Doctoral Dissertation
Doctoral Program in Computer and Control Engineering (30^{th}cycle)

Feature Fusion for Fingerprint Liveness Detection

By

Amirhosein Toosi
******

Supervisor(s):
Prof. Andrea Bottino, Supervisor

Doctoral Examination Committee:

Politecnico di Torino
2018

Declaration

I hereby declare that, the contents and organization of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Amirhosein Toosi
2018

* This dissertation is presented in partial fulfillment of the requirements for **Ph.D. degree** in the Graduate School of Politecnico di Torino (ScuDo).

To my motherland,

may stay safe from invasions, famine, and lies..

Acknowledgements

Reflecting on my years spent in Turin at Politecnico, it is undeniable that my achievements, especially this dissertation, would not have been possible without the many contributions from my insightful advisor, gracious friends, and supportive family. Although I cannot hope to repay this sizable debt within a page, I would like to take the next few lines to humbly acknowledge those that have pushed me forward and kept me from drifting astray.

First and foremost, I would like to express my sincere gratitude to my advisor Professor Andrea Bottino for the continuous support of my Ph.D study and related research, for his patience, motivation, kindness, and immense knowledge. He has been not only my supervisor, but like an older brother to me, and his endless support and energy helped me in all the past three years of research which ended up writing this thesis. I could not have imagined having a better advisor and mentor for my Ph.D study. It has been my greatest honor to be his student and I sincerely appreciate all his contributions of time, help, availability, following up, and funding for the past few months to make my Ph.D. experience productive and stimulating. The joy and enthusiasm he has for research was contagious and motivational for me, even during tough times in the Ph.D. pursuit. He is simple the nicest and the smartest man I have ever seen.

Besides, special mention goes to Dr. Sandro Cumani for not only his great contribution to this research work and sharing his wonderful expertise in the field, but also for being such a great friend. He has always been there, so kind, smart, fun and helpful.

My sincere thanks also goes to my thesis committee for their time they dedicated to read my dissertation and their insightful comments and encouragement. It is my greatest honor that they accepted to review my thesis.

In Lab 1, my habitat, I would like to thank my fellow labmates for all the tough days and the sleepless nights we were working before deadlines together, and for all the fun we have had in the last three years, Andrea, Christian, Oscar, Jacopo, Davide, Francesco, Alysson, Rifat, Ricardo, Diego and Erion. Also I thank my friends in DAUIN for all the times we spent downstairs, Alberto, Matteo, Yuka and Francesco. Life could have been indeed so hard without you all.

Back in Iran, I would like to acknowledge my master thesis' advisor Dr. ArbabTafti, My biggest inspiration that certainly influenced my decision to continue on to graduate school and to never give up on my dreams. Without him I could have never even imagined reaching where now I am and writing this thesis. Anything that I am achieving in my way would have been impossible if I had not met him.

Talking about my friends I would like to especially thank my always best friend and my best teacher, Maziyar. Although I missed him a lot, he has always been there for me through all the ups and downs of my life. In addition I would like to thank Abbas, I was so lucky to meet him and after that being away from home has not been unbearable for me. He made me always feel like home during the weekends, especially thanks to the delicious Persian foods he cooks.

I would also like to thank Prof. Shahri, head of Mechatronics Department of Qazvin University for all his support, his kindness and all the opportunities he provided me during my Masters and all his trust he had in me. Nevertheless, I am also grateful to the real inspiring tireless adventurer and the wonderful scientist, Prof. Barry Richardson. I wish I will be such inspiration that he has always been for me.

Finally, but by no means least, all of my heart goes to Mom, Dad and Amirali for their unstoppable love, support and sacrifice. Undoubtedly they are the most important people and all that I have in my whole world. I dedicate this thesis to them.

As the last word, I should mention that I have so many people to thank that I'm likely to have forgotten someone. If I've forgotten you, please accept my apologies.

Abstract

For decades, fingerprints have been the most widely used biometric trait in identity recognition systems, thanks to their natural uniqueness, even in rare cases such as identical twins. Recently, we witnessed a growth in the use of fingerprint-based recognition systems in a large variety of devices and applications. This, as a consequence, increased the benefits for offenders capable of attacking these systems. One of the main issues with the current fingerprint authentication systems is that, even though they are quite accurate in terms of identity verification, they can be easily spoofed by presenting to the input sensor an artificial replica of the fingertip skin's ridge-valley patterns.

Due to the criticality of this threat, it is crucial to develop countermeasure methods capable of facing and preventing these kind of attacks. The most effective counter–spoofing methods are those trying to distinguish between a "live" and a "fake" fingerprint before it is actually submitted to the recognition system. According to the technology used, these methods are mainly divided into hardware and software-based systems. Hardware-based methods rely on extra sensors to gain more pieces of information regarding the vitality of the fingerprint owner. On the contrary, software-based methods merely relies on analyzing the fingerprint images acquired by the scanner. Software-based methods can then be further divided into *dynamic*, aimed at analyzing sequences of images to capture those vital signs typical of a real fingerprint, and *static*, which process a single fingerprint impression. Among these different approaches, static software-based methods come with three main benefits. First, they are cheaper, since they do not require the deployment of any additional sensor to perform liveness detection. Second, they are faster since the information they require is extracted from the same input image acquired for the identification task. Third, they are potentially capable of tackling novel form of attack through an update of the software.

The interest for this type of counter–spoofing methods is at the basis of this dissertations, which addresses the fingerprint liveness detection under a peculiar perspective, which stems from the following consideration. Generally speaking, this problem has been tackled in the literature with many different approaches. Most of them are based on first identifying the most suitable image features for the problem in analysis and, then, into developing some classification system based on them. In particular, most of the published methods rely on a single type of feature to perform this task. Each of this individual features can be more or less discriminative and often highlights some peculiar characteristics of the data in analysis, often complementary with that of other feature. Thus, one possible idea to improve the classification accuracy is to find effective ways to combine them, in order to mutually exploit their individual strengths and soften, at the same time, their weakness. However, such a "multi–view" approach has been relatively overlooked in the literature.

Based on the latter observation, the first part of this work attempts to investigate proper feature fusion methods capable of improving the generalization and robustness of fingerprint liveness detection systems and enhance their classification strength. Then, in the second part, it approaches the feature fusion method in a different way, that is by first dividing the fingerprint image into smaller parts, then extracting an evidence about the liveness of each of these patch and, finally, combining all these pieces of information in order to take the final classification decision.

The different approaches have been thoroughly analyzed and assessed by comparing their results (on a large number of datasets and using the same experimental protocol) with that of other works in the literature. The experimental results discussed in this dissertation show that the proposed approaches are capable of obtaining state–of–the–art results, thus demonstrating their effectiveness.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Fingerprints are one of the biometric traits which is most frequently used as authentication system in a plethora of applications ranging from security to surveillance and forensic analysis [59]. Over the recent years they have established their place as an alternative or supplement to traditional individual identification methods (e.g., token or password-based ones). The reasons for this trend are mainly the (perceived) increase in security and the ease of use of biometric system. Today fingerprint recognition systems are cost–effective solutions that guarantee high recognition accuracy on large datasets of millions of images. Thanks to these characteristics, beside the most common applications (such as immigration regulations, international borders, classified resources or information access control and electronic transactions), they are starting to be deployed in novel scenarios, like granting access to schools, health or leisure facilities, identifying patients in hospitals and using pay–with–fingerprint systems as an alternative to cash or credit cards. Fingerprint sensors are also becoming commonly available on consumer devices like notebooks or smartphones.

However, authentication systems based on fingerprints are vulnerable to more or less sophisticated forms of spoofing that can result in granting unauthorized access. As an example, in September 2013, Apple revealed its new iPhone 5s. This device was equipped with a fingerprint sensor that could be used to unlock the device. Three days later, a German hacker group posted the detailed instructions for hacking the system. What they did was neither based on an in-depth knowledge of the internal mechanism of the fingerprint recognition module nor on a brilliant programming

proficiency. Instead, they simply created a synthetic replica of a latent fingerprint by using play-doh and vinyl glue, which was sufficient to grant access to the system when submitted to the recognition system. On September 2014, Apple presented the iPhone 6, and one might think they learned the lesson. Surprisingly, the exact same trick worked perfectly again.

This is just a case illustrating the susceptibility of biometric systems in general and of fingerprint-based identification systems in particular, to intruding attacks. This vulnerability becomes a serious issue in spite of the variety of daily uses of such systems.

Speaking in general, two main attack scenarios to a biometric-based recognition system can be considered: *direct* or *indirect* [60]. Indirect methods are regarded as the attacks carried out in order to break in the internal software or hardware mechanisms of the biometric system. Hence, they can be prevented by using encrypted information or blocked using firewalls and so on. On the contrary, direct attacks (i.e. attacks at sensor-level) are these attacks carried out on the outside of the system.

The most common form of attack is similar to the example previously introduced, i.e. a situation where the offender tries to access to the system by presenting to the input sensor a replica of the biometric traits of an authorized individual. This method, is called "*spoofing attack"*. In the same fashion, the fake biometric traits employed in direct attacks are called "*spoofs*". In the context of this work, the spoofs are simple artificial duplicated fingerprints obtained by making a mold from a real or a latent fingerprint, using simple and cheap materials such as Play-Doh, latex, gelatin and so on ([63, 105, 32]). As a result, the biometric system's security is endangered since it might not be capable of distinguishing a real from a fake input. The literature shows that the success rate of such spoofing attacks can be up to 70% of the cases [65], which makes the problem extremely critical. This is the reason why a module capable of detecting spoof attacks, telling a fake from a live sample, is sorely needed to prevent false samples to be sent to the identification module.

Several methods have been proposed in the literature to detect the liveness of a fingerprint (or, in other terms, to identify a spoofed fingerprint). One option is to exploit *hardware-based* methods, which employ peripheral (and accessory) biometric sensors capable of capturing the vital characteristics of the input sample. Some examples are the measurement temperature [22], skin electrical conductivity [75], pulse oxometry [92], skin resistance [22], the amount of light passing through the

finger in the presence of a light sensor [7] and the smell of the sample [8]. A different approach is offered by *software-based* methods, which are simply based on the analysis of the acquired fingerprint images. This analysis usually exploit different image features that are further processed in various ways.

Software–based approaches can be then further divided into *static* (using a single image) or *dynamic* (processing an image sequence). Dynamic methods are aimed at analyzing, in a temporal video sequence, phenomena like skin deformation and perspiration that are typical of a live fingerprint. This analysis comes at the cost of an increase of computational complexity and processing time. On the contrary, static methods are by far the most attractive methods since, relying on a single image, they require less data, less computational resources and are suitable for general use (e.g., they can be applied also to these sensors that cannot capture an image stream).

While hardware-based techniques are theoretically expected to obtain higher precision in spoof detection compared to software-based methods, the introduction of novel components increases as well the cost of the system, without neglecting the fact that a capable intruder can find a way to spoof even the additional sensors once their characteristics are known. Thus, software–based approaches are often preferable in the practice and, in particular, represent the only viable solution when it is not possible to modify the deployed hardware (e.g., on consumer devices, like smart–phones, already available in commerce). As other advantages, these solutions are less invasive and more flexible, since they can potentially tackle novel types of attack by a simple update of the software.

1.1 Contribution of this work

Given their characteristics of generalization and simplicity, this dissertation focuses on the development of novel software static methods for the fingerprint liveness detection. In order to introduce and clarify the approach followed in this work, along with its contribution, it is necessary to take a brief overview of the literature on the subject.

Several methods have been proposed and published, most of which focus on the analysis of individual image features. The initial works were based on the observation that the images taken from fake fingerprints are usually characterized by

a lower image quality and, thus, researchers tried to analyze some quality indexes based on a plethora of different holistic features. However, these approaches did not show a promising discriminative power, and the following approaches started analyzing various local textural features extracted from the image.

The common characteristic of all these features is that their engineering is 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 and often complementary characteristics of the samples. Given their differences, developing approaches that combine multiple features, capable of mutually exploiting their strengths and, at the same time, softening their weaknesses, could lead 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 [50] and human activity recognition [56], face [26] and facial expression recognition [110], content–based image retrieval [15] and hyperspectral image classification [108]. 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 first part of this dissertation focuses on analyzing the effectiveness of feature fusion approaches as anti–spoofing methods and to compare them with the state of the art (Chapter 3). This objective 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, the approach followed 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).

The second part of this research (Chapter 4) leverages on the recent popularity gained by Deep Learning approaches (in particular, of Convolutional Neural Networks, CNNs) in several visual recognition tasks, which has motivated researchers to apply them to the fingerprint liveness detection problem. The main idea behind this general approach is that CNNs, which works directly on the raw input image,

are capable of automatically learning the "optimal" features for the problem at hand. This is in contrast with the approaches based on "handcrafted" features, where the choice of (what can be considered as) the most suited features is often empirical and sub-optimal.

In particular, this part of the dissertation proposes a patch-based strategy whose rationale is threefold. First, since the dimension of the CNN input layers is necessarily limited, using small sized patches allows to avoid resizing the samples and, thus, to retain the original resolution and image information. Second, using patches rather than the full images as samples, allows increasing the size of the training set, thus (hopefully) making the classifier more robust and increasing its generalization capabilities. Third, the exploitation of fusion approaches, based on the combination at different levels of the pieces of information extracted from the patches, is likely (again) to improve the robustness of the final fingerprint classification process.

Both the proposed approaches (i.e., fusing handcrafted features or deep patch–based features) have been assessed on a set of publicly available benchmarks that have been widely used in the literature and, thus, enable a comparison with a great variety of methods. Overall, experimental results indicate the effectiveness of both the general approaches, which are capable of providing state–of–the–art results.

Chapter 2

Background and Literature Review

Biometric recognition, or biometrics, in short, is referred as the identity recognition of a person based on physical or behavioral attributes such as fingerprint, face, iris, and voice [44]. The uniqueness of many physical and behavioral attributes of humans, combined with the capability of capturing in digital format, these characteristics using well-designed sensors, allows the development of identity recognition systems based on the comparison of the extracted data. Thus, biometrics can be viewed as a pattern recognition problem, where the machine first learns the salient features (patterns) in the biometric attributes of an individual and then matches such patterns efficiently and effectively [44].

Among the different biometric traits used in automated identification systems, fingerprints can be considered as the most popular and successful one. Although fingerprints have been in use in forensic applications for over 100 years, the recent availability of low-cost and compact fingerprint scanners allowed the deployment of a large number of different scenarios, in such a way that fingerprint recognition has become in people's mind a synonymous of biometric recognition.

However, despite the general accuracy and ease of use of fingerprint–based recognition systems, there is a significant body of research showing their vulnerability to different types of threatening attacks. These attacks can be classified in two main groups: *obfuscation* and *impersonation*. Fingerprint obfuscation (or alteration) is the intentional attempt of a person to mask her/his identity to a biometric system by altering the fingertip skin pattern [107] (e.g., by burning or cutting the fingertip,

surgically interventions and so on). Example of these attempts have been observed in law enforcement, national identification and border control systems [106].

A more dangerous type of attack is impersonation, which corresponds to a sensor-level attack where either an attacker gains unauthorized access using the biometric identity of an enrolled person or a new identity (possibly shared between multiple individuals) is created using fake biometric traits [75]. These kinds of attacks, which are also referred as *presentation attack* or *spoofing attack*, are generally carried out with *spoofs*, i.e. objects created by making a mold of a latent or real fingerprint and then filling it with materials such as gelatin, silicone or Play-Doh. In these terms, performing a spoofing attack does not require any expert knowledge about the internal operation of the biometric system and, although extremely simple, the literature reports the (surprisingly) high success rate of these attacks [65].

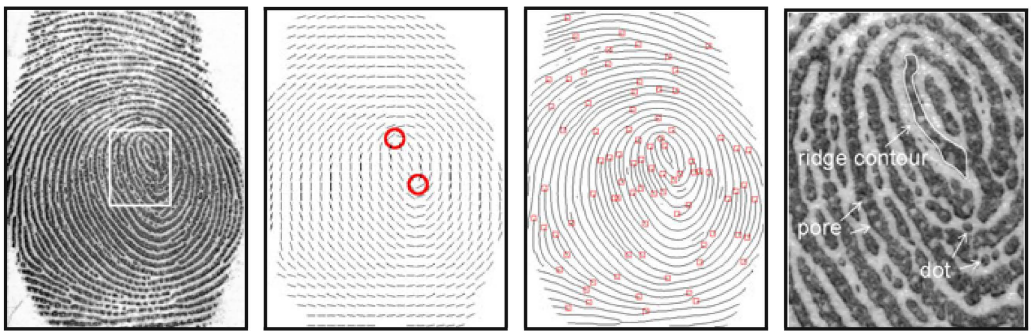
In order to reduce this vulnerability, different countermeasure methods (usually referred to as *liveness detection*) have been proposed in the literature. Their goal is to provide the biometric system with the capability of automatically telling whether the object placed on the sensor is a live or a fake finger. In the following Sections, details will be given on the characteristics of fingerprint patterns and of the different counter–spoofing methods proposed in the literature.

2.1 Fingerprint patterns

Fingerprints are characterized by an intricate pattern of interleaved ridges and valleys on the tip of the finger (friction ridges) [59]. This pattern is claimed to be unique and permanent for each finger, thus suggesting its use as a robust identifier. As a matter of facts, even identical twins (which are hardly distinguishable with face recognition systems) have different fingerprint patterns.

The process of fingerprint recognition, either done by a human expert or a machine, is mainly feature-based [44], where the features used can be represented in hierarchical order at the following three different levels, ranging from coarse to fine (see Fig 2.1).

- *Level 1.* At the global level (Level 1), the shape and structure of a fingerprint are summarized by features like ridge orientation and ridge frequency map for each location on the fingerprint. This information allows to extract a



(a) (b) (c) (d)

Fig. 2.1 (a) Grayscale fingerprint image, (b) Level 1 feature (orientation field, ridge flow and singular points), (c) Level 2 feature (ridge skeleton), and (d) Level 3 features (ridge contour, pores, and dots).

characterizing shape (such as arch, loop, delta, or whorl), whose distinctiveness is not sufficient for recognition, but is enough for fingerprint classification and indexing. Since level 1 features ignore the exact location and dimensional details of ridges, a low-resolution scanner (250 ppi) is usually enough to extract them.

