Perceptron-based Fusion of Multiple Fingerprint Matchers

Gian Luca Marcialis and Fabio Roli
Department of Electrical and Electronic Engineering – University of Cagliari
Piazza d'Armi – 09123 Cagliari - Italy
{marcialis, roli}@diee.unica.it

Abstract

In this paper, a neural fusion rule for fingerprint verification is presented. The person to be identified submits to the system her/his fingerprint and her/his identity. Multiple fingerprint matchers provide a set of verification scores, that are then fused by a perceptron-based method. The weights of such perceptron are explicitly optimised to increase the separation between genuine users and impostors (i.e., unknown users). To this end, the perceptron learning algorithm was modified. Reported experiments show that such modified perceptron allows improving the performances and the robustness of the best individual fingerprint matcher, and outperforming some simple fusion rules.

1. Introduction

Fingerprints are widely used for automatic personal authentication [1,2], in order to control the access to limited areas or resources. The person to be "authenticated" submits to the system her/his fingerprint and declares her/his identity. The system matches the input fingerprint with the one associated to the given identity and stored in its database. A degree of similarity, named "score", is computed. If the score is higher than a certain value (the so called "acceptance" threshold), the claimed identity is accepted and the person is classified as a genuine user. Otherwise, she/he is classified as an impostor and the access to the required resource is denied.

The core of any automatic fingerprint verification system is constituted by the fingerprint representation technique and the matching algorithm used. Approaches proposed in the literature [2-4] focused on the analysis of the ridge flow orientation, or the extraction of particular points, called "minutiae", that characterize the termination or the bifurcation of such ridges. Figure 1 illustrates the key aspects of these two fundamental approaches. Figure 1(a) shows the minutiae points extracted from a fingerprint image. Figure 1(b) shows the ridge flow of the fingerprint. The ridge flow defines the "texture" of fingerprint. Various methods for describing the texture of fingerprints by the orientation field have been proposed [3, 5, 6].

Matchers proposed so far use one of the two above fingerprint representations and produce authentication scores, that is, degrees of similarity between the input and the template fingerprints. Such scores take values in [0,1].

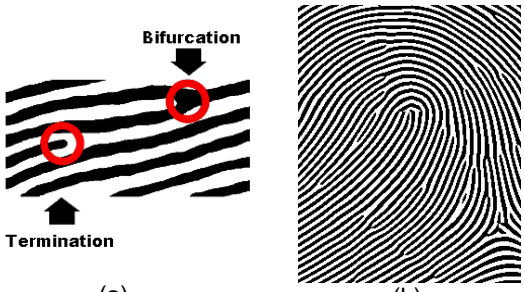

Figure 1. (a) "Minutiae" points of a fingerprint. (b) Fingerprint texture is characterized by the orientation of the ridge flow.

Authentication scores produced from matchers that use diverse fingerprint representations can be expected to exhibit a low degree of correlation. In other words, each score can be regarded as a different, and hopefully complementary, "feature" extracted from the same fingerprint pattern through different matching algorithms and/or different fingerprint representations. Therefore, two or more matching scores computed from diverse fingerprint matchers can be used to improve the performances of the best individual matcher. This strategy is called fusion of multiple matchers. Although such strategy appeared to be promising on the basis of recent experimental results [4, 7, 8], few papers investigated the potentialities of fusing multiple fingerprint matchers.

It is well known that the ability in separating genuine and impostor users is the fundamental requirement for an automatic personal verification system. High separation degree between genuine and impostor classes allows reducing the number of impostors which are classified as genuine users, and the number of genuine users which are classified as impostors.

In this paper, a fusion rule explicitly aimed to maximize the separation between genuine and impostor classes is proposed. This rule is implemented by a perceptron with a "class-separation" loss function. Our perceptron takes as input the verification scores of

multiple matchers and produces as output "fused" scores that maximize the separation between impostor and genuine users' classes.

