.5	6.4	3.0 ∗	2.7
Linear SVM	1.6	*1.6* ∗	1.0	2.0 †	1.4	3.8 ∗	2.1	0.7	3.9 ∗	1.7	2.9	2.1 ∗	1.9 ∗
Spidernet	1.6	1.8	0.3	2.4 †	1.0	3.1 †	1.5	1.0	3.7 ∗	1.0	1.6	1.7 ∗	1.5 ∗
MvDA	1.7	*1.1* †	*0.0* †	0.9 †	0.7 †	5.3 ∗	2.2	0.4	3.9 ∗	0.4 †	1.6	1.7 ∗	1.4 †
G3: SID RICLBP LCPD DSIFT WLD BSIF–5							$m = 7,816$						
AdaBoost	2.2	2.1 ∗	*0.0* †	3.3	2.9	5.9 ∗	2.1	1.1	5.6 ∗	1.6	5.3	2.9 ∗	2.7
Linear SVM	1.7	*1.5* ∗	1.0	1.8 †	1.2	4.0 ∗	2.3	0.6	4.3 ∗	1.7	2.7	2.1 ∗	1.9 ∗
Spidernet	1.6	1.8	0.3	2.3 †	1.0	2.6 †	2.1	0.9	3.3 ∗	0.7	1.8	1.7 ∗	1.5 ∗
MvDA	1.9	*1.1* †	*0.0* †	1.2 †	0.8 †	6.7 ∗	2.1	0.4	4.7 ∗	0.4 ∗	1.0 ∗	1.8 ∗	1.6 ∗
G4: SID RICLBP LCPD DSIFT WLD							$m = 6,792$						
AdaBoost	2.3	2.1 ∗	*0.0* †	3.2	3.1	5.8 ∗	2.2	1.0	5.6 ∗	1.3	5.3	2.9 ∗	2.6
Linear SVM	1.8	*1.6* ∗	1.0	2.0 †	1.3	3.5 ∗	2.2	0.7	4.7 ∗	1.8	2.8	2.1 ∗	1.9 ∗
Spidernet	1.9	1.8 ∗	0.3	2.0 †	0.7	2.8 †	1.2	1.1	3.3 ∗	1.1	1.8	1.6 ∗	1.5 ∗
MvDA	1.6	*1.1* †	*0.0* †	0.5 †	0.8 †	5.9 ∗	2.6	0.5	4.6 ∗	0.5	2.4	1.9 ∗	1.6 ∗
G5: SID RICLBP LCPD DSIFT RILPQ–3							$m = 4,168$						
AdaBoost	2.3	2.4 ∗	*0.0* †	3.5	3.5	6.0 ∗	2.9	1.3	6.3 ∗	1.4	5.1	3.2 ∗	2.8
Linear SVM	1.9	*1.7* ∗	0.9	1.5 †	1.4	4.0 ∗	2.5	0.7	4.4 ∗	1.3	2.6	2.1 ∗	1.9 ∗
Spidernet	1.4	2.0 ∗	0.4	1.7 †	1.3	2.9 †	1.7	1.0	3.6 ∗	0.9	1.9	1.7 ∗	1.5 ∗
MvDA	1.2	*1.0* †	*0.0* †	0.7 †	0.8 †	4.7 ∗	2.3	0.5	4.3 ∗	0.6	1.8	1.6 ∗	1.4 †
G6: SID RICLBP LCPD DSIFT							$m = 3,912$						
AdaBoost	2.3	2.3 ∗	*0.0* †	3.7	3.5	6.1 ∗	2.8	1.2	6.3 ∗	1.3	5.0	3.1 ∗	2.8
Linear SVM	2.1	*1.7* ∗	0.9	1.8 †	1.5	3.8 ∗	2.4	0.7	4.8 ∗	1.3	2.6	2.1 ∗	1.9 ∗
Spidernet	1.6	2.0 ∗	0.3	2.2 †	1.1	3.1 †	2.2	1.1	3.5 ∗	0.7	2.1	1.8 ∗	1.6 ∗
MvDA	1.1	*1.3* ∗	*0.0* †	0.9 †	0.8 †	4.9 ∗	2.5	0.5	5.2 ∗	0.4 ∗	2.0	1.8 ∗	1.4 †

to reduce the noise in the extracted features. The rationale of this choice was twofold. First, to keep the problem tractable since, besides optimizing the methods' hyperparameters, each combination of dataset, feature and classifier would also have required the preprocessing pipeline to be optimized. Second, our main target was to compare our results fairly with previous approaches, most of which did not rely on any pre–processing step.