- *Level 2.* The second level groups local features called minutiae, which are defined as the locations where a ridge emerges, ends, splits, or merges with another ridge. The minutiae are extracted through a detailed analysis of ridges, which involves their preliminary skeletonization. Each minutia is then characterized by its direction, type and location in the image. Level 2 features can be easily extracted from images acquired at a resolution higher than 500 ppi. Minutiae-based representations are extensively used in automated fingerprint recognition systems, primarily due to the fact that they capture most of the discriminative information of fingerprints and are reasonably robust to variations of the image quality.

- *Level 3.* At a very fine level, a fingerprint is represented by the pores and edges of the ridges acquired by high-resolution scanners (with resolution higher than 1,000 ppi). Level 3 features are receiving increased attention due to their importance in the analysis of latent fingerprints, which generally contain much fewer details compared to the rolled or plain ones.

2.2 Spoofed fingerprints

An artificial (or spoofed) fingerprint is a skin-like mask made of gelatin, latex, silicone, wood glue, clay or similar materials, which contains a fingerprint pattern on its outer surface [63]. Spoofs can be created in two ways, according to the level of participation in this process of the fingerprint owner [63] [32].

- *Cooperative spoofing.* In the first method, the enrolled user actively cooperate to create a negative impression of her/his fingertip by placing it on a mold made of various materials, like silicone or plaster. The spoof is then created by filling the mold with a thin layer of a moisture-based spoof material.
- *Non-cooperative spoofing.* When the user is not directly involved into the spoof creation process, another option is to use a latent fingerprint (e.g., one impressed on a surface such as the display of a mobile phone or a glass). This latent fingerprint is first lifted and then used for creating a spoof, either by printing its image with conductive silver ink or by etching it onto a photo-lithographic printed circuit board (PCB) and then covering it with a thin layer of a spoof material.

From this description, it is clear that there is a large difference in the spoof quality among those obtained in a consensual and unconsensual way. The former allows to create an almost perfect copy a fingerprint, which can be difficult to be detected as a fake even by a human expert. On the contrary, the non-cooperative methods results in a fingerprint image with a lower quality, due to the fact that several details are already lost in the latent fingerprint and further imperfections are introduced by the subsequent spoof creation procedure. However, it should be underlined that it is much easier for an intruder to operate on a latent fingerprint rather than convincing an enrolled user to leave a cast of her/his finger [33].

2.3 Liveness Detection Methods

According to the technology used, the liveness detection methods can be roughly divided into *hardware* and *software* based.

Hardware-based techniques try to capture the vitality attributes of the fingertip, such as finger's temperature, electrical conductivity, heartbeat and skin resistance. These characteristics are analyzed by coupling additional dedicated hardware along with the fingerprint sensor, however, this introduces several drawbacks. For instance, the new required hardware increases the costs and, then, it does not necessarily secure the system against spoofing attacks, since an expert can exploit the knowledge of the new configuration to find suitable counter–measures (e.g., placing a fake fingerprint on the fingerprint scanner and a live one on the additional hardware [92] [60]).

Software-based liveness detection methods are, instead, merely based on image processing algorithms applied to the samples acquired from the scanner [19]. These methods offer some desirable properties compared to hardware-based ones, since they do not require the deployment of any additional hardware. Thus, they are less expensive and there is no need to update or modify the hardware of any biometric systems already available. In these methods, pieces of information (features) are extracted from the fingerprint images in order to tell a live from a fake input. Based on the number of images that are required to perform liveness detection of each input fingertip, they are further divided into the two sub classes of *dynamic* and *static* methods, which are detailed in the following subsections.

2.3.1 Dynamic methods

Dynamic methods are based on the observation that live and fake fingers are characterized by different changes occurring over time in their "skin". Capturing these differences can be done analyzing a temporal image sequence during the fingerprint impression, on a time span that usually ranges between 2 and 5 seconds. As a result, there is an increase in the response time required by dynamic methods to take the decision of keeping or rejecting an input sample.

As for the possible approaches, the analysis is usually focused on identifying two distinct (dynamic) phenomena occurring in the fingerprint surface: *perspiration* and *elastic deformation*.

Perspiration Based

Perspiration is a typical behavior of the human skin, where the sweat leaks from pores and expands along ridges, causing a variation of the intensity in the regions between pores. These moisture patterns can be captured by observing multiple fingerprint images over a short time interval time. For instance, [20] captured the perspiration pattern's change from two consecutive fingerprints scanned with a difference of 5 seconds. Then, several statistics about the temporal changes occurring in the ridge signals were used to discriminate between live and fake samples. Since an excessive amount of moisture can produce a saturated signal, two new dynamic measures, respectively the dry and wet saturation percentage changes, were added in [86], showing an improvement of the overall accuracies for different devices. In [2], the use of a wavelet analysis is proposed to isolate the changes in perspiration patterns. In this case, the differences in the wavelet coefficients between two consecutive scans (taken 2 seconds apart) are used as vitality measures.

Elastic Deformation Based (Morphology-based)

Elastic deformation refers to the distortions of the fingertip skin during the scan. To analyze this penomenon, in [5] the user is asked to rotate the fingertip on the scanner surface. An image sequence acquired at high frame rate is then processed to extract the optical flow between consecutive images, whose output is encoded into a "distortion code" that is finally used for comparison. Another work [45] proposed two (simple) dynamic features to capture the fingertip skins elasticity: (i) the correlation coefficient between the fingerprint area and the signal (pixel) intensity, and (ii) the standard deviation of the fingerprint area extension along x and y. Then, the Fisher Linear Discriminant is used to tell a real skin from a spoofed one. In [109] authors modeled the skin deformation using a thin plate spline (TPS). Users are asked to put their finger on the sensor surface, and then apply minor pressure in four different directions. Authors observed that, spoof materials are usually more rigid than live skin and, thus, their deformation is lower under the same pressure condition. The minutiae displacement were used to build the TPS models, whose bending energy vector was used as the discriminative characteristic.

2.3.2 Static methods

Static software methods are based on the analysis of a single image. Thus, (i) they have a faster response time than dynamic ones, (ii) they are usually computationally lighter (since they do require the registration of multiple images) and (iii) they are more general, since they can be applied to any device (thus, including those not initially designed for acquiring an image sequence). An indication of the great interest towards these counter–spoofing methods is the large public availability of single–scan datasets such as the ones described in Section 2.4 , which have been collected to provide a common testbed that allows researchers assessing different algorithms on the same experimental conditions.

The general approach implemented in static methods is to analyze different image features. These features can be roughly divided in two main groups, based on the way they are computed: *holistic features*, which consider the image as a whole, and *local features*, which are extracted on a per–pixel (or per–region) base. As we will highlight in the following, the literature clearly shows large differences in the discriminative power expressed by these two classes. More recently, researchers started exploiting as well the advanced classification capabilities offered by deep learning approaches.

Holistic Features (Global Descriptors)

One of the options considered by holistic approaches is to derive, from a single image, some global characteristics related to the perspiration phenomenon. For instance, [98] try to discriminate between live and fake perspiration patterns by means of a combination of spatial, frequency and wavelet analyses. Different classifiers (including classification trees, neural networks and SVM) are then used to classify these features. A ridgelet [11] transform-based method is proposed in [74]. Ridgelet energy and co-occurrence signatures, compressed with PCA, are first used to characterize fingerprint textures and then fed to an ensemble classifier that combines neural network, SVM and k-NN.

Some of the holistic approaches are based on the observation that, when captured by the same sensor, fake samples produce images with lower quality than live ones. Thus, trying to capture these quality differences could result in highly discriminative features. In [1] the texture coarseness is used to highlight the blemishes present in

fakes. In [18] 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 [27], where 25 different quality measures are extracted and classified with a Quadratic Discriminant Analysis approach.

Other holistic approaches are based on textural features extracted from the images. The use of first and second-order statistical features derived from multiresolution texture analysis and the interridge frequency analysis was proposed in [1]. The features are further processed using PCA and Fuzzy c-mean classifier. In [47] authors proposed a method based on band-selective Fourier spectrum. The authors state that ridge-valley texture of the fingerprint produces ring patterns around the center in the Fourier spectral image. These rings show differences between live and fake fingerprint images in the Fourier spectral energies of certain bands. In [73] curvlet energy signatures and corresponding co–occurrences are used to represent fingerprints. First, two features based on texture measures are extracted, namely the curvlet energy signatures and their co-occurrences. After applying feature selection, the resulting vectors are tested independently on three classifiers (Adaboost, SVM, k–NN) and finally an ensamble classifier based on a majority voting rule is created, showing that the feature based on co-occurrences is slightly better. A different method [61] 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 [72] to extract textural anomalies (like structural, orientation, roughness, smoothness and regularity ones) in different image regions. The size of the feature set is reduced by Sequential Forward Floating Selection (SFFS). The resulting vectors are independently tested on three classifiers: neural network, SVM and k–NN. Finally, the two best classifiers are fused using a “sum rule”.

Local Features

Despite the efforts made by researchers, 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 ([37, 38]).

These 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 [37]. Most of these features have been initially designed for coping with different problems in computer vision and quickly found effective applications for fingerprint liveness detection. First attempts were based on Linear Binary Patterns (LBP), a popular descriptor for texture classification tasks. Basic LBPs are invariant to intensity variations and several extensions have been proposed to add more invariant properties. Combining multiple local resolutions was also found effective in improving robustness to scale variations. Examples of other local descriptors borrowed from different image processing tasks are Weber Local Descriptor (WLD), Binary Statistical Image Features (BSIF), 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) [35] 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) [38] is a joint distribution of WLD and LPQ. Convolutional Comparison Pattern (CCP) [34] is a rotation invariant descriptor based on the preliminary segmentation of the fingerprint and on its orientation into a reference direction. Then, per–pixel binary codes computed from DCT coefficients of local patches are summarized into histograms at different local scales, which are finally concatenated.

Feature fusion Approaches

Since the various image features convey different and usually complementary information on the analyzed data, an interesting perspective could be to integrate multiple features in order to obtain better accuracies compared to systems trained on a single representation. 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 [87] combines various image filters, statistic measures and quality indexes. (i.e., pore spacing, residual noise, first order statistics, intensity based, ridge strength, ridge continuity). Two classifiers, Multilayer Perceptron with one hidden layer and SVM, were compared after chaining the features and selecting the most relevant variables with the Sequential Forward Selection algorithm. Results show that SVM performs slightly better.

Finally, the study in [36] 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.

Deep Learning methods

The recent success demonstrated by deep learning approaches and of Convolutional Neural Networks (CNN) in particular in several computer vision tasks, stimulated the interest of researchers into applying them to the liveness detection problem as well. A possible taxonomy of these methods divides them into approaches that use the whole fingerprint images as samples and those who analyze separately different regions (patches) of the image and then combine the different pieces of evidence obtained.

Full image approaches. One of the early works in this direction was [24], which assessed the discriminative power of features extracted from a CNN initialized with random weights. Authors examined the effects of different preprocessing steps such as Gaussian low-pass and high-pass filtering for noise removal, image spatial reduction, region of interest (ROI) detection and data augmentation. PCA and whitening were also applied for dimensionality reduction and normalization of the extracted features. Then, authors compared, using a SVM classifier, the discriminative power of their CNN features and that of standard LBPs, showing the higher performances of the former. Two years later, Nogueira et al extended this work. In [77], they exploited a Transfer Learning (TL) approach by fine-tuning two well-known CNN architectures that were pre-trained on the Image-Net dataset.

These models were used as end-to-end classifiers, showing higher performances with respect to the random-weight CNN and the classical LBP methods analyzed in [24].

A Siamese network was proposed in [62] for learning a distance metric aimed at maximizing the inter-class distance between live and fake samples. The focus of the work is improving sensor interoperability and robustness of the liveness detection module to unseen spoofing materials. Pre–trained CaffeNet and GoogLeNet were used as basic CNN models and extensive experiments in parameter optimization during fine tuning were described.

Another interesting approach has been presented in [66], where authors combined an optimization of the hyperparameters of the CNN architecture with an optimization of the filter weights via back–propagation algorithm to construct *spoofnet*, which is able to greatly improve the results of other state–of–the–art approaches.

Patch–based approaches. One of the first works in this direction was based on a Deep Belief Network (DBN) [51]. The patches are extracted by first detecting a set of salient points in the fingerprint image and then using their average location to select a ROI which is further split into overlapping patches of size 16×16. The network architecture is composed by different layers of Restrict Boltzman Machines (RBM). After a (greedy) unsupervised training of each layer, the whole DBN network is fine-tuned using labeled training data. In testing phase, the DBN outputs of the image patches (considered as posterior probabilities) are averaged and then used to select the final label, according to a threshold learned in the training step.

In [83], authors presented a deep distance metric learning method based on triplet loss embedding. Each image patch is paired with a fake and a live patch. Then, this triplet is transferred into a common latent space where a suitable metric (aimed at maximizing inter-class distances and minimizing intra-class ones) is learned. To this end, three identical CNNs with shared weights are trained by minimizing a suitable loss function based on the distance function to learn. The trained network is then used as an end-to-end patch-based classifier. In order to determine the final image label, the distances with the reference sets of live and fake samples for each fingerprint image are analyzed and combined into a voting system.

In [85], after applying a pixel-wise background subtraction method, based on the mean and variance of the pixel's gray-scale intensity, a grid-wise patching step is applied to divide each fingerprint into non-overlapping regions. Then, different

methods (a patch-based voting, a patch-based with optimal threshold and a non-patch-based CNN) are compared.

Finally, a minutiae–based approach was proposed in [16]. In this work, patches with size 96×96 are extracted from the areas of the fingerprint image centered on the extracted minutiae points. The obtained patches are then fed to a CNN standard model (Mobilenet v1), which outputs a class probability. The final image label is computed with an average voting over all the image patches.

2.4 Experimental data

The Fingerprint Liveness Detection Competition (LivDet) [28] is a challenge organized by the Departments of Electrical and Electronic Engineering of the University of Cagliari and Clarkson University. The purpose of this competition is to assess the achievements of the state of the art in fingerprint liveness detection and it is open to academic and industrial institutions. The first edition was held in 2009 [63] and actually it takes place every two years.

The datasets collected in the various editions (three in 2009 and four in each of the following) have been made publicly available. As a result, they have been largely used in the literature and, thus, a work based on them and on the LivDet experimental protocol enable a comparison with a great variety of methods.

This observation motivated the use of the LivDet 2009 [63], LivDet 2011 [105] and LivDet 2013 [32] benchmarks in this work. Overall, these benchmarks consist in eleven sets of live and fake fingerprints acquired with different devices, 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 (Fig. 2.2). 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.

Fig. 2.2 Examples of live and fake samples (created with different materials) from the LivDet 2011 datasets.

A detailed characterization of the various datasets used can be found in Table 2.1. According to the standard LivDet protocols, the main parameter adopted for the performance evaluation is the *Average Classification Error* (ACE), which is the average between the percentage of misclassified live (*ferrlive*) and fake (*ferrfake*) fingerprints, i.e. ACE $= \frac{ferrlive+ferrfake}{2}$.

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	640x480	208x1500
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

Table 2.1 Characteristics of the datasets used in the experiments.

Chapter 3

Feature Fusion Approaches

The first general approach introduced in this Section refers to the analysis of the contribution of feature–fusion (or multi–view) methods to the liveness detection problem. Here, I briefly recall the rationale of this approach. Most of the initial software methods implemented relied on the analysis of different individual image features extracted from the input samples. These "hand-crafted" features have been engineered on the basis of an expert knowledge of the problem under analysis. As another characteristic, each of these features highlights a peculiar aspect of the analyzed data or, in other words, it offers a different view under which a sample is examined. These features have their own descriptive power and are often complementary one with each other. The main idea behind multi–view approaches is to try exploiting this complementarity and allow the different features to mutually support each other (by leveraging on their strengths and, at the same time, softening their weaknesses).

As we stated in the Introduction, while this general approach has been largely investigated in several computer vision and machine learning tasks, it has been relatively overlooked in the area of fingerprint liveness detection. Based on these preliminary observations, the rationale of the work described in this section 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 are the most effective methods for combining different views? Which are the most suitable views for such integration? Are multi–view approaches, in the specific problem, indeed capable of improving the performances of single–view approaches? Are they competitive with the current state–of–the art?

Since an exhaustive assessment of all the available features and feature fusion methods would have been clearly unfeasible, the approach followed to reply to the previous research question has been to (i) select, on the basis of the results discussed in the literature, a subset of promising features, and (ii) compare various methods capable of dealing, from a number of different perspectives, with the main 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).

The main contribution of this work can be summarized as follows:

- it extends the analysis of multi–view learning approaches discussed in the literature by providing an in–depth comparison of different feature–fusion approaches;
- it introduces *Spidernet*, a novel two-stage deep neural architecture capable of effectively combining different general image descriptors;
- it demonstrates the superiority of multi–view approaches compared to single–view ones;
- it shows that the proposed methods (and feature combinations) are indeed effective, capable of generalizing well across different datasets and, most of all, of obtaining results comparable with (if not superior to) the current state of the art.

The preliminary version of this work has been presented at the CIARP 2015 conference [102], while its extended version has been published in [101].

The rest of this Chapter is organized as follows. First, the image features considered in this work are detailed in Section 3.1, along with the rationale for picking them. Then, the various multi–view learning approaches selected for research are introduced in Section 3.2. Finally, Section 3.3 presents and discusses the experimental results.

3.1 Image Features

Given the demonstrated relevance of micro–textural and rich local features to the fingerprint liveness detection problem ([37]), this research is focused on them.

Speaking in general, features in these two classes are characterized, among many other traits, by different invariant properties (e.g. to illumination, scale, rotation, translation, blur and so on). When features have to be considered individually, it is clear that the more invariance they express the better it is. However, in a multi–view approach, this issue is less pressing since it can be expected that the mutual contribution of different features helps overcoming their individual limitations.

Thus, rather than using high invariance as a constraint, the selection process was mainly based on the discriminative power of the features, which was inferred from both the results available in the literature and those of preliminary tests conducted. Another selection criterion was the possibility to apply a descriptor to all the experimental benchmarks and with the same experimental settings. For instance, CCP [34] was ruled out since it requires a prior fingerprint segmentation.

In the following, the selected features for each of the two main categories are shortly described.

3.1.1 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, which summarize the result of the local analysis and are 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 bits used to compute them.

Several results in the literature ([14, 78, 79, 12]) show that when micro–textural features are computed at different scales (i.e. using different size of the local patches) and combined, better results are obtained with respect to the same features computed at a single resolution. Hence, when practicable, this option was considered as well (see Section 3.3.2).