Section 2 provides the basic concepts used in this paper. Section 3 presents our perceptron for fusion of multiple fingerprint matchers. Section 4 briefly describes the individual matchers used. Section 5 reports the experimental results that show the effectiveness of our approach. Section 6 draws preliminary conclusions.

2. Basic concepts

The so called verification score *"s"* provides the degree of similarity between two fingerprint patterns, and takes values in [0,1]. The minimum value indicate that the two fingerprints are definitely different, while the maximum value indicate that the two fingerprints are the same fingerprint.

Let us define the concepts of "impostor" and "genuine" classes of users. A fingerprint pattern belongs to the *genuine class* if the identity of her/his possessor corresponds to the claimed one. The opposite holds for the *impostor class*.

The design of any fingerprint verification system depends on the estimate of the two posterior probabilities $p(s|genuine)$ and $p(s|impostor)$, and the selection of the so called *acceptance threshold* s^*. If the score is higher than the acceptance threshold, the claimed identity is accepted and the person is classified as a genuine user. Otherwise, she/he is classified as an impostor and the access to the required resource is denied.

Authentication errors obviously depends on the acceptance threshold. They are called False Acceptance Rate (FAR) and False Rejection Rate (FRR), respectively. Eq. (1) and eq. (2) show the FAR and FRR definitions:

$$FAR(s^*) = \int_{s^*}^{1} p(s \mid impostor)ds \quad \textbf{(1)}$$

$$FRR(s^*) = \int_{0}^{s^*} p(s \mid genuine)ds \quad \textbf{(2)}$$

3. Perceptron-based fusion of multiple matchers

3.1. The perceptron-based fusion module

In multiple fingerprint verification systems, each individual matcher provides a score as input to the module that performs fusion. Generally speaking, the Input/Output characterization of the fusion module is the following:

(a) inputs are the verification scores of the individual matchers;
(b) the output is still a verification score, in order to consider the whole fusion module as a matcher.

The fusion model can be regarded as a transformation function which processes the input scores and produces a "fused" verification score.

Our fusion module is aimed to implement a score transformation function that maximizes the statistical separation between the posterior probabilities of genuine and impostor classes. In the following, we use the term "separation" between genuine and impostor classes for denoting the statistical separation of the related posterior probabilities.

In order to design such module, we used the perceptron classifier. In fact, using the perceptron logistic function, it is possible to satisfy easily the above I/O requirements.

Let $s_1,...,s_N$ be the matching scores provided by N individual matchers. The logistic function of perceptron implements the following fusion rule [8]:

$$s = \frac{1}{1+\exp\left[-\left(w_0 + w_1 s_1 + ... + w_N s_N\right)\right]} \quad \textbf{(3)}$$

Eq. (3) points out that the inputs of perceptron are the scores provided by the N individual matchers, and the output s is the final verification score.

The parameters $w_0, w_1,..., w_N$ are the "weights" of perceptron. For the standard perceptron, such weights are usually computed by a gradient descent algorithm with a least-squares loss (or "cost") function, or by a gradient descent algorithm with a cross-entropy loss function [9]. The so-called "delta rule" is used for weights update during the learning phase.

According to our goal, we substituted the standard loss functions with a function explicitly aimed to maximize the statistical separation between genuine and impostor classes. It is worth noting that the standard cost functions of the perceptron model are not explicitly aimed to this goal. In fact, the class separation is a consequence, but not the explicit goal of such cost functions.

It should be remarked that the use of logistic transform for fusing multiple fingerprint matchers has been previously proposed by A.K. Jain et al.. [8]. However, the goal of [8] was to estimate the parameters of the logistic transform in order to minimize FRR for a given FAR. Although Jain et al. were aware about the increase of class separation achievable by logistic transform (see Fig. 3 in [8]), their work was not focused on this aspect, and no experimental evidence on the increase of class separation was reported. In this work, we followed and expanded Jain et al. suggestion on the possible use of logistic transform for increasing class separation. We designed a perceptron-based fusion rule that estimate the parameters of the logistic transform in

order to maximize the statistical separation between genuine and impostor classes. In addition, we investigated by experiments the achievable increase of separation.