The only exception is Crossmatch 2013. In our initial tests we faced the same generalization problems experienced by other authors. However, as carried out in other works (e.g. [33], [34]), a simple reduction of the image size by a factor four dramatically improved the accuracy for all features and methods. A possible explanation is that these images have a higher resolution, higher contrast and a better quality than images of other benchmarks. This leads to higher frequency components in smaller patches, which might have a severe impact on local texture features, such as the one considered in our work. Thus, it would seem that downsizing the images helps attenuating this problem.

B. SELECTING OPTIMAL FEATURE GROUPS

One of our initial research questions was trying to understand which are the most suitable combinations of attributes for the compared algorithms. In particular, in our approach we aimed at finding attribute groups capable of generalizing well across all datasets and methods, rather than selecting the optimal combination of features and method for each case, an option that could lead to higher accuracies, as we will also discuss in the following. The rational of our choice is that the generalization property is desirable in several practical cases (e.g., when the approach has to be applied to novel sensors or classification methods, or when it has to tackle novel spoofing materials).

Clearly, the numbers involved made an exhaustive search over all possible combinations and classification methods unfeasible. Therefore, we opted for an empiric "trial and error" approach, where we initially started creating attribute groups with what we deemed to be the most appealing views. Then, we started checking several variations, such as adding or removing views, combining microtextural and rich local features in different proportion, changing feature parameters and so on. In creating such variations, we avoided including "similar" features (e.g., KSIFT and DSIFT, CoALBP and RICLBP, LPQ and RILPQ) in the same group. This choice was based on the assumption that the complementary properties of these features are minimal.

The groups were initially tested with linear SVM, which is by far the less computationally demanding method, thus

allowing us to perform many experiments in a short time. The remaining classifiers were evaluated only for the best scoring groups. This protocol allowed us to spot common trends in the results. As we will show in more detail in Section V-C, the different approaches obtain slightly different results, but their error variations are strongly consistent. Basically, this finding allowed us to increase the number of SVM–based tests and to reduce the number of validations required, which involve more computationally demanding methods.

Since we had groups with both fixed (non–parametric) and parametric views, whose parameters were included in the hyperparameters to be optimized, we ended up identifying a set of *"families"*. A family is a group with both fixed and parametric features and its members are created by varying the parameters of the non–fixed features. For instance, KSIFT–SID–BSIF represent a family, whose members have a BSIF component computed at different scales or, in a multi–scale fashion, using different numbers and values of scales.

Given the candidate view families, we first roughly identified the optimal ones according to the results achieved by the classifiers introduced in Section IV. Since we were mainly interested in finding groups that showed good and coherent behavior, we chose the mean of the average error over all the benchmarks for each classifier as the ranking score. Then, in our preliminary experiments, we noticed that the scores of the family members varied in a small range and, thus, we selected our optimal families according to the average score of a small number (usually five) of randomly picked members.

Finally, given our optimal families, we analyzed all their members and selected the one with lowest error as representative.

C. RESULTS AND DISCUSSION

The experimental results are summarized in Table 2, where, for the sake of brevity, we only reported the representatives of the best six families. The table is divided into blocks, where each block reports the error for each benchmark and method (along with their two *Avg* and Avg_{XM-} averages) of the best performer of a family. Results are sorted according to their ranking scores (*i.e.*, average error over all benchmarks and classifiers). In addition, we marked with "∗" or "†" all results with a statistically significant difference ($p < 0.05$) with, respectively, *Baseline* and SOA. Finally, for each group we added the total number m of feature variables (i.e. the sum of the length of each view composing the group) and a label (Gxx) to facilitate the following discussion. Note that parametric features are identified as well by the scale used to compute them. For instance LPQ–5 means an LPQ descriptor computed on a local patch of size 5×5 pixels.

1) FEATURES

A first remark concerns the effects of the feature parameters in a multi–view setting. We found that a multiscale version with three scales for CoALBP, RICLBP and WLD was the best option in all the cases (as also suggested in [36], [37], and [57]).

As for LPQ, the multiscale approach was indeed effective in finding a good tradeoff between its capability to discriminate small details (for smaller scales) and the sensitivity to blur (for larger scales). However, we could not find an optimal formulation for all cases. In particular, it seems that the interaction with other features requires different settings. We thus ended up with a two scale version, optimizing the scales according to the characteristics of the view group.

We expected a similar behavior for RILPQ. On the contrary, we found that a single fixed size (a 3×3 neighborhood) was the best choice for all combinations. The same observation holds true for LCPD (with a local scale of 9). For this latter case, we suggest two possible explanations. First, the orientation component of LCPD is computed with RILPQ and, in some ways, it reflects its experimental behavior; second, the size of a single–scale LCPD is 2,048 and a multi–scale version is likely to incur in overfitting issues.