Co-occurrence of Adjacent LBPs (CoALBP)

In the context of local textural features, one of the most famous descriptor is the LBP. This light weight image descriptor was initially introduced back in [80] specifically for general texture classification. In the original version, LBP descriptor encodes

the gray intensity variations of adjacent pixels placed in a circle (which is called a micro-pattern) centering each image pixel into a binary bit vector by means of thresholding the gray value intensity of each neighbor pixel around the center, and then summarize the occurrence of resulting binary patterns throughout the image by means of histograms. This is a very simple yet powerful textural descriptor, whose main advantage is its invariance to illumination changes.

However, one of the main drawbacks of the original LBP formulation is that it lacks structural information. To overcome this issue, in [78] the authors introduced a new variant where the co-occurrence among multiple LBPs (and in particular, among adjacent LBPs) is measured. CoALBP results in a high-dimensional feature vector that has been found to provide a better texture characterization compared to previous LBPs. A single-scale histogram has size 1.024. Preliminary test showed that a three-scale version (resulting in a vector of size 3072) was consistently providing better results than the single-scale one.

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 proposed by [79] 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 and (again) only a multi-scale version of RICLBP, with three scales and a final dimension of 408 elements, was used.

Weber Local Descriptor (WLD)

WLD is a dense local image descriptor, which has shown good performances on texture classification and face detection [14] and, more recently, on face recognition too [42]. WLD is based on the Weber's law, which states that the difference in a stimulus can be perceived only if the ratio between this difference and the original stimulus exceeds a certain threshold. WLD 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, encoded into a histogram of 960 elements. As with the previous features, only its multi-scale version, computed at three different resolutions, was considered.

Local Phase Quantization (LPQ)

LPQ is an operator originally proposed for texture classification [81] and later applied to face recognition tasks, achieving complementary performance to other descriptors on challenging datasets ([3, 39, 12]). The main property of this operator is its invariance to centrally symmetric blur, such as the one caused by linear motion and out of focus. Furthermore, it is contrast and illumination invariant.

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 computing, in a neighborhood of each image pixel, the phase of the 2D Short Term Fourier Transform (STFT). To maintain a compact representation, only the quantized phase of selected frequency components are extracted. Each quantized LPQ code is then encoded as an 8-bit digit and, thus, the final LPQ descriptor has size 256.

As shown in [12], a multi-scale approach helps 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. Thus, different scales were tested in our experiments.

Rotation–Invariant version of LPQ (RILPQ)

LPQs can be further improved by adding rotation invariance. As shown in [82], 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.

Local Contrast Phase Descriptors (LCPD)

This descriptor has been expressly proposed in [38] to tackle the fingerprint liveness detection problem. The idea behind LCPD is, basically, to combine the best characteristics of WLD and LPQ. WLD is characterized by two components related to contrast and orientation. In LCPD, the contrast component is first computed with a LoG (Laplacian of Gaussian) operator, which helps better dealing with the intrinsic noise of fingerprint images. Then, contrast values are quantized on N levels. The orientation component is computed with the RiLPQ descriptor, which guarantees higher robustness to noise and image rotation with respect to the gradient used in WLD. The final LCPD descriptor has size $N \times 256$, where the fixed value $N = 8$ was selected for all devices and datasets according to the suggestions in [37]. Different scales where tested and, possibly, combined.

Binary Statistical Image Features (BSIF)

BSIF [49] 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 is a relevant property that 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, which also prevent the necessity to fine-tune their parameters for each application. In order to allow the combination of different scales, 10 bits were used to represent the binary codes, resulting in an histogram of size 1.024 for each scale.

3.1.2 Rich (Dense) Local Descriptors

Compared with micro-textural features, these descriptors provide a much stronger characterization of the local image patches, which makes them better suited for tasks like image registration, object tracking and recognition. Furthermore, the extracted features are usually invariant to changes in image scale, noise and illumination. To improve their distinctiveness, they are often coupled with a feature-specific keypoint

detector. These detectors return a local measure of the feature uniqueness and usually promote high-contrast regions of the image, such as object corners.

Robust matching algorithm can then be applied to pair keypoint descriptors obtained from different images and, thus, to obtain the required registration information. When these descriptors are used for tasks like 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 is usually obtained creating a vocabulary of *visual words* by clustering training samples with vector quantization approaches and then using bag-of-words (BoW) models.

The following paragraphs provide, for the sake of brevity, a short description of the rich local descriptors used in the experiments. The interested readers are referred to the referenced articles for more details. It should be highlighted that, in all cases involving a BoW representation, this was obtained using a base of 600 codevectors, which were computed from 30 random train images picked from both live and fake sets and different spoofing materials.

Scale Invariant Feature Transform (SIFT)

SIFT [58] is a popular local image descriptor in Computer Vision for several tasks like object recognition, image registration and content-based image retrieval.SIFT works on monochromatic images, it is invariant to uniform scaling and rotation, and partially invariant to affine distortion and illumination changes.

SIFT was computed in both 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 computing robust keypoint correspondences between fingerprint images is virtually impossible. Thus, the extracted descriptors were transformed into a normalized histogram using again a BoW approach.

Dense-SIFT (DSIFT) were obtained computing a descriptor for each image pixel. It should be noted that DSIFT does not guarantee scale-invariance, since the descriptor scale, whose optimal value is obtained by applying the SIFT keypoint detector, is fixed beforehand.

DAISY

This is a descriptor specifically designed to be extracted in a pixel-wise dense way [100]. While providing distinctiveness and robustness properties similar to the SIFT ones, except for a greater robustness to rotation, 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 shape of the flower.

Scale–Invariant Descriptor (SID)

One of main issues with rich local descriptors is the way they deal with scale changes, which requires 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 [53] 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.

As for the experiments, SID was computed in a dense way, extracting a descriptor per pixel.

3.2 Feature fusion approaches to fingerprint liveness detection

After having introduced in the previous Section the various image features that we deemed interesting to investigate in the context of fingerprint liveness detection, this Section focuses on describing different multiview learning methods that can leverage on these data to improve (i) the classification accuracy and (ii) its generalization capabilities. Clearly, an exhaustive analysis of the multiview approaches would be

too broad for this work. Thus, the approach followed aimed at exploring the different challenges that appear in the multiview learning process, such as:

- when combining multiple features, which is the best strategy to follow between *early* fusion (i.e., fusion at feature level, Sections 3.2.2, 3.2.3, 3.2.5) and *late* fusion (i.e., fusion at decision level 3.2.4)?
- given that different features have different characteristics and belong to different representational spaces, how can we harmonize or normalize them (3.2.2, 3.2.3, 3.2.5)?
- since combining multiple views increases the number of variables, which data reduction techniques, such as feature selection (3.2.2, 3.2.4) or subspace transformations (3.2.3, 3.2.5), 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 (3.2.5)?

The discussion section will try to provide answers to these questions on the basis of the experimental results. For the sake of clarity, it should be underlined that in the following the terms features and views, feature fusion approaches and multi–view learning will be used interchangeably.

3.2.1 Notation

The notation that will be used throughout this section is the following. Let $y = \{y^1, \ldots, y^K\}$ be 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 $m = \sum_{k=1}^{K} m_k$.

The training set is defined 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, $X_j^k \in \mathbb{R}^{m_k \times n_{jk}}$).

3.2.2 Feature Chaining

A simple but effective way of combining multiple representations of the same sample is to concatenate the characteristic vector of each representation. Hence, $y = (y^1, \ldots, y^K)$ denote a test sample in this case. 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 ([37, 34, 38, 36, 66, 29, 30]). Before classification, as suggested in [13], each feature variable in y is linearly scaled according to the factors used to scale the same variable in the training set to the range $[0, 1]$; this avoids the variables in larger scales to dominate those in smaller ranges.

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 [104] 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 preliminary tests the effectiveness of an additional feature selection step based on the Relief algorithm [52] was tested as well.

3.2.3 Multi–view Discriminant Analysis (MvDA)

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 3.1).

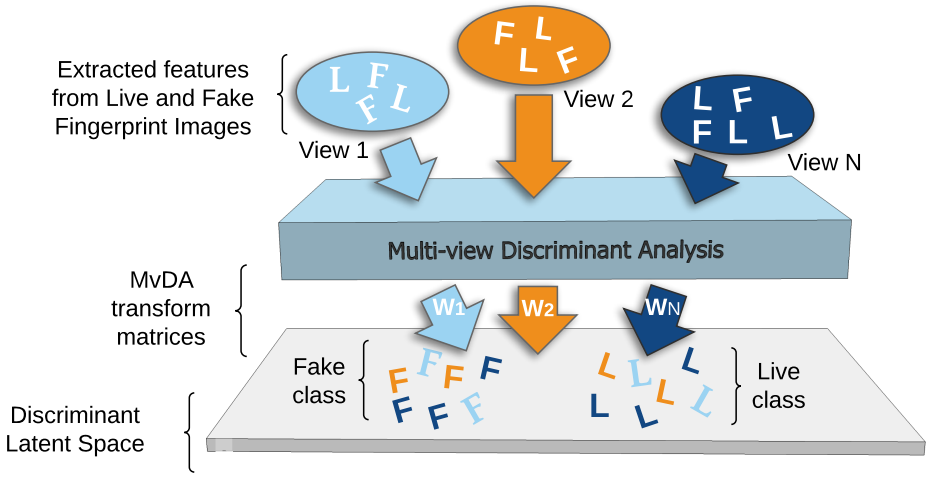


Fig. 3.1 Overview of MvDA method. The different views (i.e. features) extracted from fingerprint images are projected into a discriminant common latent space by computing a proper linear transformation for each view. Here, samples from different views are represented with distinct colors and the letters denote the view class (F for fake and L for live).

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. As introduced before, let $X_j^k = \{x_{jki}\}$ be 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:

$$(w_1, \ldots, w_K) = \arg \max_{w_1,\ldots,w_K} \frac{Tr(S_B^l)}{Tr(S_W^l)} \tag{3.1}$$

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

$$S_W^l = \sum_{j=1}^{J} \sum_{k=1}^{K} \sum_{i=1}^{n_{jk}} (l_{jki} - \mu_j)(l_{jki} - \mu_j)^T \tag{3.2}$$

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

$$S_B^l = \sum_{j=1}^{J} n_j(\mu_j - \mu)(\mu_j - \mu)^T \tag{3.3}$$

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.

As a final note, the original MvDA formulation adds the possibility to take advantage of the cross-consistencies between the views. While this could be a relevant feature in general, initial experiments showed that it was not providing substantial contribution and, thus, it was not further taken into consideration.

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, (3.2) and (3.3). Computing S and D requires to first reduce the dimensionality of all the input views to a common dimension, which is done by applying principal component analysis (PCA). Then the trace ratio problem in (3.1) 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.

3.2.4 Multi–view Real AdaBoost

Real Adaboost is an improvement of the original Adaboost algorithm [25], which outperforms the standard formulation in several practical cases and allows an effective combination of different descriptors [69, 70]. 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 [91].

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 liveness detection is a two class problem, it is assumed, without loss of generality, that $c_i \in \{-1, +1\}$. The decision rule on a test sample y is then implemented as:

$$H(y) = sign\left(\sum_{t=1}^{T} h_t(y)\right) \tag{3.4}$$

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, the two 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, are extracted. Such sets are clearly constant for all the iterations.

Finally, the weak classifiers are constructed iteratively 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 $P_g^- = P_\omega(\mathbf{v}_g^-)$; P_g^+ and P_g^- are actually obtained 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:

$$h_g(y) = \frac{1}{2} log\left(\frac{P_g^+(y)+\varepsilon}{P_g^-(y)+\varepsilon}\right) \tag{3.5}$$

   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:

$$Z_g = 1 - \chi^2(P_g^+, P_g^-) \tag{3.6}$$

   clearly, Z_g is lower when the two distributions are different (i.e., h_g is more discriminant, Fig. 3.2) 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), the one with the lowest Z measure is picked as weak classifier h_t at step t. 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:

$$\omega_i = \omega_i \cdot e^{-c_i \cdot h_t(x_i)} \tag{3.7}$$

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.

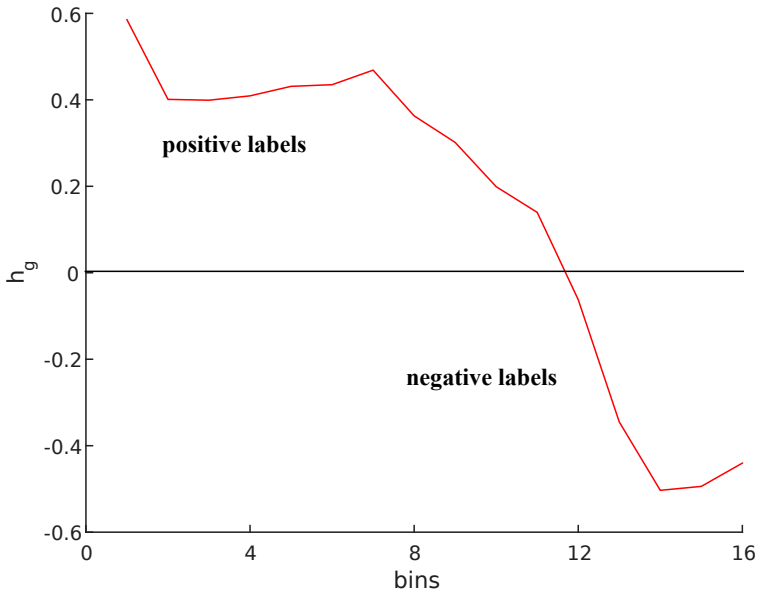
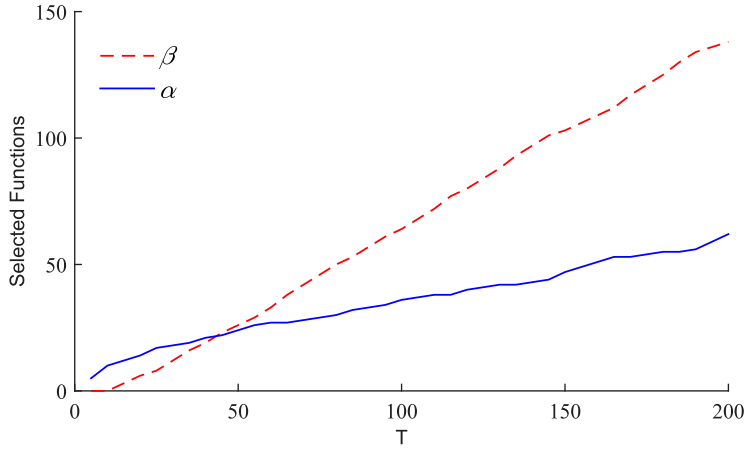


Fig. 3.2 An example of a classification function h_g showing a good discriminant power.

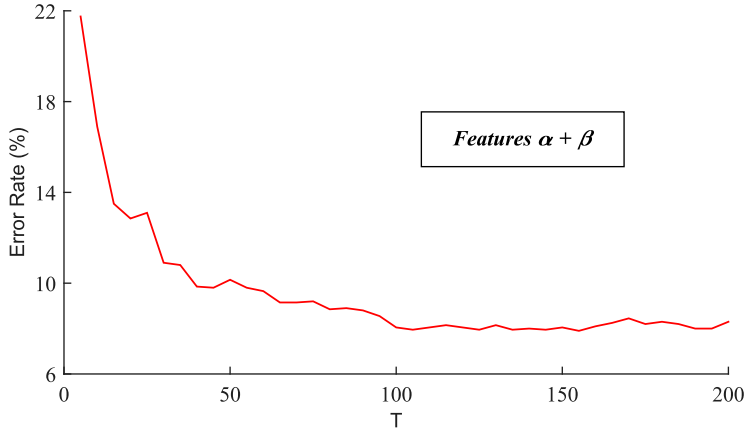
From this description, it is also clear that Real Adaboost performs an (implicit) feature selection, since only the most promising feature variables from the different views are included into the final classifier.

An example of the algorithm behavior is shown in Fig. 3.3(a), which plots the per-view number of classifiers at each iteration for a two views (α and β) experiment. It can be seen that, in this example, classification functions from feature α were selected in the first iterations while, for higher values of T, β becomes more discriminating and is selected in the majority of times.

As for the implementation details, the only parameter of the algorithm is T, the number of weak classifiers. It was experimentally verified that the algorithm has a similar behavior for different attribute groups, reaching a plateau after a certain number of iterations (see Fig. 3.3(b) for an example). It was also found that the starting value of this plateau is somewhat related to the discriminative power of the different views. This observation allowed the definition of 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).



(a) Number of Classification Functions



(b) Error Rate (%)

Fig. 3.3 (a) number of classification functions per view at each iteration and (b) behavior of the classification error in a two-view experiment for increasing values of T.

3.2.5 *Spidernet*

As a contribution to this comparative study, a novel two–stage Deep Neural Network (DNN) architecture, called *Spidernet*, is proposed. Starting from general image descriptors (fig. 3.4), *Spidernet* 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 over-fitting 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 hyper-parameters (*architecture optimization*).

