A Survey of Fingerprint Classification Part II: Experimental Analysis and Ensemble Proposal

Mikel Galar[a,*], Joaquín Derrac[g], Daniel Peralta[b], Isaac Triguero[e,f,b], Daniel Paternain[a], Carlos Lopez-Molina[a], Salvador García[b,c,d], José M. Benítez[b], Miguel Pagola[a], Edurne Barrenechea[a], Humberto Bustince[a], Francisco Herrera[b]

[a] *Departamento de Automática y Computación, Universidad Pública de Navarra, 31006, Pamplona, Spain.*
[b] *Department of Computer Science and Artificial Intelligence, University of Granada, 18071, Granada, Spain.*
[c] *Faculty of Computing and Information Technology - North Jeddah, King Abdulaziz University, 21589, Jeddah, Saudi Arabia.*
[d] *Department of Computer Science, University of Jaén, 23071, Jaén, Spain.*
[e] *Department of Respiratory Medicine, Ghent University, 9000 Gent, Belgium.*
[f] *VIB Inflammation Research Center, 9052 Zwijnaarde, Belgium.*
[g] *Affectv: Affectv Limited, 33-34 Alfred Place, London, WC1E 7DP, United Kingdom.*

Abstract

In the first part of this paper we reviewed the fingerprint classification literature from two different perspectives: the feature extraction and the classifier learning. Aiming at answering the question of which among the reviewed methods would perform better in a real implementation we end up in a discussion which showed the difficulty in answering this question. No previous comparison exists in the literature and comparisons among papers are done with different experimental frameworks. Moreover, the difficulty in implementing published methods was stated due to the lack of details in their description, parameters and the fact that no source code is shared. For this reason, in this paper we will go through a deep experimental study following the proposed double perspective. In order to do so, we have carefully implemented some of the most relevant feature extraction methods according to the explanations found in the corresponding papers and we have tested their performance with different classifiers, including those specific proposals made by the authors. Our aim is to develop an objective experimental study in a common framework, which has not been done before and which can serve as a baseline for future works on the topic. This way, we will not only test their quality, but their reusability by other researchers and will be able to indicate which proposals could be considered for future developments. Furthermore, we will show that combining different feature extraction models in an ensemble can lead to a superior performance, significantly increasing the results obtained by individual models.

Keywords: Fingerprint Classification, Feature Extraction, Classification, Fingerprint Recognition, SVM, Neural Networks, Ensembles, Orientation Map, Singular Points, Experimental Evaluation

1. Introduction

Fingerprint classification has been a hot topic since 1975. In the first part of this series of two papers [27], we reviewed the different approaches that have been presented in the specialized literature in order to address this problem from a double perspective. First, we considered the problem of feature extraction (FE)

*Corresponding author. Tel.: +34 948 166048; Fax: +34 948 168924
Email addresses: mikel.galar@unavarra.es (Mikel Galar), jderrac@decsai.ugr.es (Joaquín Derrac), dperalta@decsai.ugr.es (Daniel Peralta), Isaac.Triguero@irc.vib-UGent.be (Isaac Triguero), daniel.paternain@unavarra.es (Daniel Paternain), carlos.lopez@unavarra.es (Carlos Lopez-Molina), sglopez@ujaen.es (Salvador García), J.M.Benitez@decsai.ugr.es (José M. Benítez), miguel.pagola@unavarra.es (Miguel Pagola), edurne.barrenechea@unavarra.es (Edurne Barrenechea), bustince@unavarra.es (Humberto Bustince), herrera@decsai.ugr.es (Francisco Herrera)

[57] that aimed at obtaining a suitable representation of the fingerprint for its posterior processing. Second, we dealt with the classification problem from the machine learning point of view [19], where a classifier capable of classifying new fingerprints represented by their extracted features should be learned from a set of previously labeled fingerprints (represented by the same features).

In this context, we presented a taxonomy of FE methods and additionally other two of Singular Point (SP) and Orientation Map (OM) extraction methods. Similarly, we grouped the learning models into different categories. The revision of those works led us to try to investigate which one would perform better in a real implementation, i.e., would be more accurate in its predictions. However, currently, it is extremely difficult to answer this question by simply reviewing the existing literature. This fact is due to the different experimental set-ups that have been used in fingerprint classification. As we discussed in the first part of the paper, there are even papers using the same database (for example NIST-4 [77]) for testing their model but in different ways [35, 32, 10, 82, 72, 59, 29, 42], which make them not comparable, even though some authors continue comparing algorithms among papers by simply taking the results from them [35, 10, 67, 59, 29, 42, 43] (despite the differences in the evaluation procedures).

For this reason, our aim in this second part is to experimentally study the performance of different FE methods and learning models used for fingerprint classification. We want to investigate which method performs better in a common experimental framework, which can thereafter be used as a baseline for comparing new algorithms presented by other researchers. We are not doubting on the results presented in the corresponding papers, but since they are in many cases not comparable, we will carefully implement and test them in a common experimental framework with different databases. Moreover, carrying out these implementations would show which methods are not only better but easier to be reproduced by other researchers, which many times is overlooked despite its importance. Therefore, this study will allow us to extract meaningful conclusions about the fingerprint classification literature that will also show which methods can be recommended to practitioners when facing the problem of developing a fingerprint classification system.

In order to perform such a deep experimental study, we have implemented some of the most relevant FE methods covering all the categories in our taxonomy for FE (orientations, SPs, ridge structure and filter responses), including works from 1995 and taking into account their relevance in terms of citations received, but always with a previous analysis of the possibility of their implementation (see Discussion section in [27]). Finally, fourteen FE methods have been considered, which we show in Table 1). The works of Tan [72] and Park [59] were also implemented but due to different reasons they were finally leave out of this comparison (poor results, generate too big data-sets and the implementation may not follow the real one proposed by the authors due to a lack of details even though in our initial analysis they seemed to be reproducible). [1]

Since most of the FE methods are proposed in conjunction with a learning model by the authors, we have also implemented these models and included in the comparison in addition to the three classical classifiers that we have selected for carrying out the comparison: Support Vector Machines (SVMs) [73], C4.5 decision tree [62] and k-Nearest Neighbor (kNN) algorithm [1].

All these methods are tested in two different types of fingerprint databases: NIST-4 database [77], where most of the methods were originally tested; SFinGe (Synthetic Fingerprint Generator) tool[2] [13, 49] based fingerprint databases, which will allow us to simulate different real-world scenarios with varying fingerprint qualities.

In addition to this study, we propose the usage of an ensemble in order to improve the performance of the individual models. To do so, we create a Multiple Classifier System (MCS) [38] performing different FEs and learning models, whose diversity allows us to significantly increase the accuracy of the classification problem in exchange for increasing the rejection rate.

[1] In [72], 2496 original features were generated that should be reduced by a Genetic Programming model. However, following the implementation given by the authors the dimensionality reduction has been intractable (which may be due to the lack of some key details of this algorithm such as initialization of the chromosomes, number of generations, etc.). In the case of [59], even though it only represents OMs of 21×21 blocks, they were codified with gray scale orientations in an image, leading to 11025 features, which would lead to a high-dimensional problem. Moreover, we were not able to reduce such large dimensionality following the explanations given in the source paper.

[2] Synthetic Fingerprint Generator: `http://biolab.csr.unibo.it/research.asp?organize=Activities&select=&selObj=12&pathSubj=111||12&`

Table 1: Fingerprint classification works considered in the experimental study.

Name	Year	Features	Orientations	Singular points	Reference point	Classification technique	Test databases	Reference
Candela	1995	Orientations (Registered) Ridge structure (Ridge-tracing (Generic))	Slits sum (16×16)		Rule-based (R92)	Neural Networks	NIST-14	[8]
Karu	1996	Singular points (Number/positions)	Slits sum (8×8)	Poincarè		Fixed Classification	NIST-4/9	[35]
Cappelli99a	1999	Orientations (Segmentation)	Slits sum (16×16)			Graph Matching	NIST-4/14	[10]
Jain	1999	Filter responses (Gabor)	Gradient (16×16)	Poincarè	Covariances	Multiple Techniques	NIST-4	[32]
Cappelli02	2002	Orientations (Registered) Orientations (Segmentation)	Slits sum (16×16)	Poincarè	Rule-based (R92)	Nearest Neighbor	NIST-14	[12]
Zhang04	2004	Singular points (Number/positions) Ridge structure (Ridge-tracing (from SPs))	Gradient (16×16)	Poincarè		Fixed Classification	NIST-4	[82]
Nyongesa	2004	Singular points (Number/positions) Singular points (Relative measures) Orientations (Direct/unaltered)	Gray-level consistency/variance (5×5)	Poincarè		Neural Networks	NIST-4	[58]
Shah	2004	Orientations (Direct/unaltered) Orientations (Registered)	Line detection (16×16)		Entropy	Nearest Neighbor	Other	[67]
Wang07	2007	Singular points (Number/positions) Ridge structure (Ridge-tracing (from SPs))	Gradient (16×16)	Poincarè		Fixed Classification	NIST-4	[74]
Hong08	2008	Singular points (Number/positions) Ridge structure (Ridge-tracing (from SPs)) Filter responses (Gabor)	Gradient (16×16)	Poincarè	Covariances	Multiple Techniques	NIST-4	[29]
Li	2008	Orientations (Registered) Singular points (Number/positions) Singular points (Relative measures)	Gradient (8×8)	Complex filters		Support Vector Machines	NIST-4	[42]
Liu10	2010	Singular points (Relative measures)	Gradient (10×10)	Complex filters		Structural Models	NIST-4	[43]
Leung	2011	Filter responses (Gabor)	Gradient (16×16)		Rule-based (R92)	Other	FCV,NIST-4	[40]
Le	2012	Filter responses (Gabor)	Gradient ($6 \times 6, 12 \times 12$, 18×18)	Complex filters		Nearest Neighbor	FCV2004	[39]

The rest of this paper is organized as follows. In Section 2 the FE methods considered are described. Similarly, in Section 3 the learning models considered (both the classical ones and the specific ones) are presented. Section 4 puts forwards our experimental framework used to carry out the experiments and presents the web-page associated with this paper `http://sci2s.ugr.es/fingerprintClassification`, where the source code of our implementations and the generated data-sets can be found. The results obtained are shown and discussed in Section 5. Afterwards, the ensemble proposal and its analysis are given in Section 6. Finally, Section 7 concludes this series of two papers.

2. Feature extraction methods

In this section we briefly review each one of the FE methods that we have implemented and tested. We should remark that the complete details of our implementations can be found in the source code provided as a complementary material with this paper. We cannot assure that the implemented models are exactly the same as those proposed by the authors due to the reasons already mentioned, but we have made our best to reflect as much as possible the original works of the authors.

Each subsection in this section is devoted to a method. A brief summary of their characteristics is presented in Table 1: features used for classification, OM extraction model, SP detection method (or reference point), classification technique considered, databases used for testing and the corresponding reference. Papers are ordered by year of publication (second column). Each approach is termed by the name of its first author, following by two digits denoting the year of publication if two or more names match (the two approaches proposed by Cappelli et al. in 1999 are termed as **Cappelli99a** and **Cappelli99b**, this notation follows that used in the first part of this paper). Detailed explanations on each datum are given in the next sections.

We should mention that along the presentation of these methods we will make reference to the OM and SP extraction methods reviewed in the first part of this paper as well as to the FE categories proposed in it.

2.1. Candela - PCASYS

PCASYS (Pattern-level Classification Automation SYStem) [8] is well-known since it is the only method whose source code was made publicly available by the authors. Therefore, we have not implemented it but used the original model of the authors.

In PCASYS the orientations from the registered OM are used as features, being each orientation codified in the feature vector by its two components from the squared orientation map. Since the OM is composed of 28×30 orientation blocks, the feature vector has 1680 elements. This feature vector is thereafter reduced using the Karhunen-Loève (KL) transform [31] (also known as Principal Component Analysis, PCA, [34]) obtaining a final feature vector of 128 features (we tested this configuration even though the authors provided both configurations with 64 and 128 features).

First of all, in PCASYS the fingerprint is enhanced (after being segmented) using the Fast Fourier Transform (FFT). The OM is then obtained using the slits sum method [71]. In order to find a reference point for the OM alignment R92 algorithm [78] is considered. When carrying out the registration, the orientations of the OM are recomputed in each block, since the reference point may not be exactly centered in a block. Finally, these orientations are codified into the feature vector using their both components (sine and cosine) of the squared representation $(2 \cdot \theta)$.

2.2. Karu

The work of Karu and Jain [35] has been one of the most relevant works in fingerprint classification. In this work, a fingerprint classification model based on fixed rules was developed with good results despite its simplicity.

The model is based on the usage of the number and positions of the SPs in a fingerprint. Once these are detected, simple rules such as *"if there are no SPs then it is an arch fingerprint"* are considered. The whorl type classification is also direct if two pair of cores and deltas are found. In order to differentiate between loops and tented arch when only a pair is found, an straight line is traced from the core to the delta and the difference with the orientations crossing this line is taken into account. Finally, right and left loops are distinguished by a simple rule which depends on the positioning of the pair of SPs in the image.

Therefore, in order to define the class of the fingerprint, the SPs need to be detected. To do so, the OM is obtained by the slits sum method. Afterwards, the OM is processed giving greater weights to orientations near the center of the images in such a way that when the OM is smoothed the noise in the outer locations are removed. Similarly, the orientations in the borders are removed. Then, the localization of the SPs is carried out using the Poincarè [36] method (with a digital curve of four pixels) following several rules: cores in the borders are deleted and cores and deltas pairs which appears close to each other are also deleted. In the case that more than two pair of cores and deltas or a different number of cores and deltas are found, the OM is smoothed and the process is repeated until a valid number of pairs are found.

Notice that in this case, we do not obtain a feature vector but the classification of the fingerprint, and hence no classifier training is required. Besides the explanations given by the authors, we have included the usage of a segmentation method in order to avoid false positives in the SP detection method. This is an issue that it is no mentioned in most of the papers but in case of not using it their performance decreases significantly. The segmentation is based on the block-wise variance of the gray level and is explained in [60]; it is a combination of the ideas presented in [64, 30, 5]. Even though it is a simple segmentation it is enough for the purpose of fingerprint classification. Moreover, this segmentation is used whenever Poincarè method is considered (if it is not otherwise specified).

