Off-line handwritten word recognition using multi-stream hidden Markov models

Yousri Kessentini, Thierry Paquet, Abdelmajid Benhamadou

HAL Id: hal-00668271
https://hal.science/hal-00668271

Off-Line Handwritten Word Recognition Using Multi-Stream Hidden Markov Models

Yousri KESSENTINI[1, 2], Thierry PAQUET[1], AbdelMajid BENHAMADOU[2]

[1] Laboratoire LITIS EA 4108, université de Rouen France

Site du Madrillet 76800 Saint-Etienne du Rouvray, FRANCE.

[2] Laboratoire MIRACL, université de Sfax Tunisie

Route de Tunis km10 - B.P. n° 242 - 3021 Sfax, Tunisie

{yousri.kessentini, thierry.paquet}@univ-rouen.fr

abdelmajid.benhamadou@isimsf.rnu.tn

Abstract

In this paper, we present a multi-stream approach for off-line handwritten word recognition. The proposed approach combines low level feature streams namely, density based features extracted from 2 different sliding windows with different widths, and contour based features extracted from upper and lower contours. The multi-stream paradigm provides an interesting framework for the integration of multiple sources of information and is compared to the standard combination strategies namely fusion of representations and fusion of decisions. We investigate the extension of 2-stream approach to *N* streams *(N=2,…,4)* and analyze the improvement in the recognition performance. The computational cost of this extension is discussed. Significant experiments have been carried out on two publicly available word databases: IFN/ENIT benchmark database (Arabic script) and IRONOFF database (Latin script). The multi-stream framework improves the recognition performance in both cases. Using 2-stream approach, the best recognition performance is 79.8%, in the case of the Arabic script, on a 2100-word lexicon consisting of 946 Tunisian town/village names. In the case of the Latin script, the proposed approach achieves a recognition rate of 89.8 % using a lexicon of 196 words.

Keywords

1. Introduction

Despite the growing use of electronic documents in all the economic activities during the last years, the use of paper documents is still playing an important role. This is because the technologies now offer convenient and cheap means to capture, store, compress and transfer digitized images of paper documents, in a transparent mode. However, automating the processing of the huge amount of these particular documents requires specialized reading systems. While many of such systems are already providing good performance for several applications like OCR, Bank checks Readers, Forms readers, etc. (Srihari 2000)(Knerr et al., 1998)(D'Amato and Kuebert, 2000)(Gorski et al., 1999), the enhancement of performance is still required so as to cope with a wider range of document reading applications.

In today's business world Latin script is mostly used, but with the increasing communication among the different communities worldwide more scripts are getting integrated into information systems. In this context, we propose in this work a unified approach for the recognition of Latin and Arabic scripts. As we want the approach to be script independent, the system must proceed without explicit segmentation of handwriting into graphemes. This is because explicit segmentation methods generally rely on script specific rules to find segmentation points. According to the proposed strategy it is therefore mandatory that the system can operate on low level frame features such as directional or pixel densities. In order to achieve good discriminative power with such low level features, we attempt to combine multiple feature streams. Various combination strategies have been proposed in the literature (Günter and Bunke, 2003). They can be grouped into two broad categories: feature fusion methods and decision fusion techniques.

The first category commonly known as early integration (Okawa et al., 1998), consists in combining the input feature streams by projecting them into a unique feature space, and subsequently use traditional HMM classifier to model the combined observations in the unique feature space (see Figure. 1).
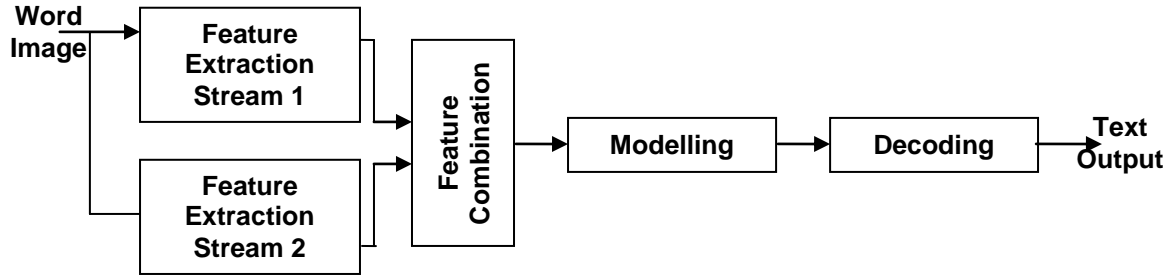
Figure. 1 : Feature combination approach

In contrast, decision fusion, known as late integration (Prevost et al., 2003), consists in combining the single stream classifier outputs (decisions). Different feature representations obtained from the word image are modelled and decoded separately by individual HMM classifiers. The decoded outputs are then combined to get the final text output (see Figure. 2). (Bertolami and Bunke, 2006) compare these two combination methods in the case of offline handwritten text line recognition and shows that both combination methods improve recognition performances compared to any recognisers built from the individual feature streams. Furthermore, in their case, the early integration approach outperforms the decision level combination.
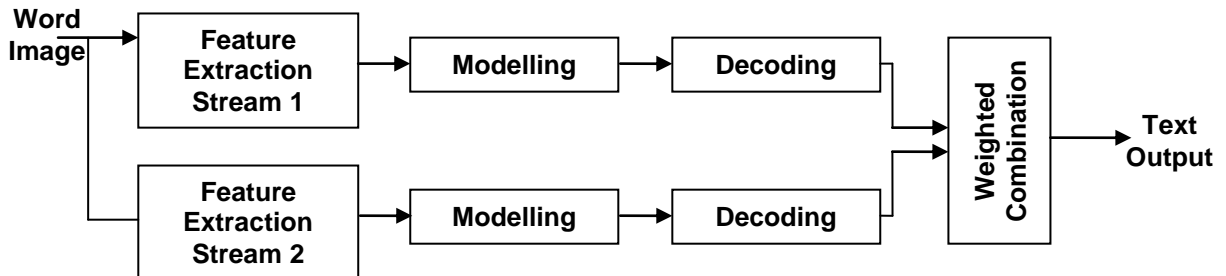
Figure. 2 : Decision fusion approach

A particular method within the decision fusion framework of sequence models falls into the multi-stream hidden Markov model paradigm (see Figure 3). Such an approach has been

particularly studied in the domain of Automatic Speech Recognition (ASR) and presents

multiple advantages (Wellekens et al, 1998) (Bourlard and Dupont, 1997):
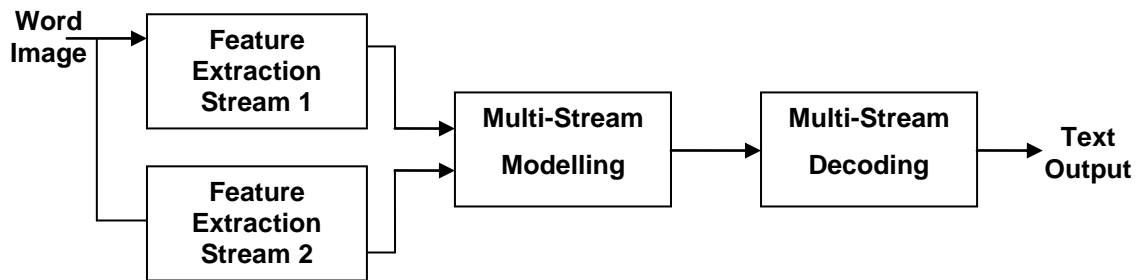
Figure 3: 2-Stream combination approach

- It offers a mean to merge different sources of information such as acoustic and visual inputs in some applications such as audio-visual automatic speech recognition (Dupont and Luettin, 2000) and automatic meeting analysis (McCowan et al. 2005).

- It can combine several kinds of independent features.

- The combination can be adaptive: some sources of information can be weighted, or even rejected if they are not reliable.

- The topology of the HMM can be adapted to each source of information.

- It allows asynchronous modelling of streams.

Despite these many possibilities, multi-stream techniques have not been studied and applied for handwriting recognition. Only the work reported in (Gauthier et al., 2001) and (Artières et al., 2003) have investigated these techniques for on-Line handwriting recognition. The authors investigate the cooperation of on-line and off-line handwriting word recognition and propose a general framework to combine temporal and spatial representation of the signal.