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) [40]. Stochastic Gradient Descent (SGD) with a decreasing learning rate and momentum is then used to fine–tune the network parameters [41]. 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 and L2 regularization were used to soften the over-fitting issues, which can affect our approach due to the quite limited number of training patterns. Dropout was introduced in [96] as a method to train a robust classifier by randomly

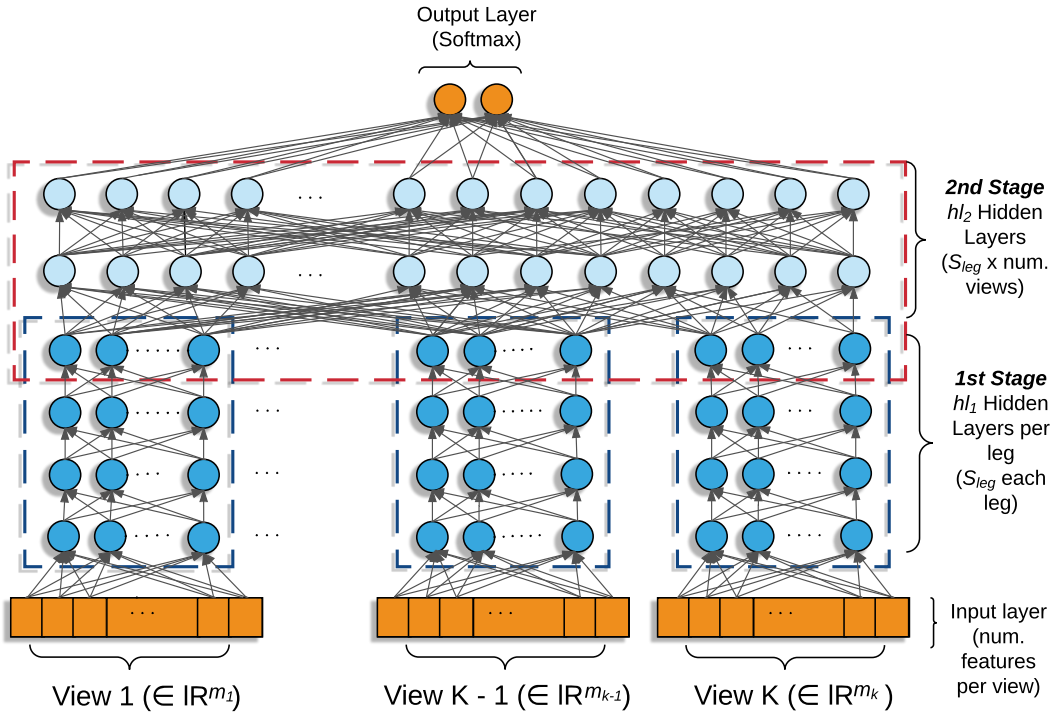


Fig. 3.4 *Spidernet* architecture. In the first stage, each view is independently processed by a network "leg" and projected into a common latent space. In the second stage, the transformed views are first combined and then classified.

removing, at each iteration, some of the neural network units. An effective dropout strategy (proposed in [23]) was used, with an initial dropout of 0.5 decreased at each epoch. The coefficient for L2 regularization was set to 0.001.

Architecture optimization

The optimization of the hl_1, hl_2 and s_{leg} hyper-parameters was carried out in the following way. For each benchmark, the training samples were equally divided 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, for each parameter, a variation interval was defined (in details, $hl_1 \in [2,4]$, $hl_2 \in [1,3]$ and $s_{leg} \in [30,200]$ with a step size of ten). Finally, the choice of the optimal architecture was based on the value provided by a functional built upon 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.

3.3 Experimental Results

The following subsections describe the experimental results. First, the baselines used to assess the results are introduced (Section 3.3). Then, the strategy used to identify the optimal feature combinations is described (Section 3.3.1). Finally, the experimental results are discussed (Section 3.3.2).

Baselines

Different baselines, summarized in Table 3.1, were used to assess our results[1]. 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, a reference to the best scoring attribute is provided in the table. The second baseline (SOA)

[1] I recall that the parameter adopted for the performance evaluation is the *Average Classification Error* (ACE), as defined in Section 2.4

collects the "best of the best results" selected from any approach following the LivDet experimental protocol. This second baseline determines if the proposed approach are indeed competitive against the state–of–the–art. Alternatively, it can highlight limitations of our work.

Since some authors (e.g. [37]) discarded the Crossmatch 2013 dataset, due to its generalization problems [32], for fair comparison, two different average results for each baseline were considered: *Avg*, which includes all eleven datasets, and Avg_{XM^-}, which rules out Crossmatch 2013.

As for these baselines, we stress the fact that they were compiled when the experiments were conducted (i.e., between late 2015 and beginning of 2016) and, thus, they reflect the state–of–the–art of that period. IF we consider the actual scenario, while no significant differences can be reported for *Baseline*, there have been recent advances (such as [83, 16]) that led to an improvement of SOA for some of the datasets, mainly for those included in the LivDet 2011 edition. Despite that, in this dissertation I deemed preferable (and fair) to base the discussion on the results available at the time the work was conceived. In the general conclusions (Chapter 5), I will further discuss this point.

Image Pre–processing and Feature Extraction

For each benchmark, all the attributes described in Section 3.1 were extracted 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 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' hyper-parameters, each combination of dataset, feature and classifier would also have required the preprocessing pipeline to be optimized. Second, the main target was to provide a fair comparison with previous approaches, most of which did not rely on any pre–processing step.

The only exception is Crossmatch 2013. The initial tests showed the same generalization problems experienced by other authors. However, as carried out in other works (e.g. [24, 66]), 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.

3.3.1 Selecting optimal feature groups

One of the initial research questions was trying to understand which are the most suitable combinations of attributes for the compared algorithms. In particular, the proposed approach aims 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 it will be discussed in the following. The rational of this 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 attribute groups were initially created with what appeared to be the most appealing views. Then, several variations were checked, 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, the inclusion of "similar" features (e.g., KSIFT and DSIFT, CoALBP and RICLBP, LPQ and RILPQ) in the same group was strictly avoided. 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 to spot common trends in the results. As it will be shown in more detail in Section 3.3.2, the different approaches obtain slightly different results, but their error variations are strongly consistent. Basically, this finding allowed to increase the number of SVM–based tests and to reduce the number of validations required, which involve more computationally demanding methods.

Since groups can have both fixed (non–parametric) and parametric views, whose parameters were included in the hyperparameters to be optimized, a set of *"families"* was identified. 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, the optimal ones were roughly identified according to the results achieved by the classifiers introduced in Section 3.2. Since the main interest was finding groups that showed good and coherent behavior, the mean of the average error over all the benchmarks for each classifier was chosen as the ranking score. Then, since in preliminary experiments it was noticed that the scores of the family members varied in a small range, the optimal families were selected according to the average score of a small number (usually five) of randomly picked members.

Finally, given the optimal families, all their members were analyzed to select the one with lowest error as representative.

3.3.2 Results and discussion

The experimental results are summarized in Table 3.1, which, for the sake of brevity, reports only the representatives of the best six families. The table is divided into blocks, where each block contains 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, all results with a statistically significant difference ($p < 0.05$) with *Baseline* and SOA are marked with, respectively, "$*$" or "†". Finally, the table lists for each group 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.

Dataset	*LivDet2009*			*LivDet2011*				*LivDet2013*					
Sensors	*Biom.*	*XMatch*	*Identix*	*Biom.*	*Digital*	*Italdata*	*Sagem*	*Biom.*	*XMatch*	*Italdata*	*Swipe*	*Avg*	Avg_{XM-}
Baselines													
Baseline	1.0 LCPD [37]	3.3 SID [37]	0.5 KSIFT [37]	4.9 LCPD [38]	2.0 SID [37]	11.0 LCPD [38]	2.7 LCPD [38]	1.1 BSIF [37]	17.5 CCP [34]	1.3 LCPD [37]	2.8 DAISY [37]	4.4	3.1
SOA	0.3 WLD+LPQ [36]	1.8 Aug CNN [24]	0.5 KSIFT [37]	4.9 LCPD [38]	1.9 CNN [24]	5.1 CNN [24]	2.7 LCPD [38]	0.2 spoofnet [66]	1.8 spoofnet [66]	0.1 spoofnet [66]	0.9 spoofnet [66]	1.8	1.8
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*** †

Table 3.1 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.

Features

A first remark concerns the effects of the feature parameters in a multi–view setting. A multiscale version with three scales for CoALBP, RICLBP and WLD was found to be the best option in all the cases (as also suggested in [78, 79, 42]).

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, an optimal formulation for all cases could not be found. In particular, it seems that the interaction with other features requires different settings. Thus, a two scale version was chosen, optimizing the scales according to the characteristics of the view group.

A similar behavior was expected for RILPQ. On the contrary, it was 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, two possible explanations can be suggested. 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 over-fitting issues.

BSIFs were also tested in both single and multi-scale versions. When tested individually, a multi-scale 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.

Experimental results offer as well some insights 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 key-points. However, the combination of noisy images together with the choice to not discard the background might have led to a non optimal choice of the key-points. 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).

Groups

All the best families in Table 3.1 are based on a common core, i.e. the combination of SID, RICLBP, LCPD and DSIFT, which is also represented by group G6. Adding more features to this core resulted in 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. Consistent accuracy drops (i.e. an average relative ACE increase ranging from 3% to 65%) were also experienced 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, it can be noticed 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 be inferred from the characteristics of these individual features?

On the basis of the results in [37], it can be seen 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 the results. As a demonstration, when the best four (average) ranking features in [37] (SID, LCPD, BSIF, CoALBP) are combined, the relative increase of ACE is 65%.

One possible explanation is 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

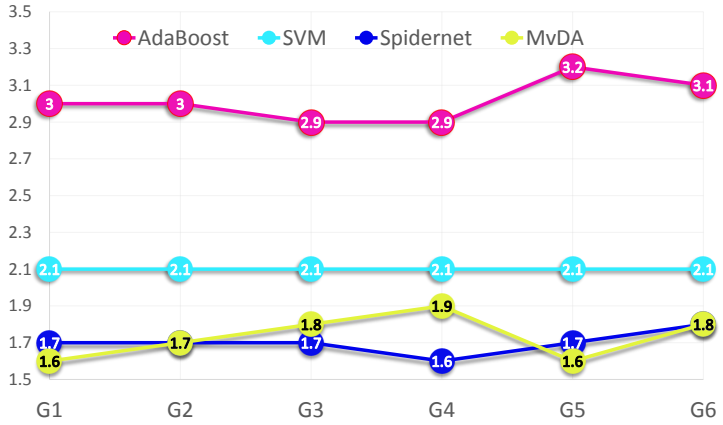


Fig. 3.5 Average classification errors (ACE) of the classifiers on the different groups described in Table 3.1.

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 multi-scale 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.

Methods

As an initial remark, the four compared methods behave consistently across the different groups, as can be appreciated from the diagram in Fig. 3.5. 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 MvDA on G1 is considered as the optimal model 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, it should be underlined that this difference is not statistically relevant.

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 over-fitting 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). However, it is likely 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 the chosen 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, the experimental results allows to provide a preliminary answer to the questions raised in Section 3.2. It should be underlined that, given the limited number of different approaches compared, further work has to be done to achieve more solid conclusions. Based on these premises, the 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, the idea to exploit deep learning approaches to automatically learn how to aggregate different features (*Spidernet*) seems to be an effective feature fusion method.

Multiple features vs. individual features

The main research question of this work was to verify the effectiveness of feature fusion approaches compared to the ones based on individual features. At least three facts allows 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 the results averaged over all the benchmarks are compared with those reported in [37], the best approach (MvDA on group G1) significantly outperforms systems trained with a single feature;
- the best model outperforms, on average, the *Baseline*.

Indeed, it can be observed 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 3.1). 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, the proposed combinations of features appear to generalize well across different experimental conditions and different sensors. It is worth noting that, using different feature groups, the results for the Biometrika 2009 dataset, which scored beyond the baseline, can be improved as well. However, this would result in higher errors over different datasets.[2]

Feature fusion approaches vs. state–of–the–art

As for the assessment of this work against the state–of–the–art, taking into account the general consideration made in Section 3.3, it can be seen that our average results are comparable with SOA and that, in the optimal case, it improves the baseline in 6 out of 11 benchmarks. We can also observe that, while we outperform the CNN–based approach in [24] in all benchmarks except Crossmatch 2013, the *spoofnet* CNN of [66], 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 [66] exploits dataset specific image pre–processing techniques, including image cropping and data augmentation that could also benefit the proposed methods (although they were not applied for the reasons explained in Section 3.3). Furthermore, it should be also noted that, while the relative improvement of *spoofnet* compared to the best result obtained looks relevant, if Crossmatch 2013 exclude, 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.

Cross–dataset evaluation

Finally, it is interesting to test 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

[2] 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 [36]).

	Test Set			
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**

Table 3.2 Cross-sensor interoperability results (obtained on the LivDet 2011 datasets with MvDA and group G1).

characteristics of the capture devices. For this analysis,cross-datasets experiments on the LivDet 2011 datasets can be performed according to the experimental protocol defined in [31] and [46]: train a classifier with the training set of sensor A (e.g. Biometrika2011) and then classy the test set of sensor B (e.g. Italdata2011).

The results are summarized in the Table 3.2 where, for the sake of conciseness, only the ACE obtained with MvDA and group G1 are reported (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 [31] (where the individual contributions of LBP, LPQ and BSIF were analyzed) and [46] (which uses Multi-Scale LBP as features). However, they also confirm other results available on cross-dataset experiments ([64, 4, 62]), which clearly show that the interoperability among different sensors is still an open issue [31].

3.4 Conclusion

This Chapter investigated the effectiveness of feature fusion approaches for fingerprint liveness detection tasks. It addressed the issue of selecting a good set of complementary features, and it assessed the capabilities of different classifiers over a wide set of publicly available datasets, comparing the results obtained with that of both single–view approaches and state–of–the–art techniques.

The experimental results described in Section 3.3 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, a consistent improvement has been found 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, even those based on CNN and/or relying 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, suggesting that subspace transformation methods are the best suited for the problem in analysis. 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, the experimental data stress the interesting results obtained by the DNN architecture proposed with *Spidernet*. Even though it did not outperformed MvDA in these tests, additional experiments show that its architecture has the potential to provide higher accuracy whenever larger training sets are available.

Chapter 4

CNN Patch–Based Approaches

Chapter 3 analyzed the contribution of different multi–view methods applied to various handcrafted image features. This Chapter aims at tackling the same fusion approach under a slightly different perspective.

First of all, the contribution of a patch–based analysis to the fingerprint liveness detection problem is investigated. In other words, rather than combining different features extracted from an individual image, the final decision is taken gathering together different pieces of (local) information collected from various parts of the fingerprint image.

Second, this work aims as well at leveraging on the recognized capabilities of Convolutional Neural Networks (CNN) in tackling different complex computer vision tasks. However, in order to be effective as classification or feature extraction tools, these models need to be trained on a huge amount of data. This can be a problem when they have to be applied to novel tasks that lack sufficient training data. A possible solution is to rely on Transfer Learning (TL) approaches, whose rationale is to exploit the knowledge learned while solving a problem and apply it to a different context. When applied to convolutional neural networks, the common transfer learning strategy starts from picking deep models that were pre–trained on ImageNet dataset [90] and then fine–tunning them to the novel task. The rationale of this procedure is that ImageNet dataset contains millions of natural images that include objects belonging to 1.000 different categories (like animals, vehicles, buildings and so on). Therefore, models trained on this dataset are capable of extracting high level features that are general enough to be "adaptable" to novel vision problems.

Thus, the aim of this work is to assess the contribution to a patch–based TL approach of different well–known existing CNN models, such as AlexNet [54], VGG [95] and GoogLeNet [97]. While similar methods have been already investigated for the liveness detection problem, previous works were all focused on the analysis of the whole fingerprint image [66]. Thus, a complete evaluation of their capabilities in the context of a patch based approach is missing in the literature, a gap that this work tries to fill.

Going into details, the proposed approach is first based on subdividing fingerprint images into non–overlapping patches, after a preliminary segmentation step aimed at discarding (noisy) background information. The patches extracted from the training set are used to adapt the various reference models to the liveness detection problem. Then, the test patches are fed to the fine–tuned CNN models and, finally, various ways of combining the outputs of the convolutional networks in order to get the fingerprint image labels are analyzed and compared.

The rationale of this approach is threefold. First, since the dimension of the network input layer is necessarily limited (and usually much lower than that of fingerprint images), using small sized patches allows avoiding to resize the samples and, thus, to retain the original image resolution and information. Second, using patches as samples rather than the full images allows increasing the size of the training set, thus (hopefully) making the classifier more robust and increasing its generalization capabilities. Third, working at patch level allows again exploiting fusion approaches, either by combining different features extracted from the patches or different pieces of evidence obtained from various image regions. This can (hopefully) lead to an improvement in the robustness of the final fingerprint classification process.

Concluding, the main contributions of this work can be summarized as follows:

- it thoroughly analyzes (CNN) TL patch-based approaches in the context of fingerprint liveness detection;
- it provides a comparison between full–image and patch–based CNN TL approaches, showing the better classification accuracies of the latter;
- it assesses the proposed methods, showing that they are indeed competitive with the current state–of–the–art.

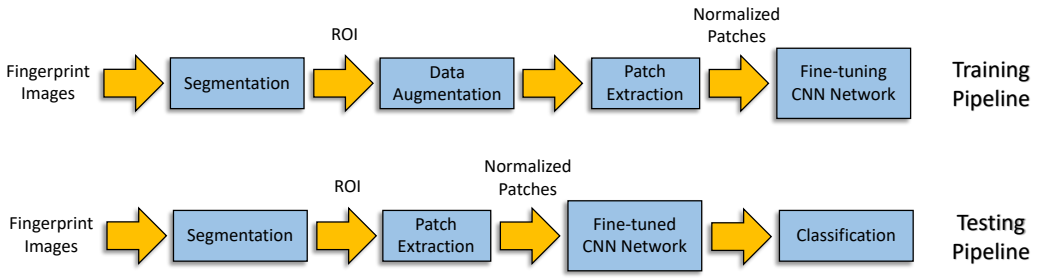


Fig. 4.1 Outline of the proposed fingerprint liveness detection approach

The preliminary version of this work, presented in [103], has been largely extended in this Chapter. In details, the current version analyzes four different models (one in [103]) and different ways of combining the network outputs (again, only one in [103]).

The remainder of this Chapter first introduces the outline of the proposed approach (Section 4.1) and the reference CNN models used (Section 4.2). Then, the different fusion approaches considered are detailed (Section 4.3) and, finally, the results of our experiments are discussed (Section 4.4) before drawing the conclusions.

4.1 Patch–based approaches to fingerprint liveness detection

In brief, the general outline of our approach comprises the following steps:

1. after a preliminary segmentation step, each fingerprint image is divided into small patches containing foreground pixels only;
2. the individual patches of the training fingerprint images are used to fine–tune a pre–trained CNN model (eventually using data augmentation techniques);
3. the individual patches of the test images are fed to the fine–tuned CNN model;
4. the final fingerprint label (i.e., *live* or *fake*) is obtained by combining, at various levels, the outputs of the network for each image patch.

These steps are summarized in Fig. 4.1 and detailed in the following subsections.

4.1.1 Dividing samples into patches

The method starts with a pre–processing phase aimed at both dividing the input sample into patches and removing irrelevant information. To this end, the image background, which is likely to introduce noise in the classification process, is first discarded. Then, the set of small-sized image patches that are fully contained into the foreground is extracted. Finally, the obtained patches are normalized before being processed by the classification pipeline.

