IJDAR (2010) 13:33–47
DOI 10.1007/s10032-009-0101-0

FULL PAPER

Improving performance of HMM-based off-line signature verification systems through a multi-hypothesis approach

Luana Batista · Eric Granger · Robert Sabourin

Received: 16 February 2009 / Revised: 8 September 2009 / Accepted: 13 October 2009 / Published online: 17 November 2009

Abstract The neural and statistical classifiers employed in off-line signature verification (SV) systems are often designed from limited and unbalanced training data. In this article, an approach based on the combination of discrete Hidden Markov Models (HMMs) in the ROC space is proposed to improve the performance of these systems. Inspired by the multiple-hypothesis principle, this approach allows the system to select, from a set of different HMMs, the most suitable solution for a given input sample. By training an ensemble of user-specific HMMs with different number of states and different codebook sizes, and then combining these models in the ROC space, it is possible to construct a composite ROC curve that provides a more accurate estimation of system performance. Moreover, in testing mode, the corresponding operating points—which may be selected dynamically according to the risk associated with input samples—can significantly reduce the error rates. Experiments performed by using a real-world off-line SV database, with random, simple and skilled forgeries, indicate that the multi-hypothesis approach can reduce the average error rates by more than 17%, as well as the number of HMM states by 48%.

Keywords Off-line signature verification · ROC curves · Codebook selection · Hidden Markov Models · Pattern recognition

L. Batista (✉) · E. Granger · R. Sabourin
Laboratoire d'imagerie, de vision et d'intelligence artificielle,
École de Technologie Supérieure, 1100 rue Notre-Dame Ouest,
Montreal, QC H3C 1K3, Canada
e-mail: lbatista@livia.etsmtl.ca

E. Granger
e-mail: eric.granger@etsmtl.ca

R. Sabourin
e-mail: robert.sabourin@etsmtl.ca

1 Introduction

Signature verification (SV) systems seek to authenticate the identity of an individual, based on the analysis of his/her signature, through a process that discriminates a genuine signature from a forgery [26]. SV systems are relevant in many situations where handwritten signatures are currently used, such as cashing checks, transactions with credit cards, and authenticating documents [15]. In off-line SV, signatures are available on sheets of paper, which are later scanned in order to obtain a digital representation. Given a digitized signature, an off-line SV system will perform preprocessing, feature extraction, and classification (also called verification). For complete and recent surveys of off-line SV, see [1] and [17].

Among several well-known classification methods used in off-line SV, the discrete Hidden Markov Model (HMM)—a finite stochastic automata used to model sequences of observations—adapts easily to the dynamic characteristics of the western handwriting [21]. Despite the fact that the HMM is a generative classifier [5], which generally requires a considerable amount of training data to achieve a high level of performance, the HMM-based off-line SV systems are often designed from limited and unbalanced data. This occurs because the dataset used to model a writer generally contains a reduced number of genuine signatures against several random forgeries [25]. Another issue with the use of discrete HMMs in off-line SV is the design of codebooks.[1] Typically, the data used to generate codebooks are the same data that are employed to train the HMMs. In the study of Ferrer et al. [8], all writers share an universal codebook generated with their training data. The main drawback of this strategy is

[1] A codebook contains a set of symbols, each one associated with a cluster of feature vectors, used to generate sequences of discrete observations in discrete HMM-based systems.

the need to reconstruct the codebook whenever a new writer is added to the system. According to Rigoll and Kosmala [28], the utilization of user-specific codebooks, generated by using only the training signatures of a particular writer, adds one more personal characteristic to the verification process. However, it was observed that this strategy leads to poor system performance whenever a limited amount of training signatures is available [6]. Finally, these systems have been evaluated through error rates calculated from a single threshold, assuming that the classification costs are always the same.

Though not fully explored in literature, it has recently been shown that the Receiver Operating Characteristic (ROC) curve—where the true positive rates (TPR) are plotted as function of the false positive rates (FPR)—provides a powerful tool for evaluating, combining, and comparing off-line SV systems [3,25]. Several interesting properties can be observed from ROC curves. First, the area under the ROC curve (AUC) is equivalent to the probability that the classifier will rank a randomly chosen positive sample higher than a randomly chosen negative sample [7]. This measure is useful to characterize the system performance by using a single scalar value. In addition, the optimal threshold for a given class distribution lies on the ROC convex hull, which is defined as being the smallest convex set containing the points of the ROC curve. Finally, by taking into account several operating points (thresholds), the ROC curve allows to analyze these systems under different classification costs [7]. This property is useful, for instance, for off-line SV systems where the operating points are selected dynamically according to the risk associated with the amount of a bank check.

In this article, an approach based on the combination of classifiers in the ROC space is proposed to improve performance of off-line SV systems designed from limited and unbalanced data. By training an ensemble of classifiers with different parameters, and then selecting the best classifier(s) for each operating point, it is possible to construct a composite ROC curve that provides a more accurate estimation of system performance during training and significantly reduces the error rates during operations. This approach follows the multiple-hypothesis principle [11], which request the system to propagate several hypothesis throughout the recognition steps, generating a hierarchical tree of possible solutions.

Although the proposed approach may be applied to any type of statistical or neural 2-class classifier, this article involves HMMs trained by using different number of states and different codebooks. In order to avoid the problems caused by the use of limited codebooks, two databases are employed by the proposed approach: the development database (DB_{dev}), which contains the signatures of writers used to generate the candidate codebooks; and the exploitation database (DB_{exp}), which contains the signatures of the final users. The latter contains samples used to train, validate and test HMMs.

The rest of this article is organized as follows. The next section presents a brief survey of off-line SV systems based on discrete HMMs. Section 3 describes some key issues that affect the performance of off-line SV systems. Then, Sect. 4 describes the proposed approach and its advantages. Finally, in Sect. 5, the experimental results are shown and discussed.

2 Off-line SV systems based on local discrete HMMs

A traditional off-line SV system based on local discrete HMMs follows the steps shown by Fig. 1. At first, the signature is scanned in order to obtain a digital representation composed of $Q \times R$ pixels. Then, the signature image is considered as a discrete 2D function $I(x, y)$, where $x = 0, 1, 2, \ldots, Q$ and $y = 0, 1, 2, \ldots, R$ denote the spatial coordinates, and the value of I in any (x, y) corresponds to the grey level (generally a value from 0 to 255) in that point [13]. After applying some corrections to the signature image (such as noise removal and centering), a set of feature vectors $V = \{V_1, V_2, \ldots, V_L\}$ are extracted and quantized in a sequence of discrete observations $O = \{o_1, o_2, \ldots, o_L\}$, where each observation is a symbol provided by the codebook. Finally, the signature is classified as genuine or forgery by using the corresponding user-specific HMM λ. Although Fig. 1 illustrates only one HMM λ, the classification subsystem is composed of several local classifiers, where each one is a discrete HMM trained with data corresponding to a specific writer.[2] Since each writer is associated with a single classifier, these systems are also referred in this article as single-hypothesis systems. Multi-hypothesis systems, on the other hand, allow a same writer to employ different classifiers depending on the input samples.

A common way to report the SV system performance is in terms of error rates. The false negative rate (FNR) is related to genuine signatures which were misclassified as forgeries. Whereas the false positive rate (FPR) is related to forgeries which were misclassified as genuine signatures. FNR and FPR, also known as type 1 and type 2 errors, respectively, can be used to generate a pair {TPR, FPR} in the ROC space, since TPR $= 1 -$ FNR. Generally, the FPR is calculated regarding three forgery types: random, simple, and skilled. The random forgery is usually a genuine signature sample belonging to a different writer. It is produced when the forger has no access to the genuine samples, not even the writer's name. In the case of simple forgeries, only the writer's name is known. Thus, the forger reproduces the signature in his/her own style [4]. A simulated forgery represents a reasonable imitation of a genuine signature [19]. Finally, the average

[2] In this article, the term *local (or user-specific) classifier* is used to differentiate from systems where a same *global classifier* is shared by all users.

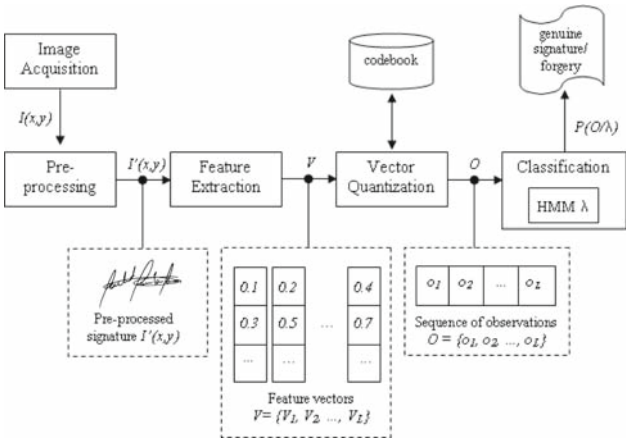

Fig. 1 Block diagram of a traditional off-line SV system based on local discrete HMMs

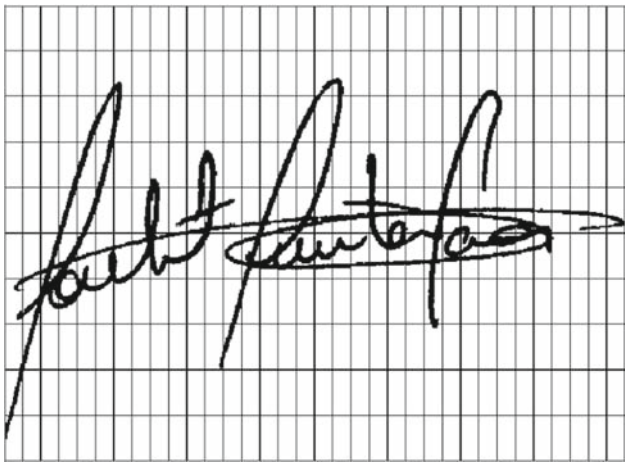

Fig. 2 Example of grid segmentation scheme

error rate (AER) is related to the total error of the system, where the type 1 and type 2 errors are averaged taking into account the a priori probabilities. On the other hand, if the decision threshold of a system is set to have the percentage of false negatives approximately equal to the percentage of false positives, the equal error rate (EER) is calculated.