2.3. Cappelli99a

In the Cappelli99a method [10] a segmentation of the OM is carried out. The segmentation consists of grouping areas with similar orientations. In this approach, a dynamic mask (template) is predefined for each fingerprint class and each one of them is adapted to match the OM as better as possible in terms of a cost function mainly defined as the orientation variance in the different areas. Finally, the feature vector is composed of the lowest segmentation cost obtained for each class, and hence five features are obtained.

In order to obtain the final feature vector a number of steps are followed: First, the fingerprint is enhanced following the model of PCASYS, i.e., using the FFT. Likewise, the OM is extracted with slits sum method and it is further processed with a enhancement step composed of three phases: regularization, attenuation and strengthening. Once the OM is obtained, it is fitted to the five dynamic masks. These masks are defined by vertices (mobile or fix), regions defined by the vertices, relationships between pairs of regions and functions which define the mobility of the vertices. The matching is carried out in a sequential manner, first the best global positioning is obtained by a greedy search (rotation and translation); then, the optimal position for each mobile vertice is obtained (also sequentially). Finally, the minimum cost obtained for each mask, normalized by the sum of costs is codified in the feature vector.

2.4. Jain - FingerCode

As we mentioned in the first part of this paper, FingerCode method [32] is the one with the greatest number of works using it. In this filter-based method, the Gabor filter with different orientations is used in order to obtain the responses of the fingerprint to them.

In first place, the region of interest is extracted. It is centered 40 pixels down from the core or reference point of the fingerprint, which is obtained using Poincarè method or the covariance-based reference point detection method if no core is found. The region is a spatial tessellation of a circular form whose sectors are distributed in different bands (a total of 48 sectors are used, i.e., 8 sectors in each of the 6 bands). After defining this region, each sector is independently normalized in order to have a common mean and variance of the grey level intensities in all of them. Then, the normalized region is decomposed into four components using Gabor filter with four orientations $(0, \pi/4, \pi/2, 3\pi/4)$. In each component the ridges and valleys in the same orientation as that of the filter become more accentuated, whereas the contrary occurs with the rest. This variation in the grey level depending on the orientation of the valleys and the ridges is indeed the one used to form the feature vector, since the standard deviation of the grey level in each sector is taken as a feature in each one of the four components. Hence, a total of 192 features are obtained. We should remark that in the FingerCode method a fingerprint is rejected in the FE mechanism whenever the whole region of interest could not be placed inside the image.

2.5. Cappelli02

In this work of Cappelli et al. [12], the authors presented the combination of two different approaches for FE and classification. On the one hand, the already mentioned Cappelli99a approach is considered as a part of the feature vector. On the other hand, the registered OM is used to fill the second part of the feature vector.

Hence, the first part of the feature vector is that obtained as described in Section 2.3. Otherwise, the second one is very similar to PCASYS approach, but the OM (obtained with blocks of 16×16) is registered using the core point obtained by Poincarè except for the case when no core is found in which the R92 method is used. Once the OM is registered, it is enhanced following the same approach as that in Cappelli99a. Finally, the orientations are stored in radians in the feature vector, whose size depends on the original image. In the case of NIST-4, 960 features are obtained, which are afterwards processed in the classification model presented by the authors with the Multi-space KL (MKL) transform [11].

2.6. Zhang04

The idea of Zhang04 [82] is similar to that of Karu96, since in this case fixed rules are also used to classify the fingerprints. However, in Zhang04 not only SPs are taken into account, but they are complemented with a pseudo-ridge tracing technique to improve the classification.

The first process in this method is to preprocess the fingerprint image using a Gaussian filter. Then, its segmentation is computed based on the coherence (similar to the variance) of the blocks. With the gradient method [63] the OM is obtained, which is thereafter smoothed following the similarity of the orientations in its neighboring blocks and the already computed coherence. Making use of Poincarè method the SPs are extracted, although differently from other methods a clustering phase is carried out in order to group SPs that are too close to each other and avoid false positives. In case of finding more than two cores or deltas,

the OM is smoothed until a valid number is obtained. Classification of arch and whorl type fingerprints is straightforward. However, when a core and delta pair is obtained, the relative angle between the direction of the core and the direction of the line connecting the core and the delta is used to differentiate among both loops and tented arch. Otherwise, the usage of pseudo-ridges is required in order to decide the class. Both when one core or one delta is found pseudo-ridges are traced from these positions and their endings are used to decide the class. Finally, if two deltas but one or no core are found, the fingerprint is rejected.

2.7. Nyongesa

In Nyongesa method [58], the authors use a feature vector composed of angle differences between the detected SPs, which are complemented with the global orientation in the fingerprint image.

Differently from Karu96 and Zhang04, the SPs are not directly encoded in the feature vector, but are used to compute the features. These features are: angle between cores, angle between deltas, and angles between each one of the cores and the deltas, making a total of six features (if the measurement cannot be taken a 0 is settled). The other feature included is computed as the mean orientation in a window of 16×16 blocks centered in the center of the image. In order to extract these features, the process starts with a preprocessing step in which the fingerprint is binarized using the mean intensity in each block of 8×8 pixels. The OM is extracted with the gray level variance method [52] and thereafter two times smoothed using a window of 15×15. Poincarè method is used to obtain the SPs, but differently from most of the methods it uses a large digital curve (28 pixels), which is decreased if too many SPs are extracted (instead of smoothing the OM, which is done if the first option is not enough to remove spurious SPs). Once these points are extracted, the feature vector computation is straightforward.

2.8. Shah

The work of Shah [67] is rather different from the previous ones. It considers orientation-based features in the feature vector in two different ways (direct/unaltered and registered) but the way the orientations are obtained and the registration method highly differ from other works, since a line detection mechanism is used from which the orientations are extracted. The final feature vector consists of 224 features of which the first 96 refer to the direct orientations and the second 128 to the registered ones.

First of all, the fingerprint is binarized in local windows of 16×16. In order to compute the OM an iterative line detection method is carried out in which eight oriented masks are considered so that the orientations can be then extracted. The line detection mechanism also includes a skeletonization process [16]. Once the lines are detected, the two parts of the feature vector are obtained. The lines are converted into an OM for the first part of the feature vector. To do so, blocks of 16×16 pixels are taken assigning their orientation as the predominant one in the lines inside the block. Then, the two most predominant orientations and their percentage of occurrence in 24 areas in which the OM is divided (taking the center of the image as the center of the fingerprint) are used as features. In the case of the second part of the feature vector a reference point is needed, which is computed as the point of maximum entropy of the orientations in windows of 120×120 pixels (from the skeletonized line image). Thereafter, the feature vector considers the two predominant orientations in the pixels of each block of 15×15 pixels in which the window is divided making a total of 128 features (from 64 blocks).

2.9. Wang07

The work of Wang [74] is based on Zhang04. The main novelty consists of the introduction of a new type of singular point named as *coredelta*, which aims to improve the detection of tented arch fingerprints. In addition, Gaussian-Hermite Moments (GHMs) are used for the computation of the segmentation (besides the detection of the coredelta).

In first place, the computation of GHMs is carried out from the original fingerprint image. These GHMs allow one to segment the fingerprint following the previous work of the authors [75] in which the energy of the GHMs (related to the variations of pixel intensities) is used. Even though the GHMs can also be used for the computation of the gradient from which the OM is extracted in the same manner as in the gradient method, they used the latter method for this purpose. Once the OM is obtained, the Poincarè index is computed,

but in this case the positive and negative cumulative changes in each block are compute independently. As a consequence, their sum gives the result of the classic Poincarè method. If in this way no core and deltas are found, the search for the coredelta starts. To do so, the map of positive cumulative changes is taken and the point reaching the maximum value is extracted. At this point is when the coherence property obtained from the GHMs is used to validate the existence of a coredelta in the region around the extracted point. The new type of SP is detected if a local minimum exists in this region of the coherence map (if it is heterogeneous). The coherence obtained with GHMs reflects whether orientations around the point are similar or not, which is computed using the eigenvalues obtained from the PCA in the neighborhood (greater differences in the eigenvalues imply greater variance of the orientations and lower values of the coherence). Obviously, if a coredelta is detected, the fingerprint is labeled as a tented arch. The rest of the cases are classified following the rules of Zhang04 (including the pseudo-ridges).

2.10. Hong08

The method presented by Hong et al. [29] can be considered as an extension of the method proposed by Jain, since FingerCode is considered together with ten new features obtained from the SPs and the usage of pseudo-ridges.

The way in which the SPs are extracted follows the method given by Karu [35] (but using the gradient method for the computation of the OM), whereas pseudo-ridges are based on Zhang04 [82] method, but in this case, two pseudo-ridges are traced from the center of the fingerprint. The center is assigned as the nearest core point to the center of the image or the center itself if no core point is found. Once the SPs and the pseudo-ridges are computed, ten features are added to the feature vector of the FingerCode (making a total of 202 features). These features are: the number of cores and deltas, and the location and distance of the deltas and the endings points of the pseudo-ridges with respect to the center of the fingerprint. It should be noted that all of these features are nominal since the distance and the relative location are discretized into four and six values, respectively.

2.11. Li

The method of Li [42] is one of the most complex ones since it involves a number of processes and inter-dependencies in order to extract the final feature vector. This feature vector is composed of two type of features: SPs (number/positions and relative measures) and orientations. However, the way in which the orientations are codified highly differs from previous approaches, since the authors propose the usage of a mathematical model to represent the OM with a number of coefficients.

As usual, the first step is the computation of the OM, in this case using the gradient method. Thereafter, the SPs are extracted using the complex-filter method [56], which uses a multi-scale model of the OM for the detection. However, this SPs are not diretly used but they are validated using the first order portrait [68, 69] which reconstructs the orientations in the area of the SP to do so. The next step consists of rotating the fingerprint so that it is aligned with respect to the main core point (that which is closest to the delta points or the reference point if there is no core point) and cropping it to obtain an image of size 320×320 pixels. The most complex part of the algorithm is the modeling of the OM using the non-linear phase portrait model [41] in which a number of coefficients are obtained to represent the OM as better as possible. This is done by following the Weighted Least Squares method using as weights the coherence of the orientations (the reconstructed OM is aimed at being as similar as possible to the originally extracted one). In this model, its order defines the number of coefficients obtained; since the authors consider the sixth order model, 56 coefficients are computed. These are in fact the ones codified in the feature vector but without considering the independent terms (54 are codified). Together with these features, four additional ones are computed: the number of cores and deltas multiplied by the confidence in their estimation (which is obtained with the complex-filters method) and the angle difference between the main core and the delta points (a zero is settled if there is no delta point).

2.12. Liu10

Liu's approach to FE [43] is rather different from previous ones, since only relative measures taken from SPs are considered as features. The same 16 features are obtained in four different scales aiming at being more robust against noise. Hence, a total of 64 features are encoded. All the features are related to the primary core point (the upper one): its position, direction, confidence in the estimation (which can be obtained due to the usage of complex filters) and the rest of the features are taken as relative measures between this core point and the other SPs (distance, direction difference, direction difference between the line connecting the pair of SPs and the direction of the primary core and the confidence in the estimation of each SP).

There are four major steps in which the method can be summarized. First, the OM is extracted using the gradient method with an additional smoothing phase which make use of the concept of consistency [44]. Thereafter, the fingerprint is segmented following the method proposed by Bazen and Gerez [6] (block-wise mean and variance of the pixel intensities are combined with the coherence of the OM). The key phase in this method is the extraction of the SPs. This process is carried out with complex filters and a multi-scale model [45] (with more coarse OMs than in Li's method). Recall that complex filters allow one to obtain a confidence in the estimation of the SPs, but also to define a reference point in the case in which no core point is found. This process is carried out using four different scales of the OM (the original plus three coarser ones) in such a way the SP detection is more reliable. In the last step, the feature vector is formed of the measures described using the SPs detected.

2.13. Leung

Leung and Leung [40] developed a modification of the FingerCode. Even though the core idea is the same, i.e., the extraction of the FingerCode, the way of doing it has several differences. The first of them is that the FingerCode has 816 features due to the 68 sectors in which the region of interest is divided and the 12 different orientations used in the Gabor filter. Moreover, the shape of the sectors is also changed and the usage of square sectors is proposed.

Besides the FingerCode itself, the differences with respect to Jain's approach starts from the initial steps. In this method, the first step consists of enhancing the fingerprint image following Hong's et al. method [30]. This preprocessing aims at improving the fingerprint quality in four phases: image normalization, OM extraction (gradient method), ridge-frequency computation and filtering using the Gabor filter (which takes into account the orientation and ridge-frequency in each block). Afterwards, the fingerprint is segmented using the variance of the gray-level intensities in the same orientations as those of the ridges [64]. The last step before extracting the FingerCode consists of establishing the reference point by R92 method (instead of Poincarè and the covariances). Once it has been obtained, the original process is followed. Nevertheless, we should note that Leung's method does not reject fingerprints whenever the region of interest does not fall in the fingerprint image (as Jain's does), but it sets the sectors in those areas in the feature vector as missing values, which are then processed in the learning phase (in this case, they are substituted by the mean value of the same sectors in the fingerprints of the same class).

2.14. Le

The method presented by Le and Van [39] is also based on FingerCode, but it introduces a series of modifications in the FE process as well as in the FingerCode extraction in order to make it invariant to the rotation of the fingerprints. Furthermore, as in the case of Leung, the number of sectors is altered although in this case the shape is maintained. In total, 1280 features form the FingerCode obtained from 10 bands with 16 sectors each and 8 different orientations of the Gabor filter. The other major modification is the detection of the reference point in order to obtain the region of interest, which is done by the combination of two techniques: complex filters and a new feature named as VORIV.