In (Kessentini et al., 2007), we have investigated the use of synchronous multi-stream HMM (without stream asynchrony) for the recognition of Latin script. In this preliminary work, tests were conducted on a private database using two contour feature streams.

In (Kessentini et al., 2008), we extended this preliminary work by enriching the feature streams as described in this paper (see section 4.2.1). Also, asynchronous multi-stream HMM

were tested for the first time for the recognition of Latin and Arabic scripts using a limited lexicon (less than 500 words) of the IFN/ENIT database and the IRONOFF database.

This paper extends these two contributions in various aspects. A comparison of the multi-stream approach to the early and late fusion is conducted. This experiment shows significant improvement of recognition performance for both Latin and Arabic scripts. New experiments have been conducted on the IFN/ENIT database considering the whole lexicon of 946 town names and including the use of set "e" so as to compare with other recently reported works. Finally, this paper explores the extension of the multi-stream paradigm to more than two streams. Experiments conducted on the IFN/ENIT as well as the IRONOFF databases using up to 4 streams show improvement of the recognition performance but with a significant increase of the computational load.

This paper is organised as follows. Section 2 reviews the multi-stream formalism and presents the training and decoding stages. Generalization to the N-streams model and some complexity problems are discussed in section 3. In section 4, the overall system organization is presented, the stages of feature extraction and modelling techniques are illustrated. Experimental results are given in section 5 using the 2-stream approach. They show that the proposed system gives promising results both on Latin and Arabic scripts and compares favourably to other published experimentations on the same data. Additional experiments have been carried out on IRONOFF-cheque and IFN/ENIT databases using the N-stream approach. We discuss the contribution and the computational costs of this extension. Conclusion and future works are addressed in section 6.

2. Multi-stream statistical framework

The multi-stream framework provides a convenient formalism to combine several information sources, namely feature streams, using cooperative Markov models (see Figure 4).

2.1. Formalism

This problem can be formulated as follows: assume an observation sequence X representing a handwritten word to be recognized, is composed of K input streams X_k and assume that the hypothesized model M (e.g. a word model) for a word is the concatenation of J sub-unit models M_j $(j=1,...,J)$ (e.g., grapheme or character models). Each sub-unit model M_j is composed of K models M_j^k (possibly with different topologies) attached to each of the K input streams. While the K stream subunit models are assumed independent of each other, they are forced to recombine at some recombination states ($\otimes$ on Figure 4).
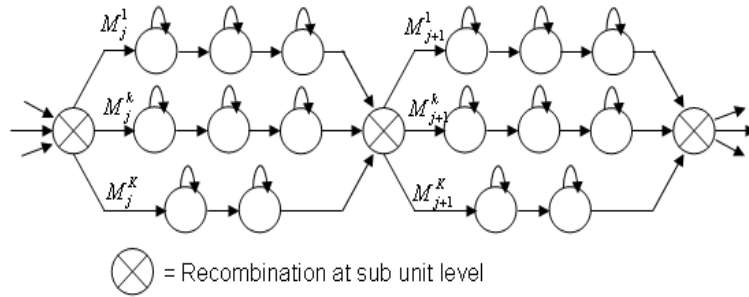
Figure 4: General form of a k-stream model with recombination states between sub-unit models

The recognition problem can be formulated as the one of finding the word model M^* that maximizes the posterior probability given a sequence of observations X:

$$M^* = \underset{M \in \theta}{argmax} \ P(M \ / \ X) \quad (1)$$

where Θ is the set of all possible word hypotheses. Bayes formula gives:

$$M^* = argmax \ \frac{P(X \ /M) \ P(M)}{P(X)} \quad (2)$$

$P(X)$ being independent of the model M it can therefore be ignored for the computation of M^*. When we assume equal prior probabilities $P(M)$ for all possible word hypotheses, then the recognition problem consists in determining the model M^* that maximizes the likelihood $P(X/M)$.

Using subunit decomposition gives:

$$P(X \mid M) = \prod_{j=1}^{J} P(X_j \mid M_j) \qquad (3)$$

Assuming that each stream is independent, each sub-model likelihood $P(X_j \mid M_j)$ can be computed as a weight sum of the K streams likelihood as depicted in equation (4).

$$P(X \mid M) = \prod_{j=1}^{J} \sum_{k=1}^{K} w_j^k P(X_j^k \mid M_j^k) \qquad (4)$$

This equation can be generalized by letting the stream combination be any function of the stream likelihoods and a set of weighting parameters, *W*. So the equation (4) can be rewritten as,

$$log P(X \mid M) = \sum_{j=1}^{J} f(W, P(X_j^k \mid M_j^k)), \ \forall k \qquad (5)$$

Most of the approaches use a linear weighted combination function of log likelihood as follows:

$$\log P(X \mid M) = \sum_{j=1}^{J} \sum_{k=1}^{K} \omega_j^k \log P(X_j^k \mid M_j^k) \qquad (6)$$

In practice, different combination rules have been proposed for multi-stream systems, including linear (sum, product …), non-linear (MLP) or others (maximum, minimum, median…) rules. More details are presented in (Hagen, 2001).

Linear combination of rules requires the estimation of the stream weights according to their relative reliability. Many weighting strategies are proposed in the literature including fixed weights which have to be trained prior to application, and adaptive weights which are estimated during recognition (Hagen, 2001).

After describing the general multi-stream formalism, we present in the next sections how to train and decode such models.

2.2. Multi-stream training

Training the multi-stream HMM consists of two tasks: the first task is the estimation of its HMM stream component parameters (mixture weights, means, variances, and state transition probabilities) and the second task is the estimation of appropriate stream exponents. Maximum likelihood parameter estimation by means of the Expectation Maximization (EM) algorithm can be used in a straightforward manner to train the first set of parameters. This can be done in two ways: either by training each stream component parameter set separately, based on single-stream observations, and subsequently combine the resulting single-stream HMMs, or train the entire parameter set (excluding the exponents) at once using the bimodal observations. In this work, the first method is used because computationally it is less complex to train the single HMMs instead of the product HMM.

In order to model the Latin characters, we built 26 uppercase character models and 26 lowercase character models). In the case of Arabic characters, we built up to 159 character models. An Arabic character may actually have different shapes according to its position within the word (beginning, middle, end word position). Other models are specified with additional marks such as "shadda". In both Latin and Arabic script, each character model is composed of 4 emitting states. The observation probabilities are modelled with Gaussian Mixtures (3 per state).

Embedded training (Rabiner et al., 1982) is used where all character models are trained in parallel using Baum-Welch algorithm applied on word examples. The system builds a word HMM by concatenation of the character HMM corresponding to the word transcription of the training sample.

The final training step concerns the optimization of the stream weights. It is shown in (Potamianos and Graf, 1998) that the maximum-likelihood estimation of the weights fails. Indeed, maximising the likelihood with respect to the weighting factor ω yields to the

selection of a stream with the highest likelihood (hence $\omega = 0$ or $\omega = 1$). The authors also show that additional constraints on the weight can yield to a satisfactory solution. Two different methods have been investigated in this work and will be detailed below. The stream combination weights are estimated using two different strategies.

- **Equal combination weights**

This strategy consists in attributing equal weights to the various streams. Its main advantage is that no data is needed for estimation of the weights and no extra time for the calculation of the weights has to be expended.

- Relative frequency weights (Hagen, 2001)

Employing the forced segmentation given by the multi-stream decoding algorithm, the ratio between the number of times an expert (ie. a stream) performs best for a given character, and the number of times this character occurs in the database is computed.

$$\omega_j^k = \frac{n_{k,j}}{n_j}$$

where $n_{k,j}$ is the number of training frames for which expert k has the largest probability, over all experts, for character j, and n_j is the number of frames for character j in the training data.

In this work we noticed that both of the weighting strategies perform similarly.

2.3. Multi-stream decoding

During recognition the best word model M^* that maximizes $P(X / M)$ has to be determined. Two solutions have been investigated in the literature:

■ Recombination at the HMM state level: Although it does not allow for asynchrony or different topologies of the stream models, it is pretty simple to implement and amounts to perform a standard Viterbi decoding (Forney, 1973) in which local probabilities are obtained from a linear or nonlinear combination of the local stream emission probabilities.