Typically, the set of random forgeries of a specific writer is composed of true signatures from the remaining writers in the system. In this article, it is assumed that only random forgeries are used to train and validate classifiers, as well as to generate codebooks. In fact, it is not always possible to obtain samples of forgeries; and, in many real-world applications (e.g., banking), it becomes impracticable [24]. On the other hand, all types of forgeries are used to test the system performance.

The rest of this section provides additional details on the feature extraction, vector quantization, and classification sub-systems considered in this article.

2.1 Feature extraction

Two classes of features are used in off-line SV [1]: (i) static, related to the signature shape and (ii) pseudo-dynamic, related to the dynamics of the writing. These features can be extracted locally, if the signature is viewed as a set of segmented regions, or globally, if the signature is viewed as a whole. Since the HMMs are used to model sequence of observations, the local approach is typically employed [6,19,20,28]. In the grid segmentation scheme of Justino et al. [19,20], all images are aligned to the left and divided in 60 horizontal cells, where each cell is a rectangle composed of 16×40 pixels. Then, to absorb the horizontal variability of the signatures, the blank cells in the end of the images are discarded. Therefore, the images may have a variable number of horizontal cells, while the number of vertical cells is always 10. Figure 2 shows an example of a signature image with its final width. Note that the segmentation can be performed both in horizontal and vertical directions.

Each column of cells is then converted into a feature vector, where each vector element contains the density of pixels in its respective cell. In other words, each vector element is a value between 0 and 1 which corresponds to the number of black pixels in a cell divided by the total number of pixels of this cell. Other static features, such as pixel distribution and gravity center, as well as pseudo-dynamic features, such as axial slant and stroke curvature, have been extracted through this segmentation scheme [19,22].

2.2 Vector quantization

In order to generate a sequence of discrete observations, each extracted feature vector is quantized as one of the previously computed symbols of the codebook. This codebook may be generated through an iterative clustering algorithm called *K*-means [23]. As explained by Algorithm 1, NV feature vectors are separated into NC clusters $\mathcal{C}_i$, where each cluster $\mathcal{C}_i$ represents a symbol in the codebook.

Algorithm 1 Description of the *K-means* algorithm, where $\mathcal{C}_i$ is the i^{th} cluster with centroid c_i.

Inputs: $V_n (1 \leq n \leq NV)$, the set of feature vectors; NC, the number of centroids/clusters

Outputs: $c_i (1 \leq i \leq NC)$, the set of updated centroids

1: from the set of feature vectors V_n, choose an initial set of NC centroids c_i

2: classify the feature vectors V_n into the clusters $\mathcal{C}_i$ by using the nearest neighbor rule:

$V_n \in \mathcal{C}_i,\ iff\ d(V_n, c_i) \leq d(V_n, c_j),\ j \neq i, 1 \leq j \leq NC$

3: update the centroid of each cluster by averaging its corresponding training vectors

4: **if** the centroids have changed **then**

5: go to step 2

6: **end if**

2.3 Classification

Once the sequences of discrete observations are obtained, they are used to train and test the HMMs. A discrete HMM can be defined as $\lambda = (\mathcal{N}, \mathcal{M}, \mathcal{A}, \mathcal{B}, \pi)$, where [27]:

- $\mathcal{N}$ is the number of distinct states in the model. The set of states is denoted by $\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_\mathcal{N}\}$, and s_τ is the state at time τ;
- $\mathcal{M}$ is the alphabet size, that is, the number of distinct observation symbols per state. The set of symbols is denoted by $\mathcal{V} = \{v_1, v_2, \ldots, v_\mathcal{M}\}$;
- $\mathcal{A}$ is the state transition probability distribution, denoted by $\mathcal{A} = \{a_{ij}\}$, where $a_{ij} = \mathcal{P}\left[q_{\tau+1} = \mathcal{S}_j | s_\tau = \mathcal{S}_i\right]$, and $1 \leq i, j \leq \mathcal{N}$;
- $\mathcal{B}$ is the observation symbol probability distribution, denoted by $\mathcal{B} = \{b_j(k)\}$, where $b_j(k) = \mathcal{P}\left[v_k \text{ at } \tau | s_\tau = \mathcal{S}_j\right]$, and $1 \leq k \leq \mathcal{M}$, $1 \leq j \leq \mathcal{N}$;
- and π is the initial state distribution, denoted by $\pi = \{\pi_i\}$, where $\pi_i = \mathcal{P}[q_1 = S_i]$, and $1 \leq i \leq \mathcal{N}$.

HMMs can be classified in two main types of topologies [27]: the *ergotic* and the *left-to-right* topologies. The *ergotic* topology is a specific case of a fully connected model in which every state can be reached from any other state in a finite number of steps. With the *left-to-right* topology, the state indices increase from left to right (that is, $a_{ij} = 0$, $j < i$), and no transitions are allowed to states whose indices are lower than the current state. As consequence, the sequence of observations must begin in $\mathcal{S}_1$ (that is, $\pi_i = 0$ when $i \neq 1$, and $\pi_i = 1$ when $i = 1$) and must end in $\mathcal{S}_N$. Finally, given a model initialized according to the constraints described so far, there are three tasks of interest [16]:

1. *The Evaluation Problem*. Given a model λ and a sequence of observations $O = \{o_1, o_2, \ldots, o_L\}$, what is the probability of the model λ to have generated the observation O, that is, $\mathcal{P}(O|\lambda)$;
2. *The Decoding Problem*. Given a model λ, what is the most likely sequence of states $\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_\mathcal{N}\}$ in the model that produces the sequence of observations $O = \{o_1, o_2, \ldots, o_L\}$;
3. *The Learning Problem*. Given a model λ and a set of observations sequences, how the model parameters can be adjusted so as to maximize the joint probability $\prod_O \mathcal{P}(O|\lambda)$.

An advantage of the discrete HMMs is that it is not necessary to have a priori knowledge about the probability distributions to model a signal [2]. With enough representative training data, it is possible to adjust the parameters for the HMM. Algorithms based on the expectation–maximization (E–M) technique (e.g., the Baum–Welch algorithm) are generally used to perform this task [14].

Typically, only genuine signatures are used for training a local HMM for SV. In this case, the decision boundary between impostor and genuine spaces is defined by using a validation set that contains samples of both classes. Another particularity of HMM-based SV systems is the use of the *left-to-right* topology. Indeed, this topology is perfectly adapted to the dynamic characteristics of the occidental handwritten, in which the hand movements are always from left to right.

3 Challenges with the single-hypothesis approach

In this article, it is assumed that the performance of the whole SV system is measured by an overall ROC curve obtained from a set of user-specific ROC curves. Averaging methods have been used to group ROC curves produced from different user-specific classifiers in single-hypothesis systems [7,18]. Jain and Ross [18], for example, proposed a method to generate an averaged ROC curve taking into account user-specific thresholds.[3] At first, the cumulative histogram of random forgery scores of each user i is computed. Then, the similarity scores (thresholds) providing a same value of cumulative frequency, γ, are used to compute the operating points $\{\text{TPR}_i(\gamma), \text{FPR}_i(\gamma)\}$. Finally, the operating points associated with a same γ are averaged. Note that γ can be viewed as the true negative rate (TNR = negatives (forgeries) correctly classified/total of negatives) and that it may be associated with different thresholds. Figure 3 shows an example where the thresholds associated with $\gamma = 0.3$ are different for users 1 and 2, that is $t_{\text{user1}}(0.3) \cong -5.6$ and $t_{\text{user2}}(0.3) \cong -6.4$.

In off-line SV, where the dataset used to model a writer signature generally contains a reduced number of genuine samples against several random forgeries, it is common to obtain ROC curves with concave areas. In general, a concave area indicates that the ranking provided by the classifier in this region is worse than random [9]. Figure 4a shows an example of score distribution in an off-line SV system. The positive class, P, contains only 10 genuine samples of a given writer, while the negative class, N, contains 100 samples of forgeries. Due the limited amount of samples in the positive class, the resulting ROC curve (see Fig. 4b) presents three concave areas, which correspond to low-quality predictions [9]. For example, the similarity scores between -1.2 and 0 provide TPRs of 90%. The result of averaging the ROC curves related to the models of 100 different writers, by using the Ross's method, is illustrated by Fig. 5.

[3] Since biometric systems typically use a common threshold across users, Jain and Ross [18] have shown that it is possible to improve system performance by setting user-specific thresholds.