The FE begins with the extraction of the OM. Although the gradient method is considered, the authors propose the usage of a three scale model [53], in which three OMs with different block sizes are extracted first (using the normalized gradients); then, these maps are combined into a new one, which is further processed in order to fill in the orientations that cannot be reliably estimated with the OM fusion. Once

the OM is obtained, the reference point detection is carried out (the main contribution of the paper). The authors emphasize the fact that only concave cores are sought. In order to do so, two different models are combined. On the one hand, complex filters are considered in the same way as in Li's method [42] where a polinomial approximation with a gaussian filter is used instead of Euler's formula (which is considered in Liu [43]). Moreover, in this case a complex filter of second order is used termed semi-radial symmetry filter. On the other hand, VORIV feature is introduced with the same aim, which is more related to Poincarè method. This feature consists of summing up the angle differences between the orientations in the same column inside each window of 4×5. That is, the vertical variation of the orientations is used to compute the reference point. Finally, the block reaching the highest mean value of both approaches is established as reference point for the FingerCode. At last, the FingerCode is extracted following the original model of Jain [32] but making it invariant to the rotation of the fingerprints [33]. This is achieved by computing the orientation of the reference point as that with the least variation with respect to the orientations in a window along the tested one [44] and rotating the fingerprint accordingly.

3. Classification techniques

In this section we describe the different classifier learning models that we have considered for the experimental study. First, general purpose classifier learning algorithms are described (Subsection 3.1). Then, specific classification models developed in conjunction with a FE method are presented (Subsection 3.2).

3.1. General purpose classification methods

The choice of these algorithm was made on the basis of their good behavior in a large number of real world problem, which is supported by the fact that all of them have been included in the list of the top-ten data mining algorithms [81].

3.1.1. SVM

Support Vector Machine (SVM) [73] maps the original input space into a higher-dimensional feature space using a certain kernel function, making possible the avoidance of the computation of the inner product of two vectors. Once the instances are mapped to the feature space, the optimal separating hyperplane (that with the maximal margin) is computed. This way, an upper bound of the expected risk is minimized instead of the empirical risk. In order to carry out the training of the SVMs, we use the Sequential Minimal Optimization (SMO) procedure [61].

3.1.2. C4.5

C4.5 [62] is a decision tree learning algorithm. In this method, classification rules are extracted in the form of decision trees starting from a set of given examples. The decision tree is generated in a top-down manner. The normalized information gain (difference in entropy) is used as a criterion to decide which attribute should be used for splitting the data in each node of the decision tree. The attribute is selected so that it maximizes the normalized information gain and the process is repeated until the tree is fully grown. Finally, the decision tree is pruned following the pessimistic pruning procedure.

3.1.3. kNN

k−Nearest Neighbours (kNN) [50] is a lazy learning algorithm because it makes use of the whole training set as a reference set to classify new instances. In order to do so, it finds the group of the k closest instances in the training set to the test instances. From these k neighbor instances a decision is made based on the predominance of a particular class. As a consequence, both the distance metric used to compute the closeness of the instances and the number of neighbors considered are key elements in this method. In order to find the best value for these parameters a cross-validation procedure can be followed using the available training data.

3.2. Specific Classification Methods for Fingerprint Classification

This subsection introduces ad-hoc classification methods designed for an specific FE method.

3.2.1. Jain

Jain's et al. proposal for classification [32] considers a two-level multi-classifier so as to ease the classification problem. In order to do so, they decompose the original 5-class problem into easier-to-solve 10 binary classification problem (all the possible pairs of classes), which is the same as the commonly used One-vs-One (OVO) strategy [24, 26]. However, instead of directly classifying the input instance with the OVO system, they carry out a preliminary phase in which the two most probable classes are obtained with the kNN algorithm using a large neighborhood ($k = 10$). Hence, the two classes with the largest number of neighbors are considered for the next phase. In the second phase, only the classifier considering both classes is used to output the final class. Neural networks are used to learn the classifiers considered in this last phase.

3.2.2. Nyongesa

Nyongesa et al. [58] proposed the usage of three types of neural networks: multi-layer perceptron (MLP), radial basis function (RBF) and fuzzy neural network (FNN). From the results of the paper, MLP was the one reaching the best accuracy, and this is why we have selected it as the specific classifier for Nyongesa's feature vector. The MLP used consists of two hidden layers and five nodes in each one. The weights are learned using the back-propagation learning technique.

3.2.3. Shah

Among the three different classification approaches presented by Shah and Sastry [67], we have considered the most general classification approach, but also the one with the best results in the paper. In this method, the authors take advantage of the SVMs learned to differentiate among the classes in order to identify the most meaningful fingerprints in the training set. In SVMs these fingerprints are selected as support vectors, in which the classification of the new instances is also based in the case of SVMs. However, the authors proposed to extract them and use the feature vectors from these fingerprints as a reference set for the kNN algorithm (with $k = 1$).

3.2.4. Hong08

Hong et al. [29] also considered the decomposition of the original problem as in Jains' work [32], but in this case the One-vs-All (OVA) [65] strategy is considered. In this strategy, the original problem is divided into as many binary problems as classes, so that in each problem a class is distinguished from the rest. These new problems are learned by SVMs in their proposal. Commonly, in OVA framework, the instance is submitted to all the classifiers and the one giving a possitive output is used to classify the instance. However, the problem arises when more than one classifier give a possitive answer in which case the tie needs to be broken. In order to do so, the authors propose to break the tie a priori by dynamically ordering the SVMs using a Naïve Bayes classifier [17]. Hence, these second classifier allows them to set an order in the execution of the SVMs (from greater to lower probable classes) and the first one outputting a possitive answer is used to label the instance. We should emphasize that different features are used for each process, since the FingerCode is considered in the SVMs, whereas the SPs and pseudo-ridge features are used in Naïve Bayes.

3.2.5. Liu10

Liu's approach [43] considers the usage of multiple classifier systems or ensembles [38, 66] for the fingerprint classification problem. More specifically, the logistic regression version of the well-known ensemble learning algorithm Adaboost [15] is considered. Recall that Adaboost performs a sequential learning of several classifiers by giving more weights to the examples that are misclassified in the previous iterations in order to learn each new classifier. As a weak classifier, Liu makes use of decision trees with a fixed number of nodes (20), which are learned using Gini split criterion [7]. However, this would not be a specific fingerprint classification model if it were not for the fact that each part of the Liu's feature vector (recall that 16 features are obtained in 4 different scales) is learned with Adaboost, making the final model an ensemble of ensembles. Moreover, since the proposed model only works with binary problems, the author uses OVA decomposition in order to address the multi-class problem.

3.2.6. Leung

Even though the original model of Leung and Leung [40] is designed for fingerprint retrieval rather than classification, it can be easily considered for this purpose, which is how we have implemented this model in the current paper. First of all, as mentioned earlier, Leung's approach for FE introduces missing values in the feature vector and they should be corrected prior to the learning phase. This process has been carried out both for this specific model but also for the general ones. The authors propose the usage of the mean imputation approach [20], in which the missing values are substituted by the mean value of the attribute in the instances of the same class. Since the class of the instance is unknown in test, the mean value of all the instances is used in this phase. With the corrected feature vectors, the authors use Fisher's Linear Discriminant Analysis [23] as a dimensionality reduction technique, allowing to represent the training data as a reduced set of very informative features. Afterwards, the classification is performed by means of a Quadratic Discriminant Analysis classifier [51].

4. Experimental framework

In this section, we introduce the experimental set-up used to carry out the experiments. Our aim is to set a common framework which will enable us to objectively compare the FE methods reviewed as well as the specific classification model developed for fingerprint classification. Hence, we describe the databases considered (Subsection 4.1), the evaluation procedure followed (Subsection 4.2), the performance measures used (Subsection 4.3) and the parameters applied in the experiments (Subsection 4.4).

4.1. Testing databases

In fingerprint classification there is a database which has been the most used benchmark in the literature. This database is the NIST (National Institute of Standards and Technology) Special Database 4 (NIST-4) [77]. Therefore, we could not leave it out of this comparison, since most of the proposed methods have been developed to work with it. However, it may be a biased benchmark due to the same reason. On this account, we have considered other three databases with fingerprints of different qualities that have been generated using SFinGe [13, 49] software tool. In this way, we can carry out a less biased comparison with two different sources of fingerprints and we are able to study the robustness of the methods against noise and the usage of different types of fingerprints. Next, we describe both types of databases and we give details on their composition.

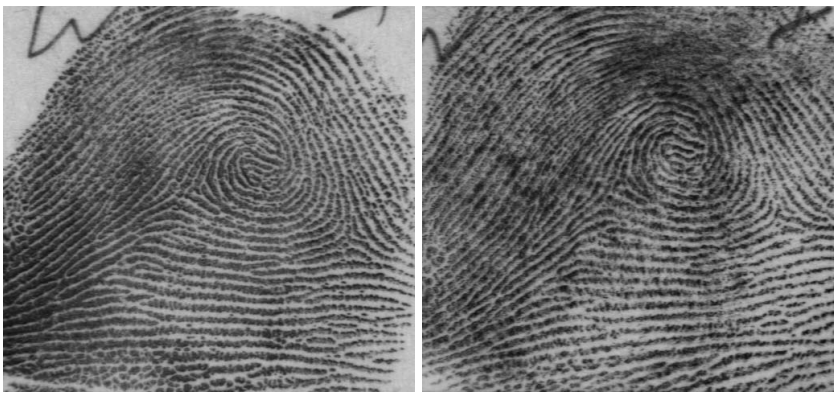
4.1.1. NIST-4

This database is the classical one for fingerprint classification papers although it may be far from the fingerprints that can be found in real systems due to the way they were obtained. The fingerprints in this database are scanned from rolled inked impressions on cards. However, the fact that it has been almost the unique database with easy access for researchers having labeled examples has made it to be the reference benchmarking tool for fingerprint classification systems.

In the NIST-4 database there are 4000 fingerprints (of 512×480 pixels) taken from 2000 fingers, that is, two captions of each fingerprint are present (denoted as F and S). Furthermore, each fingerprint is labeled with the corresponding class (A, T, L, R or W), but there are a number of fingerprints (350, that is, 17.5%) which are labeled with two classes. As we mentioned in the discussion of the first part of this paper [27] the authors proceeded differently in order to address this issue. In this work, in order to make the comparison as fair as possible and to address it as a standard classification problem, we have decided to remove those fingerprints with two class labels, both for training and testing, so that they are not taken into account in our experiments, and hence our NIST-4 databases is formed of 1650 fingers and 3300 fingerprint images. From our point of view this is the fairest method to carry out the comparison and it makes the classification problem harder, since many methods consider the classification of those ambiguous fingerprints as correct if they were classified into any of both classes (they were easier-to-classify examples).

Similarly, we have divided the original database into two databases so that the same fingers cannot be used for training and testing. Therefore, F database is formed of the first captions of the fingers and S

database with the second ones. That is, we will obtain the results for two independent executions (one for each database F and S). To give an example of the fingerprints in this database, Fig. 1 shows the two captures of the first finger that can be found in this database.



(a) F0001_01 (b) S0001_01

Figure 1: The two captions of the first fingerprint in NIST-4 database.

Finally, we should mention that the classes in NIST-4 are evenly distributed, with 400 fingerprints pairs of each class. This may be better to perform the evaluation of the classification over all the classes, but at the same time it is further from the reality where the class distribution is uneven. For this reason, in the case of SFinGe we will deal with the real class distribution both for training and testing.

4.1.2. SFiGe generated databases

In order to carry out a complete study we have considered the usage of a second source of fingerprints. The problem with fingerprint classification is that labeled fingerprints are needed to perform the evaluation. On this account, we have made use of SFinGe software tool [49, 13], which allows one to generate synthetic fingerprints with a realistic appearance and with true class labels. This fact helps in automatizing the process and makes possible to perform the evaluation of the FE and classification techniques for fingerprints with different quality levels (translations, rotations and geometric transformations).

The synthetic fingerprints generated by SFinGe simulate fingerprints obtained by optical scanners, which may be nearer to real systems than NIST-4 ones. Moreover, SFinGe's quality should be highlighted. It has been already used in several editions of the Fingerprint Verification Competition (FVC) [46, 47, 48, 9, 18] obtaining similar results to real fingerprint databases, being able to capture the nature of real fingerprints.

In order to simulate different scenarios, we have considered three different quality profiles in the generation of the fingerprints, which are provided by SFinGe. Afterwards, we briefly describe these profiles (the rest of the parameters used for the generation of the fingerprints in SFinGe tool are shown in Table 2):

- *High Quality No Perturbations* (HQNoPert): Fingerprints are generated with high quality without carrying out any kind of perturbation on the fingerprint.
- *Default*: The fingerprints generated are of middle quality and slight localization and rotation perturbations are performed.
- *Varying Quality and Perturbations* (VQandPert): The fingerprint database is composed of fingerprints captions of varying qualities. Furthermore, these fingerprints receive a perturbation treatment in which location, rotation and geometric distortions are applied to the fingerprints.

Examples of the fingerprints generated by SFinGe can be shown in Fig. 2, where the three qualities are represented from higher (2a) to lower (2c). The differences between these fingerprints and those in Fig. 1 can be easily appreciated. First of all, there is a big difference in terms of size (which in the paper is not

Table 2: Parameter specification for SFinGe tool

Scanner parameters
Acquisition area: 0.58" × 0.77" (14.6mm × 19.6mm)
Resolution: 500 dpi, Image size: 288 × 384
Background type: Optical
Background noise: Default
Crop borders: 0 x 0
Generation parameters
Seed: 1
Impression per finger: 25 (only the first one is used)
Class distribution: Natural
Generate pores: enabled
Save ISO templates: enabled
Output settings
Output file type: WSQ

greatly appreciated), whereas NIST-4 fingerprints size is 512×480, SFinGe generated ones size is 288×384. The size of the fingerprints but also the thickness of the ridges also differs. This fact makes difficult the usage of the reviewed FE methods with these fingerprints, because their properties are too different, but this also helps in studying the robustness of the methods. Consequently, the parameters of the FE methods must be adapted to this characteristics (since they are designed to work with NIST-4). On this account, we have adapted all the parameters of the FE methods related to the size of the ridges or the dimensions of the images according to an adjustment parameter δ, which we have set for all the FE methods to be the same ($\delta = 0.5625$, settled according to the ridge thickness). Even though this may not be the best parameter for all the methods, we will test their robustness against a common configuration where no method is tuned for the best performance. Our assumption is that the best method in this framework would also be the best method if they would be tuned. There is a case in which we developed a different approach, PCASYS, whose original implementation only allows for 512×480 images. For this case, we have adapted the size of the image maintaining the height to width ratio of the original one and extended the background to fit the new size. The same is done with both of Cappelli's works, since some phases are shared. In the cases of Le and Nyongesa $\delta = 1$ is settled in order to work properly given the original value of the parameters.