- Recombination at the sub-unit level: it can force the streams to be synchronous where synchrony is required at the end of character models and can allow for asynchrony where asynchrony allows the sub unit states to be independent of each other. It requires a more sophisticated decoding procedure than the Viterbi search. Two different algorithms have been proposed to solve the problem of decoding in this case:

  - Two level dynamic programming (Sakoe, 1979): Here the decoding takes place in two steps. A first dynamic programming process is applied at the sub-unit level and each sub-model is then scored against arbitrary portions of the frame data. Secondly, sub-models are merged together in order to find the best overall score.

  - HMM-recombination (Bourlard and Dupont, 1997): It is an adaptation of the HMM decomposition algorithm (Varga and Moore, 1990). The HMM decomposition algorithm is a time-synchronous Viterbi search that allows the decomposition of a single stream (speech signal) into two independent components (typically speech and noise). In the same spirit, a similar algorithm can be used to combine multiple inputs stream into a single HMM model.

It was shown in (Dupont 2000) that both algorithms are equivalent. In this work, we choose to use the HMM-recombination algorithm which is described below. With this choice, a simple pre-processing step that consists in building the product HMM is introduced prior to using a classical Viterbi decoding algorithm. Using the two-level dynamic programming approach would require much more development.

2.4. HMM-recombination algorithm

The principle of the multi-stream HMM is to model independently each stream between two pre-determined synchronization points, using multiple single-stream HMMs (two in this

study). In this study, the synchronization states are the character boundaries (see Figure 5). Decoding based on this integration method would require the computation of the best state sequence for both streams verifying the synchronous recombination rule at the same time. To avoid the computation of two best state paths, the model can be formulated as a composite or product HMM (see Figure 6) where each state is built by merging a K-tuple of states from the K stream HMMs (here, *K=2*). The topology of this composite model is defined so as to represent all possible state paths given the initial HMM topologies. Decoding under such a model requires computing a single best path using the well known Viterbi decoding algorithm.
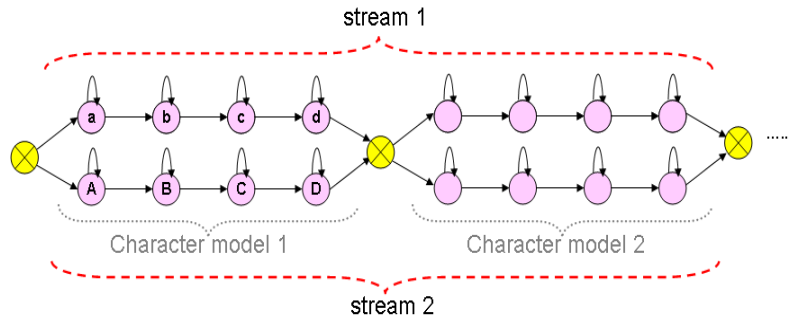
Figure 5 : Example of a multi-stream HMM with 2 streams and 4 states in each character model.
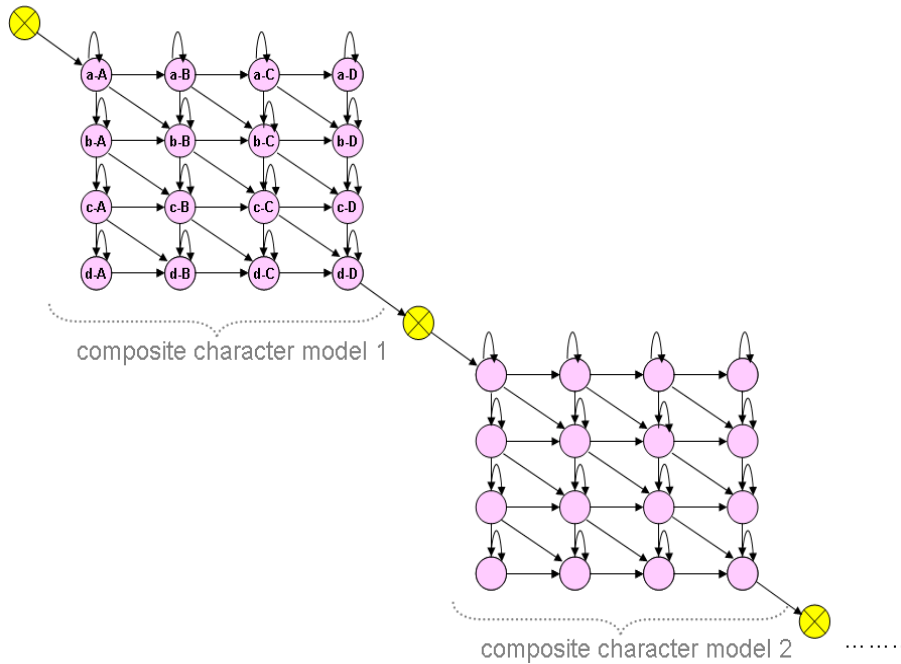
Figure 6 : The corresponding product (composite) HMM.

11

In this model, the observation log-likelihood conditioned on a composite state a-A, that is composed of the stream states a and A respectively, is given using a weighted sum of log-likelihood combination function by:

$$log\ P(X_1(t), X_2(t)\ /\ a\text{-}A) = \omega\ log\ P(X_1(t)\ |\ a) + (1\text{-}\omega)\ log\ P(X_2(t)\ |\ A)$$

where $X_1(t)\ (similarly,\ X_2(t))$ is the observation vector corresponding to stream 1 (similarly, stream 2) and ω the reliability of stream 1 ($0 \le \omega \le 1$).

The transition probabilities of the product HMM are derived from the transition probabilities of the 2 single stream HMMs assuming independence of the models between 2 recombination states. For example,

$$P(a\text{-}B\ /\ a\text{-}A) = P_1(a\ /\ a) \times P_2(B\ /\ A)$$

2.5. Generalization to more than 2 streams

Combining more than two streams is a trivial extension of the approach. The product HMM can easily be computing by combining N single-stream HMMs as follows. Let $S^i = \{s^i_j;\ j = 1,...n_i\}$ be the set of states of the single-stream model i. Then the product HMM is defined in the product state space denoted $S^{\otimes}$ as follows:

$$S^{\otimes} = S^1 \otimes S^2 ... \otimes S^N = \left\{ s^{\otimes}_i = \left(s^1_j, s^2_k, ..., s^N_l \right);\ j = 1,...n_1, k = 1,...n_2, l = 1,...n_N \right\}$$

In this model, the observation log-likelihood conditioned on a composite state $S^{\otimes}_i$ is computed using a weighted sum of the single-stream log-likelihood by:

$$log\ P\left(X_1(t), X_2(t),..., X_N(t)\ /\ S^{\otimes}_i \right) = \sum_{\alpha=1}^{N} \omega^{\alpha}_i\ log\ P\left(X_\alpha(t)\ /\ f[\alpha]\left(S^{\otimes}_i\right) \right)$$

where f is a function that identifies the state in the αth stream HMM that accounts to product state $S^{\otimes}_i$ as follows: $f[\alpha]\left(S^{\otimes}_i\right) = s^{\alpha}_k$

The transition probabilities of the product HMM are derived from the transition probabilities of the N single stream HMMs assuming conditional independence of the N stream state sets:

$$P\left(S_i^{\otimes} / S_j^{\otimes}\right) = \prod_{\alpha=1}^{N} P\left(f\left[\alpha\right]\left(S_i^{\otimes}\right) / f\left[\alpha\right]\left(S_j^{\otimes}\right)\right)$$

The number of states per character in the composite HMM is equal to the product of the number states per single-stream HMM character. For instance, a three stream character HMMs, each made of 4 states, would result in a $4 \times 4 \times 4 = 64$ states composite HMM etc…

As we can see, complexity is a major concern when dealing with multiple streams. A straightforward solution can be envisaged by reducing the number of states per character model. This possibility has been investigated in the following experiments (see section 5.4). Some other possibilities for complexity reduction have been left for further developments and will be discussed in see section 5.4.