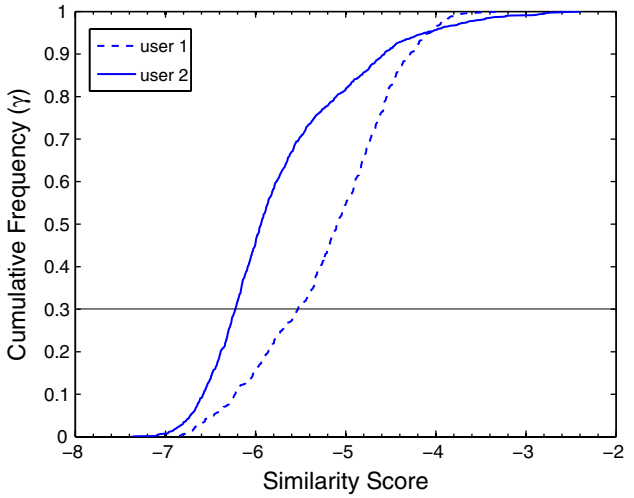

Fig. 3 Cumulative histogram of random forgery scores regarding two different users in a single-hypothesis system. The *horizontal line* indicates that $\gamma = 0.3$ is associated to two different thresholds, that is, $t_{\text{user1}}(0.3) \cong -5.6$ and $t_{\text{user2}}(0.3) \cong -6.4$

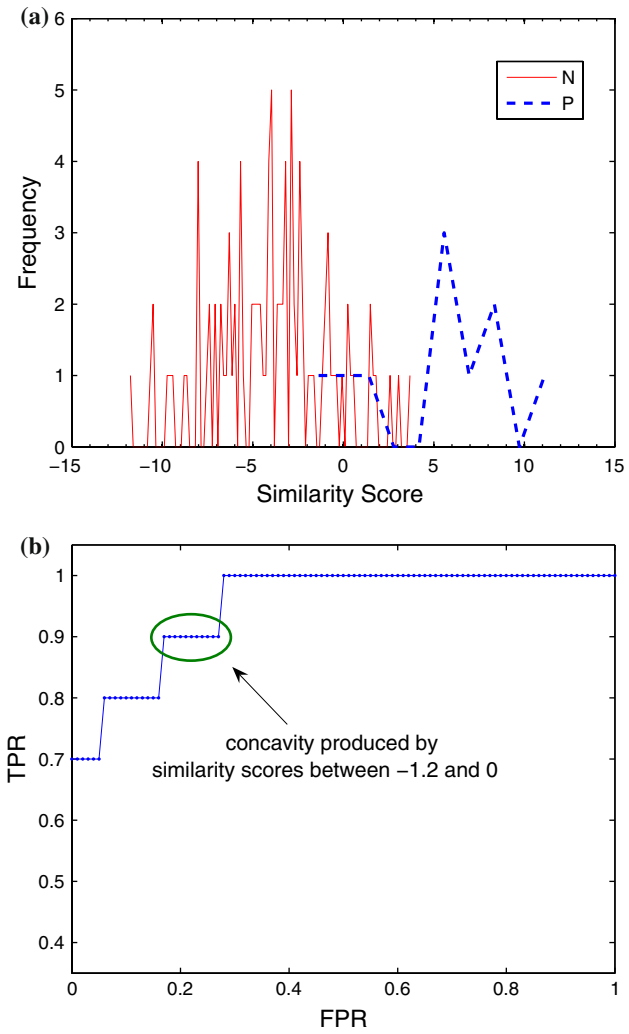

Fig. 4 **a** Typical score distribution of a writer, composed of 10 positive samples (genuine signatures) versus 100 negative samples (forgeries). **b** Due the limited amount of samples in the positive class, the corresponding ROC curve presents three concavities

Note that the imperfections of individual ROC curves are hidden within the average points, which can be observed with any averaging algorithm.

The drawback of using an averaged ROC curve can be observed during the selection of optimal thresholds in the respective convex hull. Given two γ in the convex hull, γ_1 and γ_2, where each one minimizes a different set of costs [30], γ_2 should provide a TPR higher than γ_1 whenever $\gamma_1 > \gamma_2$. However, regarding a user-specific ROC curve, γ_1 and γ_2 may fall in a same concave area, providing identical TPRs. An example is illustrated by Fig. 5, where TPR($\gamma = 0.86$) is higher than TPR($\gamma = 0.91$) in the global convex hull, but, in the user-specific ROC curve, TPR($\gamma = 0.86$) is equal to TPR($\gamma = 0.91$).

4 A multi-hypothesis approach

Based on the combination of HMMs trained with different number of states, the approach proposed in this section provides a solution to repair concavities of user-specific ROC curves while generating a high quality averaged ROC curve. The utilization of different HMMs is motivated by the fact that the superiority of a classifier over another may not occur on the whole ROC space [7]. Indeed, in off-line SV systems where the optimal number of states for a HMM is found empirically by a cross-validation process,[4] it is often observed that the best HMM is not superior than the other intermediate/ sub-optimal HMMs in all operating points of the ROC space. Three steps are involved in the proposed multi- hypothesis approach: model selection, combination, and averaging.

4.1 Model selection

This step allows to select the best classifier for each operating point such that every user's performance is optimized. In this article, a ROC outer boundary is constructed for each user in order to encapsulate the best operating points provided multiple HMMs, each one trained by using a different number of states. Given a set of ROC curves generated from different classifiers associated with a same user, the process consists in splitting the x-axis $\in [0, 1]$ into a number of bins, and within each bin finding the pair (FPR, TPR) having the

[4] Given a set of HMMs trained with different number of states, the cross-validation process selects the HMM providing the highest training probability, that is, the HMM that the best learned the training samples [6,21].

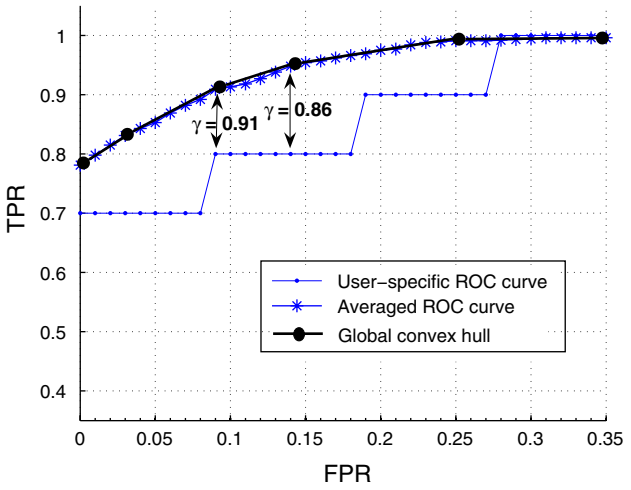

Fig. 5 Averaged ROC curve obtained by applying the Ross's method to 100 different user-specific ROC curves. The global convex hull is composed of a set of optimal thresholds that minimize different classification costs. However, these thresholds may fall in concave areas of the user-specific ROC curves, as indicated by $\gamma = 0.91$ and $\gamma = 0.86$

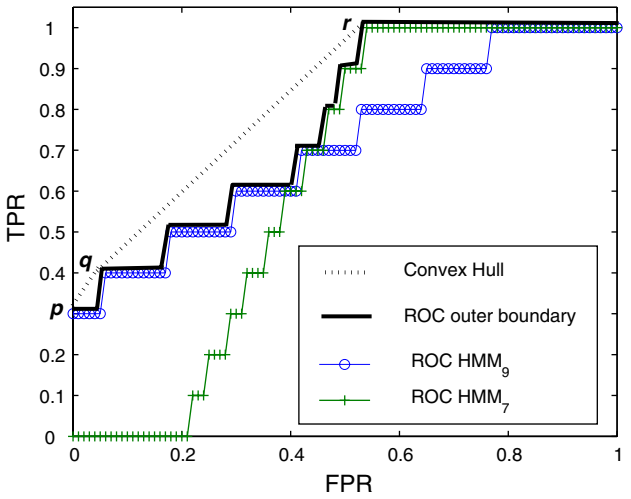

Fig. 6 ROC outer boundary constructed from ROC curves of two different HMMs. Above TPR = 0.7, the operating points are taken from HMM_7, while below TPR = 0.7, the operating points correspond to HMM_9

largest value of TPR. While the ROC outer boundary is being generated, the best HMM_j, where j is the number of states, is automatically chosen for each operating point. Figure 6 shows an example of a user-specific ROC outer boundary constructed from ROC curves of two different classifiers, HMM_7 and HMM_9. The corresponding convex hull is composed of three vertices, p, q and r, where p and q are associated with HMM_9, and r is associated with HMM_7.

Algorithm 2 presents the approach for generating the outer boundary of ROC curves produced by training $L_{\min}$ HMMs on a same dataset but with different number of states; where

Algorithm 2 Generating ROC outer boundaries from different HMMs.

Inputs: the training and validation sets
Outputs: the set of ROC outer boundaries
for each user $i = 1, 2, \ldots, M$ **do**
  for each number of states $j = 2, 3, \ldots, L_{min}$ **do**
    train an HMM with j states
    calculate the ROC curve j by using the validation set
  end for
  for each bin $k \in [0, 1]$ **do**
    let $HMM(k)$ be the classifier associated to the current bin
    set $TPR(k) \leftarrow 0$
    for each ROC curve j **do**
      if $TPR_j(k) > TPR(k)$ **then**
        $TPR(k) \leftarrow TPR_j(k)$
        $FPR(k) \leftarrow FPR_j(k)$
        $HMM(k) \leftarrow HMM_j(k)$
      end if
    end for
  end for
  use the pairs $\{\text{FPR}(k), TPR(k)\}$ to generate the ROC outer boundary i
end for

$L_{\min}$ is the number of observations in the shorter training sequence.