At last, we should mention that we have generated 10000 fingerprints of each quality, in bags of 2000, which will help us in carrying out a suitable evaluation procedure (explained in the next Subsection). Furthermore, in these databases we have considered the real distribution of the classes in the database, and hence different numbers of arch, tented arch, right loop, left loop and whorl type fingerprint are generated according to the real distribution: 3.7%, 2.9%, 31.7%, 33.8% and 27.9% ($370, 290, 3170, 3380$ and 2790 fingerprints of each quality, respectively).

4.2. Evaluation procedure

In order to evaluate the quality of the studied approaches we have followed a pure machine learning perspective. The majority of the reviewed papers estimate the quality of their methods using a part of the database for training and the rest for testing. This procedure being carried out only once. Hence, the results obtained can be dependent of the partitioning used, and hence they can be biased. A usual approach in the case of NIST-4 is to use the first (F) impressions for training and the second (S) ones for testing. Even though it might be a better approach the fact that the method are run once make them not very reliable, besides the fact that very similar fingerprints are used for training and testing (different impressions of the same fingerprint) which can ease the classification, and therefore give optimistic results.

(a) HQNoPert (b) Default (c) VQandPert

Figure 2: Three fingerprints generated by SFinGe.

On this account and aiming at performing a fairer and more complete comparison from the machine learning point of view, we have used a 5-fold Stratified Cross-Validation (SCV) scheme [55]. That is, the database is split into 5 folds, each one containing 20% of the patterns (fingerprints) of the database. For each fold, the classification model is learned with the examples (feature vectors of the fingerprints) contained in the remaining folds and then tested with the current fold. Hence, the overall results of a method in a database are obtained from averaging five executions, making it a more reliable estimation of its performance (the performance measures are described in Subsection 4.3).

An important point when evaluating the FE methods is the fact that some of them have a rejection mechanism in which some fingerprints are left without class label due to different reasons. Thus, in these cases, we will show the rejection rate near the performance obtained, that is, the percentage of fingerprints that were rejected in the FE process. Obviously, higher rejection rates should lead to greater performance. The importance of rejection should be taken into account, since rejecting too many fingerprint may not be helpful, but detecting those bad quality fingerprints that could be mismatched could help one in reducing the time that would be lost in the matching phase.

Due to the possibility of the rejection, we have carefully carried out the partitioning for the 5-fold SCV so that the same fingerprints fall in the same folds for all the FE and classification methods (all of them are evaluated in the same training and testing sets). On this account, we carry out the partitioning prior to the FE process. This is why we have performed the generation of each database of 10000 fingerprints in bags of 2000 as explained in the previous section (4.1) in the case of SFinGe (similarly, NIST-4 fingerprints are partitioned and then the rejected ones are deleted from the corresponding partitions). In this manner, we extract the feature vectors for each bag, assigning the fingerprints obtained (without the rejected ones) to a fold of the 5-fold SCV scheme. From our point of view, this is the fairest way to evaluate all the methods in the same conditions.

We should notice that carrying out the partitioning this way is much fairer in this case than considering other techniques for performing the cross-validation such as Kennard and Stone model [37] or the Distribution Optimally Balanced Stratified Cross Validation [55], which aims to avoid the data-set shift problem in the evaluation of the classification problems (when the training and testing data do not follow the same distribution) [54]. Using these types of methods would need to perform the partitioning once the features are extracted, and hence the partitioning would depend on the FE method. As a consequence, different FE methods would be evaluated in different training and test sets, which is not desirable.

4.3. Performance measures

Regarding the performance measures used to evaluate the different methods, we contemplate two different metrics considering the properties of the problem we are dealing with (multiple classes and uneven class

distribution in the case of SFinGe). As it is done in all the classification works, we must use the classification rate, however, we complement it with Cohen's kappa metric in such a way that more complete conclusions can be yielded:

- *Classification rate:* also called accuracy rate, is the number of correctly classified examples (successful hits) relative to the total number of classified examples. It is the most commonly used metric for assessing the performance of classifiers [4, 80].
- *Cohen's kappa:* is an alternative measure to *classification rate*, which compensates for random hits [14, 70]. In contrast to classification rate, kappa evaluates the portion of hits that can be attributed to the classifier itself (i.e., not to mere chance), relative to all the classifications that cannot be attributed to chance alone. An easy way of computing Cohen's kappa is by making use of the resulting confusion matrix (3) in a classification task. From this matrix, Cohen's kappa is computed as follows:

Table 3: Confusion matrix for an m-class problem

	Predicted Class				
Correct Class	C_1	C_2	$\dots$	C_m	Total
C_1	h_{11}	h_{12}	$\dots$	h_{1m}	T_{r1}
C_2	h_{21}	h_{22}	$\dots$	h_{2m}	T_{r2}
$\vdots$			$\ddots$		$\vdots$
C_m	h_{m1}	h_{m2}	$\dots$	h_{mm}	T_{rm}
$Total$	T_{c1}	T_{c2}	$\dots$	T_{cm}	T

$$kappa = \frac{n\sum_{i=1}^{m} h_{ii} - \sum_{i=1}^{m} T_{ri}T_{ci}}{n^2 - \sum_{i=1}^{m} T_{ri}T_{ci}}, \tag{1}$$

where h_{ii} is the cell count in the main diagonal (the number of true positives for each class), n is the number of examples, m is the number of class labels, and T_{ri}. T_{ci} are the rows' and columns' total counts, respectively ($T_{ri} = \sum_{j=1}^{m} h_{ij}$, $T_{ci} = \sum_{j=1}^{m} h_{ji}$). Cohen's kappa ranges from -1 (total disagreement) through 0 (random classification) to 1 (perfect agreement). For multi-class problems, kappa is a very useful, yet simple, meter for measuring a classifier's classification rate while compensating for random successes. The main difference between the classification rate and Cohen's kappa is the scoring of the correct classifications. Classification rate scores all the successes over all classes, whereas Cohen's kappa scores the successes independently for each class and aggregates them. The second way of scoring is less sensitive to randomness caused by a different number of examples in each class. This measure has also been widely used for assessing classification performance [21, 24, 28, 25].

4.4. Parameter specification

In this subsection we give details on the parameters used in the different learning algorithms considered along the experimental study, both for the general and the specific ones. These parameters can be observed in Table 4 (the parameters of the specific models are established according to the recommendations of the authors). Notice that we have used three different kernel configurations for SVMs, since results can highly differ due to this parameter and depending on the features used. This way we are able to obtain a useful configuration for all the methods. Similarly, we have considered two different values of k (number of neighbors) in kNN algorithm and two different distance metrics to compute the neighborhoods (Euclidean and HVDM [79], which is more suitable to work with nominal values). Notice that we do not tune the parameters of the classifiers for each FE method as in the case of the adjust parameter δ. Even though the tuning of the parameters for each method on each particular problem could lead to better results, we preferred to maintain a baseline performance on each method as the basis for comparison. Since we are not comparing base classifiers among them, our hypothesis is that the FE methods that perform better with

this configuration, would also do so if a better setting would have been performed. Moreover, when methods are not tuned, winner methods tend to correspond to the most robust ones, which is also desirable. All the specific classification models have been developed under KEEL software [3, 2] and the implementations of the classical classifiers used are also from this toolkit.

Table 4: Parameter specification for the classifiers employed in the experimentation.

General purpose classification methods		
Algorithm		*Parameters*
C4.5		Prune = True, Confidence level = 0.25 Minimum number of item-sets per leaf = 2
SVM	Poly	C = 1.0, Tolerance Parameter = 0.001, Epsilon = 1.0E-12 Kernel Type = Polynomial, Polynomial Degree = 1 Fit Logistic Models = True
	RBF	C = 1.0, Tolerance Parameter = 0.001, Epsilon = 1.0E-12 Kernel Type = RBF, RBFKernel $\gamma = 0.01$ Fit Logistic Models = True
	Puk	C = 1.0, Tolerance Parameter = 0.001, Epsilon = 1.0E-12 Kernel Type = Puk, PukKernel $\omega = 1.0$, PukKernel $\sigma = 1.0$ Fit Logistic Models = True
kNN	$k = 5$	Distance metric = Euclidean Distance metric = HVDM
	$k = 10$	Distance metric = Euclidean Distance metric = HVDM
Specific classification methods for fingerprint classification		
Algorithm		*Parameters*
Jain		$k = 10$, Distance metric = Euclidean, Hidden layers = 1 Hidden nodes = 20, Learning rate = 0.01, Momentum = 0.7
Nyongesa		Hidden layers = 2, Hidden nodes = 5, Transfer function = Log Momentum terms $\eta = 0.01$ and $\alpha = 0.01$, Decay term $\lambda = 0.0$ Iterations = 2500000, Typify inputs = True
Shah		C = 1.0, Tolerance Parameter = 0.001, Epsilon = 1.0E-12 Kernel Type = RBF, RBFKernel $\gamma = 0.0005$ Fit Logistic Models = False
Hong08		C = 1.0, Tolerance Parameter = 0.001, Epsilon = 1.0E-12 Kernel Type = RBF, RBFKernel $\gamma = 0.0625$ Fit Logistic Models = False
Liu10		Maximum leafs = 20, Number of trees = 4
Leung		Fisher LDA dimensions = 4, Regulariztion eigenvalue = -1 Regularization value = 0, QDA regularization value = 0

In addition, as a brief summary of the FE methods considered, we include the type of features used, the number of features ("–" if they perform a fixed classification) that are extracted in each method and whether it can reject fingerprints or not. This information is given in Table 5. Finally, due to the large number of parameters involved in the development of the FE methods, we cannot include them in this paper, but we refer the reader to the original papers or to the source code provided in order to check them.

Table 5: Brief summary of the FE methods tested.

Name	Features	No. features	Reject?	Ref.
Candela	OM	128	No	[8]
Karu	SPs	–	No	[35]
Cappelli99a	OM	5	No	[10]
Jain	Filter resp. (Gabor)	192	Yes	[32]
Cappelli02	OM (seg. + reg.)	357 (5 + 352)	No	[12]
Zhang04	SPs + Ridge trac.	–	Yes	[82]
Nyongesa	SPs + OM	7 (6 + 1)	No	[58]
Shah	OM (dir. + reg.)	224 (96 + 128)	No	[67]
Wang07	SPs + Ridge trac.	–	Yes	[74]
Hong08	Jain + SPs + Ridge trac.	202 (192 + 10)	Yes	[29]
Li	OM + SPs	58 (54 + 4)	No	[42]
Liu10	SPs	64 ($16 \cdot 4$)	No	[43]
Leung	Filter resp. (Gabor)	816	No	[40]
Le	Filter resp. (Gabor)	1280	Yes	[39]

4.5. Web page associated with the paper

In order to provide additional material to the paper content, we have developed a Web page at (http://sci2s.ugr.es/fingerprintClassification) in which we have included the following complementary material:

- The source code of the FE module developed for fingerprint classification, including all the FE methods tested.
- The source code of the specific classification models developed to deal with the fingerprint classification problem (the classical ones can be directly found in `http://www.keel.es`).
- Finally, we include all the datasets obtained by the FE methods for the different databases tested (SFinGe and NIST-4). This way, other researchers will be able to use them as benchmarks when testing new classification approaches.

5. Experimental analysis and discussion

Once we have properly established the experimental framework, we can present the results obtained with each FE method in each database with the different classifiers considered. With this aim, we have divided this section into five subsections. In the first four ones we will analyze the results in each database considered (both NIST-4 F and S are analyzed together, since their difficulty is supposed to be similar). Additionaly, the last subsection will summarize the results obtained in each database as a whole, showing which are the best and most robust methods.

5.1. Results on HQNoPert database

HQNoPert database is a priori the easiest one, since no perturbations are applied to the fingerprints. The testing results in terms of accuracy and kappa obtained for each FE method and classifier can be observed in Table 6. The results obtained with accuracy measure are presented on the left part of the table, whereas those obtained with kappa are shown on the right part. The best performance in each row (in each FE method) is highlighted in **bold-face**, whereas the best result for each classifier (each column) is underlined (independently for accuracy and kappa). The column in the middle between both accuracy and kappa results presents the rejection rates produced by the corresponding FE method if it has this option. Otherwise a "–" is shown, the same as in the case that the combination between the FE method and the classifier does not make sense (fixed classifiers with trainable features and FE methods with fixed classification with the

rest of the classifiers). Notice that in the case of Cappelli99a the fixed classification result as well as the results for the rest of the classifiers are shown since, as noted by the authors, a fixed classification can also be performed by taking the class of the mask with the minimum matching cost as the predicted one.

Table 6: Accuracy and kappa results in High Quality and No Perturbations (HQNoPert) database.