BSIFs were also tested in both single and multiscale versions. When tested individually, a multiscale version with three scales, spanning uniformly the interval between 5 and 17, consistently outperformed other options on all benchmarks. On the contrary, when combined with other features, single scales provided better results. We assume this behavior is related to the interplay with other group members.

Our results also offer an insight into the relevance of the individual features in a multi–view setting. First, CoALBP and KSIFT were consistently outperformed, in any possible group, by their direct peers (respectively, RICLBP and DSIFT). This fact can be explained in terms of their characteristics. Both CoALBP and RICLBP exploit the rich descriptive characteristics of LBP, with the addition of rotation invariance for RICLBP. KSIFT computation relies on the preliminary detection of the optimal keypoints. However, the combination of noisy images together with our choice to not discard the background might have led to a non optimal choice of the keypoints. On the contrary, the dense approach of DSIFT seems to be effective in softening the noise effects.

Final remarks concern the limited relevance of BSIFs (which, apparently, were not factually contributing to other views) and the lack of contribution of DAISY (indeed, substituting DAISY with any other rich local descriptor always improved the accuracy).

2) GROUPS

All the best families in Table 2 are based on a common core, i.e. the combination of SID, RICLBP, LCPD and DSIFT, which is also represented by group G6. When adding more features to this core, we noticed a saturation effect and, in some cases, even an error increase. The best reduction of the optimal average error was a mere 0.2% with MvDA when LPQ based features where added. We also experienced consistent accuracy drops (i.e. an average relative HTER increase ranging from 3% to 65%) for any combination where one or two core elements were removed or substituted with other views.

These findings could suggest that the members of the G6 kernel are indeed the ones, among those analyzed, which express the most complementary information. Thus, their combination appears effective in capturing the essence of the liveness detection problem. Analyzing the core features, we can notice that micro–textural (RICLBP and LCPD) and rich local descriptors (DSIFT and SID) are equally represented, which supports our choice of combining elements from the two classes. What can we infer from the characteristics of these individual features?

On the basis of the results in [27], we can see that, on average, LCPD and SID express the best liveness detection capabilities among the analyzed features, while the rank of DSIFT and RICLBP is quite low. In other words, individual performances are not sufficient to explain our results. As a demonstration, if we combine the best four (average) ranking features in [27] (SID, LCPD, BSIF, CoALBP), we obtain a 65% relative increase of the optimal HTER.

We can speculate that the combination of DSIFT and SID allows exploiting both the descriptive strength of SIFT and the higher robustness to rotation of SID. This observation might also explain the (lack of) contribution of DAISY, which has a lower rotation invariance than SID and a lower descriptive power than SIFT. In addition, (i) LCPD brings as dowry the fact of being conceived specifically for fingerprint images and, thus, of best exploiting expert knowledge on the specific domain, and (ii) RICLBP contributes with its rotation invariance, high descriptive ability and capability to adapt well to images with different resolutions, when used in its multiscale version.

The simpler explanation for the lack of significant improvements expanding the G6 core is the minimal or null complementarity of the added views. LCPD is a stack of eight RILPQ histograms, one for each quantization level of the contrast component. Thus, it conveys somewhat similar information to the LPQs–like views (G1, G2 and G5). The combination of WLD and LPQ (G2) is already summarized by LCPD, and BSIF (G3) was already noted for its limited contribution.

3) METHODS

As an initial remark, the four compared methods behave consistently across the different groups, as can be appreciated from the diagram in Fig. 3. Furthermore, with the exception of AdaBoost, these techniques provide similar degrees of accuracy, which suggests that the proper selection and engineering of the feature group is more relevant than the choice of the classification method.

The lowest average error (1.6) is obtained with both MvDA and *Spidernet*, which consistently outperform other methods. In the discussion we picked as optimal model MvDA on G1 since it provides a slightly lower number of absolute errors when compared to *Spidernet* on G4 and MvDA on G5 (respectively, 9 and 10 over a total number of 29,896 samples across all test datasets). However, we underline that this difference is not statistically relevant.
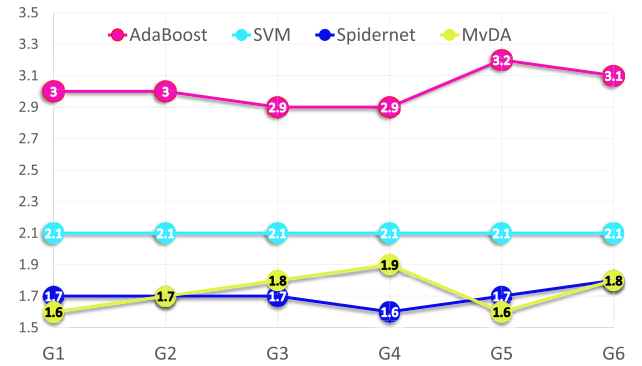

FIGURE 3. Average classification errors (HTER) of the classifiers on the different groups described in Table 2.