Note that this process can also be extended for multiple codebooks. In other words, by training an ensemble of HMMs with different codebook sizes and different number of states, each bin can be associated with the pair {codebook, state} providing the highest TPR. Therefore, depending on the operating point, a same individual may use a different model, denoted as HMM_j^{NC}, trained with j states and with a codebook of NC clusters.

4.2 Combination

This step allows to interpolate between two consecutive classifiers on the ROC curve in order to obtain not yet reached operating points. In this article, the method proposed by Scott et. al [29] is applied to the ROC outer boundaries in order to repair concavities. Given two vertices A and B on the convex hull, it is possible to realize a point C, located between A and B, by randomly choosing A or B. The probability of selecting one of the two operating points is determined by the distance of C regarding A and B. Equations 1 and 2 indicate how a given FPR_C can be obtained. The expected operating point (FPR_C, TPR_C) is given by Eqs. 3 and 4 [29].

By using the Eqs. 1–4, Algorithm 3 can realize any FPR_C located between two consecutive classifiers, A and B, where each classifier corresponds to a hull vertex of a user-specific ROC outer boundary. Figure 7 presents an example of a maximum realizable ROC (MRROC) curve.

Algorithm 3 Generating MRROC curves.

Inputs: the set of ROC outer boundaries
Outputs: the set of MRROC curves
for each user $i = 1, 2, \ldots, M$ **do**
  for each $FPR_C \in [0, 1]$ **do**
    find, in the convex hull of his/her ROC outer boundary, the pair of classifiers (A, B) able to realize FPR_C
    calculate:
    //the probability of A to be chosen (Eq. 1):
    $\mathcal{P}(A) \leftarrow \frac{(FPR_C - FPR_i(B))}{(FPR_i(A) - FPR_i(B))}$
    //the probability of B to be chosen (Eq. 2):
    $\mathcal{P}(B) \leftarrow 1 - \mathcal{P}(A)$
    //the expected operating points (Eqs. 3 and 4):
    $FPR_C \leftarrow (\mathcal{P}(A) \cdot FPR(A)) + (\mathcal{P}(B) \cdot FPR_i(B))$
    $TPR_C \leftarrow (\mathcal{P}(A) \cdot TPR(A)) + (\mathcal{P}(B) \cdot TPR_i(B))$
  end for
  use the pairs $\{FPR_C, TPR_C\}$ to generate the MRROC curve i
end for

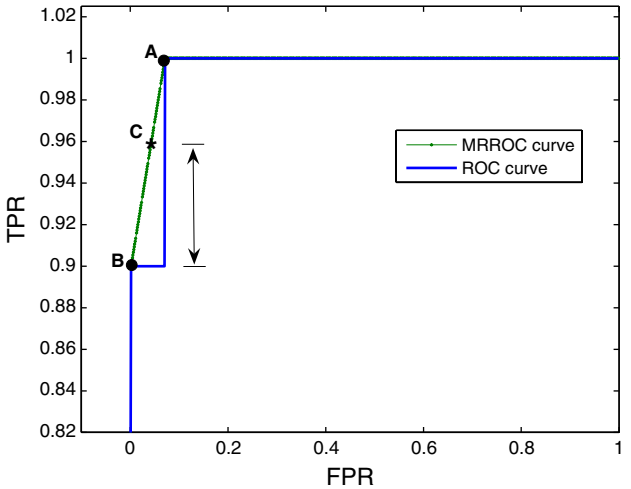

Fig. 7 Example of MRROC curve. By using combination, any operating point C between A and B is realizable. In this example, for a same FPR, the TPR associated with C could be improved from 90 to 96%

4.3 Averaging

Finally, a process based on the Jain and Ross's method [18] is used to group the operating points already computed during the combination step. Given a γ, the Algorithm 4 searches the pairs $\{\text{TPR}_i(\gamma), \text{FPR}_i(\gamma)\}$ where $\text{FPR}_i(\gamma) = 1-\gamma$ (recalling that γ can be viewed as the TNR, and that $\text{FPR} = 1 - \text{TNR}$). Then, the operating points corresponding to a same γ are averaged and used to generate the averaged ROC curve.

4.4 Testing

During operations, γ is used to retrieve the set of user-specific HMMs/thres- holds which will be applied on input samples. Figure 8 illustrates two possible situations linking the averaged ROC curve and a user-specific MRROC curve. In the first case, γ falls directly on HMM_7. Thus, the user-specific threshold associated to this γ will be used to classify the input samples. In the second case, the requested γ is obtained by combining classifiers HMM_7 and HMM_9. That is, each test sample must be randomly assigned to either HMM_7 or to HMM_9, according to the probabilities given by Eqs. 1 and 2. Note that a test sample is not assigned to both classifiers at the same time. However, since a fusion strategy is incorporated to the process, there are no restrictions to the use of combination in this level.

Algorithm 4 Generating the averaged ROC curve.

Inputs: the set of MRROC curves
Outputs: the averaged ROC curve
for each value of $\gamma \in [0, 1]$ **do**
  set $\{FPR_i(\gamma), TPR_i(\gamma)\} \leftarrow 0$
  for each user $i = 1, 2, \ldots, M$ **do**
    find, in his/her MRROC curve, the operating point associated with γ, that is:
    $\{FPR_i(\gamma), TPR_i(\gamma)\} | FPR_i(\gamma) = 1 - \gamma$
    update $FPR(\gamma)$ and $TPR(\gamma)$ as:
    $FPR(\gamma) \leftarrow FPR(\gamma) + FPR_i(\gamma)$
    $TPR(\gamma) \leftarrow TPR(\gamma) + TPR_i(\gamma)$
  end for
  divide $FPR(\gamma)$ and $TPR(\gamma)$ by the number of users M

  $FPR(\gamma) \leftarrow FPR(\gamma)/M$

  $TPR(\gamma) \leftarrow TPR(\gamma)/M$
end for
use the pairs $\{FPR(\gamma), TPR(\gamma)\}$ to generate the averaged ROC curve

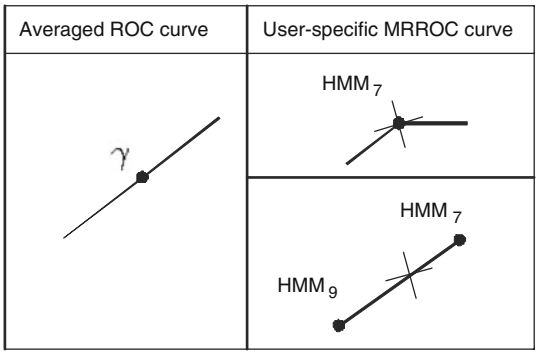

Fig. 8 Retrieving user-specific HMMs from an averaged ROC curve

5 Simulation results

Two types of codebook have been employed in off-line SV systems based on discrete HMMs: the *universal* codebook, shared by all writers in the population, and the *user-specific* codebook, adapted to a particular writer. However, as discussed before, these codebooks have typically been constructed by using the same data that are used to train the HMMs [8,28]. Instead, this section investigates the use of

an independent SV database to generate *universal* and *user-specific* codebooks. In both cases, the impact of applying the multi-hypothesis approach to improve the system performance is analysed.

The Brazilian signature database [10] is used for proof-of-concept computer simulations. It contains 7,920 samples of signatures that were digitized as 8-bit greyscale images over $400 \times 1{,}000$ pixels, at resolution of 300 dpi. The signatures were provided by 168 writers and are organized as follows:

- DB_{dev} is composed by 4,320 genuine samples supplied by 108 individuals. For each writer, there are 20 genuine samples for training, 10 genuine samples for validation and 10 genuine samples for test. Moreover, the genuine signatures of the other writers (i.e., 10 samples from 107 writers) are used as random forgeries in validation in order to perform ROC analysis.
- DB_{exp} contains 60 writers, each one with 40 samples of genuine signatures, 10 samples of simple forgery and 10 samples of skilled forgery. For each writer, 20 genuine samples are used for training, 10 genuine samples for validation, and 30 samples for test (10 genuine samples, 10 simple forgeries and 10 skilled forgeries). Moreover, 10 genuine samples are randomly selected from the other 59 writers and used as random forgeries to test the current model. For ROC analysis, however, the random forgeries are taken from DB_{dev} (i.e., 10 samples from 108 writers).

Given DB_{dev} and DB_{exp}, two experiments are performed:

1. *Off-line SV based on an universal codebook.* As proposed by Justino et al. [20], an evaluation system is generated with DB_{dev} by trying different candidate codebooks obtained from this same dataset. Then, the codebook which performs the best in DB_{dev} is selected to develop the final system, that is, a system for the writers in DB_{exp}. This strategy assumes that if DB_{dev} is representative of the whole population, the codebook selected to represent this database will also work well for DB_{exp}.
2. *Off-line SV based on user-specific codebooks.* By using the same candidate codebooks of the previous experiment, the idea is to find, for each writer in DB_{exp}, the codebook which the best adapts to his/ her individual characteristics. Thus, a same off-line SV system may work with multiple codebooks.

Therefore, while a DB_{dev} is used to construct codebooks, as well as to develop SV systems used to evaluate these codebooks, a DB_{exp} is employed in training, testing, and validation of the final SV system. As mentioned, the use of DB_{dev} allows to construct codebooks earlier, with sufficient data, independently of the writers in DB_{exp}.