2.6. Discussion

Now some comments on using the multi-stream framework as opposed to the classical single stream approach can be given. First, the multi-stream framework gives the possibility to combine feature streams asynchronously. This means that some features detected at the same location in the image can be desynchronised during decoding. Such possibility is impossible with standard single-stream HMM and is especially suited when dealing with 2D features as is the case in this study dedicated to Off-Line recognition. Second, because a multi-stream HMM can be implemented with the product HMM of the N stream models, the formalism is applicable to any kind of topology, ranging from left/right models to fully connected ergodic models. In addition, we can notice that the formalism can easily be extended to variable duration state models straightforwardly (Rabiner, 1989). Indeed, assuming independence of

13

each stream model, the duration probability of the product state $S_i^\otimes$ is the product of the

duration probability of each state that account for $S_i^\otimes$ as follows:

$$p_{S_i^\otimes}(d) = \prod_{\alpha=1}^{N} p_{f[\alpha](S_i^\otimes)}(d)$$

The product HMM framework also allows the multi-stream model to be adapted to a new

dataset in a straightforward way by using standard adaptation approaches. Notice however

that adaptation of a product HMM is computationally demanding due to its complexity

especially with a large number of streams. This particular aspect of multi-stream adaptation

would require more research efforts to receive efficient low complexity implementations.

3. A script-independent recognition system

Earlier there have been a few efforts behind designing multi-script recognition systems

(Malaviya and Leja, 1996) (Chaudhuri and Pal, 1997). However, these systems do not use

HMMs, the framework that have lately been successfully applied to develop many

handwriting recognition systems. An HMM based framework offers several advantages

allowing us to propose a unified script-independent recognition system. These advantages are

mainly due to automatic training of character models on non-segmented words (embedded

training), and the segmentation-free recognition paradigm that fits particularly well to a

script-independent approach. The proposed system (see Figure 7) is based on a multi-stream

HMM.

In the following sub-section, we describe the different stages of our approach. In the first step,

pre-processing is applied to the word image. Two types of features are considered in this

work: (i) contour based features and (ii) density based features. Contour based features are

extracted from the lower and the upper contours, and density based features are computed on

two different sliding windows with varying width (see Figure 8). Therefore, each feature type

14

(contour or density feature) defines two feature streams representing the input word image. Each stream model is then separately trained using Baum Welch algorithm. It can be noted that for assigning weights to the multi-streams, both the equal weight and relative frequency strategies (refer section 2.2) perform similarly. The results reported in section-5 are based on using equal weight strategy. The last step is the recognition during which the HMM models are simultaneously decoded according to the multi-stream formalism presented above.
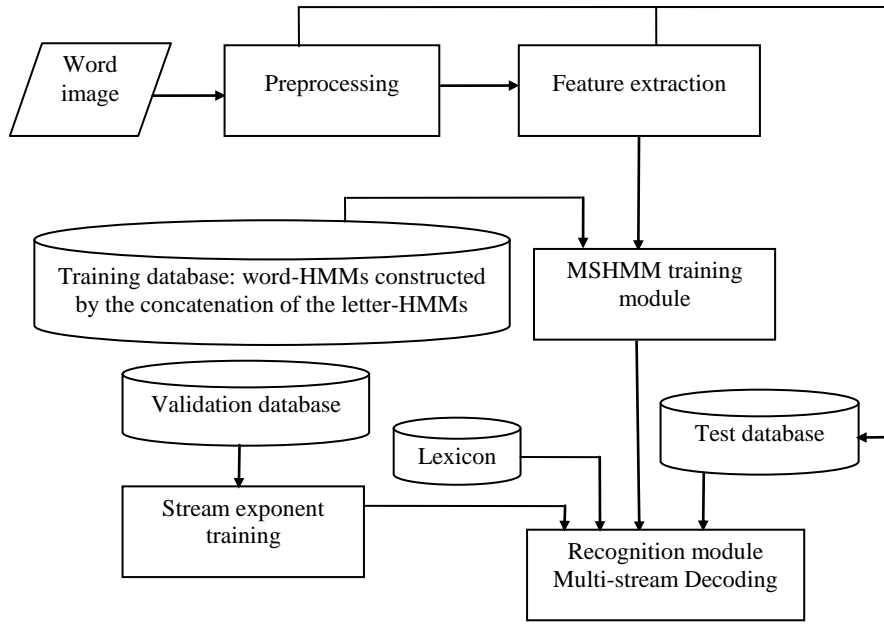
Figure 7. Methodology for the 2-stream training and decoding

3.1. Pre-processing

Pre-processing is applied to word images in order to eliminate noise and to simplify the procedure of feature extraction. It is worth noticing that these pre-processing methods are script independent.

- Normalization: In an ideal model of handwriting, a word is supposed to be written horizontally and with ascenders and descenders aligned along the vertical direction. In real data, such conditions are rarely respected. We use slant and slope correction so as to normalize the word image (Kimura et al., 1994).

- Contour smoothing: Smoothing eliminates small blobs on the contour.

- Base line detection: Our approach uses the algorithm described in (Vinciarelli et al., 2004) based on the horizontal projection curve that is computed with respect to the horizontal pixel density (see Figure 8). Baseline position is used to extract baseline dependent features that emphasize the presence of descenders and ascenders.

3.2. Features extraction

An important task in multi-stream combination is to identify features that carry complementary information. In order to build the feature vector sequence, the image is divided into vertical overlapping windows or frames. The sliding window is shifted along the word image from right to left (in case of Arabic words) or left to right (in case of Latin words) and a feature vector is computed for each frame.
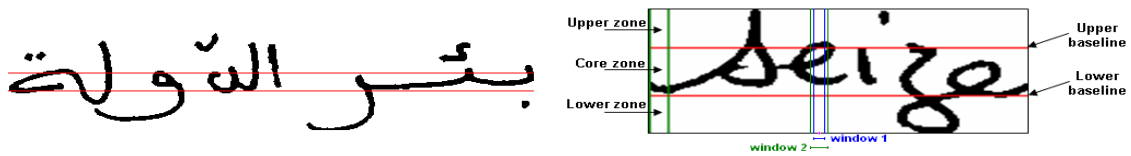
Figure 8. Upper and lower baselines detection

Two feature sets are proposed in this work. The first one is based on directional density features. This kind of features has proved to be discriminative for off-Line handwriting recognition, especially for Latin script (Kimura et al., 1994). The second one is based on foreground (black) pixel densities and has been tested only on Arabic script (El-Hajj et al., 2005). In this study, we will thus evaluate the discriminative power of these features using two different scripts (Latin and Arabic scripts).

3.2.1. Contour features

These features are extracted from the word contour representation. Each word image is represented by its lower and upper contours (see Figure 9). A sliding window is shifted along the word image, two parameters characterize a window: window width (8 pixels) and window

16

overlap between two successive positions (5 pixels). For each position of a window, we extract the upper contour points (similarly, the lower contour points). For every point in this window, we determine the corresponding Freeman direction (Freeman, 1961) and the directions points are accumulated in the directional histogram (8 features).
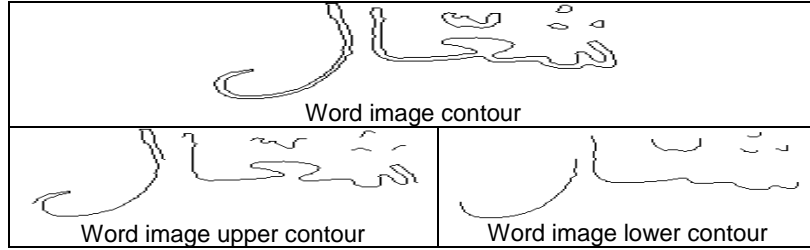
Figure 9. Word image contours

In addition to the directional density features, a second feature set is computed at every point of the upper contour (similarly, it is done for every points on lower contour). The last (black) point (say, p') in the vertical black run started at an upper contour point (say, p) is considered and depending on the location of p', one of the four situations may arise. The point (p') can belong to a:

- Lower contour (see corresponding p points as marked red in Figure 10).

- Interior contour on closure (see blue points in Figure 10).

- Upper contour (see yellow points in Figure 10).

- No point found (see green points in Figure 10).

The black points in Figure 10 represent the lower contour.

Figure 10: Contour feature extraction

The histogram of the four kinds of points is computed in each window. This second feature set provides additional information about structure of the contour like the loops, the turning

points (for instance, one in letter 't'), the simple lines, and the end points on the word image (altogether, four different features).

The third feature set indicates the position of the upper contour (similarly, lower contour) points in the window. For this purpose, we localize the core zone of the word image. More precisely, we extract the lower and upper baselines of word images. These baselines divide the image into 3 zones: 1) a middle zone, 2) the lower zone, 3) the upper zone. This feature set (3 features) provides additional information about the ascending and the descending characters, which are salient characteristics for recognition of the Latin script, as well as of the Arabic script. Hence, in each window we generate a 15-dimensional (8 features from chain code, 4 features from the structure of the contour and 3 features from the position of the contour) contour (for upper or lower contour) based feature vector.