Fingerprint segmentation

Fingerprint segmentation is based on the method proposed in [99]. This approach is built upon the preliminary observation that the patterns of fingerprint images have frequencies only in specific bands of the Fourier spectrum. In order to preserve these frequencies, 16 directional sub-bands are obtained by convolving the Fourier transform of the original image with a directional Hilbert transform of a Butterworth bandpass filter. Then, soft-thresholding is applied to remove spurious patterns. Finally, the feature image is binarized and morphological operators are applied to reduce segmentation noise. The method is characterized by a set of hyperparameters that should be fine tuned per benchmark. This is done by optimizing the segmentation error on a small set of manually segmented images (around 30), which are taken from the training set to include both live and fake samples created with different spoofing materials. Some examples of the segmentation results can be seen in Fig. 4.2.

The only exception to this procedure is the Swipe 2013 dataset (see Section 2.4), whose images are obtained by swiping the fingerprint on a linear scanner. In some cases, these images include other finger parts beyond the pulp (the finger extremity). When this happens, the segmentation algorithm might be "attracted" by these parts discarding the pulp. Thus, a slightly different procedure is adopted for these images.

First, the fingerprint is "cleaned" by (i) removing the blank rows at the image bottom and (ii) detecting the beginning and the end of the impressed fingerprint by identifying large peaks of the gradient between consecutive image lines. The resulting region is then processed with the segmentation algorithm and its results are further analyzed. Clearly, a successful segmentation should start at the beginning of this region. If, on the contrary, it starts below a certain line (which was heuristically fixed at the value 300), the starting line of the (incorrectly) segmented area is taken

Fig. 4.2 Examples of segmented fingerprint images from different sensors: (a) Sagem 2011 (b) Italdata 2011 (c) Biometrika 2013 (d) Italdata 2013 (e) Digital 2011 (f) Biometrika 2011 and (g) Swipe 2013

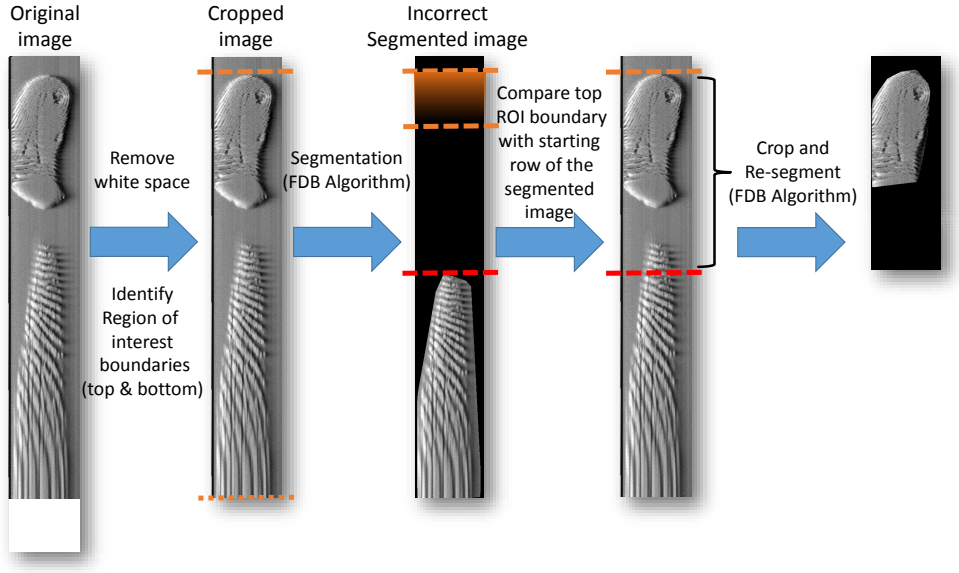


Fig. 4.3 An example showing the segmentation algorithm applied to Swipe 2013 images.

as lower boundary of the actual fingerprint region and the segmentation is applied again to obtain the final foreground mask (see Fig.4.3 for an example).

Patch extraction and normalization

The segmentation mask defines the ROI where the next computation steps are focused. This region is divided into patches of size $w \times w$ pixels, where w is a parameter of the method. In order to avoid any influence of background pixels, only those patches whose pixels are all labeled as foreground are extracted. The algorithm works in the following way.

The ROI is scanned line by line from its top–left corner and each (i, j) pixel is treated as the top–left corner of a candidate patch of size w. If all pixels of this patch are labeled as foreground, the patch is stored and the ROI scan restarts at pixel $(i+w, j)$. When the scan of line j is concluded, if no patches have been found, the scan restarts at line $j+1$, otherwise it restarts at line $j+w$ (see Fig. 4.4).

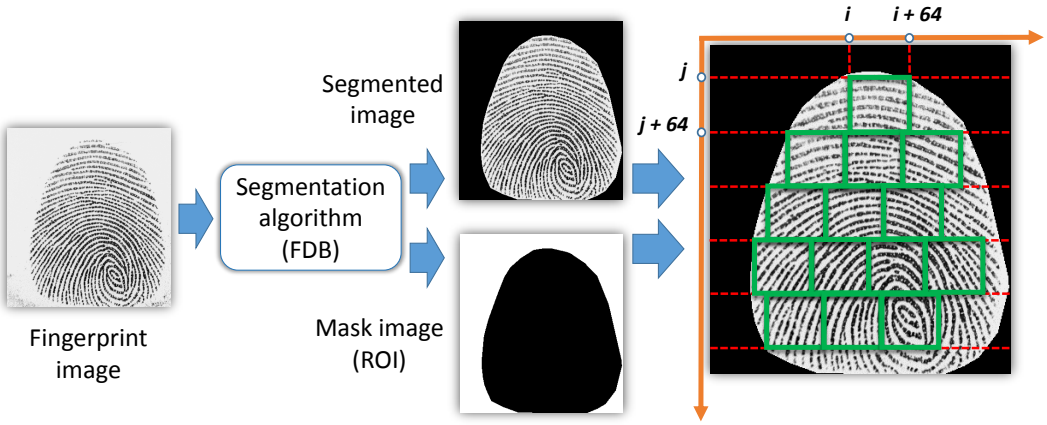


Fig. 4.4 Example of the subdivision in patches of a segmented fingerprint for a patch size $w = 64$.

Finally, each patch is normalized to zero mean and unit variance. It should be underlined that, in the following, if no patches can be extracted from a test sample, the "fake" label is arbitrarily assigned to the fingerprint. This choice derives from the observation that having a false fake is better than a false live, which could result in granting unauthorized access to the system.

4.2 Architectures

Concerning the use of CNNs, as stated in the Introduction of this Chapter, this work explores methods based on Transfer Learning (TL), whose rationale is to exploit the knowledge learned while solving a problem and apply it to a similar problem in a different context. The general approach that was followed is to adapt to the problem at hand models that have demonstrate state–of–the–art performances in a variety of image recognition benchmarks. Specifically, this work focuses on several pre-trained models, like AlexNet [54], VGG [95] and GoogLeNet [97]. Their fine–tuning is performed through a further learning step for few more epochs, which exploits the patches extracted from the fingerprint training datasets as input.

4.2.1 AlexNet

This is a well known model that showed state–of–the–art results in the ILSVRC-2012 competition (whose benchmark was ImageNet). The overall AlexNet model used in our work is substantially equivalent to the one described in [54] and is summarized in Fig. 4.5. In brief, the network architecture contains five convolutional layers, interwoven with three sub sampling layers, followed by three fully–connected layers. The receptive field of each convolutional layer is decreased from 11 in first layer to 5 in the second and 3 in the remaining ones. The network uses Rectified Linear Unit (ReLU) as activation function. The size of the input layer is $w \times w \times 1$, where w is the patch size. In this work, the original 1.000–unit soft–max classification layer (designed to predict 1.000 different classes, [54]) is replaced by a 2–unit soft–max layer, which provides an estimation of posterior probabilities of live and fake classes.

Dropout regularization [96], with probability of 0.5, has been applied to the first two fully connected layers to soften the over-fitting issues. As suggested in [94], Batch Normalization (BN) is applied as well in order to improve the network performances. BN, first proposed in [43], aims at stabilizing the learning process and decreasing the learning rates by reducing the internal covariance shift.

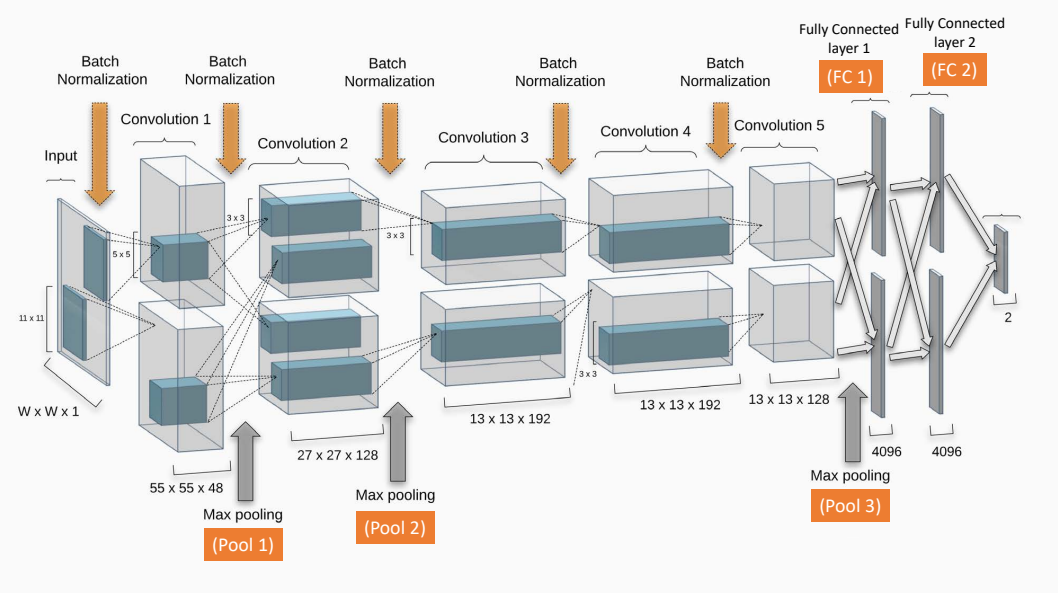


Fig. 4.5 AlexNet-BN Architecture

4.2.2 VGG

VGG network was introduced by Simonyan and Zisserman's in [95] with the aim of improving the original AlexNet architecture. The core idea of VGG is to replace the large perceptive field of the first convolutional layers of AlexNet with multiple cascaded small-sized (3×3) kernel filters. This choice has two effects. First, smaller kernels help to extract image features at a finer grain. Second, the use of a larger number of stacked filters with respect to AlexNet, increases the depth of the network and, thus, it enables the learning of more complex features.

While various versions of VGG achieved the best performance in ILSVRC2014 classification task and the second best in the localization task, it has been shown that they are also able to attain state-of-the-art performances in several image recognition datasets, even when used as feature extractors [68, 57, 76, 67].

The two best performing architectures released by Simonyan and Zisserman are VGG-16 and VGG-19, which, as the name suggests, have 16 and 19 layers, respectively. The original architecture of VGG-16 (VGG-19) accepts 224×224 RGB images as input, followed by a stack of 13 (16) 3×3 convolution layers, with 4 max-pooling layers that divide the convolution layers into 5 convolution blocks. The architecture is followed by 3 fully-connected layers, two of them with the size

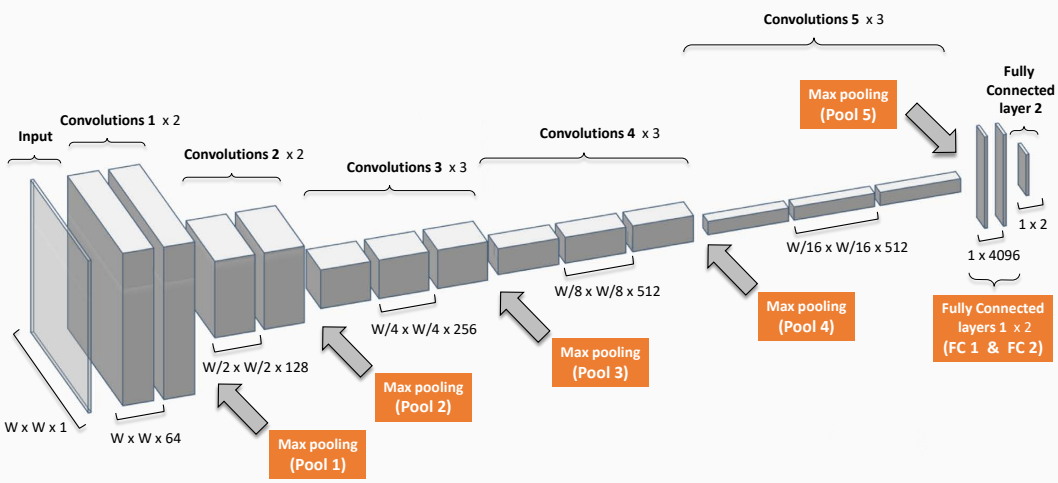


Fig. 4.6 VGG-16 Architecture

of 4096 and the last one with 1000 nodes for classification using softmax activations (See Fig. 4.6). In our work, both VGG versions are considered in order to analyze the effect of network depth on the discriminative strength of the extracted features. As for AlexNet, the size of the input layer was changed to $w \times w \times 1$, where w is the patch size, and that of the last fully connected layer to 2 (the number of classes of the liveness detection problem).

4.2.3 GoogLeNet (Inception V-1)

This deep CNN architecture was proposed in 2014 by Szegedy et al. [97]. It was a successful attempt along the recent trend to increase the network size, both in terms of width and depth, in order to achieve better classification power. However, this idea has two main drawbacks. First, an increased depth typically means a larger number of parameters, which can increase the risks of over-fitting. Second, it also results in huge computational requirements.

The key to tackle both problems was the introduction of a novel module called "*inception*", which is then stacked into a 22 layer network. Inception derives from the Network-in-Network approach proposed in [55]. This module exploits, at the same time, pooling and multiple convolutional filters, whose outputs are concatenated and made available to the following Inception module. In particular, Inception comprises three convolution blocks with a size of 1×1, 3×3 and 5×5 where the last two are preceded by a 1×1 convolution layer for dimensionality reduction(see Fig. 4.7).

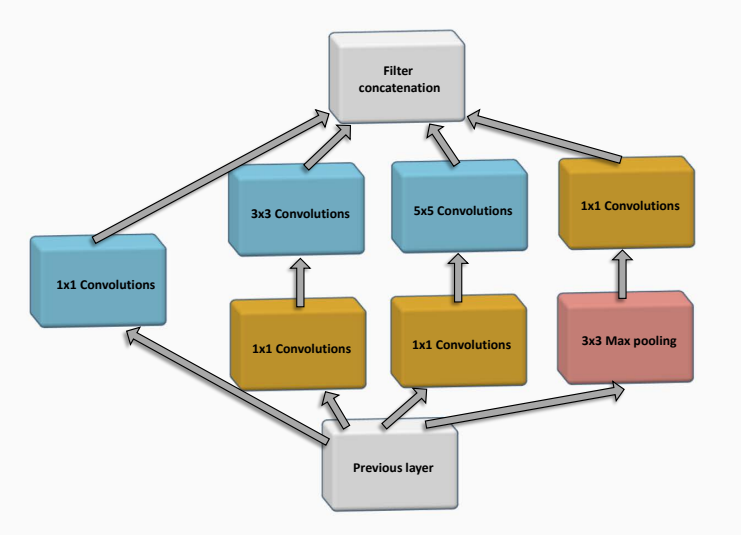


Fig. 4.7 Inception module

This structure, basically, allows benefitting from features extracted at different scales for each input.

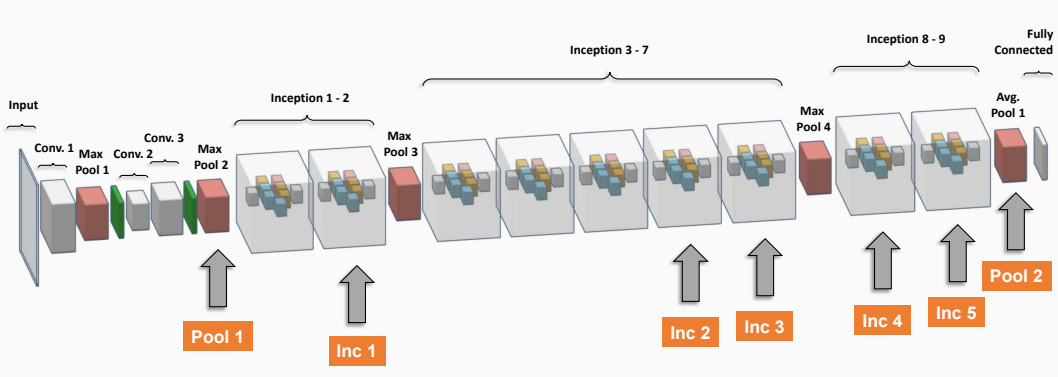


Fig. 4.8 GoogLeNet Architecture

In GoogLeNet, 9 Inception modules are stacked after 2 blocks of traditional convolution and max-pooling layers. Two max-pooling layers divide these 9 modules into three groups including 2, 5 and 2 inception modules respectively. An average pooling along with a dropout layer and a fully connected layer complete the network architecture. ReLU is used as activation function. Finally, a softmax unit is used for classification. As it was done for VGG and AlexNet, the original size of both the input layer and the last fully-connected layer before the softmax unit have been updated in this work.

4.2.4 Data augmentation

Data Augmentation (DA) is exploited during the fine–tuning of the reference CNN models in order to cope with the (relatively) limited amount of training data. DA is a well–known technique that consists in creating synthetic training samples by applying small variations to the original data. In the case of images, such variations are usually obtained by applying various combinations of affine transformations and image cropping [54]. The advantage of DA is that it "forces" the classifier to learn small variations of the input data, thus making it (possibly) more robust to unseen data, and it can also act as a regularizer in preventing over-fitting in deep neural networks [93].