The signature images are represented by means of density of pixels, extracted through the grid segmentation scheme described in Sect. 2. By varying the number of clusters from 10 to 150, in steps of 5, 29 candidate codebooks are obtained; where each one is constructed by using the first 30 signatures (training set + validation set) of each writer in DB_{dev}.

In the first experiment, in order to select the universal codebook, an averaged ROC curve is produced for each different version of the evaluation system. Then, the codebook providing the averaged ROC curve with highest AUC is chosen. In the second experiment, the user-specific codebooks are selected before the averaging step; that is, by choosing the codebook providing the MRROC curve with highest AUC for each writer. Then, the individual MRROC curves are averaged. Since high values of FPR are rarely useful in real situations, the AUC may be calculated regarding a specific region of interest of the ROC space.

Finally, in the first experiment, a ROC outer boundary is obtained for each writer by using the same universal codebook and a set of HMMs trained with a different number of states. In the second experiment, a ROC outer boundary is obtained for each writer by using his/her own codebook and a set of HMMs trained with a different number of states. Note that, in both experiments, the ROC curves are generated by using a validation set which contains only genuine samples and random forgeries.

5.1 Off-line SV based on an universal codebook

In a first step, the multi-hypothesis approach (Algorithms 2–4) was applied for each one of the candidate codebooks by using the whole validation set. Among the 29 averaged ROC curves, the codebook of 35 clusters (CB_{35}) provided the highest AUC (i.e., 0.997). Figure 9a shows the averaged ROC curve and AUC associated with CB_{35}. Then, in order to confirm this result, 10 different averaged ROC curves were generated for each codebook by randomly selecting a subset of signatures in the validation set. Since there are much more random forgeries than genuine samples, only the forgery space was changed for each averaged ROC curve. Figure 9b presents the relation between the number of clusters (NC) used in each codebook and the corresponding AUC, calculated on the entire averaged ROC space. This graphic confirms that the codebook with 35 clusters provides the highest AUCs for this population. Moreover, the Kruskal–Wallis test [12] indicates that the AUC values corresponding to CB_{35} are significantly different to the data regarding any other codebook in the graphic.

The next step consisted in applying the codebook CB_{35} to DB_{exp}, which is composed of 60 writers. As expected, CB_{35} also performed well for this population, providing an AUC

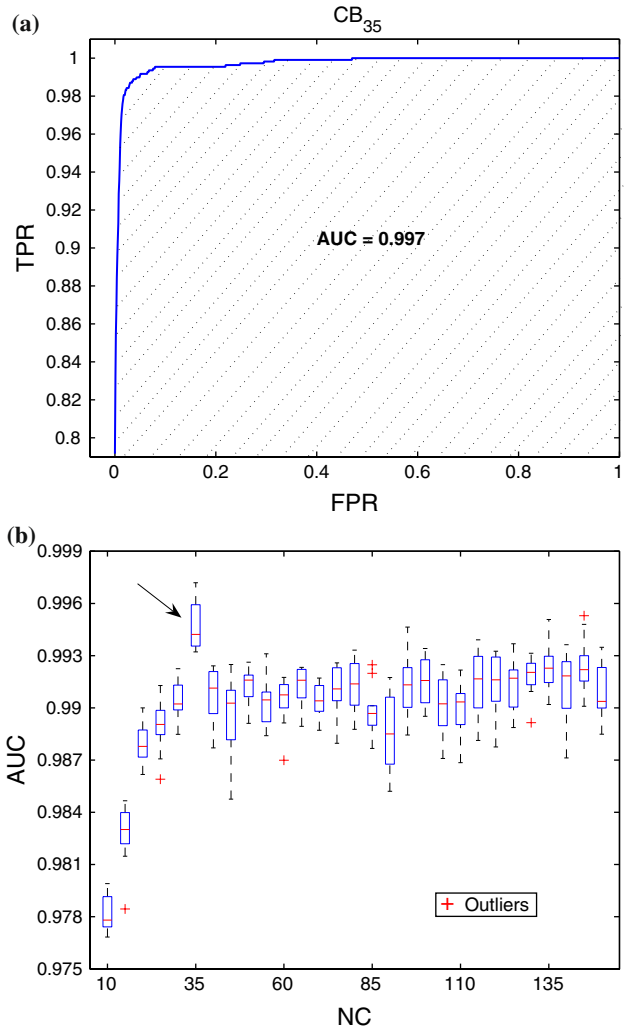

Fig. 9 **a** Composite ROC curve, with AUC = 0.997, provided by CB_{35} on the whole validation set of DB_{dev}. **b** AUC versus NC. As indicated by the *arrow*, CB_{35} represents the best codebook for DB_{dev} over 10 averaged ROC curves generated with different validation subsets of DB_{dev}

of 0.998 in the region of interest (i.e., between FPR = 0 and FPR = 0.1) of the averaged ROC space. The ROC curve of the multi-hypothesis system is indicated by the star-dashed line in Fig. 10. Whereas the circle-dashed line represents the baseline or single-hypothesis system used for comparisons. In this system, only the HMM which performs the best in the cross-validation process [6,21] is considered for generating a user-specific ROC curve. This means that all operating points of an individual ROC curve are associated with a same HMM. Moreover, the user-specific ROC curves are directly averaged, without repairing, by using the standard Ross's method [18] (see the inner graphic in Fig. 10). As expected, the multi-hypothesis system provided a higher ROC curve, and, due the Scott's method [29], a superior number of operating points was obtained.

Depending on the operating point, a same writer could employ a different HMM, as shows Fig. 11. It is worth noting

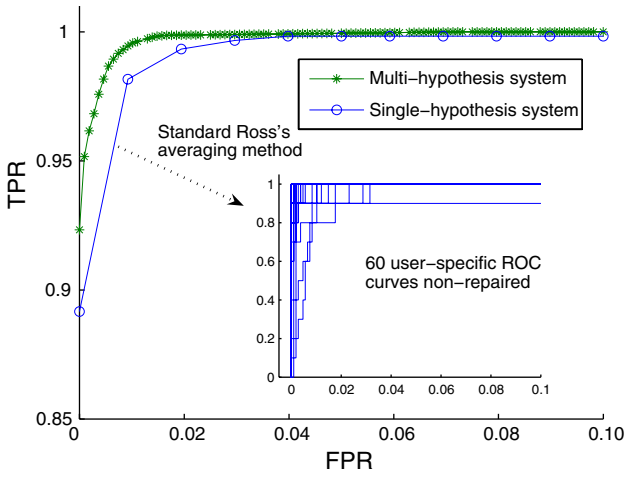

Fig. 10 Averaged ROC curves when applying CB_{35} to DB_{exp}. Before using the Ross's averaging method, the proposed system used the steps of model selection and combination in order to obtain smoother user-specific ROC curves; while the baseline system directly averaged the 60 user-specific ROC curves obtained with the cross-validation process, as shows the *inner figure*

that the complexity of the HMMs increases with the value of γ; which indicates that the operating points in the best region of the ROC space (i.e., the upper-left part) are achieved with a greater number of states. Figure 12 shows the user-specific MRROC curves of two writers. While writer 1 employs different HMMs in the MRROC space, writer 3 uses the same HMM in all operating points. The fact that writer 3 obtained AUC=1 by using an HMM with only 2 states (i.e., HMM_2) indicates that his/her corresponding genuine and forgery spaces in the validation set are easily separable. In other words, high inter-class variability demands less complex models.

In the following phase, the operating points of the averaged ROC space (given by γ) were used to retrieve the user-specific HMMs/thresholds, and apply them to the test set. Table 1 presents the test set error rates for some γ values in both baseline and proposed systems. In order to observe the impact on system performance at each step of the multi-hypothesis approach, results are first shown without the combination step (that is, by performing model selection followed by averaging), while the last results correspond to the whole multi-hypothesis approach. Since there are three types of forgeries in the test set, the average error rate is calculated as

$$\mathrm{AER}(\gamma) = (\mathrm{FNR}(\gamma) + \mathrm{FPR}(\gamma)_{\mathrm{random}} + \mathrm{FPR}(\gamma)_{\mathrm{simple}} + \mathrm{FPR}(\gamma)_{\mathrm{skilled}})/4$$

Which is equivalent to consider equal a priori probabilities, that is, $\mathcal{P}(\mathrm{genuine}) = \mathcal{P}(\mathrm{random}) = \mathcal{P}(\mathrm{simple}) = \mathcal{P}(\mathrm{skilled}) = 0.25$, since the test set of each writer is composed of 10 samples per category of signature.