Accuracy			SVM			5NN		10NN				Kappa			SVM			5NN		10NN		
	Fixed	C4.5	Poly	RBF	Puk	Eucl.	HVDM	Eucl.	HVDM	Specific	**Rej. %**		Fixed	C4.5	Poly	RBF	Puk	Eucl.	HVDM	Eucl.	HVDM	Specific
Candela	–	87.61	**95.82**	93.68	93.72	88.48	87.38	86.79	85.93	–	–	Candela	–	.8244	**.9407**	.9103	.9116	.8345	.8187	.8098	.7974	–
Karu	**74.49**	–	–	–	–	–	–	–	–	–	–	Karu	**.6685**	–	–	–	–	–	–	–	–	–
Cappelli99a	74.64	80.77	82.82	81.18	**83.55**	81.65	81.87	82.69	82.91	–	–	Cappelli99a	.6506	.7256	.7538	.7306	**.7642**	.7382	.7414	.7527	.7559	–
Jain	–	93.67	96.23	95.94	95.00	**96.46**	96.30	96.31	96.17	96.42	1.44	Jain	–	.9102	.9464	.9423	.9294	**.9496**	.9473	.9474	.9455	.9490
Cappelli02	–	92.40	89.87	**93.87**	59.50	74.84	88.72	74.51	87.91	–	–	Cappelli02	–	.8922	.8561	**.9131**	.4963	.6418	.8382	.6359	.8263	–
Zhang04	**89.57**	–	–	–	–	–	–	–	–	–	0.70	Zhang04	**.8555**	–	–	–	–	–	–	–	–	–
Nyongesa	–	85.45	76.59	79.67	85.24	84.25	84.14	85.28	85.15	**85.53**	–	Nyongesa	–	.7932	.6567	.7088	.7898	.7759	.7743	.7904	.7885	**.7938**
Shah	–	85.24	**91.55**	88.14	60.73	79.36	88.50	79.15	87.93	76.21	–	Shah	–	.7899	**.8792**	.8349	.4983	.7014	.8352	.6975	.8269	.6600
Wang07	**87.81**	–	–	–	–	–	–	–	–	–	0.96	Wang07	**.8318**	–	–	–	–	–	–	–	–	–
Hong08	–	95.95	**97.32**	97.12	96.09	96.67	96.68	96.55	96.58	96.99	1.44	Hong08	–	.9425	**.9620**	.9591	.9446	.9527	.9528	.9509	.9514	.9574
Li	–	**89.20**	74.26	75.05	83.50	85.55	88.67	85.73	87.94	–	–	Li	–	**.8465**	.6314	.6432	.7644	.7938	.8383	.7962	.8276	–
Liu10	–	**94.75**	93.09	92.26	94.20	94.00	94.30	94.04	94.40	94.55	–	Liu10	–	**.9254**	.9012	.8893	.9176	.9142	.9186	.9148	.9200	.9225
Leung	–	57.26	86.39	84.74	72.87	**92.73**	88.21	92.40	88.74	87.38	–	Leung	–	.3984	.8062	.7840	.6481	**.8962**	.8319	.8915	.8393	.8225
Le	–	74.12	77.65	**86.54**	60.38	81.40	81.95	81.44	82.28	–	3.42	Le	–	.6297	.6800	**.8069**	.4747	.7313	.7394	.7314	.7437	–

Looking at Table 6 the following conclusions can be drawn:

- Regarding fixed classification approaches, Zhang04 is the best performing one (89.57% accuracy) and Wang07 has a lower performance (87.81% accuracy) even though it was proposed as an improvement of Zhang's work. This difference is even greater with kappa (.8555 vs. .8318). Karu and Cappelli99a are not able to obtain competitive results.

- Overall, Hong08 is the most dominant method outperforming the rest of the approaches with any classifier both with accuracy and kappa and having a low rejection rate (1.44%). It is also the one reaching the best performance (97.32% accuracy and .9620 kappa using SVM with polynomial kernel). Moreover, its robustness must be highlighted, being able to obtain good results with every classifier. Recall that Hong08 uses three types of features (FingerCode + SPs + Ridge-tracing), which seems to be a good strategy.

- Specific classification approaches were properly selected by most of the authors, reaching similar performances to those obtained by the classical classifiers. However, only in the case of Nyongesa the result of the specific classifier is better than all the rest. Other exceptions are Shah and Leung in which the results obtained by these classifiers do not produce good performances.

- With respect to similar methods, Cappelli02, which uses the features of Cappelli99a, make a difference with respect to the latter. Nevertheless, this method is highly dependent of the classifier used (kNN does not work well and neither does SVM with Puk kernel due to how the orientation features are codified, since these classifiers are not able to model the non-existing discontinuity in the angles codified in the feature vector). Similarly, the new features introduced by Hong08 in Jain's FingerCode has led to an improvement (but not so large in this case).

- Other methods such as PCASYS (Candela) or Leung also perform well with the proper classifier (SVM with polynomial kernel and kNN with Euclidean distance, respectively). Liu10 also reaches a very competitive result (94.75% accuracy, .9254 kappa) but being much more robust with almost the same result with every classifier.

- Le method does not work as expected despite being the one with the largest rejection rate (3.42%). The new reference point and the translation invariant FigerCode together with the large number of

features seem not to be beneficial for the classification. Nyongesa, Shah and Li also perform worse than the rest (the first one seems to be too simple with only 7 features, the approach to FE of Shah does not properly capture the features needed for fingerprint classification probably due to the line detection-based orientation computation, whereas the complexity of Li's approach may be responsible of its results).

- With respect to the different configurations used for SVMs and kNN, we should mention that there are major differences in some cases due to the type of features used, or better said, due to what they represent. For instance, this fact can be clearly observed in the cases where HVDM distance works much better than Euclidean one (Cappelli02, Shah, Li).

- Finally, we should point out that similar conclusions are obtained with accuracy and kappa, but the latter one shows a greater difference between methods, mainly highlighting the good behavior of Hong08.

5.2. Results on Default database

The results obtained on Default database created by SFinGe software are shown in Table 7. The structure of the table follows the same as that of Table 6, where the results for HQNoPert have been presented.

Table 7: Accuracy and kappa results in Default database.

Accuracy			SVM			5NN		10NN				Kappa			SVM			5NN		10NN		
	Fixed	C4.5	Poly	RBF	Puk	Eucl.	HVDM	Eucl.	HVDM	Specific	**Rej. %**		Fixed	C4.5	Poly	RBF	Puk	Eucl.	HVDM	Eucl.	HVDM	Specific
Candela	–	84.58	**92.21**	90.27	88.08	85.41	83.81	84.11	83.07	–	–	Candela	–	.7810	**.8895**	.8617	.8364	.7899	.7667	.7709	.7557	–
Karu	**69.40**	–	–	–	–	–	–	–	–	–	–	Karu	**.6049**	–	–	–	–	–	–	–	–	–
Cappelli99a	70.43	77.84	79.64	78.34	**80.67**	78.63	78.55	79.69	79.65	–	–	Cappelli99a	.5989	.6840	.7078	.6889	**.7222**	.6950	.6941	.7098	.7092	–
Jain	–	91.65	95.50	94.92	93.92	**95.66**	95.65	95.58	95.49	95.60	5.38	Jain	–	.8816	.9360	.9276	.9145	**.9381**	.9380	.9370	.9356	.9375
Cappelli02	–	91.25	86.76	**92.10**	59.93	73.83	87.32	74.59	86.53	–	–	Cappelli02	–	.8759	.8125	**.8878**	.4999	.6248	.8180	.6341	.8064	–
Zhang04	**84.09**	–	–	–	–	–	–	–	–	–	1.27	Zhang04	**.7820**	–	–	–	–	–	–	–	–	–
Nyongesa	–	**81.31**	72.41	73.38	81.29	79.96	80.10	81.00	80.92	81.18	–	Nyongesa	–	**.7324**	.5952	.6133	.7307	.7140	.7160	.7284	.7272	.7303
Shah	–	76.21	79.67	**85.30**	52.86	66.64	81.58	65.74	81.92	62.91	–	Shah	–	.6602	.7101	**.7914**	.3979	.5125	.7361	.4970	.7401	.4672
Wang07	**76.27**	–	–	–	–	–	–	–	–	–	2.74	Wang07	**.6822**	–	–	–	–	–	–	–	–	–
Hong08	–	94.02	**96.29**	95.80	94.71	96.07	96.00	96.00	95.93	95.94	5.38	Hong08	–	.9150	**.9473**	.9404	.9250	.9440	.9430	.9429	.9421	.9426
Li	–	83.28	68.49	69.62	77.44	80.50	**83.78**	81.13	83.31	–	–	Li	–	.7622	.5481	.5638	.6773	.7217	**.7682**	.7304	.7610	–
Liu10	–	**93.96**	93.17	92.12	93.75	93.20	93.75	93.17	93.78	93.39	–	Liu10	–	**.9142**	.9025	.8875	.9111	.9028	.9107	.9023	.9111	.9063
Leung	–	50.65	82.53	82.15	70.76	**91.50**	86.74	91.40	87.68	83.94	–	Leung	–	.3079	.7510	.7468	.6224	**.8787**	.8112	.8773	.8244	.7730
Le	–	66.83	71.17	**82.17**	57.62	76.94	78.16	77.52	78.64	–	5.68	Le	–	.5260	.5872	**.7441**	.4371	.6667	.6846	.6747	.6908	–

Following the previous analysis, the next facts can be observed in these results (results are worse given that the database is supposed to be more difficult):

- Regarding fixed classification, Zhang04 continues being the best method, now with a greater advantage with respect to Wang07, which has had a noticeable drop in accuracy and kappa. Anyway, these methods are far from learned classifiers.

- Hong08 is again the best method in the comparison. It has a robust behavior and it outstands with every classifier. The best result in this database is 96.26% accuracy and .9473 kappa (using SVM with polynomial kernel).

- Overall, three groups of methods can be formed:

    1. *Competitive and robust methods:* Hong08, Jain and Liu10 behave well with almost every classifier and reach competitive performances. Notice also that Liu10 obtains such a competitive behavior without rejecting any fingerprint (all of them are classified and taken into account when computing accuracy and kappa). Otherwise, Hong08 and Jain use the reject option, and hence the accuracy and kappa values are obtained over the classified fingerprints.

2. *Competitive methods:* Candela, Cappelli02 and Leung obtain good results with the proper classifier. Results are not as good as those of the previous methods and are also less robust but maintain a competitive behavior.
3. *Non-competitive methods:* Cappelli99a, Nyongesa, Shah, Li and the fixed classification methods do not reach the performance obtained by the rest of the FE methods, showing a less competitive behavior.

- At last, comparing the rejection rates obtained in this database (Table 7) with those obtained in HQNoPert database (Table 6) it can be observed that they have increased due to the worse quality of the fingerprints. The rejection rates have been almost tripled with respect to the previous results.

5.3. Results on VQAndPert database

Table 8 presents the results for the most difficult database: VQAndPert. The same structure as that of the previous tables is maintained.

Table 8: Accuracy and kappa results in Varying Quality and Perturbations (VQAndPert) database.

Accuracy			SVM			5NN		10NN				Kappa			SVM			5NN		10NN		
	Fixed	C4.5	Poly	RBF	Puk	Eucl.	HVDM	Eucl.	HVDM	Specific	**Rej. %**		Fixed	C4.5	Poly	RBF	Puk	Eucl.	HVDM	Eucl.	HVDM	Specific
Candela	–	77.40	**85.26**	83.39	81.25	81.78	80.03	81.05	79.43	–	–	Candela	–	.6793	**.7902**	.7628	.7464	.7371	.7115	.7261	.7024	–
Karu	**58.63**	–	–	–	–	–	–	–	–	–	–	Karu	**.4840**	–	–	–	–	–	–	–	–	–
Cappelli99a	65.89	72.59	74.85	74.00	**75.30**	72.67	72.48	74.08	74.29	–	–	Cappelli99a	.5398	.6091	.6381	.6257	**.6447**	.6094	.6067	.6289	.6319	–
Jain	–	87.65	92.24	91.36	90.44	**93.47**	92.97	93.17	92.91	92.04	15.90	Jain	–	.8255	.8900	.8774	.8672	**.9072**	.9001	.9029	.8991	.8870
Cappelli02	–	85.07	79.94	**86.21**	58.45	69.89	82.85	70.90	81.61	–	–	Cappelli02	–	.7881	.7157	**.8044**	.4797	.5672	.7531	.5785	.7344	–
Zhang04	**75.56**	–	–	–	–	–	–	–	–	–	4.09	Zhang04	**.6747**	–	–	–	–	–	–	–	–	–
Nyongesa	–	73.25	65.19	66.00	**73.76**	70.24	70.25	72.11	72.13	73.41	–	Nyongesa	–	.6078	.4879	.5001	**.6142**	.5713	.5714	.5951	.5953	.6095
Shah	–	65.04	70.36	70.64	53.71	60.56	76.40	59.78	**77.05**	58.18	–	Shah	–	.4999	.5786	.6081	.3983	.4205	.6615	.4062	**.6696**	.3961
Wang07	**69.67**	–	–	–	–	–	–	–	–	–	5.84	Wang07	**.6030**	–	–	–	–	–	–	–	–	–
Hong08	–	88.89	92.92	92.39	91.78	**93.55**	93.39	93.15	93.24	92.98	15.90	Hong08	–	.8427	.8997	.8922	.8836	**.9083**	.9062	.9026	.9041	.9010
Li	–	74.81	59.31	60.73	68.63	71.68	**75.48**	72.35	75.20	–	–	Li	–	.6426	.4099	.4304	.5507	.5955	**.6490**	.6047	.6442	–
Liu10	–	90.15	88.81	87.06	90.28	89.59	**90.95**	89.61	90.79	89.27	–	Liu10	–	.8601	.8401	.8154	.8622	.8515	**.8710**	.8518	.8687	.8481
Leung	–	39.48	74.01	75.22	63.29	**87.51**	83.32	87.21	84.09	75.06	–	Leung	–	.1659	.6298	.6494	.5344	**.8219**	.7623	.8175	.7730	.6510
Le	–	62.93	66.59	**79.03**	53.60	73.96	74.80	74.43	75.10	–	15.22	Le	–	.4721	.5216	**.7002**	.3881	.6242	.6362	.6306	.6402	–

An analysis of Table 8 has led us to the following conclusions:

- Fixed classification approaches accuracy and kappa values have had a large drop off, mainly in the case of Zhang04, even though it continues being the best one among them.