3.2.2. Density features

Here we recall the definition proposed in (El-Hajj et al., 2005). From each frame 26 features are extracted for window of 8-pixel width (and 32 features for window of 14-pixel width). There are two types of features: features based on foreground (black) pixel densities, and features based on concavity. In order to compute some of these features (for example, f_2 and f_{15} as described next) the window is divided into cells where the cell height is fixed (4 pixels in our experiments) see Figure 11.

Let H be the height of the frame in an image, h be the fixed height of a cell, w the width of a frame (see figure 11). The number of cells in a frame n_c is equal to $n_c=H/h$. Let $n_t(i)$ the number of foreground pixels in cell i in frame t, and $b_t(i)$ the density level of cell i in frame t, then:

$$b_t(i)=0 \text{ if } n_t(i)=0 \text{ else } b_t(i)=1$$

Let LB be the position of the lower baseline, UB be the position of the upper baseline. For each frame t, the features are the following:

Stream direction (right to left)
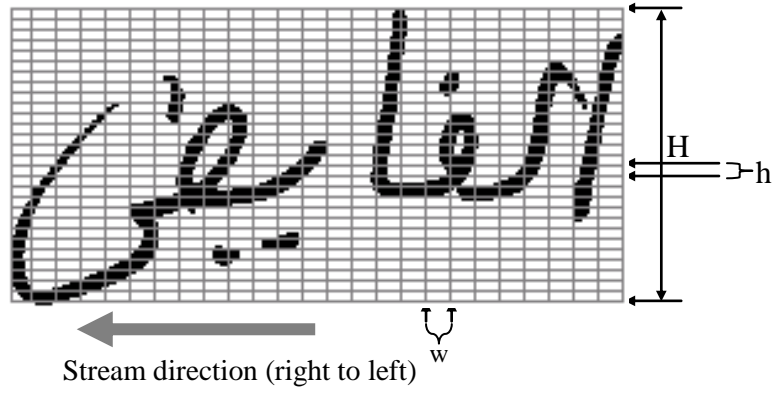
Figure 11. Word image divided into vertical frames (here without overlap)

- f_1: density of foreground (black) pixels.

$$f_1 = \sum_{i=1}^{n_c} n_t(i)$$

- f_2: number of transitions between two consecutive cells of different density levels.

$$f_2 = \sum_{i=2}^{n_c} \left| b_t(i) - b_t(i-1) \right|$$

- f_3: difference in y position of gravity centers of foreground pixels in the current frame and in the previous one.

$$f_3 = g(t) - g(t-1)$$

where g is computed as $g(t) = \dfrac{\sum\limits_{j=1}^{H} j \cdot r_t(j)}{\sum\limits_{j=1}^{H} r_t(j)}$ where $r_t(j)$ denotes the number of foreground pixels

in the j-th row in the frame t,

- f_4 to f_{11}: densities of black pixels for each vertical column of pixels in each frame (note that the frames here are of 8-pixel width).

- f_{12}: vertical position of the center of gravity of the foreground pixels in the whole frame with respect to the lower baseline. The result is then normalized by the height H of the frame.

19

$$f_{12} = \frac{g(t) - LB}{H}$$

- f_{13}-f_{14}: density of foreground pixels over and under the lower baselines for each frame.

$$f_{13} = \frac{\sum\limits_{j=LB+1}^{H} r_t(j)}{H.w} , \quad f_{14} = \frac{\sum\limits_{j=1}^{LB-1} r_t(j)}{H.w}$$

- f_{15}: number of transitions between two consecutive cells of different density levels above the lower baseline.

$$f_{15} = \sum\limits_{i=k}^{n_c} \left| b_t(i) - b_t(i-1) \right|$$

where k is the cell that contains the lower baseline.

- f_{16}: zone to which the gravity center of black pixels belongs with respect to the upper and lower baselines (above upper baseline, a middle zone, and below lower baseline).

- f_{17} to f_{26}: five concavity features in each frame and another five concavity features in the core zone of a word, that is, the zone bounded by the upper and lower baselines. They are extracted by using a 3×3 grid as shown in Figure 12.

Left-Up   Up-Right   Right-Down   Down- Left   Vertical
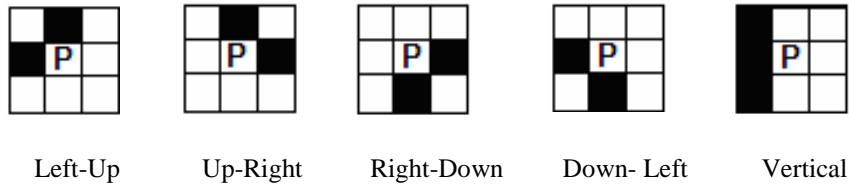
Figure 12. Five types of concavity configurations for a background pixel P

The density feature set has been chosen in order to capture the presence of ascenders, descenders and dots in the word image. Concavity features are added to reflect local concavity and stroke directions. Although this set of features has been originally used for Arabic script, we will see that it can be useful for the recognition of Latin script.

4. Experiments and Results

The proposed recognition system is script independent. It proceeds without explicit segmentation of handwriting into graphemes. This is because explicit segmentation methods generally rely on script specific rules to find segmentation points. According to the proposed strategy it is therefore mandatory that the system can operate on low level frame features such as directional, contour or pixel densities. In order to achieve good discriminative power of such low level features, the proposed approach is based on the multi-stream paradigm that provides a interesting way of combining individual feature streams.

To evaluate the performance of our recognition system, experiments have been conducted on two publicly available databases: IFN/ENIT benchmark database of Arabic words and IRONOFF database for Latin words (French and English). In all experiments, word-level recognition accuracies have been computed.

4.1. IFN/ENIT database

The IFN/ENIT (Pechwitz et al., 2002) contains a total of 32,492 handwritten words of 946 Tunisian town/villages names written by 411 different writers. Some town/village names occur in the database with slightly different writing style according to the presence or absence of "shadda" for example. It follows that our lexicon is made of about 2,100 valid entries. Four different sets (a, b, c, d) are predefined in the database for training and one set (e) for testing.

Table1 shows the experimental results of the performance of our recognition system using 4 different single streams (upper contour, lower contour and density with two windows varying in their widths; Density1 and Density2 correspond to the windows of 8-pixel and 14-pixel widths, respectively) as a function of the size of the list of word hypothesis. The best recognition rate is 70.5 % obtained using upper contour feature. From these results it appears that the upper contour is significantly better than the three other feature stream for the recognition of Arabic script.

Models	TOP			
	1	2	5	10
1) U. contour	70.5	78.6	86.3	90.4
2) L. contour	63.5	73.1	82.6	86.4
3) Density1	65.1	73	80.6	83.2
4) Density2	68.7	78.1	83.3	86.9

To improve the performance given in Table 1, we try to combine the 4 single streams according to the multi-stream formalism. Six possible pairs of streams can be formed. The recognition results of the 2-streams HMM are presented in Figure 13.
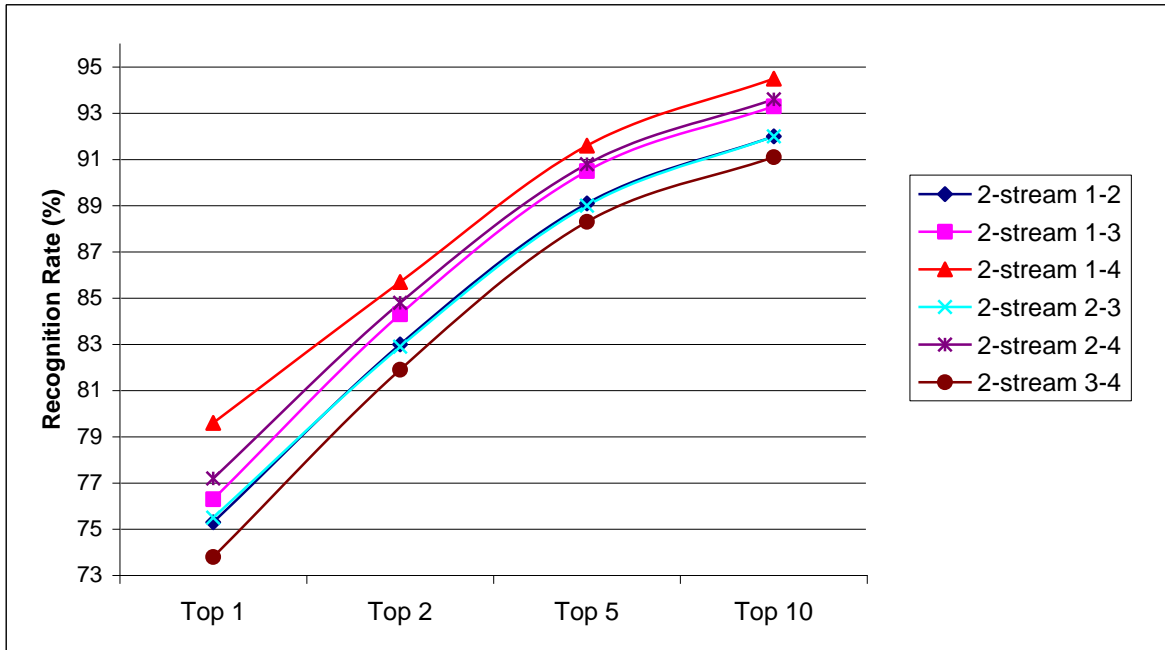
Figure 13: The IFN/ENIT 2-stream recognition performances