Concerning MvDA, these results are likely due to the fact that MvDA projects all the features into a common latent subspace taking into account not only inter–view variations, but also intra–view variations. This has both the effect of removing directions that are not useful for classification and normalizing the different features, thus mitigating the issues due to the presence of in–homogeneous features.

As for *Spidernet*, the two–stage architecture allows reducing overfitting issues by decoupling feature processing and classification. Combined with dropout and L2 regularization, this allows *Spidernet* to match the performance of MvDA, even though the amount of training data is very limited (around one thousand patterns per class). We believe that larger amount of training data would allow *Spidernet* to outperform the other methods. One possible question is whether Spidernet does effectively exploit all the input features. A view is totally discarded if (i) all the input weights of any of the hidden layers of the corresponding network leg (first stage) are null, or (ii) all the weights connecting the last hidden layer of the view leg and the first hidden layer of the second network stage are null. In all the experiments reported, none of these conditions is ever verified.

SVM with feature chaining has the advantage of providing a simple yet effective method for estimating cross–correlations among different features. However, the fact that our features can have very different characteristics seems to adversely affect the classification. Experiments also show that linear SVM is effective in controlling the influence of non–discriminative features by imposing a penalty on the combination weights, while the use of explicit feature selection has a detrimental effect on accuracy and was, thus, discarded.

As for Adaboost, its main advantage remains the fact that it uses only a few number of features from the multi–view space and the sample classification is computationally light.

Concluding, our experimental results allows us to provide a preliminary answer to the questions raised in Section IV. We underline that, given the limited number of different approaches compared, further work has to be done to achieve more solid conclusions. Based on these premises, our results

seem to suggest the following answers:

- as for the fusion level, fusion at feature level appears to be more effective than fusion at decision level;
- as for the harmonization or normalization of the aggregated features, the most effective methods seem to be those based on a proper transformation of the different views into a common latent space (i.e., the approach followed by MvDA and *Spidernet*);
- in order to deal with the curse of dimensionality problem, subspace transformations appear to be more useful for reducing the dimension of the classification space than feature selection techniques, which, in some cases, even appear to have a detrimental effect;
- finally, our idea to exploit deep learning approaches to automatically learn how to aggregate different features (*Spidernet*) seems to be an effective feature fusion method.

4) MULTIPLE FEATURES VS. INDIVIDUAL FEATURES

Our main research question was to verify the effectiveness of feature fusion approaches compared to the ones based on individual features. At least three facts allow us to provide an affirmative answer:

- in general, the compared multi–view approaches perform consistently better than those based on individual features (these results were not reported for the sake of brevity, although references can be obtained by observing that *Baseline* elements are most of the time components of the groups G1–G6);
- if we compare our results averaged over all the benchmarks with those reported in [27], our best approach (MvDA on group G1) significantly outperforms systems trained with a single feature;
- our best model outperforms, on average, the *Baseline*.

Indeed, we can observe that MvDA on G1 provides a 64% relative reduction of average error (58% excluding Crossmatch 2013) compared to models trained using the optimal feature for each dataset (row *Baseline* of Table 2). This improvement is robust across different groups. Furthermore, this approach consistently improves single–view performances on 10 out of 11 benchmarks without requiring to hand–pick different features for different datasets. In other words, our combinations of features appear to generalize well across different experimental conditions and different sensors. It is worth noting that, using different feature groups, we can also improve the results for the Biometrika 2009 dataset, where we score beyond the baseline. However, this would result in higher errors over different datasets.[1]

5) FEATURE FUSION APPROACHES VS. STATE–OF–THE–ART

As for the assessment of our work against the state–of–the–art, it can be seen that our average results are comparable

---

[1]A simple evidence of this statement is the SOA baseline for Biometrika 2009, i.e. the group WLD+LPQ classified with linear SVM, whose overall accuracy drops significantly in the other benchmarks (see [32]).

TABLE 3. Cross-sensor interoperability results (obtained on the LivDet 2011 datasets with MvDA and group G1).