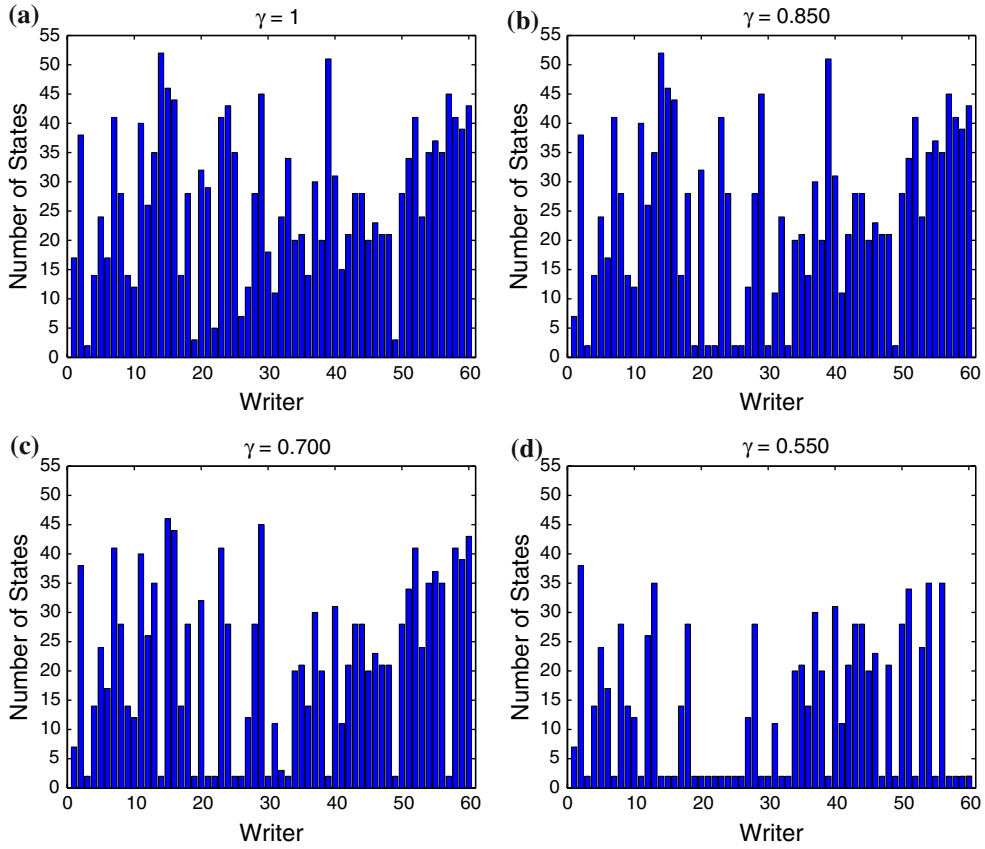

Fig. 11 Number of HMM states used by different operating points in the averaged ROC space. The complexity of the HMMs increases with the value of γ, indicating that the operating points in the upper-left part of the ROC space are harder achieved. In other words, the best operating points are achieved with more number of states

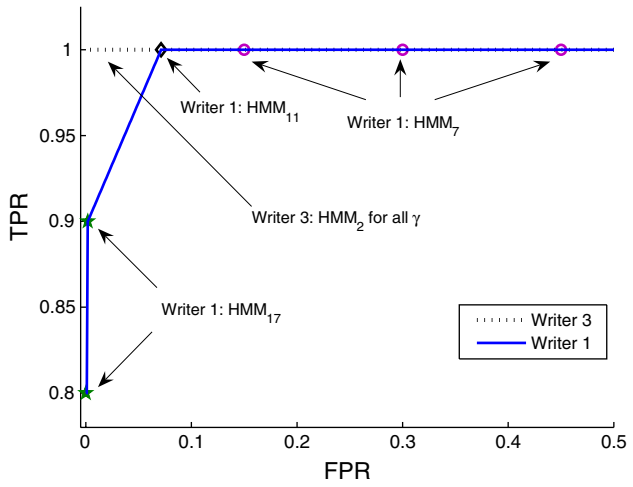

Fig. 12 User-specific MRROC curves of two writers in DB_{exp}. While writer 1 can use HMMs with 7, 11, or 17 states depending on the operating point, writer 3 employs the same HMM all the time. Note that writer 3 obtained a curve with AUC = 1, which indicates a perfect separation between genuine signatures and random forgeries in the validation set

In general, the multi-hypothesis system provided smaller error rates. Moreover, the $\text{FPR}(\gamma)_{\text{random}}$ are closer to the expected error rates given by $1 - \gamma$ (recalling that γ can be viewed as the TNR, and that $\text{FPR} = 1 - \text{TNR}$).

In order to analyze the impact of repairing individual ROC curves, the proposed approach was applied only to the 20 writers having ROC curves with concavities. On average, 75.91% of the problematic writers had their AERs on test enhanced with the multi-hypothesis system in the region between $\gamma = 0.9$ and $\gamma = 1$; while 18.64% performed better with the single-hypothesis system. For the remaining 5.45%, both systems performed equally. Figure 13 presents the results for some γ values, where the improvements obtained with the multi-hypothesis system are located in the positive side, that is, below the dotted line. With the single-hypothesis system, the writer indicated by the arrow in (a) had an AER of 22.5%. When using the multi-hypothesis system, the respective AER was reduced to 4.9%, that is, 17.6% lower.

Finally, the multi-hypothesis system required fewer HMM states than the single-hypothesis system. Figure 14 shows the number of HMM states used by each writer in the

Table 1 Error rates on test

γ	FNR(%)	FPR_{random}(%)	FPR_{simple}(%)	$FPR_{skilled}$(%)	AER(%)
Single-hypothesis system					
0.96	0.50	6.00	10.83	64.83	20.54
0.97	0.83	5.67	9.00	60.17	18.92
0.98	1.17	4.00	5.67	52.50	15.83
0.99	2.33	2.67	4.00	42.67	12.92
1	12.67	0.33	1.17	19.83	8.50
Multi-hypothesis system without combination					
0.96	3.17	5.17	11.00	62.33	20.42
0.97	3.33	4.17	9.33	58.50	18.83
0.98	3.83	1.83	6.67	51.17	15.88
0.99	5.00	1.17	4.17	41.17	12.88
1	12.83	0.33	1.17	20.50	8.71
Multi-hypothesis system					
0.96	4.13	4.67	8.70	54.81	18.07
0.97	4.15	3.17	7.35	52.23	16.72
0.98	4.48	2.50	5.51	47.05	14.88
0.99	5.55	1.17	4.00	39.63	12.58
1	12.83	0	1.17	20.50	8.62

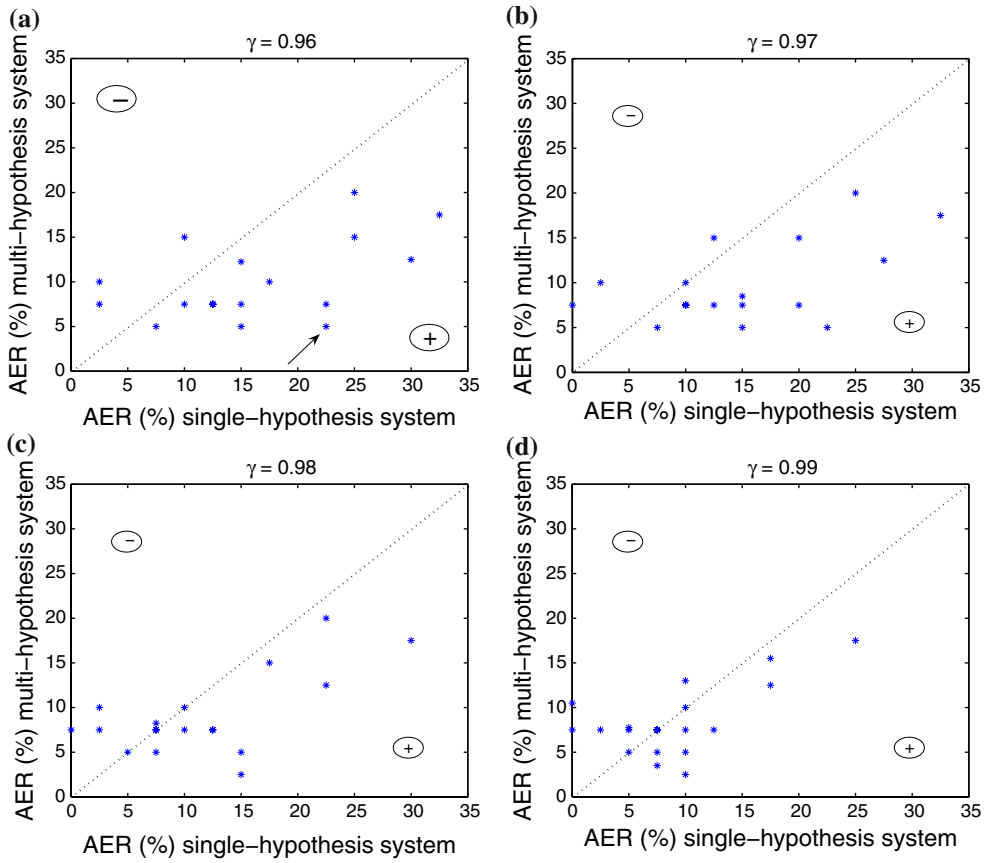

Fig. 13 User-specific AERs obtained on test with the single- and multi-hypothesis systems. The *stars* falling below the *dotted lines* represent the writers who improved their performances with multi-hypothesis system

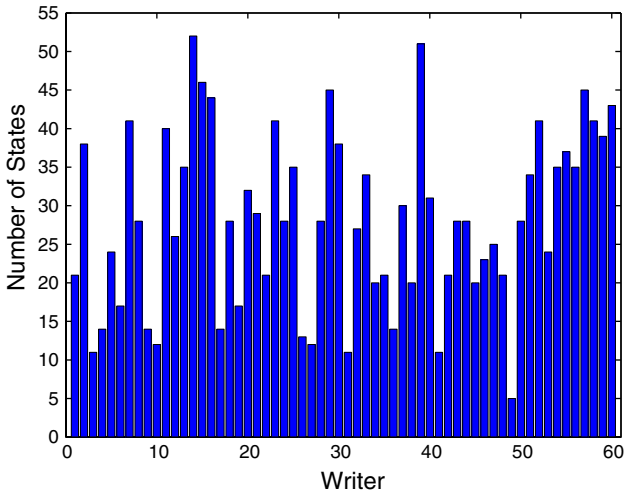

Fig. 14 Number of HMM states selected by the cross-validation process in the single-hypothesis system. These models are used in all operating points of the ROC space