Finally, it should be noted that, although loss functions based on statistic separation measures have been already investigated for neural network learning [10], no previous work dealt with fingerprint recognition.

In the following, the loss function and the learning algorithm are described.

3.2. The learning algorithm

At this stage of our work, we used a parametric distance, called Fisher distance (FD), as measure of statistical separation between the genuine and impostor classes.

Let us indicate with μ_{gen} and σ^2_{gen} the mean and the variance of the "fused" scores of the genuine class, and with μ_{imp} and σ^2_{imp} the mean and the variance of the fused scores of the impostor class. With the term "fused" score, we mean the output of the fusion module, i.e., the output of our perceptron. The FD expression is as follows:

$$FD = \frac{(\mu_{gen} - \mu_{imp})^2}{\sigma^2_{gen} + \sigma^2_{imp}} \quad (4)$$

In our learning algorithm, the weights update is performed by the delta-rule:

$$w_i \leftarrow w_i + \eta \cdot \frac{\partial FD}{\partial w_i} \quad (5)$$

Using the Fisher distance, the gradient computation is simple, as it depends only on the four parameters in equation 4. According to eq. (4), the "contribute" of w_i to the FD-based gradient expression can be written as follows:

$$\frac{\partial FD}{\partial w_i} = \frac{2(\mu_{gen} - \mu_{imp})}{\sigma^2_{gen} + \sigma^2_{imp}} \cdot \frac{\partial(\mu_{gen} - \mu_{imp})}{\partial w_i} + \\ - \frac{(\mu_{gen} - \mu_{imp})^2}{(\sigma^2_{gen} + \sigma^2_{imp})^2} \cdot \frac{\partial(\sigma^2_{gen} + \sigma^2_{imp})}{\partial w_i} \quad (6)$$

A "batch" learning method is required for the computation related to eq. (5). Means and variances of the fused scores belonging to genuine and impostor classes can be computed only after the transformation of all samples by the logistic function.

For the sake of brevity, details of the gradient computation are omitted. However, the complete expression of the gradient is not difficult to obtain.

From eq. 6 it is easy to see that our learning algorithm looks for weights of logistic transform that separate distributions in terms of their means and reduce distributions variances. Accordingly, the distributions of genuine and impostor classes will be separated in terms of distance between their means, while their variances will be reduced. This is very important for personal identification purposes. As a consequence, it can be expected that the acceptance threshold of our combined matcher exhibits good generalisation performances, with small variations in terms of FAR and FRR around such threshold. Experiments reported in section 5 support this conclusion.

It is worth noting that other parametrical distances could be used. For example, the Bhattacharya and the Kullback-Leibler distances [10]. However, it is easy to show that such distances produce values always larger than those produced by the Fisher distance.

4. Fusion of Multiple Fingerprint Matchers

4.1. Methodology

In this preliminary stage of our research, we considered only two fingerprint verification algorithms.

Given the input fingerprint image associated to the claimed identity *i*:

- For each algorithm, compute the verification score (a real value on the interval [0,1]) between the given fingerprint and the "template" fingerprint stored in the database and associated to the identity *i*. Let s_m and s_t be the matching scores provided by the two individual algorithms.
- Apply the following transformation to the above scores s_m and s_t:

$$s = f(s_m, s_t) \quad (7)$$

All the fusion rules investigated in this paper can be regarded as the application of a particular transformation rule [7].

- Compare the obtained score value *s* with a threshold. The claimed identity is classified as "genuine" if:

$$s > th \quad (8)$$

otherwise it is classified as "impostor".

It is easy to see that the above methodology can be also used for the case of more than two verification algorithms.

4.2. The selected matching algorithms

In the literature, two main types of fingerprint verification algorithms have been proposed: the so called minutiae-based and texture-based algorithms [2-4]. Each type uses representations of fingerprints that are

substantially different, so that they can be expected to provide reasonably uncorrelated matching scores. As it is well known that fusion of multiple pattern classifiers is effective under the assumption of uncorrelated classifiers [11], for our experiments, we selected one algorithm for each type.