In all these experiments, we notice that the multi-stream approach improves the performance obtained with any of the single stream HMM. The best 2-stream recognition rate is 79.6% in Top 1 and is obtained by combining upper contour and density2 features. The gain is 9.1%

compared to the best single stream recognition rate. Also combining density and contour feature streams performs better than combining 2 contour streams or 2 density streams.

To compare the multi-stream approach to the standard combination strategies namely fusion of features and fusion of decisions, we report the best 2-stream result obtained by combining the features corresponding to the upper contour and the Density2. As shown in Table 2, the multi-stream approach performs better than the two others standard combination strategies of each individual stream model.

Table 2. Multi-stream results vs. decision and feature fusion approaches by combining upper contour and density2 features

Models	TOP			
	1	2	5	10
2-stream	**79.6**	**85.7**	**91.6**	**94.5**
Decision Fusion	75.4	83.2	89.5	92.2
Feature Fusion	74.1	82.6	88.4	90.8

Notice that the late decision fusion is performed here using equal weights of the two stream likelihoods using the same combination operator as in the multi-stream approach.

In order to compare our results to the works presented in the literature, we report on table 3 the results obtained on the same database (learning sets: a, b, c, d; test set: e) during the international competition in Arabic handwriting recognition systems at ICDAR 2005 (Märgner et al., 2005). In this table we have reported the performance of the best proposed multi-stream system which shows a gain of 3.67% for the Top 1 solution compared to the winner system of the ICDAR'05 competition.

Table 3: Comparison with other word recognition systems which are presented in (Mätgner et al., 2005): recognition results in % with the IFN/ENIT dataset e (6033 images)

System	Top 1	Top 5	Top 10
ICRA	65.74	83.95	87.75
SHOCRAN	35.70	51.62	51.62
TH-OCR	29.62	43.96	50.14
UOB	75.93	87.99	90.88
REAM	15.36	18.52	19.86
ARAB-IFN	74.69	87.07	89.77
Proposed system	**79.6**	**91.6**	**94.5**

These results should also be compared to those obtained during the last ICDAR'07 contest (Märgner and El Abed, 2007). We report in table 4 the results obtained during this competition using a learning sets "a, b, c, d, and e" and sets "f" and "s" for testing. Our system shows the best performance on set "s" and the third best performance on set "f".

Table 4. Recognition results in % of correct recognized images on references d, e (Märgner and El Abed, 2007). (ID 01: MITRE; IDs 02-04: CACI; ID 05: CEDAR; ID 06: MIE; IDs 07-08: SIEMENS; IDs 09-12: UOB-ENST; ID 13: ICRA; ID 14: PARIS V)

| ID	set f	set s
01	61.70	49.91
02	11.95	8.01
03	15.79	14.24
04	14.28	10.68
05	59.01	41.32
06	**83.34**	68.40
07	82.77	68.09
08	**87.22**	73.94
09	79.10	64.97
10	81.65	69.61
11	81.93	69.93
12	81.81	70.57
13	81.47	72.22
14	80.18	64.38
Proposed system	**82.09**	**74.51**

4.2. IRONOFF database

The IRONOFF database (Viard-Gaudin et al., 1999) contains a total of 31,346 isolated word images from a 196-word lexicon (French and English). Although the database contains both on-Line and off-line information of the handwriting signals, only the off-line information is used for our experiments. The offline handwriting signals are sampled with spatial resolution of 300 dots per inch (DPI), with 8 bits per pixel (256 gray levels). For the experiments reported in this paper, we have used a training set of 20,898 words and a test set of 10,448 words.

Table 5 reports the performance of our recognition system using 4 different single streams presented earlier in section 4.1 (upper contour, lower contour and density with two window width). One can notice that the lower contour stream performs significantly poorer than the other three features streams, which perform similarly. This can be explained by the fact that lower contours in Latin words are not generally very informative. Compared to the single stream performance on Arabic script where the upper contour performs significantly better than the other three streams, no single stream feature significantly outperforms the others streams.

Table 5. IRONOFF recognition performance using single stream features

Models	TOP			
	1	2	5	10
1) U. contour	81.2	86.8	91.4	94.6
2) L. contour	69.8	78.4	88.6	93.4
3) Density1	81.8	87.6	92.6	95.6
4) Density2	80	86.2	92.8	95.8

Similar to the experiment conducted on the IFN/ENIT database, we have conducted several experiments using the 6 possible 2-stream combinations of HMMs. The results are presented in Figure 14.
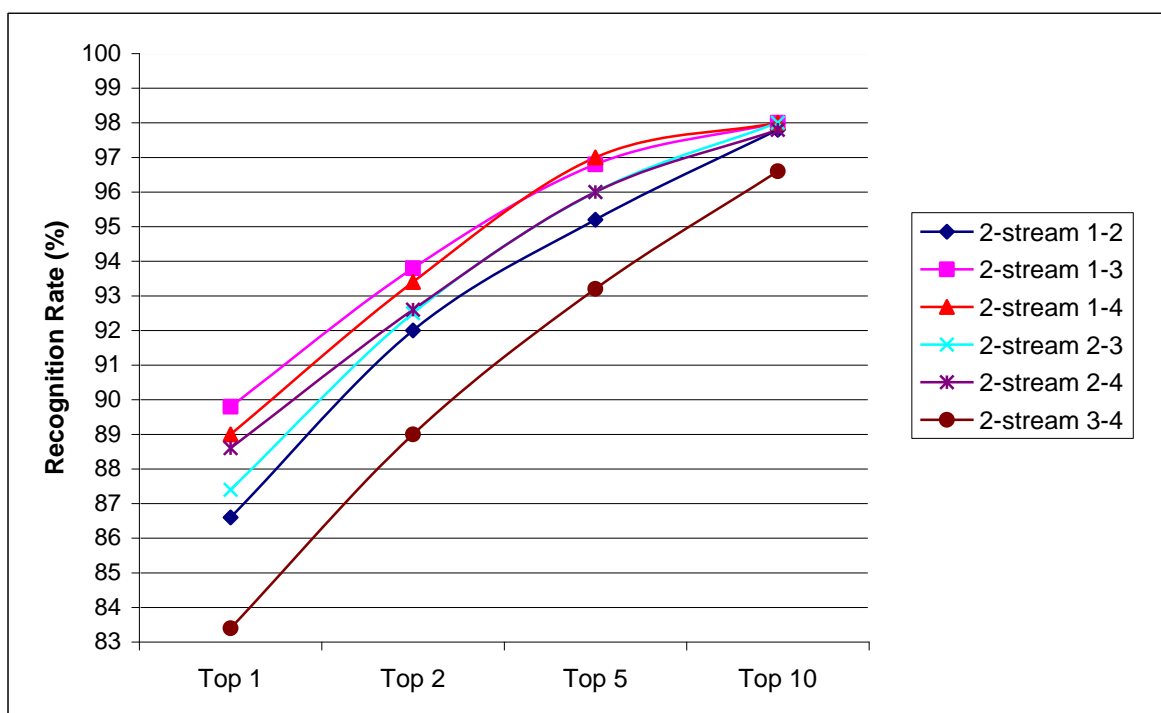
Figure 14 : The IRONOFF 2-stream performances

As already observed on IFN/ENIT database experiments, we once again notice that the multi-stream combination approach outperforms the performance obtained with the single stream HMM. The best 2-stream recognition rate is 89.8 % by combining upper contour and density1 features. The gain is 8 % compared to the best single stream recognition rate. Similar to what has been observed on the Arabic script, we notice that the best two stream features is also the combination of complementary feature streams (upper contour and density features).