- Hong08 is still the best method (93.55% accuracy and .9083 kappa), but in this case using kNN classifier ($k = 5$ and Euclidean distance). This fact shows that kNN may be more suitable to deal with low quality fingerprint classification than SVMs. It should be stressed that in this case, Jain is much closer to Hong08 with 93.47% accuracy and .9072 kappa also using kNN. It seems like it is more difficult to take advantage of the new features introduced by Hong08 (SPs and pseudo-ridges) when the difficulty of the database increases, probably because their extraction becomes more difficult and less accurate. It is also interesting to note that Hong08 is no longer the best in all classifiers, since Jain performs better with 10NN using Euclidean distance and Liu10 with C4.5.

- The results obtained by Liu10 are remarkable. Its behavior is competitive (90.95% accuracy and .8710 kappa using kNN with $k = 5$ and HVDM distance) but which is more impressive, this result is obtained without rejecting any fingerprint, whereas Hong08 and Jain are discarding 15.90% of the fingerprints.

- The same groups as those in Default database can be observed. In the second one, Cappelli02 and Leung remain to be competitive as well as Candela. However, in this database the latter one has had a larger drop off and is the worst among them, whereas it was the best in the Default database. Hence, it

is more affected by the quality of the fingerprints. Regarding the rest of the methods (non-competitive ones), their behavior is neither good in this database, falling to the level of the fixed classification approach of Zhang04, despite carrying out a learning phase.

- As noticed previously, the reject rates have also increased, once again tripling those of the Default database.

5.4. Results on NIST-4 F/S databases

The results obtained on NIST-4 F and S databases are presented in Tables 9 and 10, respectively. We should recall that NIST-4 database highly differ from those from SFinGe, since fingerprints have been scanned from rolled-ink captures, producing a different image size, with wider fingerprints (with more information) and thicker ridges. There are also different artifacts in these images such as printed letter and notes, which may hinder the FE process. Moreover, the fingerprint classes are evenly distributed, whereas in SFinGe we generated the databases with the natural distribution of fingerprint classes. For these reasons, the results obtained with the FE methods in these databases may differ from those obtained with the previous ones.

Table 9: Accuracy and kappa results in NIST-4_F database.

Accuracy			SVM			5NN		10NN				Kappa			SVM			5NN		10NN		
	Fixed	C4.5	Poly	RBF	Puk	Eucl.	HVDM	Eucl.	HVDM	Specific	**Rej. %**		Fixed	C4.5	Poly	RBF	Puk	Eucl.	HVDM	Eucl.	HVDM	Specific
Candela	–	71.58	86.55	**86.61**	80.55	66.97	62.61	67.15	61.58	–	–	Candela	–	.6358	.8278	**.8283**	.7566	.5732	.5162	.5748	.5029	–
Karu	**84.55**	–	–	–	–	–	–	–	–	–	–	Karu	**.8004**	–	–	–	–	–	–	–	–	–
Cappelli99a	78.67	78.12	**81.58**	79.21	81.52	80.73	80.06	80.67	80.30	–	–	Cappelli99a	.7230	.7183	**.7621**	.7315	.7614	.7513	.7424	.7502	.7455	–
Jain	–	77.37	88.13	87.64	80.93	**88.38**	87.33	87.45	87.27	87.52	1.45	Jain	–	.7103	.8480	.8412	.7620	**.8500**	.8364	.8379	.8353	.8391
Cappelli02	–	83.03	83.39	**87.45**	71.70	71.94	75.70	68.85	75.21	–	–	Cappelli02	–	.7826	.7880	**.8388**	.6466	.6387	.6861	.5987	.6802	–
Zhang04	**87.63**	–	–	–	–	–	–	–	–	–	1.09	Zhang04	**.8404**	–	–	–	–	–	–	–	–	–
Nyongesa	–	78.73	73.03	73.03	78.18	77.09	77.15	78.55	78.36	**79.03**	–	Nyongesa	–	.7254	.6493	.6493	.7163	.7038	.7045	.7222	.7197	**.7285**
Shah	–	59.76	58.55	63.45	61.58	48.24	64.73	50.55	**65.21**	42.18	–	Shah	–	.4822	.4913	.5451	.4999	.3298	.5434	.3584	**.5493**	.2594
Wang07	**75.56**	–	–	–	–	–	–	–	–	–	2.73	Wang07	**.6862**	–	–	–	–	–	–	–	–	–
Hong08	–	83.82	**89.49**	88.93	82.47	88.44	88.56	88.01	87.89	88.87	1.45	Hong08	–	.7929	**.8655**	.8573	.7802	.8508	.8523	.8451	.8434	.8580
Li	–	64.48	59.27	57.88	66.97	67.03	**73.27**	66.79	72.55	–	–	Li	–	.5438	.4711	.4524	.5753	.5752	**.6554**	.5724	.6462	–
Liu10	–	82.67	82.85	80.42	83.58	83.39	**84.97**	82.67	84.48	82.97	–	Liu10	–	.7774	.7786	.7461	.7884	.7858	**.8061**	.7759	.7997	.7822
Leung	–	77.70	79.82	**80.67**	68.67	69.52	71.70	69.21	72.24	79.03	–	Leung	–	.7143	.7418	**.7524**	.6042	.6059	.6344	.6010	.6407	.7326
Le	–	55.05	56.72	**78.23**	58.73	76.66	75.82	74.38	73.74	–	6.73	Le	–	.4236	.4630	**.7202**	.4763	.6991	.6882	.6690	.6606	–

Table 10: Accuracy and kappa results in NIST-4_S database.

Accuracy			SVM			5NN		10NN				Kappa			SVM			5NN		10NN		
	Fixed	C4.5	Poly	RBF	Puk	Eucl.	HVDM	Eucl.	HVDM	Specific	**Rej. %**		Fixed	C4.5	Poly	RBF	Puk	Eucl.	HVDM	Eucl.	HVDM	Specific
Candela	–	70.36	**85.45**	83.33	79.27	64.18	57.64	63.64	57.58	–	–	Candela	–	.6200	**.8136**	.7867	.7414	.5363	.4513	.5286	.4504	–
Karu	**82.73**	–	–	–	–	–	–	–	–	–	–	Karu	**.7771**	–	–	–	–	–	–	–	–	–
Cappelli99a	79.39	78.36	**81.82**	78.36	**81.82**	80.18	79.27	80.67	80.73	–	–	Cappelli99a	.7323	.7220	.7655	.7209	**.7657**	.7444	.7326	.7502	.7512	–
Jain	–	78.55	85.16	**86.28**	79.79	86.09	85.05	85.85	84.92	86.15	1.94	Jain	–	.7247	.8099	**.8233**	.7472	.8205	.8067	.8171	.8048	.8215
Cappelli02	–	82.91	82.06	**87.03**	71.39	72.73	75.45	69.33	75.39	–	–	Cappelli02	–	.7809	.7713	**.8337**	.6430	.6484	.6826	.6048	.6817	–
Zhang04	**84.39**	–	–	–	–	–	–	–	–	–	1.03	Zhang04	**.7988**	–	–	–	–	–	–	–	–	–
Nyongesa	–	**77.45**	70.67	71.70	76.85	75.94	75.52	76.24	75.64	76.61	–	Nyongesa	–	**.7082**	.6185	.6319	.6990	.6887	.6830	.6922	.6843	.6967
Shah	–	55.58	59.39	**62.85**	60.48	44.91	61.64	45.39	62.24	41.21	–	Shah	–	.4278	.5008	**.5412**	.4858	.2858	.5032	.2909	.5100	.2426
Wang07	**72.07**	–	–	–	–	–	–	–	–	–	3.21	Wang07	**.6409**	–	–	–	–	–	–	–	–	–
Hong08	–	80.41	85.29	**86.83**	80.97	86.28	86.15	86.65	86.77	86.59	1.94	Hong08	–	.7493	.8119	**.8299**	.7614	.8228	.8211	.8273	.8289	.8294
Li	–	60.79	56.85	55.33	64.91	65.88	**70.12**	66.42	70.06	–	–	Li	–	.4966	.4397	.4190	.5471	.5615	**.6143**	.5678	.6131	–
Liu10	–	80.61	81.76	78.36	82.48	81.52	82.36	81.27	**82.73**	81.82	–	Liu10	–	.7513	.7640	.7187	.7742	.7613	.7722	.7579	**.7771**	.7676
Leung	–	76.79	74.91	**78.06**	66.55	68.67	71.94	69.70	72.61	76.73	–	Leung	–	.7021	.6794	**.7183**	.5776	.5939	.6369	.6072	.6452	.7050
Le	–	50.76	53.05	**75.28**	57.87	74.46	74.01	71.77	71.57	–	7.94	Le	–	.3685	.4152	**.6817**	.4649	.6704	.6646	.6350	.6322	–

Observing the results on both NIST-4 databases the following facts should be highlighted:

- Among fixed classification approaches Karu shows a much better performance than in previous databases compared with the rest of the methods. However, it does not reach the accuracy and kappa values of Zhang04, whose behavior is near the best performing FE methods with learned features. Such an important improvement may be expected due to the fact that these methods were designed to work with these type of fingerprints, even though the same occurs with Wang07 and Cappelli99a whose results are not as accurate.

- The superiority of Hong08 is once again remarkable, being only Cappelli02 capable of improving it in the case of NITS-4 S database. In these two databases Hong08 performs better than the rest when considering the kNN classifier or the specific proposal but there are other methods with better performance with C4.5 (in S database) and SVMs (mainly in S database).

- Hong08 and Cappelli02 are nearly followed by PCASYS (Candela) and Jain methods, but also by Zhang04, which can be considered in this group of competitive methods. Surprisingly, Liu10, whose performance in SFinGe's databases has excelled, performs worse than the fixed classification model of Zhang04. The usage of the features obtained with Cappelli99a method as input for different classifiers neither produced competitive results, but it almost reaches Liu10, on the contrary to the behavior found on SFinGe's databases.

- There are a number of methods with lower accuracy and kappa values which cannot be considered as competitive in NIST-4 databases: Leung (which in the previous databases performed better), Nyongesa, Shah, Wang07, Li and Le.

- Comparing both databases of NIST-4, it can be concluded that S database may be more difficult for most of the methods, even though they share similar characteristics, since lower accuracy and kappa values are obtained in it. Moreover, the rejection rates are greater, indicating that more distorted fingerprints are present. Nevertheless, similar conclusion can be drawn from both of them.

5.5. Summary of results

In this subsection our aim is to sum up the results obtained on the five databases and to show which methods are the best in the framework considered. In this way, we will be able to conclude which type of features are the most adequate for classification purposes. Once again, we should recall that these conclusions are drawn from the experiments carried out from our implementations of the proposed methods. Hence, we do not only test their quality, but we also deal with their implementability and usefulness for other researchers, since all these methods have been developed following their original source papers.

Regarding the FE methods analyzed it can be concluded that:

- FE methods based on fixed classification are not able to overcome FE methods considering learning models. However, Zhang04, which is the best fixed classification model in all databases, is competitive in NIST-4. Similarly, Karu's results are much better in this database. Between these two methods, Zhang04 works better in all cases, and hence the computation of the orientations (gradient method instead of slits sums) and the introduction of pseudo-ridges have made a difference, since Karu only works with the SPs obtained. The reason why they perform better in NIST-4 is that they were specifically designed for the properties of the fingerprints in this databases and some of them are not translated to SFinGe generated ones (rolled vs. non-rolled fingerprints). In non-rolled fingerprints cores and deltas are sometimes out of the fingerprint area or on the border, making the classification only based on these points more difficult.

- The robustness of the FE methods of Hong08 and Jain must be pointed out. They perform well in all databases and with every classifier. Hence, it can be concluded that FingerCode features are robust, providing that the proper method is used to set the center of the region of interest and the recommended parameters are used.

- Leung and Le are also FingerCode-based methods but they are not as accurate as the previous ones. In the case of Le, the FingerCode does not work as expected, at least in this experimental framework. A new invariant to rotation FingerCode was introduced together with a new reference point localization algorithm. Different parameters from the recommended ones were also used, leading to a high number of features (1280 vs. 192 of Jain). The complexity added by these modifications has been translated into a less accurate model. The case of Leung is rather different. Its performance could be considered as competitive in SFinGe's databases with kNN and Euclidean distance, but the same does not occur in NIST-4. Besides the different shape of the region of interest (squared sectors) and the localization of the reference point (R92), Leung's model major change with respect to FingerCode is the usage of missing values instead of rejecting fingerprints. Apparently, these values are not hindering the classification in SFinGe, but they do so in NIST-4. This could be related with the fact that many more fingerprints are considered in SFinGe's databases for training, and hence for the imputation which may result in better feature vectors. However, it should also be mentioned that Leung uses 816 features and many of them may contain missing values making the learning more difficult.

- Liu10 method is also one of the best performing ones, but has a major difference with Hong08 and Jain: it does not reject fingerprints in the FE process. The trade-off between rejection and accuracy should be considered when comparing this method with the mentioned ones. However, its drawback is that it does not perform in NIST-4 as well as it does in SFinGe's databases. Anyway, it is by far the best method among those not rejecting fingerprints in the case of SFinGe.

- Candela (PCASYS) and Cappelli02 work well in almost all databases (with the appropriate classifier). They are more competitive on NIST-4 databases and suffer a loss of accuracy in SFinGe's databases as the difficulty increases. Hence, it can be concluded that these orientation-based models (registered in both cases and in combination with the segmentation of the OM in Cappelli02) perform reasonably well, but they are more affected by noise than FingerCode-based methods.

- Otherwise, there are orientation-based models not performing as well as the previous ones. Cappelli99a is based on OM segmentation and by itself is not enough to properly perform the classification, whereas in conjunction with registered orientations (Cappelli02) reaches a competitive performance. The methods of Li and Shah are also orientation-based but the added complexity (the mathematical model for describing the OM and the line detection for computing the OM) has not come along with an improved classification.

- Other FE methods using SPs apart from Hong08, Zhang04 and Karu have not shown a competitive behavior. Nyongesa only considered 7 features obtained as relative measures between SPs but it has been shown that their description is not suitable for classification. The new SP introduced by Wang07 aiming at improving Zhang04 has neither had a positive effect always being less accurate than the original model.