The selected minutiae-based algorithm is commonly referred as "String" algorithm [2, 4]. The ridge bifurcations and endings, usually called "minutiae" (Figure 2), are extracted from the input fingerprint image. Such "minutiae" set is compared with that of the template fingerprint. Such comparison is performed by considering each set as a "string". A "string" distance is computed and converted into a matching score. It is worth noting that other minutiae-based algorithms have been proposed [4], but "String" has shown the best performances [4].

Figure 2. Fingerprint representation based on the "map" of the minutiae-points locations and orientations.

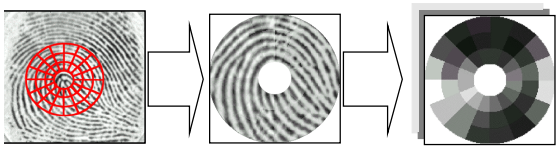

Figure 3. Fingerprint texture representation by the so called "finger-code"[3]. See Reference 3 for further details.

The selected texture-based algorithm is also known as "Filter" algorithm [3, 4]. The input fingerprint image is "partitioned" around its centre (the so called "core" point) by a tessellation [3] (Figure 3). A set of Gabor filters is applied to each sector of such tessellation. The outcome of each Gabor filter is evaluated for all pixels of each sector and the related standard deviation is computed. The feature vector made up of the standard deviations computed for each Gabor filter and for each sector is called "finger-code". The verification score is obtained by computing the Euclidean distance between the finger-code of the input image and the one of the template image. See Reference [3] for details.

Other texture-based algorithms have been proposed, but their use is often limited to the fingerprint classification task [5], for which the fingercode has shown the best performance [6]. In general, the characteristics of these approaches also constitute their main limitations, as the fingerprint description is less detailed with respect to the minutiae-based one. Consequently, performances are often lower.

4.3. Fusion of String and Filter algorithms

At this stage of our work, we compared our perceptron with simple fusion rules: the maximum score value (Max), the minimum score value (Min), the average of the scores (Mean), the product of the scores (Product). These transformations are known as "fixed" fusion rules, as they do not require any training phase [11].

Our aim was to compare the perceptron-based fusion rule, that requires a training phase for setting the weights of the logistic fusion transform, with simple and fixed fusion rules.

5. Experimental results

5.1. The Data Set

For our experiments, we used the FVC-DB1 data base that was recently introduced as a benchmark data set for fingerprint verification algorithms [12]. This data set is made up of 800 fingerprint images acquired from a low-cost optical sensor. The image size is 300x300 pixels and the resolution is 500 dpi. The number of identities is 100, and the number of images per identity is eight. In our experiments, 276 fingerprints were disregarded because the "core" point requested by the Filter algorithm could not be extracted in a reliable way.

5.2. Experimental plan

First of all, let us remark that, in fingerprint verification systems, the choice of the acceptance threshold (eq. (8)) is very critical. The rates of impostors (FAR) and the rate of genuine users rejected (FRR) strictly depend on such threshold. Typically, the criterion for selecting such threshold is to assess the score value that produces the same number of accepted impostors and rejected genuine users on the training set. This score value is called point of "Equal Error Rate" (EER).

In our experiments, the following evaluation protocol was used:

– For each matcher, or any combination of the two selected matching algorithms, we computed two sets of scores. The first one is the "genuine-matching scores" set ***G***, made up of all matchs among fingerprints of the same identity (in our experiments, all images were compared with all the other images of a given identity). A score of the set ***G*** belongs to the "genuine pattern" class. The second set is the "impostor matching scores" set ***I***, made up of all

comparisons among fingerprints of different identities. A score of the set ***I*** belongs to the "impostor" class. The total number of genuine matching scores was 1,440, and the total number of impostor matching scores was 135,586.