In order to compare the multi-stream approach with the standard combination strategies, we report in Table 6 the best 2-stream result obtained by combining upper contour and Density1 single streams. We notice that the multi-stream approach performs better than the standard combination approaches. The gain is 3.8 % in Top 1.

Table 6. Multi-stream results vs. decision and feature fusion approaches by combining upper contour and density1 features

Models	TOP			
	1	**2**	**5**	**10**
2-stream	**89.8**	**93.8**	**96.8**	**98**
Decision Fusion	86	91.4	95	97.4
Feature Fusion	85.2	90.6	94.4	97

To compare our results to the related works, we report in table 7 the results obtained on the same database. We notice that our system provides good results on Latin script by achieving the second best performance reported on this database.

Table 7: IRONOFF: Comparison with other word recognition systems

Authors	Performances (%)	
	TOP 1	**TOP 5**
(Tay et al., 2001) system 1	86.6	94.2
(Viard-Gaudin et al., 2005) system 1	87.4	95.8
(Viard-Gaudin et al., 2005) system 2	89.8	97
(Tay et al., 2001) system 2	**96.1**	**99.1**
Proposed system	89.8	96.8

During these experiments, the multi-stream based system have shown significant recognition results for both Latin and Arabic scripts. The comparison of the multi-stream performance to the classical combination strategies namely, fusion of features and fusion of decisions shows the superiority of the multi-stream approach. We investigate in the next section how the multi-stream approach performs using more than two streams.

4.3 Results with N-streams

The previous results have demonstrated the interest of using a multi-stream approach for the recognition of both Arabic and Latin scripts. Following this direction, we are now interested in assessing the superiority of this framework when using more than two streams.

As already stated in section 2.5, extension to N-streams is straightforward but significantly increases the computational cost. In fact, the complexity of the Viterbi algorithm is $O(T.(C.N)^2)$. N being the number of states per character model, C is the number of characters in the model and T the number of distinct observation frames per state. In the product HMM, the number of states increase to $C.N^S$ with S being the number of streams. Therefore, the complexity of the Viterbi algorithm becomes $O(T(C.N^S)^2)= O(T.C^2.N^{2S})$. As a consequence, the complexity exponentially increases with the number of streams. This is actually a severe limitation and this is why we have experimented this framework using small lexicons. We have conducted some experiments on IRONOFF-cheque database, a subset of the IRONOFF database, which is made of only French cheque words (30 word lexicons). Table 8 shows the 2-stream, 3-stream and 4-stream recognition results. We notice that adding one feature stream generally improves the performance especially when the single stream HMMs are less accurate. Nevertheless it considerably slows down the decoding process. During these experimentations we have also investigated the influence of reducing the number of states per character on the recognition performance. The use of 3 states per character model (3s/c) speeds up the decoding procedure but slightly decreases the recognition results as shown in table 8.

Table 8: IRONOFF-Cheque performances

Models	4 s/c		3 s/c	
	Top1	Top5	Top1	Top5
2-stream 1)2)	86	95.4	87	97.6
2-stream 1)3)	92.8	99.2	91	99.4
2-stream 2)3)	94	99.6	89.4	97.8
2-stream 3)4)	91	99	86.2	97.2
3-stream1)2)3)	94	99.6	91.6	**99.6**
3-stream1)3)4)	92.6	99.2	90.8	98.8
3-stream2)3)4)	93.4	99	89.4	98.4
3-stream1)2)4)	93.6	**99.8**	92	99
4-stream	**94.2**	99.2	**92.8**	99.2

N-stream performances on the IFN/ENIT (testing on set d, learning on sets a, b, c) are given in Table 9 using 2 lexicon sizes and 4 states per character model. We use a random selection of dictionaries for each word test sample. Once again we observe a slight increase of the recognition performance by increasing the number of streams but for a significant increase of the complexity of the decoding process. More investigations should be conducted so as to reduce the complexity of the N-stream decoding process. This could be studied by introducing some of the known techniques that have been already proposed for large lexicon decoding (Koerich et al., 2003). Various strategies should be tested such as beam search decoding, pruning, A* search etc… together with the use of a lexical tree search.

Table 9: IFN/ENIT performances on set d)

Models	Lexicon size			
	30		100	
	Top1	Top5	Top1	Top5
1) U. contour	97.8	99.8	95.4	99
2) L. contour	97.8	99.8	93.4	99.2
3) Density1	92	97.4	88.4	94
4) Density2	92.6	98	88.8	94.4
2-stream 1)2)	98	99.8	95.6	99.8
2-stream 1)3)	98.6	99.8	98.4	99.4
2-stream 1)4)	98.7	100	98.6	99.8
2-stream 2)3)	98	100	96.8	99.8
2-stream 3)4)	97.6	99.6	96.4	98.6
3-stream1)2)3)	98.6	99.8	98.4	99.8
4-stream	**99.2**	**100**	**98.8**	**100**

It should be noted that due to complexity reasons, the N-stream experiments have been carried out on limited lexicons. Choosing a particular limited lexicon may biased the results. For this reason, the random selection has been chosen on the IFN/ENIT database. On the contrary, because there already exists a small bank-cheque lexicon on the IRONOFF database, random election of dictionaries has not been used. This is why the experimental settings on the IRONOFF database are probably biased by this bank-cheque lexicon. This

may explain why the best 2-stream results are obtained in this case using a different stream combination (stream 2-3) than those obtained in the previous experiment when using the whole IRONOFF lexicon (stream 1-3). This phenomenon is not observed on the N-stream IFN/ENIT experiment, probably due to the random selection of the lexicon that makes the performance more comparable to those obtained in the previous experiments with the whole IFN/ENIT lexicon.

As a conclusion, it must be stated that the main purpose of these experiments was to establish the contribution of additional streams to the recognition performances. Regarding this investigation, the experimental results show in both cases (IRONOFF and IFN/ENIT) the same moderate improvement of recognition performance when using additional streams.

5. Conclusion and Perspectives

This paper presents a multi-stream HMM-based approach for off-line handwritten word recognition. The proposed system is script independent. It proceeds without explicit segmentation of handwriting into graphemes and makes use of low level feature sets (directional features and colour density features) irrespective of the scripts. Features are then combined according to the multi-stream paradigm. The developed system has been tested on two publicly available databases: the benchmark database IFN/ENIT (for Arabic script) and IRONOFF database (for Latin script). For both scripts the results show significant improvement while using a multi-stream approach. The comparison of the multi-stream performances to the classical combination strategies namely, fusion of features and fusion of decisions shows the superiority of the multi-stream approach. Moreover, the proposed recognition system provides significant results comparable to the best results reported in the literature on both databases. Future works will consist of testing a new combination rules especially the non-linear ones and a joint training method using a composite HMM. We will

be also focused on proposing a unified multi-script recognition system operating independently of the script nature.

Exploring the Transferable Belief Model (TBM) (Smets and Kennes, 1994) to improve the combination rules and test a credal HMM (Ramasso et al., 2007) which combines the generality of TBM and mechanisms of HMM are also one of the interesting futures works. We are also interested in combining other kinds of features providing more complementary information to further improve the results using a N-streams approach. Feature stream selection should also be addressed using this kind of multi-stream framework.

References

A. Hagen, Robust speech recognition based on multi-stream processing, PHD thesis, Ecole polytechnique fédérale de Lausanne, 2001.

A. L. Koerich, R. Sabourin, C.Y. Suen, "Lexicon-driven HMM decoding for large vocabulary handwriting recognition with multiple character models", International Journal on Document Analysis and Recognition, IJDAR ,Vol. 6, N°.2, pp.126-144, October 2003.

A. Malaviya, C. Leja, L. Peters, Multi-script handwriting recognition with FOHDEL. New Frontiers in Fuzzy Logic and Soft Computing Biennial Conference of the North American Fuzzy Information Processing Society- NAFIPS. IEEE, Piscataway, NJ, USA, pp. 147-151, 1996.