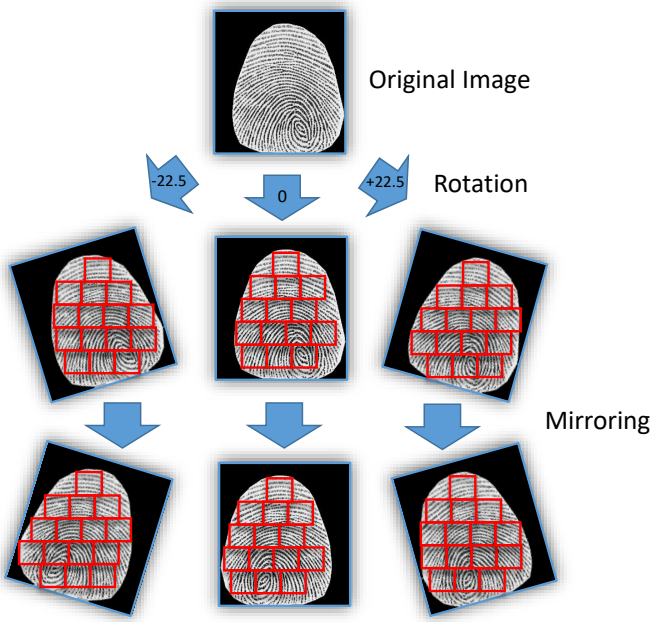


Fig. 4.9 Data Augmentation

In this work, five different variations of each fingerprint image are first created by (i) mirroring the image, (ii) rotating the image by -22.5 and $+22.5$ degrees, and (iii) mirroring the rotated images. Then, after applying the same transformations to the segmentation masks, all augmented version of the input samples are divided in patches according to the process described in Section 4.1.1 (Fig. 4.9).

As a result, the total number of training patches after the DA step is listed, for each benchmark and patch size, in Table 4.1. It should be underlined that the augmentation process is applied to the training set only and not to the test samples.

4.3 Fusion approaches

Once fingerprint patches have been extracted and processed by the reference CNN architectures, two decisions should be taken. First, which network output should be associated with each patch? Second, how the different pieces of information describing the whole patch set of a fingerprint image can be combined in order to obtain the final classification label?

Concerning the network outputs, the CNNs can extract both a probability score and different features, obtained from the intermediate outputs of the inner convolutional layers, each of which capture different characteristics of the data in analysis. Then, according to the type of output chosen to describe the patches, we can apply different strategies to combine them, such as *early* fusion (i.e., fusion at feature level) and *late* fusion (i.e., fusion at decision level).

In the following, the different fusion methods that have been approached in this work are detailed.

4.3.1 Fusion of end–to–end patch scores (E2EF)

Irrespective of the reference network architecture used to process the fingerprint patches, the output of the last layer of the model provides an estimation of posterior probabilities of live and fake classes for each patch, from which it is possible to compute a *patch score*, corresponding to the log–likelihood ratio between live and fake class hyposthes for each patch.

These patch scores can be combined, by averaging them[1], to compute an *image score*. The image score can be interpreted as log–likelihood ratio between live and fake hypotheses, and the image can be labeled by simply comparing the score to a

[1] If patches were independent, the image log–likelihood ratio should be computed from the *sum* of patch scores. However, since patches are correlated, in practice averaging the log–likelihood ratios achieves better results

threshold τ. Theoretically, the optimal accuracy should be obtained by setting $\tau = 0$. In practice, it was experimentally observed that the scores are not well calibrated, i.e., the optimal accuracy is achieved with a different value of τ. In order to tackle this issue, a strategy that has been successfully employed in speaker verification tasks [10] has been adopted to "recalibrate" the scores. The method assumes that the scores for live and fake images can be modeled by means of Gaussian distributions, whose parameters can be estimated on a validation set. Given an image score s, the calibrated score s_{cal} is obtained by computing the following log–likelihood ratio:

$$s_{cal} = \log \frac{\mathcal{N}(s; \mu_L, \sigma_L)}{\mathcal{N}(s; \mu_F, \sigma_F)} \tag{4.1}$$

where μ_L, σ_L and μ_F, σ_F denote the mean and standard deviation for the live and fake uncalibrated scores, respectively. The final sample label is then obtain by comparing the calibrated score s_{cal} with the theoretical threshold $\tau = 0$.

Multi–resolution E2EF (MR-E2EF)

Since the local patch size is likely to impact the liveness detection performances, another possibility is to explore the use of multi–resolution approaches. This is similar to what has been done with micro–textural handcrafted features (Section 3.1.1). Experimental results in Chapter 3 showed that their multi–resolution versions usually performed better than the single resolution ones.

The multi–scale version of E2EF considers jointly different patch sizes for training independent CNN models. These models are used for patch classification. Their scores are fused by first computing a calibrated image score for each size according to equation 4.1. The image scores at different size are then averaged and thresholded to zero.

4.3.2 Deep patch features fusion (DFF)

Convolutional Neural Networks can also be used as feature extractors. In this case, given the fact that features computed at different levels of the architecture highlight different properties of the analyzed samples, a first problem to face is which features to select for the classification.

As reported in the literature, this choice is problem dependent. For instance, in [67] authors compared the discriminative power of features extracted from different layers of VGG-16 for the iris recognition task. While it can be expected that by moving higher in the network's hierarchy the accuracy increases, the best results were achieved from an intermediate layer, while going higher resulted in large drops of the accuracy. A possible explanation of this phenomenon is that filters learned at higher levels start to capture more abstract (high-level) and domain-specific information, while lower layers tend to capture basic level features. As a result, in tasks where there are no extreme differences between samples, higher level features may not be as discriminative as mid-level ones. On the contrary, other works demonstrate that for different computer vision tasks the higher levels are preferable [88, 21].

Since (i) it is not possible to determinate a priori which are the "optimal" layers to pick in the context of this work and (ii) using as candidates all the available ones would result in heavy computational burdens, a subset of the most (potentially) promising layers has been selected for each reference model. The chosen layers have been highlighted in the figures that describe each architecture (Fig. 4.5, 4.6 and 4.8).

For AlexNet and VGG, the features corresponding to the last two fully connected layers (labeled as FC1 and FC2 in the figures) and the pooling layers[2] (labeled as Pool$x$) were picked. The rationale for preferring pooling to convolutional layers is that the former (i) reduce the spatial size of the learned representation behind them while keeping the depth size untouched, (ii) they provide more significant features compared to convolutional layers (as shown in [6]), and (iii) the placement of the pooling layers in the hierarchy of AlexNet and VGG guarantees a good overview of the different transformations the original signals are subject to along the network's depth.

Concerning GoogLeNet, various pooling layers at the early and late stages of the network and different "well-placed" Inception modules (whose outputs have a great descriptivity richness, since they are a concatenation of information learned at multiple resolutions) were picked as feature extractors.

Beside solving the problem of how to pick the optimal features for our case, there are other issues to face. The first is related to the dimensionality of the features that can be extracted at different levels, whose size range from a minimum of 4,096

[2] As for VGG, both VGG-16 and VGG-19 have the same number of pooling layers, which are simply connected, in the two cases, by a different number of convolutional layers

to tens of thousands variables. It is clear that, in these conditions, it is necessary to apply data reduction techniques in order to soften the curse of dimensionality problem. To this end, in this work the extracted features are reduced to a suitable dimension by means of Principal Component Analysis (PCA). However, both the number of samples and the dimensionality of the feature vectors are very large, causing computational and memory problems. In order to address this issue, the EM–PCA algorithm [89] was used. As the name suggests, this method is based on Expectation Maximization algorithm and is detailed in the following.

Let $\mathbf{D}$ denote the $P \times N$ matrix that contains the N samples, each of which is described by a characteristic vector of size P. EM–PCA allows computing the first K principal dimensions of the feature space in $O(KNP)$ time, without requiring the computation of the $O(N^2)$ sample covariance matrix.

The algorithm iteratively estimate the PCA projection matrix through a sequence of Expectation (E) and Maximization (M) steps. In particular, given the current estimate $\mathbf{T}$ of the PCA matrix, the E–step computes a projection of the feature vectors as:

$$\mathbf{Y} = \left(\mathbf{T}^T\mathbf{T}\right)^{-1}\mathbf{T}^T\mathbf{D} \tag{4.2}$$

This matrix can be easily stored in memory, since $\mathbf{Y}$ is a $K \times N$ matrix. Furthermore, $\mathbf{Y}$ can be computed online, i.e. without requiring to store the whole set of sample features. As for the first iteration, $\mathbf{T}$ can be initialized with random values.

Once the new estimate of $\mathbf{Y}$ is computed, the M–step updates the estimate of $\mathbf{T}$ as follows:

$$\mathbf{T}_{new} = \mathbf{D}\mathbf{Y}^T\left(\mathbf{Y}\mathbf{Y}^T\right)^{-1} \tag{4.3}$$

The M–step can be computed dividing $\mathbf{Y}$ in chunks and thus, similarly to the E–step, it does not require to store the whole matrix in memory.

It can be proven that the EM–PCA algorithm converges to the PCA solution [89]. However, preliminary experiments with this method showed a different convergence rate among the K directions. In particular, irrespective of the actual value of K, the ones with lower variance had a much slower convergence. In order to address this issue, the following heuristic is applied. First, the number of eigenvectors to compute is increased by a factor δ (e.g., $\delta = 100$). In this way, the directions showing slow convergence are likely to be all included in the "extra" set of size δ. Then, the termination criterion is defined through a threshold on the difference between the

first K eigenvectors computed in two consecutive iterations. When convergence is reached, only the first K vectors are kept. This heuristic allows (greatly) improving the convergence rate of the algorithm, which usually requires just few iterations.

Once PCA reduced features have been obtained, a last problem to address is how to define a (patch–based) classification approach for telling a live from a fake fingerprint images. In our case, the extracted (and compressed) patch features are first classified with a linear SVM. Then the decision values of the classifier are treated as patch scores and the final image labels is obtained by combining them according to the procedure described in Section 4.3.1 (i.e., first averaging them and then re–calibrating the obtained image scores according to equation 4.1).

Fusion of multiple Deep patch features (MDFF)

Another possibility that has been experimented with in this work, in a way similar to the one proposed in Chapter 3, is the analysis of multi–view methods based on deep features. The rationale is, again, that features extracted from different layers of the networks can be seen as different observations of the samples in analysis, each of which highlights certain peculiar aspects of the data. Thus, another possibility to exploit these pieces of information it is to combine them in order to (possibly) strengthen the robustness and generalization capabilities of the classifier.

In the specific context of this work, a possible multi–view approach can be defined as follows:

- given a benchmark and a reference model, select a suitable subset of layers where features can be extracted;
- reduce the feature dimensions with EM–PCA;
- combine the features in a multi–view classifier.

Given the computational complexity already involved with feature extraction and reduction, in this work only a simple feature chaining followed by a classification with a linear SVM has been used as multi–view classifier. However, it can be underlined that this option, although simple, is not the optimal one, as shown in the experimental section of Chapter 3. The final fingerprint label is obtained, as in the

DFF approach (Section 4.3.2), by averaging and recalibrating the per–patch SVM decision values.

4.4 Experimental Results

The following subsections describe the outcome of the experiments. First, the baselines used to assess the results are introduced (Section 4.4.1). Then, the experimental results obtained by the different approaches described in Section 4.3 are discussed (Sections 4.4.2 and 4.4.3).

4.4.1 Baselines

The experimental results can be assessed comparing them with those obtained, on the same datasets and with the same experimental protocol, with other state–of–the–art deep learning methods.

These methods can be roughly classified along two different dimensions. The first is the type of classifier input. According to this taxonomy, there are methods analyzing either the full image (CIFAR-10 [66], *Spoofnet* [66], CNN–Random [77], AlexNet and VGG-19 [77]) or the individual fingerprint patches (DBN [51], *TripleNet* [83] and *Spoof Buster* [16]). As for the second dimension, there are methods either based on Transfer Learning approaches (CIFAR-10 [66], AlexNet and VGG-19 [77]), or non TL–based ones (*Spoofnet* [66], CNN–Random [77], DBN [51], *TripleNet* [83] and *Spoof Buster* [16]).

As explained in Section 2.4, the metric used for comparison, according to the LivDet protocol ([105, 32]), is the Average Classification Error (ACE). It should be underlined that, while all methods have been tested with LivDet2013, some results are not available for LivDet2011.

Dataset	*LivDet2011*				*LivDet2013*			
Patch size	Biom.	Digital	Italdata	Sagem	Biom.	XMatch	Italdata	Swipe
32×32 patches	605,582	688,225	703,702	748,368	582,306	848,384	643,363	1,454,649
48×48 patches	225,946	264,104	265,867	283,104	216,914	320,821	240,534	549,862
64×64 patches	106,952	123,659	125,344	132,120	99,272	151,142	112,298	256,472

Table 4.1 Total number of patches for each dataset and for $w = 32, 48, 64$.

Dataset	*LivDet2011*				*LivDet2013*			
Method	Biom.	Digital	Italdata	Sagem	Biom.	XMatch	Italdata	Swipe
AlexNet [77]	5.6	4.6	9.1	3.1	1.9	4.7	0.5	4.3
VGG-19 [77]	5.2	3.2	8	1.7	1.8	3.4	0.4	3.7
CIFAR-10 [66]	–	–	–	–	1.5	2.7	2.7	1.3
CNN-Random [77]	8.2	3.6	9.2	4.6	0.8	3.2	2.4	7.6
Spoofnet [66]	–	–	–	–	**0.2**	**1.7**	**0.1**	0.9
DBN [51]	–	–	–	–	1.2	7.0	0.6	2.9
TripletNet [83]	5.2	1.9	5.1	**1.2**	0.7	–	0.5	**0.7**
Spoof Buster [16]	**1.2**	**1.6**	**2.5**	1.4	**0.2**	–	0.3	–

Table 4.2 Baselines. Bold values represent the best accuracies per benchmark.

		LivDet2011				*LivDet2013*			
Nets\Datasets		Biom.	Digital	Italdata	Sagem	Biom.	XMatch	Italdata	Swipe
AlexNet	*E2EF*	$4.0_{/64}$	$3.0_{/48}$	$\mathbf{5.2}_{/48}$	$3.0_{/64}$	$0.4_{/48}$	$\mathbf{5.4}_{/64}$	$0.2_{/48}$	$1.3_{/64}$
	MR-E2EF	$3.8_{/48-64}$	$2.5_{/32-48}$	$\mathbf{5.3}_{/48-64}$	$3.3_{/48-64}$	$\mathbf{0.1}^{*}_{/32-48}$	$\mathbf{7.4}_{/48-64}$	$0.1_{/48-64}$	$4.4_{/48-64}$
VGG-16	*E2EF*	$3.1_{/48}$	$0.9^{*}_{/32}$	$5.9_{/48}$	$2.0_{/64}$	$0.2^{*}_{/48}$	$8.0_{/64}$	$\mathbf{0.1}^{*}_{/32}$	$\mathbf{0.5}^{*}_{/64}$
	MR-E2EF	$3.0_{/32-64}$	$1.0^{*}_{/32-48}$	$5.9_{/48-64}$	$\mathbf{1.7}_{/48-64}$	$\mathbf{0.1}^{*}_{/32-48}$	$8.3_{/48-64}$	$\mathbf{0.1}^{*}_{/48-64}$	$3.4_{/48-64}$
VGG-19	*E2EF*	$\mathbf{2.8}_{/64}$	$1.0^{*}_{/32}$	$5.8_{/64}$	$\mathbf{1.8}_{/64}$	$\mathbf{0.1}^{*}_{/32}$	$10.1_{/64}$	$\mathbf{0.1}^{*}_{/32}$	$0.6^{*}_{/64}$
	MR-E2EF	$\mathbf{2.6}_{/48-64}$	$1.1^{*}_{/32-64}$	$5.5_{/48-64}$	$\mathbf{1.7}_{/32-64}$	$\mathbf{0.1}^{*}_{/48-64}$	$10.3_{/48-64}$	$\mathbf{0.1}^{*}_{/48-64}$	$3.8_{/48-64}$
GoogLeNet	*E2EF*	$4.0_{/64}$	$\mathbf{0.7}^{*}_{/48}$	$7.1_{/48}$	$3.0_{/64}$	$0.3_{/48}$	$15.4_{/32}$	$0.2_{/32}$	$1.2_{/64}$
	MR-E2EF	$4.2_{/48-64}$	$\mathbf{0.8}^{*}_{/all}$	$7.2_{/48-64}$	$3.2_{/all}$	$\mathbf{0.1}^{*}_{/48-64}$	$14.2_{/32-64}$	$\mathbf{0.1}^{*}_{/48-64}$	$\mathbf{1.1}_{/48-64}$

Table 4.3 E2EF and Multi-resolution E2EF results. For each benchmark and model, the ACEs of the two methods are reported (along with the optimal patch size for E2EF). For each method, bold values represent the best accuracies per benchmark. "*" indicates an improvement of (or equivalence with) the state–of–the–art.

4.4.2 Evaluation of E2EF approaches

Before analyzing and assessing the effectiveness of the patch score fusion approach, it should be recalled that the patch size w influences each of the chosen reference models (AlexNet, VGG-16, VGG-19 and GoogLeNet). This parameter controls the granularity of the data, and it is clear that the final accuracies would certainly benefit from a fine–tuning of w per dataset and method. Since training the models is a computational extensive task, only three different values were experimented, namely 32×32, 48×48 and 64×64. The resulting number of patches in the various cases is illustrated in Table 4.1

The "optimal" patch size (per benchmark and model) is heuristically picked by analyzing the classification accuracy computed with a 5-fold cross validation on the

training set, taking care into putting all samples of an individual into the same fold to enforce robustness to cross—individual variations.

Results are summarized in Table 4.3, which reports for each model the final ACE along with the picked w value. A first remark concerns the classification accuracies of the different models. The results show that there is no model outperforming the others on all datasets although, on the average and excluding XMatch 2013, VGG-19 appears to have slightly superior results when compared to the others. Another interesting finding is that these results do not seems to be affected by the depth of the architecture. As a matter of facts, VGG-19 offers slight improvements with respect to VGG-16 (with a top increase of 0.3% in Biometrica 2011, backed by a loss of 0.1% in Swipe 2013 and Italdata 2011) and GoogLeNet performs optimally only in Digital 2011 (while having accuracies comparable with AlexNet, whose depth is much lower).

A second comment relates to the optimal patch size. Analyzing the distribution of w values for each benchmark, it can be seen that, in general, most of the datasets are characterized by a "preferred" size. For instance, the optimal size is the same for all models in two cases (Swipe 2013 and Sagem 2011), the same for all except one in five cases and for only one dataset (Digital 2011) the models are equally divided among two choices (32 and 48). In order to gain better insights into these results, more tests were performed (whose results are not reported in the table for the sake of clarity). For each of the five cases where only one model picked a different size (Biometrica and Italdata 2011, Biometrica, XMatch and Italdata 2013), changing that size to the common one (e.g., using size 64 for VGG-16 and Biometrica 2011) resulted in minimal losses in the accuracies (an average -0.2% if we do not consider XMatch 2013). This is an indication that the selection of a sub–optimal patch size does not have dramatic consequences on the final results.