Train set	Biom.	Italdata	Digital	Sagem
Biom.	**0.7**	42.1	32.5	29.8
Italdata	22.2	**4.9**	33.2	30.8
Digital	34.1	35.1	**0.7**	22.3
Sagem	22.5	39.7	29.0	**2.3**

with SOA and that, in the optimal case, we improve the baseline in 6 out of 11 benchmarks. We can also observe that, while we outperform the CNN–based approach in [33] in all benchmarks except Crossmatch 2013, the *spoofnet* CNN of [34], which was tested only on LivDet2013, achieves a significant reduction in terms of error rates. However, we should recall that (i) we looked for a solution capable of generalizing well across all datasets and methods, although better accuracies could be obtained selecting an optimal group for each benchmark, and (ii) the approach in [34] exploits dataset specific image pre–processing techniques, including image cropping and data augmentation that could also benefit our methods (although we did not apply them for the reasons explained in Section V-A.2). Furthermore, it should be also noted that, while the relative improvement of *spoofnet* compared to our best result looks relevant, if we exclude Crossmatch 2013, it actually corresponds to a very small difference in terms of absolute number of errors (11, over a total of 6,157 test samples across 3 datasets), and thus has little statistical significance.

6) CROSS–DATASET EVALUATION

Finally, we tested the interoperability performance of feature fusion approaches, i.e. the capability of handling variations in the biometric data introduced by different sensors. This is a difficult task, due to the different hardware characteristics of the capture devices. For this analysis, we performed cross-datasets experiments on the LivDet 2011 datasets according to the experimental protocol defined in [58] and [59]: we trained a classifier with the training set of sensor A (e.g. Biometrika2011) and then we classified the test set of sensor B (e.g. Italdata2011).

The results are summarized in the Table 3 where, for the sake of conciseness, we simply reported the HTERs obtained with MvDA and group G1 (which are anyway consistent with those obtained by other combinations of classifier and feature group). These results show large improvements with respect to the one showed in [58] (where the individual contributions of LBP, LPQ and BSIF were analyzed) and [59] (which uses Multi-Scale LBP as features). However, they also confirm other results available on cross-dataset experiments ([60]–[62]), which clearly show that the interoperability among different sensors is still an open issue [58].

VI. CONCLUSION

In this work we investigated the effectiveness of feature fusion approaches for fingerprint liveness detection tasks.

We addressed the issue of selecting a good set of complementary features, and we assessed the capabilities of different classifiers over a wide set of publicly available datasets, comparing our results with both single–view approaches and state–of–the–art techniques. Our results show that feature fusion approaches are effective and able to generalize well, without the need for dataset–specific image pre–processing and without requiring hand–picking of different features for different datasets. Indeed, we found a consistent improvement in terms of accuracy over single–view methods, even when such systems are trained using the optimal feature for each dataset. Furthermore, feature fusion methods are also competitive with other state–of–the–art approaches based on CNN, which rely on intensive image pre–processing steps.

Concerning the compared classifiers, both MvDA and *Spidernet* proved to be the most effective for combining the different features. As for the features, care has to be taken when designing the groups, since the selection of features should not be merely based on their individual performance, but should also consider their ability to mutually complement each other.

Finally, we stress the interesting results obtained by the DNN architecture proposed with *Spidernet*. Even though it did not outperformed MvDA in our tests, additional experiments show that our architecture has the potential to provide higher accuracy whenever larger training sets are available. Future work will be devoted to improve the optimization of the DNN architecture and the learning algorithms. Further work will be also devoted to tackling the liveness detection of other biometric traits, such as iris and faces, through feature fusion approaches.

ACKNOWLEDGMENT

REFERENCES

[1] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, *Handbook of Fingerprint Recognition*, 2nd ed. Berlin, Germany: Springer, 2009.

[2] G. L. Marcialis *et al.*, "First international fingerprint liveness detection competition—LivDet 2009," in *Image Analysis and Processing—ICIAP*. Berlin, Germany: Springer, 2009, pp. 12–23.

[3] D. Yambay, L. Ghiani, P. Denti, G. L. Marcialis, F. Roli, and S. Schuckers, "LivDet 2011—Fingerprint liveness detection competition 2011," in *Proc. 5th IAPR Int. Conf. Biometrics (ICB)*, Mar. 2012, pp. 208–215.

[4] L. Ghiani *et al.*, "Livdet 2013 fingerprint liveness detection competition 2013," in *Proc. Int. Conf. Biometrics (ICB)*, Jun. 2013, pp. 1–6.

[5] T. Matsumoto, H. Matsumoto, K. Yamada, and S. Hoshino, "Impact of artificial gummy fingers on fingerprint systems," *Proc. SPIE*, vol. 4677, pp. 275–289, Jan. 2002.