A. P. Varga and R. K. Moore, Hidden Markov model decomposition of speech and noise. Proc. IEEE Internat. Conf. Acoust. Speech and Signal Process, pp 845-848, 1990.

A. Vinciarelli, S.Bengio, H.Bunke, Offline Recognition of Unconstrained Handwritten Texts Using HMMs and Statistical Language Models, IEEE Transactions on PAMI, vol. 26, No 6, pp. 709-720, 2004.

A. Vinciarelli. A survey on off-line Cursive Word Recognition. Pattern Recognition, vol. 35, num. 07, p. 1433-1446, 2002.

B. B. Chaudhuri, U. Pal, An OCR system to read two Indian language scripts: Bangla and Devnagari (Hindi), ICDAR '97, vol 2, pp. 1011 -1015, 1997.

C. J. Wellekens, J. Kangasharju, C. Milesi, The use of meta-HMM in multistream HMM training for automatic speech recognition, Proc. of Intl. Conference on Spoken Language Processing (Sydney), pp. 2991-2994, 1998.

C. Viard-Gaudin , P. Lallican , S. Knerr, Recognition-directed recovering of temporal information from handwriting images, Pattern Recognition Letters, v.26 n.16, p.2537-2548, December 2005.

C. Viard-Gaudin, P.M. Lallican, S. Knerr, P. Binter, The IRESTE On/Off (IRONOFF) Dual Handwriting Database, ICDAR'99, pp 455 – 458, 1999.

D. D'Amato, E. Kuebert, A. Lawson, Results from a performance evaluation of handwritten address recognition systems for the United States Postal Service, in: IWFHR, Amsterdam, pp.189–198, 2000.

E. Ramasso, M. Rombaut & D. Pellerin. Forward-Backward-Viterbi procedures in the Transferable Belief Model for state sequence analysis using belief functions. European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty, pp 405 – 417, 2007.

F. Kimura, S. Tsuruoka, Y. Miyake & M. Shridhar, A Lexicon Directed Algorithm for Recognition of Un-constrained Handwritten Words. IEICE Trans. Inf. &Syst., vol. E77-D, no. 7, 1994.

G. Forney, "The Viterbi algorithm", Proc. IEEE 61 (3), pp. 268–278, 1973.

G. Nagy, S. Seth, and X. Zhang, Multi-Character Field Recognition for Arabic and Chinese Handwriting, Proceedings of the Summit on Arabic and Chinese Handwriting Recognition, College Park, MD, pp. 93-100, 2006.

G. Potamianos, H.P Graf. Discriminative training of HMM stream exponents for audio-visual speech recognition. Proceedings IEEE Conference Acoustics, Speech, and Signal Processing. pp. 3733-3736 vol.6, Seattle, WA, 1998.

H. Bourlard, S.Dupont. Sub-band-based Speech Recognition. In IEEE Int. Conf. on Acoust., Speech, and Signal Processing, pp. 1251-1254, 1997.

H. Freeman. On the encoding of arbitrary Geometric Configurations. IRE Transaction on ElectronicComputers, 10 :260-268, 1961.

H. Sakoe, Two-level DP matching - A dynamic programming-based pattern matching algorithm for connected word recognition, IEEE Transactions of the IECE of Japan, vol 27, pp 588- 595, 1979.

I. McCowan , D. Gatica-Perez , S. Bengio , G. Lathoud , M. Barnard , D. Zhang, Automatic Analysis of Multimodal Group Actions in Meetings, IEEE Transactions on Pattern Analysis and Machine Intelligence, v.27 n.3, p.305-317, 2005.

J. J. Lee, M. Nakajima and J. Kim, A Hierarchical HMM Network-based Approach for On-Line Recognition of Multi-Lingual Cursive Handwritings, IEICE Transactions on Information and Systems, vol. E81-D, No. 8, pp. 881-888, 1998.

L. Prevost, C. Michel-Sendis, A. Moises, L. Oudot, and M. Milgram, Combining model-based and discriminative classifiers: application to hadwritten character recognition. In 7th ICDAR'03, vol 1, pages 31-35, 2003.

L. R. Rabiner, A. Bergh and J. R. Wilpon. An Embedded Word Training Procedure for Connected Digit Recognition, Conference Record 1982 International Conference on Acoustics, Speech, and Signal Processing, pp. 1621-1624, May 1982.

L. R. Rabiner, A tutorial on Hidden Markov Model and selected applications in speech recognition, IEEE proceedings, Vol. 77, pp. 257-286, 1989.

M. Pechwitz, S. Maddouri, V. Maegner, N. Ellouze, IFN/ENIT–DataBase for Handwritten Arabic words, CIFED'02, pp. 129-136, 2002.

M. Schambach, J.Rottland, T. Alary, How to Convert a Latin Handwriting Recognition System to Arabic, ICFHR'08. vol ,pp ,2008.

N. Gauthier, T. Artieres, B. Dorizzi, and P. Gallinari, "Strategies for combining on-Line and off-line information in an on-Line handwriting recognition system". ICDAR'01, pp 412-416, 2001.

N. Gorski, V. Anisimov, E. Augustin, O. Baret, D. Price, J. Simon, A2iA check reader: a family of bank check recognition systems, in Proceedings of ICDAR, Vol. 1, Bangalore, pp. 523–526, 1999.

N. S. Srihari: Handwritten Address Interpretation: A Task of Many Pattern Recognition Problems. IJPRAI, vol 14, n° 5, pp 663-674, 2000.

Ph. Smets and R. Kennes. The Transferable Belief Model. Artificial Intelligence, vol 66, n° 2 pp 191–234, 1994.

R. Bertolami, H. Bunke: Early feature stream integration versus decision level combination in a multiple classifier system for text line recognition. ICPR (2): 845-848, 2006.

R. El-Hajj, L. Likforman-Sulem, C. Mokbel, Arabic handwriting recognition using baseline dependent features and Hidden Markov Modeling, ICDAR'05, Seoul, South korea, vol 2, pp. 893- 897, 2005.

S. Dupont and J. Luettin. Audio-Visual Speech Modelling for Continuous Speech Recognition. IEEE Transactions on Multimedia, vol 2 (3) : 141-151, 2000.

S. Dupont. Etude et développement d'architectures multi-bandes et multi-modales pour la reconnaissance robuste de la parole. PhD thesis, Faculté Polytechnique de Mons, Juin 2000.

S. Günter, H. Bunke. Ensembles of classifiers for handwritten word recognition, Int. Journal on Document Analysis and Recognition, Vol. 5, No. 4, 224–232, 2003.

S. Knerr, E. Augustin, O. Baret and D. Price: Hidden Markov Model Based Word Recognition and Its Application to Legal Amount Reading on French Checks. Computer Vision and Image Understanding 70(3): 404-419 , 1998.

S. Okawa, E. Bocchieri, and A. Potamianos. Multi-band speech recognition in noisy environments. In Proceedings of IEEE International Conference on Acoustic, Speech, and Signal Processing, pages 641.644, Seattle, Washington, 1998.

T. Artières, N.Gauthier, P.Gallinari, B.Dorizzi, "A Hidden Markov Models combination framework for handwriting recognition", International Journal on Document Anlysis and Recognition (IJDAR), vol 5, N° 4 pp 233-243 / July, 2003.

V. Märgner, H. El Abed: Arabic Handwriting Recognition Competition. ICDAR'07, vol. 2 pp. 1274-1278, 2007.

V. Märgner, M. Pechwitz, H. El Abed. Arabic handwriting recognition competition. In: Proceedings of the ICDAR'05 (1), pp. 70-74, 2005.

Y. H.Tay, P.M.Lallican, M.Khalid, C.Viard-Gaudin and S.Knerr, An Offline Cursive Handwritten Word Recognition System, IEEE Region 10 International Conference on Electrical and Electronic Technology (TENCON), 2001. vol.2 pp. 519 - 524.

Y. Kessentini, T. Paquet, A. Benhamadou, A Multi-Stream HMM-based Approach for Off-line Multi-Script Handwritten Word Recognition. ICFHR'08, pp ,2008.

Y. Kessentini, T. Paquet, A. Benhamadou. A Multi-stream Approach to Off-Line Handwritten Word Recognition, ICDAR'2007, vol. 1, pp. 317-321, 2007.