- We randomly subdivided the above sets in two parts, so that: ***G***=***G1***U***G2***, ***I***=***I1***U***I2***. ***G1*** and ***G2***, as well as ***I1*** and ***I2***, are disjoint sets, and have the same size. In order to increase the statistical significance of our results, five different partitions were used and reported results refer to the average.
- The training set ***Tr***={***G1***, ***I1***} was used to estimate the EER point and to train our modified perceptron, that is, to compute the weights of the logistic fusion function.
- The test set ***Tx***={***G2***, ***I2***} was used to assess the performances of algorithms, or combinations of algorithms, on novel patterns, using the acceptance threshold computed on the training set ***Tr***.

The individual fingerprint matchers and the different combinations of the two selected algorithms were assessed and compared in terms of EER on the training set, FAR, FRR (Tables 1-3), and Receiver Operating Characteristic curves (Figure 4) on the test set. All reported results are averaged on the five trials carried out for each experiment.

5.3. Experiments

The effectiveness of the multiple matchers fusion depends on the correlation of the individual matchers and on the differences in accuracy [11]. In order to show the effectiveness of our perceptron-based fusion rule for different degrees of correlation and performances of matchers, we simulated three experimental conditions:

(a) low degree of correlation among matchers, and different verification accuracy of the individual matchers;
(b) high degree of correlation among matchers and different verification accuracy of the individual matchers;
(c) low degree of correlation among matchers and similar verification accuracy of the individual matchers.

The different degrees of correlation allowed us to evaluate the capability of the various fusion rules to handle various degrees of error correlation among matchers. We remark that fixed fusion rules, like average or product of verification scores, work well for fusing uncorrelated matchers, while trainable fusion rules are usually requested for correlated matchers.

The differences in matchers performances allowed us to point out the advantages of our perceptron over fixed fusion rules. In fact, the benefits of the low correlation among matching scores could be reduced by the "scissor" in the performance of the individual matchers. As a consequence, the effectiveness of the fixed fusion rules could be also reduced. On the contrary, a trained rule like the perceptron-based one could be effective even in this case.

In order to create the above three experimental conditions, we generated two different sets of scores for the String matcher. This matcher has two parameters that allow varying the precision degree of the computed scores. The first parameter d_{max} tunes the maximum Euclidean distance that is allowed between the template minutia and the input minutiae. The second parameter α_{max} tunes the maximum angular distance that is allowed between the template minutia and the input minutiae. The "angular" distance is the difference between the template minutia orientation and the input minutia orientation. Therefore, the increase of such parameters decreases the reliability of the matching score.

The first set of matching scores was produced by reducing d_{max}. This means that the template minutia and the input minutia were considered similar if their Euclidean distance was very "low". The String algorithm so tuned was called "String 1".

The second set of matching scores was produced by reducing α_{max}. This means that the template minutia and the input minutia were considered similar if their angular distance was very "low". The String algorithm so tuned was called "String 2".

String 1 and String 2 exhibited a very high degree of correlation, as they use the same approach to score computation, and they also exhibited very different performances. On the other hand, verification scores produced by Filter matcher exhibited a low degree of correlation with those produced by String 1 and String 2, and also exhibited different performances.

Accordingly, we realised the above experimental conditions indicated with (a-c) by the following experiments:

(a) fusion of String 1 and Filter;
(b) fusion of String 1 and String 2;
(c) fusion of String 2 and Filter.

5.4. Results

Table 1 show the EER on the training set, FAR and FRR on the test set for the experiment (a), i.e., the fusion of String 1 and Filter. For this experiment, the fixed fusion rules effectiveness has been reduced by the "scissor" in performance between String 1 and Filter. In fact, the low correlation between String and Filter scores (the value of average correlation coefficient is 0.32) is not sufficient to guarantee an improvement with respect to the best individual matcher. However, our perceptron outperformed String 1, although the improvement is small. It is also worth noting that the reliability of a fingerprint verification system strongly depends on the difference between training set and test set performances.

Therefore, a "robust" fingerprint verification system should exhibit test set performances very close to the training set performances, that is, to exhibit a low generalisation error. Table 1 shows that our perceptron-based fusion rule has a good reliability and robustness. This is more evident from the FAR value, that is very close to the training set performance (EER).