[6] M. Drahansky, "Experiments with skin resistance and temperature for liveness detection," in *Proc. IIH-MSP*, Aug. 2008, pp. 1075–1079.

[7] K. A. Nixon, V. Aimale, and R. K. Rowe, *Spoof Detection Schemes*. Boston, MA, USA: Springer, 2008, pp. 403–423.

[8] S. A. C. Schuckers, "Spoofing and anti-spoofing measures," *Inf. Secur. Tech. Rep.*, vol. 7, no. 4, pp. 56–62, 2002.

[9] E. Marasco and A. Ross, "A survey on antispoofing schemes for fingerprint recognition systems," *ACM Comput. Surv.*, vol. 47, no. 2, pp. 28:1–28:36, Nov. 2014.

[10] A. Al-Ajlan, "Survey on fingerprint liveness detection," in *Proc. IWBF*, Apr. 2013, pp. 1–5.

[11] A. Antonelli, R. Cappelli, D. Maio, and D. Maltoni, "Fake finger detection by skin distortion analysis," *IEEE Trans. Inf. Forensics Security*, vol. 1, no. 3, pp. 360–373, Sep. 2006.

[12] S. A. C. Schuckers, S. T. V. Parthasaradhi, R. Derakshani, and L. A. Hornak, "Comparison of classification methods for time-series detection of perspiration as a liveness test in fingerprint devices," in *Biometric Authentication* (Lecture Notes in Computer Science), vol. 3072, D. Zhang and A. K. Jain, Eds. Berlin, Germany: Springer, 2004, pp. 256–263.

[13] A. Abhyankar and S. Schuckers, "Fingerprint liveness detection using local ridge frequencies and multiresolution texture analysis techniques," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 2006, pp. 321–324.

[14] E. Marasco and C. Sansone, "Combining perspiration- and morphology-based static features for fingerprint liveness detection," *Pattern Recognit. Lett.*, vol. 33, no. 9, pp. 1148–1156, Jul. 2012.

[15] S. B. Nikam and S. Agarwal, "Co-occurrence probabilities and wavelet-based spoof fingerprint detection," *Int. J. Image Graph.*, vol. 9, no. 2, pp. 171–199, 2009.

[16] D. Gragnaniello, G. Poggi, C. Sansone, and L. Verdoliva, "Local contrast phase descriptor for fingerprint liveness detection," *Pattern Recognit.*, vol. 48, no. 4, pp. 1050–1058, 2015.

[17] A. Kembhavi, B. Siddiquie, R. Miezianko, S. McCloskey, and L. S. Davis, "Incremental multiple kernel learning for object recognition," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 638–645.

[18] C. Liu and P. C. Yuen, "A boosted co-training algorithm for human action recognition," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 9, pp. 1203–1213, Sep. 2011.

[19] Y. Fu, L. Cao, G. Guo, and T. S. Huang, "Multiple feature fusion by subspace learning," in *Proc. ACM Int. Conf. Content-Based Image Video Retr. (CIVR)*, New York, NY, USA, 2008, pp. 127–134.

[20] W. Zheng, X. Zhou, C. Zou, and L. Zhao, "Facial expression recognition using kernel canonical correlation analysis (KCCA)," *IEEE Trans. Neural Netw.*, vol. 17, no. 1, pp. 233–238, Jan. 2006.

[21] J. Cheng and K. Wang, "Active learning for image retrieval with Co-SVM," *Pattern Recognit.*, vol. 40, no. 1, pp. 330–334, 2007.

[22] L. Zhang, L. Zhang, D. Tao, and X. Huang, "On combining multiple features for hyperspectral remote sensing image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 3, pp. 879–893, Mar. 2012.

[23] A. Toosi, S. Cumani, and A. Bottino, "On multiview analysis for fingerprint liveness detection," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications* (Lecture Notes in Computer Science), vol. 9423, A. Pardo and J. Kittler, Eds. Cham, Switzerland: Springer, 2015.

[24] P. Coli, G. L. Marcialis, and F. Roli, "Power spectrum-based fingerprint vitality detection," in *Proc. IEEE Workshop Autom. Identificat. Adv. Technol.* Jun. 2007, pp. 169–173.

[25] J. Galbally, F. Alonso-Fernandez, J. Fierrez, and J. Ortega-Garcia, "A high performance fingerprint liveness detection method based on quality related features," *Future Generat. Comput. Syst.*, vol. 28, no. 1, pp. 311–321, 2012.

[26] S. B. Nikam and S. Agarwal, "Fingerprint liveness detection using curvelet energy and co-occurrence signatures," in *Proc. 5th Int. Conf. Comput. Graph., Imag. Vis. (CGIV)*, Aug. 2008, pp. 217–222.