The comparison with full–image methods shows the effectiveness of our patch–based approach, as confirmed by previous results in the literature (see Table 4.2). This is evident in particular when TL–based methods are compared. As a matter of facts, analyzing the results of both AlexNet and VGG-19, it can be seen that the full image approach is winning merely in Sagem 2011 (-0.6% for AlexNet and -0.1% for VGG-19) and XMatch 2013 (-0.7% for AlexNet and -6.7% for VGG-19) while the proposed patch—based approach shows larger improvements everywhere else (-2.0% on average for both AlexNet and VGG-19). Similar results

can be observed comparing CIFAR-10 results. When compared with other non–TL based approaches, E2EF method is the best performer on almost all the 2013 datasets, with the exception of (again) XMatch, while it is not able to improve state–of–the–art results in LivDet 2011. There, Digital 2011 is the only benchmark where E2EF is the top performer (using GoogLeNet as model). As for Biometrica and Italdata, E2EF improves both CNN-random and *TripleNet*, but *Spoof Buster* has much better accuracy (on average, 2.1% higher than that achieved by the optimal E2EF model). Finally, in Sagem the E2EF approach could only outperform CNN-Random.

In case of multi-resolution E2EF, all possible combinations of the three patch sizes ($32, 48$ and 64) were analyzed. The "optimal" combination was selected based on the 5 fold cross-validation accuracy the training set.

If we analyze the results and compare them with those obtained by E2EF (see Table 4.3), MR–E2EF does not provide any substantial improvement of the results. In general, if XMatch and Swipe 2013 are ruled out, the differences per dataset are minimal and range in the interval $[-0.3\%, 0.3\%]$. It can also be seen that the greatest benefits are obtained by the VGG networks on the 2011 datasets, which are capable of outperforming full image TL approaches even of Sagem, while their results in 2013 are almost identical to E2EF ones, except for Swipe 2013. The large decrease obtained on Swipe 2013 by all models except GoogLeNet, although negative, is another result worth of interest. A possible explanation is related to the low resolution of this device (96 dpi) compared with the ones of the other scanners (higher than 500 dpi). This evidence is somewhat supported by the E2EF results, which shows that all models prefer the largest possible patch size (64×64). In other words, it appears that smaller sizes are not capable of capturing enough fingerprint details to guarantee an accurate analysis and, consequently, satisfactory live and spoof detection rates. On the contrary, the performance of GoogLeNet on this dataset can be somewhat explained in terms of the characteristics of the model, which already incorporates in its analysis the selection of pieces of information extracted at different resolutions.

Concluding, the results of these experiments show that the proposed E2EF approaches, both in their single and multi–resolution versions, (i) improve the accuracies of the same CNN models when they analyze the full fingerprint images, and (ii) are competitive with (when not superior to) the current state–of–the–art. We finally highlight that a possible drawback of the MR–E2EF approach proposed is

related to the facts that, as for E2EF, only a limited set of patch sizes have been analyzed, which might have led to a sub–optimal choice per dataset.

4.4.3 Evaluation of DFF approaches

		LivDet2011				*LivDet2013*			
Nets\Datasets		Biom.	Digital	Italdata	Sagem	Biom.	XMatch	Italdata	Swipe
AlexNet	*DFF*	4.0 FC2-439	1.8 FC2-327	5.7 FC2-386	3.8 FC1-101	0.4 FC1-178	**6.2** FC2-143	0.2 FC2-340	1.4 FC2-141
	MDFF	4.0 Pool5-FC1-FC2	2.0 Pool2-FC2	5.8 Pool5-FC1	3.5 Pool1-FC1	0.4 Pool2-FC1	**6.7** Pool2-FC2	0.2 Pool2-FC2	1.1 Pool5-FC1-FC2
VGG-16	*DFF*	**2.7** FC1-112	0.9* FC2-104	5.8 Pool5-240	**1.6** Pool4-211	0.2* Pool5-101	9.1 Pool5-241	**0.1*** Pool4-129	**0.5*** FC1-450
	MDFF	**2.9** Pool5-FC1	0.9* Pool3-FC2	**5.6** Pool5-FC1-FC2	1.9 Pool4-FC2	0.2* Pool5-FC1	9.1 Pool5-FC2	**0.1*** Pool4-Pool5	**0.5*** FC1-FC2
VGG-19	*DFF*	3.1 Pool5-165	1* FC2-100	**5.4** Pool5-119	1.8 Pool5-114	**0.1*** FC1-101	10.2 Pool5-138	**0.1*** FC2-121	0.6* FC1-104
	MDFF	3.4 Pool4-Pool5-FC1	1* Pool5-FC2	**5.6** Pool5-FC1	**1.8** Pool5-FC2	**0.1*** Pool5-FC2	10 Pool3-FC1	**0.1*** Pool3-FC2	0.6* Pool5-FC1
GoogleNet	*DFF*	3.7 Inc5-288	**0.8*** Inc2-236	6.9 Inc5-155	3.2 Inc3-279	0.3 Inc3-225	14 Inc1-112	**0.1*** Inc1-138	1.4 Pool2-179
	MDFF	3.7 Inc4-Pool2	**0.8*** Inc2-Pool2	6.5 Inc1-Inc5	3.1 Inc3-Inc5-Pool2	0.4 Inc1-Inc5	14.8 Inc2-Inc3	0.2 Inc1-Inc2	1.6 Inc1-Inc5

Table 4.4 DFF and MDFF results. For each benchmark and model, the ACE is reported. Each cell reports as well the optimal layer and the PCA size used to reduce its features (for DFF) and the optimal deep feature group (for MDFF). For each method, bold values represent the best accuracies per benchmark. "*" indicates an improvement of (or equivalence with) the state–of–the–art.

As for the individual patch feature fusion approach, the hyper-parameters of the method are three: the patch size w, the layer used to extract the features and their reduced size. Concerning w, while it should indeed be jointly optimized with the other parameters for each dataset and each architecture, a simpler procedure was followed in order to reduce the computational burden: the optimal sizes were picked as the ones identified during the analysis of the E2EF approach (see Section 4.4.2).

Concerning the other two parameters, they are heuristically computed by analyzing, as before, the patch level accuracies of a 5-fold cross validation (CV) over the training set for different layers and different PCA sizes spanning the interval $[100, 450]$. The combination providing the highest accuracies is then used in the final classifier.

When tackling a multi–view approach, beyond the hyperparameters involved in the DFF method, it is also necessary to select the subset of features to be combined. The algorithm followed to approach this task has been the following:

- given a benchmark and a reference model, compute for each of the selected network layers (see Section 4.3.2) the optimal PCA size for reducing the feature dimensions in the same way explained in Section 4.3.2 (i.e., using a linear SVM classifier with a 5–fold cross validation on the training set);
- sort the features according to their descending accuracy;
- select the first 4 features in the list and create with them groups of n_{feat} features (where $n_{feat} \in [2,3]$ is a parameter of the method);
- select the group providing the optimal 5–fold cross validation accuracy on the training set using the feature chaining approach described in Section 3.2.2;

Results for both DFF and MDFF are summarized in Table 4.4, which reports for each model and benchmark the final ACE along with an indication of the layer and PCA size picked, for DFF, and the feature combination, for MDFF.

Comparing these results with those presented in Section 4.4.2, it can be noticed that there are no large differences in the overall accuracies. Results averaged over all the benchmarks are basically identical and, despite some notable exception (such as AlexNet on Digital 2011 scoring a 1.2% improvement), there are no clear indications that DFF or MDFF approaches are better suited with respect to the previous E2EF methods to address the liveness detection problem.

Concluding, these results confirm the previous conclusions about (i) the effectiveness of patch–based approaches with respect to full–image ones, especially when TL is concerned, and (ii) their capabilities to be competitive with and even superior to the current state–of–the–art (in particular in Digital 2011 and most of the LivDet 2013 datasets).

4.5 Conclusion

This work investigated the effectiveness of the combination of Transfer Learning and patch–based approaches for fingerprint liveness detection tasks. Different well

known CNN reference models were analyzed and different ways of combining their outputs were investigated and assessed through a comparison with state–of–the–art techniques (either full image TL based methods or deep learning approaches analyzing individual patches).

The experimental results described in Section 4.4 support the validity of the initial hypothesis, according to which the analysis of the local features extracted at patch level provides better liveness detection accuracies than that performed on the whole image.

Concerning the different fusion methods analyzed, no one demonstrated its superiority with respect to the others. On the contrary, they all showed somewhat similar accuracies over all the datasets, with some few notable positive and negative exceptions. That said, some general observations can be made.

First, CNN patch–based TL approaches are indeed effective and, in particular, capable of improving similar approaches based on the analysis of the whole image. Second, they show their capabilities to tackle the fingerprint liveness detection problem and (substantially) to generalize well across different datasets. Third, for each combination of model and fusion approach, the results achieved are comparable with the current optimal results on the literature and even improve of the baselines over different datasets.

Concerning the reference models analyzed, VGG (irrespective of the actual depth, which was found to have a very limited effect on the experimental results) appears to be the most fit to the problem analyzed. As a matter of facts, all the fusion approaches relying on VGGs were able to improve the results on Digital 2011 and Biometrica, Italdata and Swipe 2013 (with the only exception of MR-E2EF on Swipe 2013). Results were slightly worse, but still interesting, for the GoogLeNet model, while AlexNet showed its (relatively) limited relevance.

Concluding, interesting results have been obtained, but there is room for further improvements. For instance, in E2EF based methods it could be investigated the possibility to select an optimized patch size (or set) for each sensor. Another point is that none of the proposed methods and architectures consistently outperforms the other on all datasets. On the contrary, our results show in some cases a large variance between the different approaches on the same dataset. Therefore, one option worth to be investigated is the possibility to automatically select a model and a fusion approach for each dataset. This choice could be for instance based on the same heuristic used

throughout this chapter to select the method hyperparameters. Although (extremely) computationally extensive, this selection would be made in a pre–processing step and, thus, it would have limited effects on the online performances.

Chapter 5

Conclusions

The work presented in this thesis aimed at analyzing the contribution of fusion methods to the fingerprint liveness detection problem. The basic idea of these methods is to combine different pieces of information extracted from the input samples in order to improve both the final accuracies and, the generalization properties of the classifiers built on them.

A thorough analysis of fusion methods would require to tackle different and complex issues, such as deciding which type of information to associate to each sample, at which level this information should be combined (e.g., at feature level, at score/decision level) and which are the most effective methods for such combination. The extent of such analysis would have been too broad for this dissertation, which, thus, focused on two distinct approaches.

The work started with an analysis of methods capable of effectively combining multiple handcrafted features. The rationale of this approach is that these features have been engineered exploiting expert domain knowledge and, thus, they provide a detailed characterization of the samples under a specific perspective. Such perspectives are aimed at highlighting peculiar aspects of the data in analysis and the different features often provide pieces of information that complement each other. Therefore, finding effective methods of combining these features, which are capable of mutually exploiting their individual strengths and softening their weaknesses, might indeed provide benefits for the classification tasks.

On the basis of these observations, the followed 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). As a further contribution, this work introduced *Spidernet*, a novel two-stage deep neural architecture capable of effectively combining different general image descriptors.

The experiments were focused on finding feature groups capable of generalizing well across all datasets and methods analyzed. While an approach based on the selection of the optimal combination of features and method for each case would have certainly led to higher accuracies, the rational of this choice was 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).

The experimental results showed that the proposed multi–view methods are capable of (i) outperforming those based on individual handcrafted features on each dataset and (ii) providing state–of–the–art results. Concerning these results, it should be stressed again that the actual performances can be indeed improved by fine–tuning the chosen feature groups for each dataset and method.

In the second part of this work, we analyzed the fusion problem under a different perspective. Rather than combining features extracted from the whole image, the combination of different pieces of information extracted from local image patches was investigated. In approaching this analysis, the contribution of the recent advances in the field of deep learning was also exploited. The proposed method relied on Transfer Learning approaches, aimed at transferring the knowledge learned while solving a problem to a different context. In particular, the work leveraged on the recognized capabilities of several reference CNN models to tackle various and different computer vision tasks. The general approach is to take the pre–trained version of these models (which were computed to identify thousands of different object classes and with the support of millions of training images) and, then, fine–tune them to the problem at hand with a further training step aimed at "adapting" their characteristic to the novel context they are applied to.

It should be also underlined that, while TL approaches have been already investigated in the liveness detection context, previous works merely focused on using the whole image as network input. To the best of my knowledge, this is the first work in the literature that provides a thorough analysis of TL patch–based methods.

The rationale of the proposed approach is threefold. First, the use of small sized patches allows avoiding to resize the samples before they are fed to the network and, thus, retaining the original resolution and image information. Second, using patches rather than the full images as samples allows increasing the size of the training set, thus improving the fine–tuning of the reference models (and even allowing their re–training from scratch, although this option has not been considered in this work). Third, working at patch level allows again exploiting fusion approaches that can (hopefully) lead an improved robustness of the final fingerprint classification process.

Several fusion methods were analyzed and compared. They are based on exploiting (i) different pieces of information extracted from each patch (either decision scores or deep features extracted from the inner network layers) and (ii) different ways of combining them (e.g., score fusion, multi-resolution score fusion, fusion of individual or multiple deep features). Although no method outperformed the other ones, as a general statement, it should be observed the effectiveness of CNN patch–based TL approaches, which showed (i) improved accuracies when compared with similar approaches based on the analysis of the whole image as input and (ii) the state–of–the–art results.

The assessment of the results of both general approaches (fusion of handcrafted or deep features) could not lead to identify one of them as being optimal to tackle all the benchmarks analyzed. However, an overall view of the fusion approaches analyzed supports the evidence of their effectiveness. Indeed, when the optimal accuracy for each dataset and each approach is considered (in a way similar to what has been done to compile the baselines used during the method assessments), the results, summarized in Table 5.1, clearly show that fusion methods are capable of improving the current state–of–the–art in all cases except XMatch 2013[1].

Concerning future perspectives about the patch–based approach, given the promising results of MvDA and Spidernet methods on hand-crafted features, we are planning to apply these multi–view methods to the deep patch features extracted from different layers of trained convolutional neural networks. Furthermore, custom patch sizes can be picked for each dataset to improve the results. Another possibility for expanding the current work is to apply the idea of fusing deep features extracted from different reference CNN models.

[1]This observation clearly indicates as well that the proposed approaches were not able to fully address the complexity in the analysis of this specific dataset, which is the same problem highlighted by several other authors [32] leading them to discard this dataset in their works.

	LivDet2011				*LivDet2013*			
Nets\Datasets	Biom.	Digital	Italdata	Sagem	Biom.	XMatch	Italdata	Swipe
SOA	1.2 *F.S.Buster* [17]	1.6 *F.S.Buster* [17]	2.5 *F.S.Buster* [17]	**1.2** *TripletNet* [84]	0.2 *spoofnet* [66]	**1.8** *spoofnet* [66]	**0.1** *spoofnet* [66]	0.7 *TripletNet* [84]
Feature Fusion approach	**0.5** MvDA-G4	**0.7** MvDA-G1	**2.1** Spidernet-G1	**1.2** Spidernet-G4	0.4 MvDA-G2	3.3 Spidernet-G4	0.4 MvDA-G2	1.0 MvDA-G3
Patch-based approach	2.6 MR–E2EF-VGG 19	**0.7** E2EF-GNet	5.2 E2EF-AlexNet	1.6 DFF-VGG 16	**0.1** E2EF-VGG 19	5.4 E2EF-AlexNet	**0.1** DFF-VGG 16	**0.5** MDFF-VGG 16

Table 5.1 Comparing the best results in the literature with the best obtained with the two general approaches analyzed in this dissertation. **Bold** values represent the optimal result for each benchmark. Underlined labels indicate that for the specific dataset there are several combinations of methods and features providing the same result.

As a final comment I would like to underline that the software liveness detection methods described in my work are sufficiently robust when the operating conditions are similar to the ones used to create the experimental datasets of my work. These conditions are based on the assumption that an individual is trying to fool the fingerprint recognition system by presenting a replica of a real sample to the fingerprint scanner. In other words, this method is robust to attacks at sensor level. However, a capable intruder could access the system at different levels of the processing chain and find ways to bypass or make virtually useless any liveness detection scheme. Concerning that, there is the problem, especially for deep learning approaches, of images purposely crafted to fool a classifier to ignore actual evidence and report a predefined target class [71, 9]. Taking advantage of these methods, the intruder could intercept the communication between the scanner and the liveness detector and submit one of such images to grant access to the system. Therefore, future work stemming from this thesis should probably keep this problem into account as well.

References

[1] Abhyankar, A. and Schuckers, S. (2006). Fingerprint liveness detection using local ridge frequencies and multiresolution texture analysis techniques. In *Image Processing, 2006 IEEE International Conference on*, pages 321–324. IEEE.

[2] Abhyankar, A. and Schuckers, S. (2009). Integrating a wavelet based perspiration liveness check with fingerprint recognition. *Pattern Recognition*, 42(3):452–464.

[3] Ahonen, T., Rahtu, E., Ojansivu, V., and Heikkila, J. (2008). Recognition of blurred faces using local phase quantization. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–4. IEEE.

[4] Akhtar, Z., Micheloni, C., and Foresti, G. L. (2015). Correlation based fingerprint liveness detection. In *2015 International Conference on Biometrics (ICB)*, pages 305–310.

[5] Antonelli, A., Cappelli, R., Maio, D., and Maltoni, D. (2006). Fake finger detection by skin distortion analysis. *IEEE Transactions on Information Forensics and Security*, 1(3):360–373.

[6] Athiwaratkun, B. and Kang, K. (2015). Feature representation in convolutional neural networks. *arXiv preprint arXiv:1507.02313*.

[7] Balaji, A., HS, V., and OK, S. (2016). Multimodal fingerprint spoof detection using white light. *Procedia Computer Science*, 78:330–335.

[8] Baldisserra, D., Franco, A., Maio, D., and Maltoni, D. (2006). Fake fingerprint detection by odor analysis. In *International Conference on Biometrics*, pages 265–272. Springer.

[9] Brown, T. B., Mané, D., Roy, A., Abadi, M., and Gilmer, J. (2017). Adversarial patch. *CoRR*, abs/1712.09665.

[10] Brümmer, N., Swart, A., and Van Leeuwen, D. (2014). A comparison of linear and non-linear calibrations for speaker recognition. In *Odyssey 2014: The Speaker and Language Recognition Workshop*.