Table 2 shows the EER on the training set, FAR and FRR on the test set for the experiment (b). The high degree of correlation (0.75) and the low accuracy of String 2 affected all fusion rules. In this case the fusion did not appear to be effective. The generalisation error of our perceptron-based fusion rule is a bit lower than that of String 1 (see the FRR value).

Table 3 shows the EER on the training set, FAR and FRR on the test set for the experiment (c). In this case the benefits of the low average correlation (0.33) and the same performance degree among individual matchers allowed fixed rules to improve the performance with respect to that of the best individual matcher. However, the best improvement is obtained with the perceptron. In this case, differences in performance are very significant (more than 1% for the perceptron-based fusion rule). Perceptron outperformed all individual and combined matchers used in our experiments and also exhibited very low generalisation errors.

	EER	FAR	FRR
String 1	1,5	1,6	1,8
Filter	6,5	6,5	7,2
Product	1,4	1,5	**1,6**
Mean	2,5	2,6	2,8
Min	1,5	1,6	1,8
Max	6,4	6,4	6,9
Perceptron	**1,2**	**1,3**	**1,6**

Table 1. Experiment (a). Average performances of individual and combined matchers in terms of EER percentage values on the training set, FAR and FRR percentage values on the test set.

	EER	FAR	FRR
String 1	**1,5**	1,6	1,8
String 2	7,1	7,0	7,7
Product	2,0	2,0	2,1
Mean	1,9	1,9	2,1
Min	4,2	4,2	4,8
Max	1,9	1,9	2,5
Perceptron	**1,5**	**1,5**	**1,6**

Table 2. Experiment (b). Average performances of individual and combined matchers in terms of EER percentage values on the training set, FAR and FRR percentage values on the test set.

	EER	FAR	FRR
Filter	6,5	6,5	7,2
String 2	7,1	7,0	7,7
Product	4,8	4,8	5,2
Mean	4,6	4,6	4,7
Min	7,1	7,0	7,7
Max	6,5	6,5	7,2
Perceptron	**3,3**	**3,3**	**3,6**

Table 3. Experiment (c). Average performances of individual and combined matchers in terms of EER percentage values on the training set, FAR and FRR percentage values on the test set.

5.5 Effectiveness of the class separation-based fusion rule

Figure 4 shows the ROC curve (FAR vs. FRR) on the test set for a variation of ±0.01 of the acceptance threshold computed on the training set.

The aim of such figure is to show that the behavior of our perceptron-based fusion rule *around* the acceptance threshold is more "stable" than that of the other fusion rules investigated in this paper. This is a property very important for personal verification systems. The usual performance parameters say nothing about the probability of correct classification of patterns that are close to the acceptance threshold. In fact, a pattern with a matching score close to such threshold could fall in a "uncertainty" region that does not allow to classify such pattern with sufficient reliability, due to the high number of patterns of the opposite class falling in such region. Performances could decrease dramatically, and the reliability of the personal verification system could be compromised. However, if the distributions are well separated, the acceptance threshold should be estimated near to the intersection of the "queues" of genuine and impostor distributions. Hence, the performance variation around the estimated acceptance threshold should be "smaller" than that produced by a transformation not explicitly aimed to increase the separation between genuine and impostor classes. The "uncertainty region" should also be smaller. In order to point out this effect, we computed the value *th* of the acceptance threshold at the EER on the training set. Then, we considered the interval [*th*-0.01, *th*+0.01]. We evaluated the FAR and the FRR on such interval and *on the test set* for the perceptron and the best fixed rule of the experiments a-c. FAR and FRR are plotted in Figure 4. For sake of brevity, we reported such curves only for the experiment (a). But we noticed the same behaviour in experiments (b) and (c). The modified perceptron presents FAR and FRR variations smaller than those of the best fixed rule. This result points out that the class separation allows to improve the degree of reliability of the personal identification system, for which small errors variations around the acceptance threshold are required. In particular, the small variation of the FAR allows us to

show the effectiveness of the perceptron-based fusion rule.