[27] D. Gragnaniello, G. Poggi, C. Sansone, and L. Verdoliva, "An investigation of local descriptors for biometric spoofing detection," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 4, pp. 849–863, Apr. 2015.

[28] C. Gottschlich, E. Marasco, A. Y. Yang, and B. Cukic, "Fingerprint liveness detection based on histograms of invariant gradients," in *Proc. IEEE IJCB*, Sep./Oct. 2014, pp. 1–7.

[29] C. Gottschlich, "Convolution comparison pattern: An efficient local image descriptor for fingerprint liveness detection," *PLoS ONE*, vol. 11, no. 2, p. e0148552, 2016.

[30] L. Ghiani, G. L. Marcialis, and F. Roli, "Experimental results on the feature-level fusion of multiple fingerprint liveness detection algorithms," in *Proc. ACM Multimedia Secur.*, New York, NY, USA, 2012, pp. 157–164.

[31] L. F. A. Pereira *et al.*, "A fingerprint spoof detection based on MLP and SVM," in *Proc. IJCNN*, Jun. 2012, pp. 1–7.

[32] D. Gragnaniello, G. Poggi, C. Sansone, and L. Verdoliva, "Fingerprint liveness detection based on Weber local image descriptor," in *Proc. IEEE BIOMS*, Sep. 2013, pp. 46–50.

[33] R. F. Nogueira, R. de Alencar Lotufo, and R. C. Machado, "Evaluating software-based fingerprint liveness detection using convolutional networks and local binary patterns," in *Proc. IEEE Workshop Biometric Meas. Syst. Secur. Med. Appl. (BIOMS)*, Oct. 2014, pp. 22–29.

[34] D. Menotti *et al.*, "Deep representations for iris, face, and fingerprint spoofing detection," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 4, pp. 864–879, Apr. 2015.

[35] J. Chen *et al.*, "WLD: A robust local image descriptor," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1705–1720, Sep. 2010.

[36] R. Nosaka, Y. Ohkawa, and K. Fukui, "Feature extraction based on Co-occurrence of adjacent local binary patterns," in *Advances in Image and Video Technology*. Berlin, Germany: Springer, 2011, pp. 82–91.

[37] R. Nosaka, C. H. Suryanto, and K. Fukui, "Rotation invariant Co-occurrence among adjacent LBPs," in *Proc. Workshops Comput. Vis. (ACCV)*, 2012, pp. 15–25.

[38] C. H. Chan, M. A. Tahir, J. Kittler, and M. Pietikäinen, "Multiscale local phase quantization for robust component-based face recognition using kernel fusion of multiple descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 5, pp. 1164–1177, May 2013.

[39] V. Ojansivu and J. Heikkilä, "Blur insensitive texture classification using local phase quantization," in *Image and Signal Processing*. Berlin, Germany: Springer, 2008, pp. 236–243.

[40] V. Ojansivu, E. Rahtu, and J. Heikkilä, "Rotation invariant local phase quantization for blur insensitive texture analysis," in *Proc. 19th Int. Conf. Pattern Recognit.*, vol. 4. Dec. 2008, pp. 1–4.

[41] J. Kannala and E. Rahtu, "BSIF: Binarized statistical image features," in *Proc. IEEE 21st Int. Conf. Pattern Recognit. (ICPR)*, Nov. 2012, pp. 1363–1366.

[42] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[43] E. Tola, V. Lepetit, and P. Fua, "DAISY: An efficient dense descriptor applied to wide-baseline stereo," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 5, pp. 815–830, May 2010.

[44] I. Kokkinos and A. Yuille, "Scale invariance without scale selection," in *Proc. IEEE CVPR*, Jun. 2008, pp. 1–8.

[45] L. Ghiani, G. L. Marcialis, and F. Roli, "Fingerprint liveness detection by local phase quantization," in *Proc. ICPR*, Nov. 2012, pp. 537–540.

[46] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, "A practical guide to support vector classification," Dept. Comput. Sci., Nat. Taiwan Univ., Taipei, Taiwan, Tech. Rep., 2010.

[47] K. Kira and L. A. Rendell, "The feature selection problem: Traditional methods and a new algorithm," in *Proc. AAAI*, 1992, pp. 129–134.

[48] M. Kan, S. Shan, H. Zhang, S. Lao, and X. Chen, "Multi-view discriminant analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 1, pp. 188–194, Jan. 2016.

[49] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, Aug. 1997.