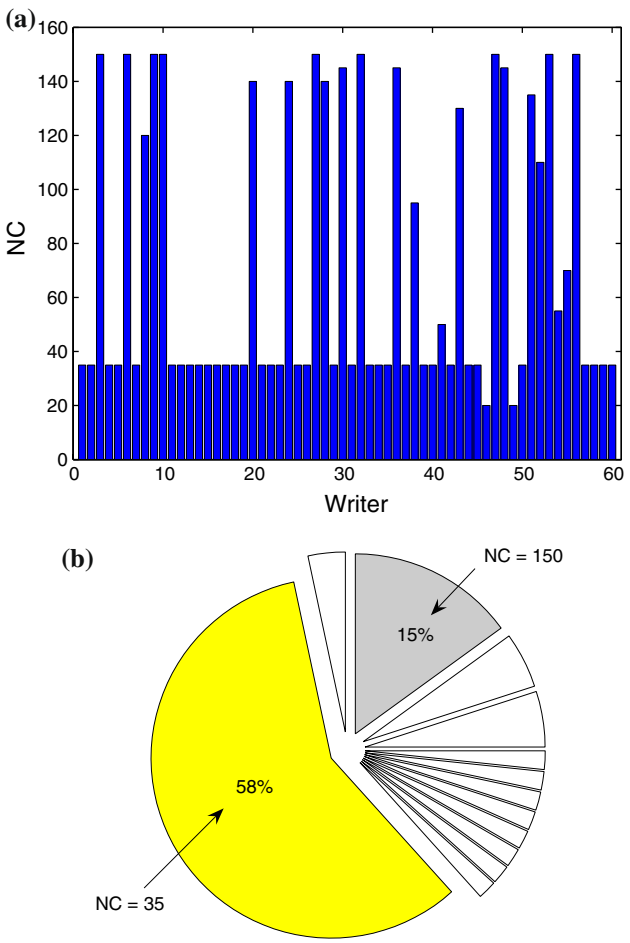

Fig. 15 **a** Distribution of codebooks per writer. In the cases where all codebooks provided AUC = 1, the universal codebook CB_{35} was used. **b** Among the 13 codebooks selected by this experiment, CB_{35} was employed by 58% of the population, while 15% of the writers used a codebook with 150 clusters

single-hypothesis system. Note that these models, applied to all operating points, are more complex than those ones previously shown in Fig. 11. Regarding the entire ROC space, the proposed approach reduced by 48.09% the number of states employed by the HMMs. This occurs because during the model selection step (see Algorithm 2), when two or more HMMs achieve the same TPR, the model with fewer states is chosen.

5.2 Off-line SV based on user-specific codebooks

This experiment explored the idea of using user-specific codebooks in order to reduce individual error rates. Since each writer must be evaluated separately, the selection of the best user-specific codebook was performed after the combination step, by choosing that one providing the MRROC curve with greatest AUC in the region of interest (i.e., between FPR = 0 and FPR = 0.1). Figure 15a presents the codebook selected for each writer. For some writers, due the high variability inter-class on validation (i.e., genuine signatures vs. random forgeries), all codebooks provided AUC = 1. In these cases, the universal codebook CB_{35}, found in the previous experiment, was used. Figure 15b indicates that only 13 codebooks (out of 29) were selected by this experiment; where 58% of the writers used the universal codebook CB_{35}, and 15% used a codebook with 150 clusters.

After selecting the best codebook per user, the user-specific MRROC curves were averaged by using Algorithm 4. The dash-dot line in Fig. 16 shows the resulting averaged ROC curve (AUC = 0.9989), which is better than the curve obtained with the previous version of

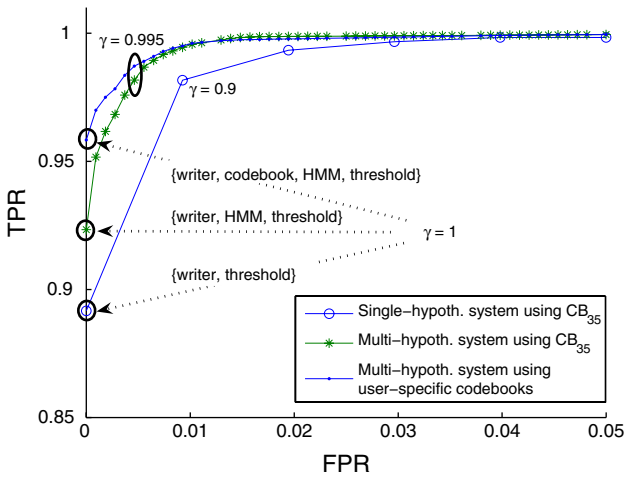

Fig. 16 Averaged ROC curves obtained with three different systems, where the multi-hypothesis system with user-specific codebooks provided the greatest AUC. While the single-hypothesis system stores only the user-specific thresholds in the operating points, the multi-hypothesis systems can store information about user-specific thresholds, classifiers and codebooks in each operating point of the composite ROC space

Table 2 Error rates on test

γ	FNR(%)	FPR_{random}(%)	FPR_{simple}(%)	$FPR_{skilled}$(%)	AER(%)
Results obtained with the multi-hypothesis system when using user-specific codebooks					
0.995	5.93	1.00	3.22	35.10	11.31
0.996	6.60	0.83	2.85	33.15	10.86
0.997	6.87	0.67	2.83	31.82	10.55
0.998	7.73	0.67	2.60	29.22	10.05
0.999	8.63	0.17	1.60	26.00	9.10
1	9.83	0	1.00	20.33	7.79
Results obtained with the multi-hypothesis system when using CB_{35}					
0.995	6.30	0.67	3.17	34.80	11.23
0.996	7.10	0.67	2.67	33.73	11.04
0.997	7.63	0.67	2.67	32.48	10.86
0.998	8.21	0.67	2.67	30.60	10.54
0.999	9.37	0.17	1.91	26.83	9.57
1	12.83	0	1.17	20.50	8.62

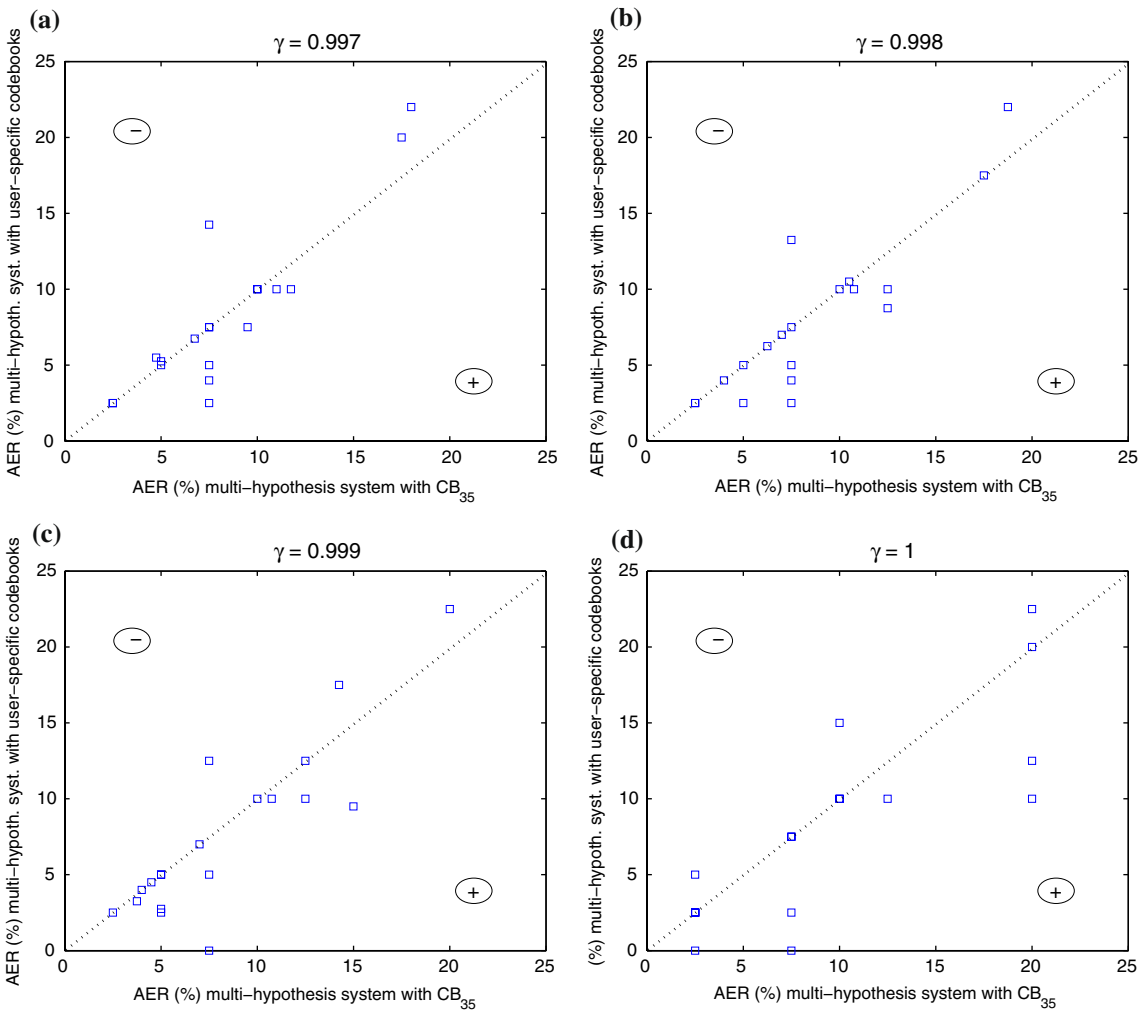

Fig. 17 User-specific AERs obtained on test with two versions of the multi-hypothesis system. The *squares* falling below the *dotted lines* represent the writers who improved their performances with multi-hypothesis system based on user-specific codebooks

multi-hypothesis system (i.e., with CB_{35}) in the region between $\gamma = 0.995$ and $\gamma = 1$. This improvement was also observed on test, as shows Table 2; specially for $\gamma = 1$. The results for $\gamma = 0.96$ to 0.99 are not shown in Table 2 since both systems performed similarly for $\gamma < 0.995$. Figure 17 presents the AERs of the 20 problematic writers obtained with the two multi-hypothesis systems. On average, 36.25% of these writers had their AERs enhanced with the use of user-specific codebooks. This represents 12.08% of the whole population, which may indicate a considerable amount of users in a real world application. Only 16.25% performed better with CB_{35}; and for the remaining 47.50%, both systems performed equally.