[11] Candes, E. J. (1998). *Ridgelets: theory and applications*. PhD thesis, Stanford University Stanford.

[12] Chan, C. H., Tahir, M. A., Kittler, J., and Pietikainen, M. (2013). Multiscale local phase quantization for robust component-based face recognition using kernel fusion of multiple descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(5):1164–1177.

[13] Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27.

[14] Chen, J., Shan, S., He, C., Zhao, G., Pietikainen, M., Chen, X., and Gao, W. (2010). Wld: A robust local image descriptor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1705–1720.

[15] Cheng, J. and Wang, K. (2007). Active learning for image retrieval with co-svm. *Pattern Recognition*, 40(1):330 – 334.

[16] Chugh, T., Cao, K., and Jain, A. K. (2018a). Fingerprint spoof buster: Use of minutiae-centered patches. *IEEE Transactions on Information Forensics and Security*.

[17] Chugh, T., Cao, K., and Jain, A. K. (2018b). Fingerprint spoof buster: Use of minutiae-centered patches. *IEEE Transactions on Information Forensics and Security*.

[18] Coli, P., Marcialis, G. L., and Roli, F. (2007a). Power spectrum-based fingerprint vitality detection. In *Automatic Identification Advanced Technologies, 2007 IEEE Workshop on*, pages 169–173.

[19] Coli, P., Marcialis, G. L., and Roli, F. (2007b). Vitality detection from fingerprint images: a critical survey. In *International Conference on Biometrics*, pages 722–731. Springer.

[20] Derakhshani, R., Schuckers, S. A., Hornak, L. A., and O'Gorman, L. (2003). Determination of vitality from a non-invasive biomedical measurement for use in fingerprint scanners. *Pattern recognition*, 36(2):383–396.

[21] Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T. (2014). Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655.

[22] Drahansky, M. (2008). Experiments with skin resistance and temperature for liveness detection. In *Proceedings of IIH-MSP 2008*, pages 1075–1079.

[23] Duyck, J., Lee, M. H., and Lei, E. (2014). Modified dropout for training neural network. Technical report.

[24] Frassetto Nogueira, R., de Alencar Lotufo, R., and Campos Machado, R. (2014). Evaluating software-based fingerprint liveness detection using convolutional networks and local binary patterns. In *Biometric Measurements and Systems for*

Security and Medical Applications (BIOMS) Proceedings, 2014 IEEE Workshop on, pages 22–29. IEEE.

[25] Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119 – 139.

[26] Fu, Y., Cao, L., Guo, G., and Huang, T. S. (2008). Multiple feature fusion by subspace learning. In *Proceedings of the 2008 International Conference on Content-based Image and Video Retrieval*, CIVR '08, pages 127–134, New York, NY, USA. ACM.

[27] Galbally, J., Alonso-Fernandez, F., Fierrez, J., and Ortega-Garcia, J. (2012). A high performance fingerprint liveness detection method based on quality related features. *Future Generation Computer Systems*, 28(1):311 – 321.

[28] Ghiani, L. (2015). *Textural features for fingerprint liveness detection*. PhD thesis, Universita'degli Studi di Cagliari.

[29] Ghiani, L., Marcialis, G., and Roli, F. (2012a). Fingerprint liveness detection by local phase quantization. In *ICPR 2012*, pages 537–540.

[30] Ghiani, L., Marcialis, G. L., and Roli, F. (2012b). Experimental results on the feature-level fusion of multiple fingerprint liveness detection algorithms. In *Proceedings of the on Multimedia and Security*, MM&Sec '12, pages 157–164, New York, NY, USA. ACM.

[31] Ghiani, L., Mura, V., Tuveri, P., and Marcialis, G. L. (2017a). On the interoperability of capture devices in fingerprint presentation attacks detection. In *Proceedings of ITASEC17*, pages 66–75.

[32] Ghiani, L., Yambay, D., Mura, V., Tocco, S., Marcialis, G. L., Roli, F., and Schuckcrs, S. (2013). Livdet 2013 fingerprint liveness detection competition 2013. In *Biometrics (ICB), 2013 International Conference on*, pages 1–6.

[33] Ghiani, L., Yambay, D. A., Mura, V., Marcialis, G. L., Roli, F., and Schuckers, S. A. (2017b). Review of the fingerprint liveness detection (livdet) competition series: 2009 to 2015. *Image and Vision Computing*, 58:110–128.

[34] Gottschlich, C. (2016). Convolution comparison pattern: An efficient local image descriptor for fingerprint liveness detection. *PLoS ONE*, 11(2):1–12.

[35] Gottschlich, C., Marasco, E., Yang, A. Y., and Cukic, B. (2014). Fingerprint liveness detection based on histograms of invariant gradients. In *Proceeding of IEEE IJCB 2014*, pages 1–7.

[36] Gragnaniello, D., Poggi, G., Sansone, C., and Verdoliva, L. (2013). Fingerprint liveness detection based on weber local image descriptor. In *IEEE BIOMS 2013*, pages 46–50.

[37] Gragnaniello, D., Poggi, G., Sansone, C., and Verdoliva, L. (2015a). An investigation of local descriptors for biometric spoofing detection. *IEEE Transactions on Information Forensics and Security*, 10(4):849–863.

[38] Gragnaniello, D., Poggi, G., Sansone, C., and Verdoliva, L. (2015b). Local contrast phase descriptor for fingerprint liveness detection. *Pattern Recognition*, 48(4):1050 – 1058.

[39] Heikkila, J. and Ojansivu, V. (2009). Methods for local phase quantization in blur-insensitive image analysis. In *Local and Non-Local Approximation in Image Processing, 2009. LNLA 2009. International Workshop on*, pages 104–111. IEEE.

[40] Hinton, G. E. (2012). *A Practical Guide to Training Restricted Boltzmann Machines*, pages 599–619. Springer.

[41] Hinton, G. E. (2007). Learning multiple layers of representation. *Trends in Cognitive Sciences*, 11:428–434.

[42] Hussain, M., Muhammad, G., and Bebis, G. (2012). Face recognition using multiscale and spatially enhanced weber law descriptor. In *Proceedings of the IEEE SITIS 2012*, pages 85–89.

[43] Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456.

[44] Jain, A. K., Ross, A. A., and Nandakumar, K. (2011). Introduction. In *Introduction to Biometrics*, pages 1–49. Springer.

[45] Jia, J., Cai, L., Zhang, K., and Chen, D. (2007). A new approach to fake finger detection based on skin elasticity analysis. In *International Conference on Biometrics*, pages 309–318. Springer.

[46] Jia, X., Yang, X., Cao, K., Zang, Y., Zhang, N., Dai, R., Zhu, X., and Tian, J. (2014). Multi-scale local binary pattern with filters for spoof fingerprint detection. *Information Sciences*, 268(0):91 – 102.

[47] Jin, C., Kim, H., and Elliott, S. (2007). Liveness detection of fingerprint based on band-selective fourier spectrum. In *International Conference on Information Security and Cryptology*, pages 168–179. Springer.

[48] Kan, M., Shan, S., Zhang, H., Lao, S., and Chen, X. (2016). Multi-view discriminant analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(1):188–194.

[49] Kannala, J. and Rahtu, E. (2012). Bsif: Binarized statistical image features. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 1363–1366. IEEE.

[50] Kembhavi, A., Siddiquie, B., Miezianko, R., McCloskey, S., and Davis, L. S. (2009). Incremental multiple kernel learning for object recognition. In *2009 IEEE 12th International Conference on Computer Vision*, pages 638–645.

[51] Kim, S., Park, B., Song, B. S., and Yang, S. (2016). Deep belief network based statistical feature learning for fingerprint liveness detection. *Pattern Recognition Letters*, 77:58 – 65.

[52] Kira, K. and Rendell, L. A. (1992). The feature selection problem: Traditional methods and a new algorithm. In *Proceedings of AAAI'92*, pages 129–134. AAAI Press.

[53] Kokkinos, I. and Yuille, A. (2008). Scale invariance without scale selection. In *IEEE CVPR*.

[54] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

[55] Lin, M., Chen, Q., and Yan, S. (2013). Network in network. *arXiv preprint arXiv:1312.4400*.

[56] Liu, C. and Yuen, P. C. (2011). A boosted co-training algorithm for human action recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(9):1203–1213.

[57] Liu, T., Xie, S., Yu, J., Niu, L., and Sun, W. (2017). Classification of thyroid nodules in ultrasound images using deep model based transfer learning and hybrid features. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 919–923. IEEE.

[58] Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110.

[59] Maltoni, D., Maio, D., Jain, A. K., and Prabhakar, S. (2009). *Handbook of fingerprint recognition*. Springer Science & Business Media.

[60] Marasco, E. and Ross, A. (2015). A survey on antispoofing schemes for fingerprint recognition systems. *ACM Computing Surveys (CSUR)*, 47(2):28.

[61] Marasco, E. and Sansone, C. (2012). Combining perspiration- and morphology-based static features for fingerprint liveness detection. *Pattern Recogn. Lett.*, 33(9):1148–1156.

[62] Marasco, E., Wild, P., and Cukic, B. (2016). Robust and interoperable fingerprint spoof detection via convolutional neural networks. In *Technologies for Homeland Security (HST), 2016 IEEE Symposium on*, pages 1–6. IEEE.

[63] Marcialis, G. L., Lewicke, A., Tan, B., Coli, P., Grimberg, D., Congiu, A., Tidu, A., Roli, F., and Schuckers, S. (2009). *Proceedings of ICIAP 2009*, chapter First International Fingerprint Liveness Detection Competition—LivDet 2009, pages 12–23. Springer.

[64] Mason, S., Gashi, I., Lugini, L., Marasco, E., and Cukic, B. (2014). Interoperability between fingerprint biometric systems: An empirical study. In *Proceedings of the 2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, DSN '14, pages 586–597, Washington, DC, USA. IEEE Computer Society.

[65] Matsumoto, T., Matsumoto, H., Yamada, K., and Hoshino, S. (2002). Impact of artificial "gummy" fingers on fingerprint systems. *Proceedings of SPIE Vol. 4677*, 4677.

[66] Menotti, D., Chiachia, G., Pinto, A., Schwartz, W. R., Pedrini, H., Falcao, A. X., and Rocha, A. (2015). Deep representations for iris, face, and fingerprint spoofing detection. *IEEE Transactions on Information Forensics and Security*, 10(4):864–879.

[67] Minaee, S., Abdolrashidiy, A., and Wang, Y. (2016). An experimental study of deep convolutional features for iris recognition. In *Signal Processing in Medicine and Biology Symposium (SPMB), 2016 IEEE*, pages 1–6. IEEE.

[68] Nakada, M., Wang, H., and Terzopoulos, D. (2017). Acfr: active face recognition using convolutional neural networks. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*, pages 35–40. IEEE.

[69] Negri, P., Clady, X., Hanif, S., and Prevost, L. (2008). A cascade of boosted generative and discriminative classifiers for vehicle detection. *EURASIP JASP*, 2008:1–12.

[70] Negri, P., Goussies, N., and Lotito, P. (2014). Detecting pedestrians on a movement feature space. *Pattern Recognition*, 47(1):56 – 71.

[71] Nguyen, A. M., Yosinski, J., and Clune, J. (2014). Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *CoRR*, abs/1412.1897.

[72] Nikam, S. and Agarwal, S. (2009a). Co-occurrence probabilities and wavelet-based spoof fingerprint detection. *Int. Journal of Image and Graphics*, 09(02):171–199.

[73] Nikam, S. B. and Agarwal, S. (2008). Fingerprint liveness detection using curvelet energy and co-occurrence signatures. In *Computer Graphics, Imaging and Visualisation, 2008. CGIV '08. Fifth International Conference on*, pages 217–222.

[74] Nikam, S. B. and Agarwal, S. (2009b). Ridgelet-based fake fingerprint detection. *Neurocomputing*, 72(10-12):2491–2506.

[75] Nixon, K. A., Aimale, V., and Rowe, R. K. (2008). Spoof detection schemes. In *Handbook of biometrics*, pages 403–423. Springer.

[76] Nogueira, K., Penatti, O. A., and dos Santos, J. A. (2017). Towards better exploiting convolutional neural networks for remote sensing scene classification. *Pattern Recognition*, 61:539–556.

[77] Nogueira, R. F., de Alencar Lotufo, R., and Machado, R. C. (2016). Fingerprint liveness detection using convolutional neural networks. *IEEE Transactions on Information Forensics and Security*, 11(6):1206–1213.

[78] Nosaka, R., Ohkawa, Y., and Fukui, K. (2011). Feature extraction based on co-occurrence of adjacent local binary patterns. In *Advances in image and video technology*, pages 82–91. Springer.

[79] Nosaka, R., Suryanto, C. H., and Fukui, K. (2012). Rotation invariant co-occurrence among adjacent lbps. In *Computer Vision-ACCV 2012 Workshops*, pages 15–25. Springer.

[80] Ojala, T., Pietikäinen, M., and Harwood, D. (1996). A comparative study of texture measures with classification based on featured distributions. *Pattern recognition*, 29(1):51–59.

[81] Ojansivu, V. and Heikkilä, J. (2008). Blur insensitive texture classification using local phase quantization. In *Image and signal processing*, pages 236–243. Springer.

[82] Ojansivu, V., Rahtu, E., and Heikkilä, J. (2008). Rotation invariant blur insensitive texture analysis using local phase quantization. In *Proceedings of 19th International Conference on Pattern Recognition*, volume 4.

[83] Pala, F. and Bhanu, B. (2017a). Deep triplet embedding representations for liveness detection. In *Deep Learning for Biometrics*, pages 287–307. Springer.

[84] Pala, F. and Bhanu, B. (2017b). Deep triplet embedding representations for liveness detection. In *Deep Learning for Biometrics*, pages 287–307. Springer.

[85] Park, E., Cui, X., Kim, W., Liu, J., and Kim, H. (2018). Patch-based fake fingerprint detection using a fully convolutional neural network with a small number of parameters and an optimal threshold. *arXiv preprint arXiv:1803.07817*.

[86] Parthasaradhi, S. T., Derakhshani, R., Hornak, L. A., and Schuckers, S. A. (2005). Time-series detection of perspiration as a liveness test in fingerprint devices. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 35(3):335–343.

[87] Pereira, L. F. A., Pinheiro, H. N., Silva, J. I. S., Silva, A. G., Pina, T. M., Cavalcanti, G. D., Ren, T. I., and de Oliveira, J. P. N. (2012). A fingerprint spoof detection based on mlp and svm. In *Neural Networks (IJCNN), The 2012 International Joint Conference on*, pages 1–7. IEEE.

[88] Razavian, A. S., Azizpour, H., Sullivan, J., and Carlsson, S. (2014). Cnn features off-the-shelf: an astounding baseline for recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, pages 512–519. IEEE.

[89] Roweis, S. (1998). Em algorithms for pca and spca. In *Proceedings of the 1997 Conference on Advances in Neural Information Processing Systems 10*, NIPS '97, pages 626–632, Cambridge, MA, USA. MIT Press.

[90] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252.

[91] Schapire, R. and Singer, Y. (1999). Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336.

[92] Schuckers, S. A. (2002). Spoofing and anti-spoofing measures. *Information Security technical report*, 7(4):56–62.

[93] Simard, P. Y., Steinkraus, D., and Platt, J. C. (2003). Best practices for convolutional neural networks applied to visual document analysis. In *Proceedings of the Seventh International Conference on Document Analysis and Recognition - Volume 2*, ICDAR '03, pages 958–, Washington, DC, USA. IEEE Computer Society.

[94] Simon, M., Rodner, E., and Denzler, J. (2016). Imagenet pre-trained models with batch normalization. *arXiv preprint arXiv:1612.01452*.

[95] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

[96] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

[97] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., et al. (2015). Going deeper with convolutions. Cvpr.

[98] Tan, B. and Schuckers, S. (2006). Liveness detection for fingerprint scanners based on the statistics of wavelet signal processing. In *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW'06. Conference on*, pages 26–26. IEEE.

[99] Thai, D. H., Huckemann, S., and Gottschlich, C. (2015). Filter design and performance evaluation for fingerprint image segmentation. *CoRR*, abs/1501.02113.

[100] Tola, E., Lepetit, V., and Fua, P. (2010). Daisy: An efficient dense descriptor applied to wide-baseline stereo. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(5):815–830.

[101] Toosi, A., Bottino, A., Cumani, S., Negri, P., and Sottile, P. L. (2017a). Feature fusion for fingerprint liveness detection: a comparative study. *IEEE Access*, 5:23695–23709.

[102] Toosi, A., Cumani, S., and Bottino, A. (2015). On multiview analysis for fingerprint liveness detection. In *Proceeidngs of CIARP 2015*, volume 9423, pages 143–150. Springer.

[103] Toosi, A., Cumani, S., and Bottino, A. (2017b). Cnn patch–based voting for fingerprint liveness detection. In *Proceedings of the 9th International Joint Conference on Computational Intelligence - Volume 1: IJCCI,*, pages 158–165. INSTICC, SciTePress.

[104] wei Hsu, C., chung Chang, C., and jen Lin, C. (2010). A practical guide to support vector classification.

[105] Yambay, D., Ghiani, L., Denti, P., Marcialis, G., Roli, F., and Schuckers, S. (2012). Livdet 2011 - fingerprint liveness detection competition 2011. In *Biometrics (ICB), 2012 5th IAPR International Conference on*, pages 208–215.

[106] Yoon, S., Feng, J., and Jain, A. (2010). Fingerprint alteration. In *Proceedings of the 95th International Educational Conference of the International Association for Identification (IAI'10)*, pages 11–17.

[107] Yoon, S., Feng, J., and Jain, A. K. (2012). Altered fingerprints: Analysis and detection. *IEEE transactions on pattern analysis and machine intelligence*, 34(3):451–464.

[108] Zhang, L., Zhang, L., Tao, D., and Huang, X. (2012). On combining multiple features for hyperspectral remote sensing image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 50(3):879–893.

[109] Zhang, Y., Tian, J., Chen, X., Yang, X., and Shi, P. (2007). Fake finger detection based on thin-plate spline distortion model. In *International Conference on Biometrics*, pages 742–749. Springer.

[110] Zheng, W., Zhou, X., Zou, C., and Zhao, L. (2006). Facial expression recognition using kernel canonical correlation analysis (kcca). *IEEE Transactions on Neural Networks*, 17(1):233–238.