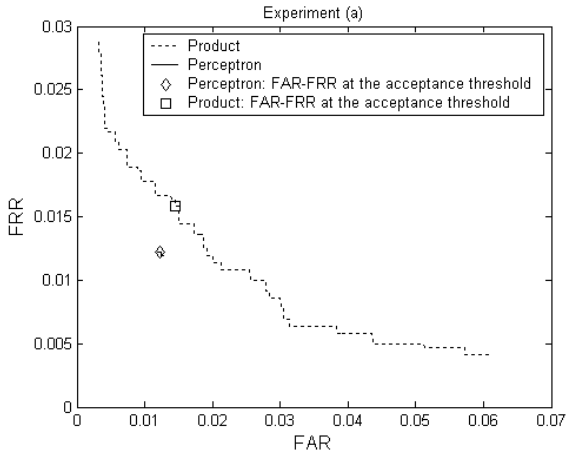

Figure 4. FAR and FRR values for the acceptance threshold evaluated on the training set (diamond for the modified perceptron, square for the best fixed rule), and the plot of FAR and FRR related to a variation of 0.01 of the acceptance threshold. See Section 5.5 for details.

6. Conclusions

In this paper, we presented a perceptron with a class-separation loss function for fusing multiple fingerprint matchers. This fusion model is aimed maximising the separation between genuine and impostor classes.

Experimental results showed that our perceptron-based fusion rule is more effective and reliable than simple fusion rules. Although definitive conclusions cannot be drawn on the basis of these limited set of experiments, our perceptron appears to perform better than other fusion rules and exhibits a "stable" behaviour around the estimated acceptance threshold, as required for personal verification purposes.

References

[1] A.K. Jain, R. Bolle, S. Pankanti Eds., *BIOMETRIC – Personal Identification in Networked Society*, Kluwer Academic Publishers, Boston/Dordrecht/London, 1999.

[2] A.K. Jain, L. Hong, and R. Bolle, "On-line Fingerprint Verification", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4) 302-314, 1997.

[3] A.K. Jain, S. Prabhakar, L. Hong, and S. Pankanti, "Filterbank-based Fingerprint Matching", *IEEE Transactions on Image Processing*, 9(5) 846-859, 2000.

[4] S. Prabhakar and A.K. Jain, "Decision-level Fusion in Fingerprint Verification", *Pattern Recognition*, 35(4) 861-874, 2002.

[5] G.T. Candela, P.J. Grother, C.I. Watson, R.A. Wilkinson and C.L. Wilson, "PCASYS - A Pattern-Level Classification Automation System for Fingerprints", NIST tech. Report NISTIR 5647, 1995.

[6] A.K. Jain, S. Prabhakar, L. Hong, "A Multichannel Approach to Fingerprint Classification", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(4) 348-358, 1999.

[7] G.L. Marcialis and F. Roli, "Experimental results on fusion of multiple fingerprint matchers", J.Kittler and M.S.Nixon Eds., Proc. of the 4th Int. Conf. on Audio- and Video- Based Person Authentication AVBPA 2003 (Guildford, U.K., June, 9-11, 2003), Springer LNCS 2688, 814-820.

[8] A.K. Jain, S. Prabhakar, and S. Chen, "Combining Multiple Matchers for a High Security Fingerprint Verification System", *Pattern Recognition Letters*, 20(11-13) 1371-1379, 1999.

[9] C.M. Bishop, *Neural Networks for Pattern Recognition*, Oxford Univ. Press 1995.

[10] S. Haykin, *Neural Networks: A comprehensive foundation*, 2nd edition, Prentice-Hall, Englewood Cliffs, NJ, 1999.

[11] F. Roli and J. Kittler Eds., *Multiple Classifier Systems*, Springer-Verlag, Lecture Notes in Computer Science, Vol. 2364, 2002.

[12] D. Maio, D. Maltoni, R. Cappelli, J.L. Wayman, and A.K. Jain, "FVC-2000: Fingerprint Verification Competition", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3) 402-412, 2002.