- Finally, looking at the different rejection rates of the methods in the tested databases, it can be observed that in SFinGe's databases they start with a low rate in the easiest database (HQNoPert) with rates between 1-2%, but these values are tripled in each database when the difficulty increases (reaching a maximum of 15.90% in VQAndPert). The rejection rates in NIST-4 are similar to those of HQNoPert database, even though the performance has more similarity with VQAndPert database (notice that the distribution of the classes differ between these databases).

With respect to the classification models considered (both specific and general purpose ones) the following points must be highlighted:

- SVM is the classifier obtaining the best results for almost all the FE methods in most of the databases. However, it outstands in the easiest databases, whereas kNN classifier performance increases with greater difficulty. For 6 FE methods SVM is the best classifier in HQNoPert, whereas this only occurs twice with kNN (and twice with C4.5). In Default database the same results are maintained except for

kNN, which becomes the best method for Nyongesa (whose best result in HQNoPert was the specific classifier). The greatest change appears in VQAndNoPert where SVM is the best method for 5 FE methods, whereas kNN performs better in 6 of the FE methods, showing a tendency in its favor. With respect to NIST-4 databases, SVM makes a difference, being the best for 6 and 8 FE methods, whereas kNN only performs better with 3 and 2 methods in F and S, respectively.

- Regarding the best kernel used in SVMs, it directly depends on the type of features to be learned and what they represent, but also on the quality of the fingerprints (i.e., quality of the features extracted). In some cases it does not highly affect the results, whereas in other large differences can be obtained.

- In the case of kNN, the value of k neither have a big influence on the results (between 5 and 10), whereas the selection of the distance becomes an important decision with several FE methods (for instance Leung, Li, Shah and Cappelli02).

- Overall, specific classification approaches do not provide the improvement expected. They are properly selected in most of the cases (except for Shah and Leung) obtaining similar results to the best classifier in the corresponding FE method, but hardly ever perform better than the best classifier.

To conclude this analysis, we want to rank FE methods tested according to the results obtained. This ranking follows a similar structure to the groups created in previous sections:

1. Hong08, Jain and Liu10: Hence, original FingerCode-based methods (better using SPs with Poincarè and pseudo-ridges as a complementary features) and complex-filters-based SP detection with relative features (even though Liu10 performs poorly in NIST-4).
2. Cappelli02 and Candela (PCASYS): Features obtained from registered orientations.
3. Leung and Zhang04: These methods depend more on the database. They used a new FingerCode approach without rejection and a fixed classification with SPs and pseudo-ridges, respectively.
4. Karu, Cappelli99a, Nyongesa, Le, Li, Shah and Wang07: These are the least accurate models due to different reasons as we have already described.

Therefore, it can be observed that in the first two categories (the most competitive ones) there are methods using all types of features from the taxonomy presented in the first part of this paper [27]: Orientations, SPs, Ridge structure and Filter-based. However, we have shown that registered orientations perform better than other approaches in that category. Similarly, FingerCode is the best approach from filter-based ones. SPs can be helpful as a complementary features (Hong08) but also as direct features using relative measures (Liu10). Otherwise, ridge structure features, and more specifically pseudo-ridges, are more appropriate to complement other features as Hong08 and Zhang04 have shown.

6. Ensemble proposal: Consensus vs. majority for combining existing approaches

After carrying out the experimental analysis of the different FE methods, we have realized that we could take advantage of the different results obtained with different FE methods. Therefore, in this section our aim is to put forward a new way of overcoming the fingerprint classification problem by the usage of several FE methods and classifiers, fusing the classifications obtained in different models. We present how the performance obtained in the classification problem can be boosted by this approach and different trade-offs between performance and rejection rate can be obtained so that the best combination can be selected depending on the final application.

In order to do so, we propose to combine the classifiers learned with different features obtained with different FE methods in a MCS, which has not been previously considered. Notice that instead of considering different type of features in the same feature vector as for instance Hong08 does, we combine the results obtained with different FE methods after the classification has been performed. In fact, in the case of the specific classifier of Hong08 the features of the SPs and pseudo-ridges were used to order the SVMs trained with FingerCode. However, their results were not combined but used to improve the classification of the SVMs using OVA strategy. The other approach considering ensembles of classifiers is the specific classifier of

Liu10, which considered Adaboost. Nevertheless, it neither combined different types of features but learned an Adaboost classifier for each scale where the SPs were detected (features are related).

Our aim is to combine different classifiers learned with different features. To do so, we consider two combination models so that one can decide the trade-off between performance and rejection rate sought. These combinations only consider the class label given for the fingerprint by each classifier of the ensemble:

- **Consensus:** If all the classifiers agree with the class label assigned, it is used to label the fingerprint. Otherwise, if any of the classifiers disagrees the fingerprint is rejected.
- **Majority:** Each classifier gives a vote for the class label predicted and the class with the greatest number of votes is used as final prediction of the MCS. If there is a tie of votes, the fingerprint is rejected.

It is clear that the first approach is more restrictive, and hence greater performance is expected at the expenses of an increase in the rejection rates. The second approach does not need a total agreement among the classifiers, rejecting less fingerprints, but consequently more fingerprints should be misclassified.

In order to show the performance improvement that this type of MCS can lead to fingerprint classification, we have selected some of the FE methods tested before and which have shown a competitive behavior. Furthermore, they should be diverse, which is always required in ensembles and MCSs [38]. For these reasons, we have considered Hong08 (H), Liu10 (L), Zhang04 (Z) and Cappelli02 (C) methods. Even though Zhang04 is the less accurate among them, the fact that it uses a fixed classification approach makes it appropriate for the ensemble, since greater diversity is expected from such a different model. For the sake of simplicity, we have considered the specific classifiers for all cases except for Cappelli02, which does not have a specific classifier, and hence we have used SVM with RFB kernel results.

The results obtained by the different combinations of these methods are presented in Table 11 (the components of the MCS are used to label the column). We show all the possible combinations among them. Notice that when only two methods are considered Consensus and Majority models are the same, since no majority could be reached without consensus. The table shows the results in testing and it is horizontally divided into three parts: the upper part presents the accuracy results, the part in the middle the kappa values and the lower one the rejection rates obtained for each one of the five databases considered. We should remark that all the fingerprints rejected by the FingerCode are also rejected by the MCS and the same is done with Zhang04. The best result in each row is highlighted in **bold-face** (the greatest rejection rate in the case of the lower part of the table).

Looking at Table 11 it can be observed that the best single classification model in each database from the previous study is always improved in every combination (in exchange for increasing the rejection rate). It is clear that the best results of the MCS are obtained when all the FE methods are combined using Consensus model, but also the greatest rejection rates are obtained. The least restrictive model is Majority when three classifiers are combined. In these cases, low rejection rates are obtained but the performance is improved with respect to single classifiers. Using the four methods instead of three of them produces more rejection rates (ties might occur in this case) and consequently the performance is improved. The combination of a pair of classifiers leads to the rejection of more fingerprints, which allows one to obtain good performances. Consensus model with three classifiers does not reach the performance of the combination of the four methods, but a good trade-off between performance and rejection rate is achieved.

As we have shown in the previous section, the best combination would first depend on the database we are working with (for example, Zhang04 performs better in NIST-4, whereas Liu10 works better with SFinGe's databases). Hence, a combination of Hong08, Liu10 and Cappelli02 may perform well in SFinGe's databases without a large increase in the rejection rate, whereas a combination of Hong08, Zhang04 and Cappelli02 may be more adequate for NIST-4. Anyway, our aim was to show that these types of combinations can lead to an improved performance with different rejection rates depending on the methods and the model considered. It should be taken into account that different combination approaches can be developed but from our point of view, this type of MCS combination can lead to better fingerprint classification models. There are many FE methods in the literature as we have reviewed in this series of two papers, but no combination

Table 11: Accuracy and kappa results and the corresponding rejection rates in all databases with different ensemble models in test.

Test							HLZ		HLC		HZC		LZC		HLZC	
	HL	HZ	LZ	HC	LC	ZC	Cons.	Maj.	Cons.	Maj.	Cons.	Maj.	Cons.	Maj.	Cons.	Maj.
HQNoPert	98.76	98.49	98.06	98.74	98.46	98.86	99.11	97.27	99.27	97.53	99.26	97.71	99.22	97.14	**99.47**	98.00
Default	98.54	98.06	97.93	98.84	98.31	98.51	99.09	96.73	99.29	97.29	99.35	97.08	99.15	96.83	**99.53**	97.64
VQAndPert	97.63	97.60	96.92	97.29	97.34	97.65	98.62	95.56	98.65	95.38	98.82	95.76	98.70	95.37	**99.10**	96.00
NIST-4_F	95.81	96.33	95.91	96.15	95.17	96.01	98.25	92.58	97.57	92.84	98.33	92.80	97.50	92.97	**98.69**	94.73
NIST-4_S	93.86	96.34	96.16	95.46	95.26	96.03	98.28	91.28	96.99	91.61	97.64	93.35	97.79	92.89	**98.45**	93.43
HQNoPert	.9823	.9785	.9723	.9820	.9779	.9837	.9873	.9613	.9894	.9650	.9894	.9675	.9888	.9592	**.9923**	.9715
Default	.9790	.9724	.9705	.9834	.9757	.9786	.9870	.9536	.9897	.9615	.9906	.9586	.9878	.9549	**.9932**	.9664
VQAndPert	.9662	.9660	.9566	.9611	.9618	.9665	.9804	.9374	.9805	.9346	.9830	.9401	.9814	.9347	**.9871**	.9435
NIST-4_F	.9459	.9521	.9466	.9501	.9373	.9480	.9771	.9045	.9683	.9081	.9781	.9071	.9672	.9092	**.9828**	.9320
NIST-4_S	.9208	.9520	.9495	.9413	.9384	.9480	.9773	.8880	.9608	.8926	.9689	.9145	.9708	.9082	**.9796**	.9154
HQNoPert	6.49	11.88	12.83	7.66	8.54	14.91	14.55	2.59	10.21	2.27	15.97	2.94	16.75	3.25	**17.49**	3.73
Default	12.50	20.11	22.12	13.73	15.25	23.68	23.97	7.56	17.46	6.56	25.23	7.74	26.85	8.08	**27.78**	8.49
VQAndPert	24.53	35.25	36.08	27.82	28.72	39.46	38.48	20.83	31.59	17.89	41.27	21.56	42.27	21.34	**43.34**	21.30
NIST-4_F	19.21	17.58	21.52	16.67	19.39	18.00	26.73	5.52	25.45	4.36	24.00	5.03	26.85	6.06	**30.24**	8.06
NIST-4_S	20.30	22.55	26.55	18.36	22.42	22.06	31.27	7.52	27.82	5.45	28.48	6.79	31.94	7.88	**34.97**	8.67

between approaches has been previously considered, which can be an interesting future research line in the topic.

7. Conclusions and future research lines

In this second paper of the series of two papers on fingerprint classification we have performed a deep experimental study of the most relevant FE methods developed in the literature as well as the specific classifiers developed. The FE methods cover most of the categories included in our taxonomy in the first part of this paper. In order to carry out the experimental study, we have carefully prepared a common and objective experimental framework where our implementations of these methods could be tested. This way, we analyze their performance and reimplementability, which is usually overlooked despite of its importance.

From the experimental study carried out the following conclusions can be drawn:

- FingerCode-based Hong08 and Jain methods together with Liu10 approach based on SPs (complex-filters) and relative measures are the most robust and accurate methods.
- Orientation-based approaches (Cappelli02 and Candela) can obtain competitive results, but without reaching the former ones.
- A number of methods does not work as expected due to different reasons such as complexity, which makes difficult to exactly recover the author's implemented version but also makes them worse when the difficulty of the database increases.
- Classifiers specifically designed for the fingerprint classification problem do not perform better than general purpose classification models, even though they were properly selected in most of the cases.

Looking at the variety and diversity of the results obtained with the different FE methods, we have also proposed a MCS combination model with two fusion strategies (Majority and Consensus), which allows one to highly improve the classification results obtained in the different databases at the expenses of increasing the rejection ratio. This trade-off can be selected depending on the application by using the different fusion strategies and different FE methods.

Before ending this paper, we want to highlight the difficulty that the implementation of the different FE methods and specific classifiers has had, since as highlighted in the first part of this paper, most of the papers on fingerprint recognition lack of details in their implementations. This may be the reason why some of the methods with goods results in the original source papers obtain poorer results in our experiments. Nevertheless, we should notice that this mainly occurs with the most complex methods, whose implementation becomes a real challenge.

As future research lines, first of all, we would like to see more researchers sharing their source codes along with the papers of the topic. This fact would help in the development of this area. Moreover, we have shown that many FE methods exist but there has not been a major improvement since FingerCode (with few exceptions such as Hong08 and Liu10). Hence, with respect to FE methods simpler approaches need to be developed, which really capture the nature of the fingerprints for classification. However, instead of only focusing on the FE methods, much more effort should be made on the development of specific fingerprint classification approaches and the diversity provided by different FE methods should be exploited in MCSs as the one that we have proposed in this paper as an example, which can be further improved.

Furthermore, fingerprint classification poses several challenges from the ML point of view, which remain to be studied. Multi-class problems are usually harder to solve than their binary counterparts, and hence specific models for multi-class problems should be tested [24, 25]. But the difficulty resides not only in the fact that the problem consider multiple classes but also in the presence of the class imbalance problem [22, 76, 26]. Another issue that may enhance fingerprint classification is the introduction of more complex rejection mechanisms that do not only take into account whether the features can be obtained (as in FingerCode) but analyze their quality and decide whether the classification is going to be successful or not.

Acknowledgment

This work was supported by the Research Projects CAB(CDTI), TIN2011-28488, and TIN2013-40765-P. D. Peralta holds an FPU scholarship from the Spanish Ministry of Education and Science (FPU12/04902).