Besides being used to retrieve the set of user-specific classifiers and thresholds, the operating points of the last multi-hypothesis system also stores information about what codebook to use; taking into account that a same codebook may be shared by different writers. In the single-hypothesis system, on the other hand, only the user-specific thresholds are stored, since each writer employs his/her single classifier in all operating points. It is worth noting that another variation of the proposed system could be developed by finding the best codebook at each operating point. However, since the 13 codebooks selected by this experiment already provided composite ROC curves with AUC equal or very close to 1, this approach was not investigated.

6 Conclusions

In this article, an approach based on the combination of classifiers in the ROC space was proposed in order to improve performance of off-line SV systems designed from limited and unbalanced data. By training an ensemble of HMMs with different number of states and different codebooks, and then selecting the best model(s) for each operating point, it is possible to construct a composite ROC curve that provides a more accurate estimation of system performance during training and significantly reduces the error rates during operations.

Experiments carried out on the Brazilian SV database [10], with random, simple, and skilled forgeries, show that the multi-hypothesis approach leads to a significant improvement in overall performance, besides reducing the number of HMM states by up to 48%. The results also show that the use of user-specific codebooks can improve class separability. Moreover, the multi-hypothesis system with universal codebook obtained a considerable reduction in the number of codebook clusters and in the error rates[5] when comparing with another off-line SV system developed with the same SV database [21]. Therefore, since ROC concavities are observed, the proposed approach is suitable to improve system performance. Of course, given a set of features able to provide adequate class separation, the multi-hypothesis approach would give no advantage.

The multi-hypothesis approach can be used for dynamic selection of the best classification model—that is, the best codebook, HMM and threshold—according to the risk linked to an input sample. In banking applications, for example, the decision to use a specific operating point may be associated with the amount of the check. In the simplest case, for a user that rarely signs high value checks, big amounts would require operating points related to low FPRs, such as would be provided by a γ close to 1; while lower amounts would require operating points related to low FNRs, since the user would not feel comfortable with frequent rejections.

The proposed approach may require greater computational complexity (training time and memory consumption) than a single-hypothesis approach due to the generation of a set of candidate codebooks and HMMs. However, once the multi-hypothesis ROC space is obtained, all sub-optimal solutions can be discarded, and only the HMMs associated with useful operating points should be stored. During the test phase, no additional time is required, since the process consists in the use of one HMM at a time.

Finally, this strategy can be easily adapted to any neural or statistical classifier designed to solve similar two-class problems. It is useful to note that the choice of classifiers should consider the number of training samples. In the cases where a validation set is not available, the ROC curves may be generated by using the training set.

[5] Error rates reported in [21] with pixel density features: FNR = 2.17%, FPR_{random} = 1.23%, FPR_{simple} = 3.17%, $FPR_{skilled}$ = 36.57%, AER = 10.78%.

References

1. Batista, L., Rivard, D., Sabourin, R., Granger, E., Maupin, P.: State of the art in off-line signature verification. In: Verma, B., Blumenstein, M. (eds.) Pattern Recognition Technologies and Applications: Recent Advances, 1st edn. IGI Global, Hershey (2007)
2. Britto, A.: A two-stage HMM-based method for recognizing handwritten numeral strings. PhD thesis, PUC-PR, Brasil (2001)
3. Coetzer, H., Sabourin, R.: A human-centric off-line signature verification system. In: International Conference on Document Analysis and Recognition (ICDAR 2007), pp. 153–157 (2007)
4. Coetzer, J., Herbst, B., du Preez, A.: Off-line signature verification using the discrete radon transform and a hidden markov model. EURASIP J. Appl. Signal Process. **4**, 559–571 (2004)
5. Drummond, C.: Discriminative vs. generative classifiers for cost sensitive learning. In: Canadian Conference on Artificial Intelligence. Lecture Notes in Artificial Intelligence, pp. 479–490 (2006)
6. El-Yacoubi, A., Justino, E., Sabourin, R., Bortolozzi, F.: Off-line signature verification using hmms and cross-validation. In: IEEE Workshop on Neural Networks for Signal Processing, pp. 859–868 (2000)

7. Fawcett, T.: An introduction to roc analysis. Mach. Learn. **27**, 861–874 (2006)
8. Ferrer, M., Alonso, J., Travieso, C.: Offline geometric parameters for automatic signature verification using fixed-point arithmetic. IEEE Trans. Pattern Anal. Mach. Intell. **27**(6), 993–997 (2005)
9. Flach, P., Wu, S.: Repairing concavities in roc curves. In: Proceedings 2003 UK Workshop on Computational Intelligence, pp. 38–44, August 2003
10. Freitas, C., Morita, M., Oliveira, L., Justino, E., Yacoubi, A., Lethelier, E., Bortolozzi, F., Sabourin, R.: Bases de dados de cheques bancarios brasileiros. In: XXVI Conferencia Latinoamericana de Informatica (2000)
11. Fujisawa, H.: Robustness Design of Industrial Strength Recognition Systems. Springer, New York (2007)
12. Gibbons, J.: Nonparametric Statistical Inference, 2nd edn. M Dekker, New York (1985)
13. Gonzalez, R., Woods, R. (eds.): Digital Image Processing, 2nd edn. Prentice Hall, Upper Saddle River (2002)
14. Gotoh, Y., Hochberg, M., Silverman, H.: Efficient training algorithms for HMMs using incremental estimation. IEEE Trans. Speech Audio Process. **6**(6), 539–548 (1998)
15. Griess, F., Jain, A.: On-line signature verification. Pattern Recognit. **35**, 2963–2972 (2002)
16. Huang, X., Acero, A., Hon, H. (eds.): Spoken Language Processing. Prentice Hall, Upper Saddle River (2001)
17. Impedovo, D., Pirlo, G.: Automatic signature verification: the state of the art. IEEE Trans. Syst. Man Cybern. C **38**(5), 609–635 (2008)
18. Jain, A., Ross, A.: Learning user-specific parameters in a multibiometric system. In: International Conference on Image Processing (ICIP), pp. 57–60 (2002)
19. Justino, E.: O Grafismo e os Modelos Escondidos de Markov na Verificacao Automatica de Assinaturas. PhD thesis, PUC-PR, Brasil (2001)
20. Justino, E., El-Yacoubi, A., Bortolozzi, F., Sabourin, R.: An off-line signature verification system using hmm and graphometric features. In: International Workshop on Document Analysis Systems, pp. 211–222 (2000)
21. Justino, E., Bortolozzi, F., Sabourin, R.: Off-line signature verification using hmm for random, simple and skilled forgeries. In: International Conference on Document Analysis and Recognition, pp. 105–110 (2001)
22. Justino, E., Bortolozzi, F., Sabourin, R.: A comparison of svm and hmm classifiers in the off-line signature verification. Pattern Recogn. Lett. **26**(9), 1377–1385 (2005)
23. Makhoul, J., Roucos, S., Gish, H.: Vector quantization in speech coding. IEEE **73**, 1551–1558 (1985)
24. Murshed, N., Bortolozzi, F., Sabourin, R.: Off-line signature verification using fuzzy artmap neural networks. In: IEEE International Conference on Neural Networks, pp. 2179–2184 (1995)
25. Oliveira, L., Justino, E., Sabourin, R.: Off-line signature verification using writer-independent approach. In: International Joint Conference on Neural Networks (IJCNN), pp. 2539–2544, 2007
26. Plamondon, R.: Progress in Automatic Signature Verification. Word Scientific, Singapore (1994)
27. Rabiner, L.: A tutorial on hidden markov models and selected applications in speech recognition. IEEE **77**(2), 257–286 (1989)
28. Rigoll, G., Kosmala, A.: A systematic comparison between on-line and off-line methods for signature verification with hidden markov models. In: International Conference on Pattern Recognition, vol. 2, pp. 1755–1757 (1998)
29. Scott, M., Niranjan, M., Prager, R.: Realisable Classifiers: Improving Operating Performance on Variable Cost Problems. University of Southampton, UK (1998)
30. Tortorella, F.: A ROC-based reject rule for dichotomizers. Pattern Recogn. Lett. **26**, 167–180 (2005)