[50] P. Negri, X. Clady, S. M. Hanif, and L. Prevost, "A cascade of boosted generative and discriminative classifiers for vehicle detection," *EURASIP J. Adv. Signal Process.*, vol. 2008, p. 782432, Dec. 2008. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.395.6769&rep=rep1&type=pdf

[51] P. Negri, N. Goussies, and P. Lotito, "Detecting pedestrians on a movement feature space," *Pattern Recognit.*, vol. 47, no. 1, pp. 56–71, 2014.

[52] R. E. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions," *Mach. Learn.*, vol. 37, no. 3, pp. 297–336, 1999.

[53] G. E. Hinton, *A Practical Guide to Training Restricted Boltzmann Machines*. Berlin, Germany: Springer, 2012, pp. 599–619.

[54] G. E. Hinton, "Learning multiple layers of representation," *Trends Cognit. Sci.*, vol. 11, no. 10, pp. 428–434, Oct. 2007.

[55] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[56] J. Duyck, M. H. Lee, and E. Lei, "Modified dropout for training neural network," School Comput. Sci., Carnegie-Mellon Univ., Pittsburgh, PA, USA, Advanced Introduction to Machine Learning Course, Tech. Rep. 10-715, Fall 2014.

[57] M. Hussain, G. Muhammad, and G. Bebis, "Face recognition using multi-scale and spatially enhanced weber law descriptor," in *Proc. IEEE SITIS*, Nov. 2012, pp. 85–89.

[58] L. Ghiani, V. Mura, P. Tuveri, and G. L. Marcialis, "On the interoperability of capture devices in fingerprint presentation attacks detection," in *Proc. ITASEC*, Jan. 2017, pp. 66–75.

[59] X. Jia *et al.*, "Multi-scale local binary pattern with filters for spoof fingerprint detection," *Inf. Sci.*, vol. 268, pp. 91–102, Jun. 2014.

[60] S. Mason, I. Gashi, L. Lugini, E. Marasco, and B. Cukic, "Interoperability between fingerprint biometric systems: An empirical study," in *Proc. 44th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Washington, DC, USA, Jun. 2014, pp. 586–597.

[61] Z. Akhtar, C. Micheloni, and G. L. Foresti, "Correlation based fingerprint liveness detection," in *Proc. Int. Conf. Biometrics (ICB)*, May 2015, pp. 305–310.

[62] E. Marasco, P. Wild, and B. Cukic, "Robust and interoperable fingerprint spoof detection via convolutional neural networks," in *Proc. IEEE Symp. Technol. Homeland Secur. (HST)*, May 2016, pp. 1–6.

AMIRHOSEIN TOOSI received the B.S. degree in software engineering from the Azad University of Tehran, Iran, in 2008, and the M.S. degree in mechatronics engineering from Qazvin Azad University, Iran, in 2013. He is currently pursuing the Ph.D. degree in computer and control engineering with Computer Graphics and Vision Research Group, Politecnico di Torino, Italy. His research interests are in the area of robotics, haptics, computer graphics, computer vision, and deep learning.

ANDREA BOTTINO (M'08) is currently a Professor of computer science and Head of the Computer Graphics and Vision Research Group, Department of Control and Computer Engineering of the Politecnico di Torino. He is the author of several journal and conference papers. His current research interests include computer vision, machine learning, human computer interaction, computer graphics, virtual, and augmented reality.

SANDRO CUMANI received the M.S. degree in computer engineering from the Politecnico di Torino, Torino, Italy, in 2008, and the Ph.D. degree in computer and system engineering from the Politecnico di Torino in 2011. He was with the Brno University of Technology, Czech Republic, and is also a Research Fellow with the Department of Control and Computer Engineering, Politecnico di Torino. His current research interests include machine learning, speech processing and biometrics, in particular speaker, and language recognition.

PABLO NEGRI received the degree in electronic engineering from the Universidad Nacional de La Plata in 1998 and the Ph.D. degree in computer vision from Université Pierre et Marie Curie-Paris VI in 2008. Since 2010, he has been a Researcher of CONICET. He joined the Institute of Technologies, Universidad Argentina de la Empresa, before moving to the National Scientific and Technical Research Council, University of Buenos Aires, in 2017. His research interests are machine learning and pattern recognition applied to computer vision for intelligent traffic or biometric applications. He received the Best Paper Award in CIARP 2012.

PIETRO LUCA SOTTILE received the M.S. degree in computer engineering from the Polytechnic of Turin, Italy, in 2016. His thesis project was focused primarily on deep neural networks for spoofing detection in biometric systems. He is currently a Data Scientist and his research interests involve data mining, computer vision, and deep learning.

● ● ●