References

[1] D. W. Aha, D. Kibler, M. K. Albert, Instance-based learning algorithms, Machine Learning 6 (1991) 37–66.
[2] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework, Journal of Multiple-Valued Logic and Soft Computing 17 (2011) 255 – 287.
[3] J. Alcalá-Fdez, L. Sánchez, S. García, M. J. del Jesus, S. Ventura, J. M. Garrell, J. Otero, C. Romero, J. Bacardit, V. M. Rivas, J. Fernández, F. Herrera, KEEL: a software tool to assess evolutionary algorithms for data mining problems, Soft Computing 13 (3) (2009) 307–318.
[4] E. Alpaydin, Introduction to Machine Learning, The MIT Press, 2004.
[5] A. M. Bazen, S. H. Gerez, Segmentation of fingerprint images, in: ProRISC 2001 Workshop on Circuits, Systems and Signal Processing, 2001.
[6] A. M. Bazen, S. H. Gerez, Systematic methods for the computation of the directional fields and singular points of fingerprints, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (7) (2002) 905–919.
[7] L. Breiman, J. H. Friedman, R. A. Olshen, C. J. Stone, Classification and Regression Trees, Chapman and Hall (Wadsworth and Inc.), 1984.
[8] G. T. Candela, P. J. Grother, C. I. Watson, R. A. Wilkinson, C. L. Wilson, PCASYS - A pattern-level classification automation system for fingerprints, Tech. rep., UNIST Interagency/Internal Report (NISTIR) - 5647 (1995).
[9] R. Cappelli, M. Ferrara, A. Franco, D. Maltoni, Fingerprint verification competition 2006, Biometric Technology Today 15 (7-8) (2007) 7–9.
[10] R. Cappelli, A. Lumini, D. Maio, D. Maltoni, Fingerprint classification by directional image partitioning, IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (5) (1999) 402–421.
[11] R. Cappelli, D. Maio, D. Maltoni, Multispace kl for pattern representation and classification, IEEE Transactions on Pattern Analysis and Machine Intelligence 23 (9) (2001) 977–996.
[12] R. Cappelli, D. Maio, D. Maltoni, A multi-classifier approach to fingerprint classification, Pattern Analysis & Applications 5 (2002) 136–144.
[13] R. Cappelli, D. Maio, D. Maltoni, Sfinge: an approach to synthetic fingerprint generation, in: Eighth International Conference on Control, Automation, Robotics and Vision (ICARCV2004), Kunming, China, 2004.
[14] J. Cohen, A coefficient of agreement for nominal scales, Educational and Psychological Measurement 20 (1) (1960) 37–46.
[15] M. Collins, R. E. Schapire, Y. Singer, Logistic regression, adaboost and bregman distances, Machine Learning 48 (1-3) (2002) 253–285.

[16] E. R. Davies, A. P. N. Plummer, Thinning algorithms: A critique and a new methodology, Pattern Recognition 14 (1-6) (1981) 53–63.
[17] P. Domingos, M. Pazzani, On the optimality of the simple bayesian classifier under zero-one loss, Machine Learning 29 (2-3) (1997) 103–130.
[18] B. Dorizzi, R. Cappelli, M. Ferrara, D. Maio, D. Maltoni, N. Houmani, S. Garcia-Salicetti, A. Mayoue, Fingerprint and on-line signature verification competitions at icb 2009, in: M. Tistarelli, M. Nixon (eds.), Advances in Biometrics, vol. 5558 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2009, pp. 725–732.
[19] R. O. Duda, P. E. Hart, D. G. Stork, Pattern Classification, 2nd ed., John Wiley, 2001.
[20] A. Farhangfar, L. Kurgan, J. Dy, Impact of imputation of missing values on classification error for discrete data, Pattern Recognition 41 (2008) 3692–3705.
[21] A. Fernández, S. García, J. Luengo, E. Bernadó-Mansilla, F. Herrera, Genetics-based machine learning for rule induction: State of the art and taxonomy and comparative study, IEEE Transactions on Evolutionary Computation 14 (6) (2010) 913 – 941.
[22] A. Fernández, V. López, M. Galar, M. J. del Jesus, F. Herrera, Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches, Knowledge-Based Systems 42 (2013) 97–110.
[23] R. A. Fisher, The use of multiple measurements in taxonomic problems, Annals of eugenics 7 (2) (1936) 179–188.
[24] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, F. Herrera, An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes, Pattern Recognition 44 (8) (2011) 1761 – 1776.
[25] M. Galar, A. Fernández, E. Barrenechea, F. Herrera, DRCW-OVO: Distance-based relative competence weighting combination for one-vs-one strategy in multi-class problems, Pattern Recognition 48 (1) (2015) 28–42.
[26] M. Galar, A. Fernndez, E. Barrenechea, F. Herrera, Empowering difficult classes with a similarity-based aggregation in multi-class classification problems, Information Sciences 264 (0) (2014) 135–157.
[27] M. Galar et al., A survey of fingerprint classification I: Taxonomies on feature extraction methods and learning models, Submitted.
[28] S. García, J. Derrac, J. Cano, F. Herrera, Prototype selection for nearest neighbor classification: Taxonomy and empirical study, IEEE Transactions on Pattern Analysis and Machine Intelligence 34 (3) (2012) 417–435.
[29] J.-H. Hong, J.-K. Min, U.-K. Cho, S.-B. Cho, C. H. Leung, Fingerprint classification using one-vs-all support vector machines dynamically ordered with naive bayes classifiers, Pattern Recognition 41 (2008) 662–671.
[30] L. Hong, Y. Wan, A. Jain, Fingerprint image enhancement: algorithm and performance evaluation, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (8) (1998) 777–789.
[31] A. K. Jain, Fundamentals of digital image processing, vol. 3, Prentice-Hall Englewood Cliffs, 1989.
[32] A. K. Jain, S. Prabhakar, L. Hong, A multichannel approach to fingerprint classification, IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (4) (1999) 348–359.
[33] A. K. Jain, S. Prabhakar, L. Hong, S. Pankanti, Filterbank-based fingerprint matching, IEEE Transactions on Image Processing 9 (5) (2000) 846–859.
[34] I. Jolliffe, Principal component analysis, Wiley Online Library, 2005.
[35] K. Karu, A. K. Jain, Fingerprint classification, Pattern Recognition 29 (3) (1996) 389–404.
[36] M. Kawagoe, A. Tojo, Fingerprint pattern classification, Pattern Recognition 17 (1984) 295–303.
[37] R. W. Kennard, L. A. Stone, Computer aided design of experiments, Technometrics 11 (1969) 137–148.
[38] L. I. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, Wiley-Interscience, 2004.
[39] T. H. Le, H. T. Van, Fingerprint reference point detection for image retrieval based on symmetry and variation, Pattern Recognition 45 (9) (2012) 3360 – 3372.
[40] K. Leung, C. H. Leung, Improvement of fingerprint retrieval by a statistical classifier, IEEE Transactions on Information Forensics and Security 6 (1) (2011) 59–69.
[41] J. Li, W. Y. Yau, H. Wang, Constrained nonlinear models of fingerprint orientations with prediction, Pattern Recognition 39 (1) (2006) 102–114.
[42] J. Li, W.-Y. Yau, H. Wang, Combining singular points and orientation image information for fingerprint classification, Pattern Recognition 41 (1) (2008) 353–366.
[43] M. Liu, Fingerprint classification based on adaboost learning from singularity features, Pattern Recognition 43 (2010) 1062–1070.
[44] M. Liu, X. Jiang, A. C. Kot, Fingerprint reference-point detection, EURASIP Journal on Applied Signal Processing 2005 (2005) 498–509.
[45] M. Liu, X. Jiang, A. C. Kot, Fingerprint retrieval by complex filter responses, in: ICPR 2006. 18th International Conference on Pattern Recognition, vol. 1, 2006.
[46] D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman, A. K. Jain, FVC2000: Fingerprint verification competition, IEEE Transactions on Pattern Analysis Machine Intelligence 24 (3) (2002) 402–412.
[47] D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman, A. K. Jain, FVC2002: Second fingerprint verification competition, in: Proceedings of the 16th International Conference on Pattern Recognition (ICPR'02), Quebec City, Canada, August 11-15, 2002.
[48] D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman, A. K. Jain, FVC2004: Third fingerprint verification competition, in: Proceedings of the First International Conference on Biometric Authentication (ICBA'04), Hong Kong, China, July 15-17, 2004.
[49] D. Maltoni, D. Maio, A. Jain, S. Prabhakar, Handbook of fingerprint recognition, Springer-Verlag New York Inc, 2009.
[50] G. J. McLachlan, Discriminant Analysis and Statistical Pattern Recognition, John Wiley and Sons, 2004.

[51] G. J. Mclachlan, Discriminant Analysis and Statistical Pattern Recognition (Wiley Series in Probability and Statistics), Wiley-Interscience, 2004.
[52] B. M. Mehtre, N. N. Murthy, S. Kapoor, B. Chatterjee, Segmentation of fingerprint images using the directional image, Pattern Recognition 20 (4) (1987) 429–435.
[53] Y. Mei, H. Sun, D. Xia, A gradient-based combined method for the computation of fingerprints orientation field, Image and Vision Computing 27 (8) (2009) 1169–1177.
[54] J. G. Moreno-Torres, T. Raeder, R. Alaiz-Rodríguez, N. V. Chawla, F. Herrera, A unifying view on dataset shift in classification, Pattern Recognition 45 (1) (2012) 521–530.
[55] J. G. Moreno-Torres, J. A. Saez, F. Herrera, Study on the impact of partition-induced dataset shift on k-fold cross-validation, IEEE Transactions on Neural Networks and Learning Systems 23 (8) (2012) 1304–1312.
[56] K. Nilsson, J. Bigun, Localization of corresponding points in fingerprints by complex filtering, Pattern Recognition Letters 24 (13) (2003) 2135–2144.
[57] M. Nixon, A. S. Aguado, Feature extraction & image processing, Elsevier, 2008.
[58] H. O. Nyongesa, S. Al-Khayatt, S. M. Mohamed, M. Mahmoud, Fast robust fingerprint feature extraction and classification, Journal of Intelligent and Robotic Systems 40 (1) (2004) 103–112.
[59] C. H. Park, H. Park, Fingerprint classification using fast fourier transform and nonlinear discriminant analysis, Pattern Recognition 38 (4) (2005) 495–503.
[60] D. Peralta, M. Galar, I. Triguero, O. Miguel-Hurtado, J. M. Benitez, F. Herrera, Minutiae filtering to improve both efficacy and efficiency of fingerprint matching algorithms, Engineering Applications of Artificial Intelligence 32 (2014) 37–53.
[61] J. C. Platt, Fast training of support vector machines using sequential minimal optimization, MIT Press, Cambridge, MA, USA, 1999.
[62] J. R. Quinlan, C4.5: Programs for Machine Learning, 1st ed., San Mateo-California: Morgan Kaufmann Publishers, 1993.
[63] A. R. Rao, A Taxonomy for Texture Description and Identification, Springer Verlag, 1990.
[64] N. K. Ratha, S. Chen, A. K. Jain, Adaptive flow orientation-based feature extraction in fingerprint images, Pattern Recognition 28 (11) (1995) 1657–1672.
[65] R. Rifkin, A. Klautau, In defense of one-vs-all classification, Journal of Machine Learning Research 5 (2004) 101–141.
[66] J. A. Sáez, M. Galar, J. Luengo, F. Herrera, Analyzing the presence of noise in multi-class problems: alleviating its influence with the one-vs-one decomposition, Knowledge and Information Systems 38 (1) (2014) 179–206.
[67] S. Shah, P. S. Sastry, Fingerprint classification using a feedback-based line detector, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 34 (1) (2004) 85–94.
[68] C. F. Shu, R. C. Jain, Direct estimation and error analysis for oriented patterns, CVGIP: Image Understanding 58 (3) (1993) 383–398.
[69] C. F. Shu, R. C. Jain, Vector field analysis for oriented patterns, IEEE Transactions on Pattern Analysis and Machine Intelligence 16 (9) (1994) 946–950.
[70] S. F. Smith, Flexible learning of problem solving heuristics through adaptive search, in: IJCAI'83: Proceedings of the Eighth international joint conference on Artificial intelligence, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1983.
[71] R. M. Stock, C. W. Swonger, Development and evaluation of a reader of fingerprint minutiae, Tech. rep., Cornell Aeronautical Laboratory, Technical Report CAL no. XM-2478-X-1 (1969).
[72] X. Tan, B. Bhanu, Y. Lin, Fingerprint classification based on learned features, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 35 (3) (2005) 287–300.
[73] V. Vapnik, Statistical Learning Theory, New York: Wiley, 1998.
[74] L. Wang, M. Dai, Application of a new type of singular points in fingerprint classification, Pattern Recognition Letters 28 (13) (2007) 1640–1650.
[75] L. Wang, M. Dai, G. Geng, Fingerprint image segmentation by energy of gaussian-hermite moments, in: S. Z. Li, J. Lai, T. Tan, G. Feng, Y. Wang (eds.), Advances in Biometric Person Authentication, 5th Chinese Conference on Biometric Recognition, SINOBIOMETRICS 2004, vol. 3338 of Lecture Notes in Computer Science, Springer, 2005.
[76] S. Wang, X. Yao, Multiclass imbalance problems: Analysis and potential solutions, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 42 (4) (2012) 1119–1130.
[77] C. I. Watson, C. L. Wilson, NIST Special Database 4, Fingerprint Database, Tech. rep., U.S. National Institute of Standards and Technology (1992).
[78] J. H. Wegstein, An automated fingerprint identification system, NBS Special Publication 500-89.
[79] D. R. Wilson, T. R. Martinez, Improved heterogeneous distance functions, Journal of Artificial Intelligence Research 6 (1997) 1–34.
[80] I. H. Witten, E. Frank, Data mining: practical machine learning tools and techniques with Java implementations, 2nd ed., Morgan Kaufmann, San Francisco, 2005.
[81] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z.-H. Zhou, M. Steinbach, D. J. Hand, D. Steinberg, Top 10 algorithms in data mining, Knowl. Inf. Syst. 14 (2007) 1–37.
[82] Q. Zhang, H. Yan, Fingerprint classification based on extraction and analysis of singularities and pseudo ridges, Pattern Recognition 37 (11) (2004) 2233–2243.