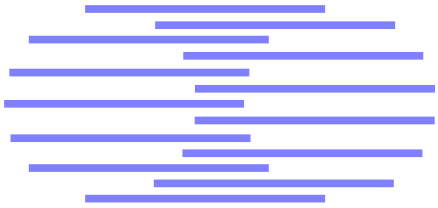
IDIAP
Martigny - Valais - Suisse



Offline Cursive Word Recognition using Continuous Density Hidden Markov Models trained with PCA or ICA Features

Alessandro Vinciarelli [a] Samy Bengio [a]

IDIAP–RR 01-46

December 2001

submitted for publication

Dalle Molle Institute for Perceptual Artificial Intelligence • P.O.Box 592 • Martigny • Valais • Switzerland

phone +41 – 27 – 721 77 11
fax +41 – 27 – 721 77 12
e-mail secretariat@idiap.ch
internet http://www.idiap.ch

[a] Institut Dalle Molle d'Intelligence Artificielle Perceptive, CP592 - Rue du Simplon 4, 1920 Martigny (Switzerland)

IDIAP Research Report 01-46

Offline Cursive Word Recognition using Continuous Density Hidden Markov Models trained with PCA or ICA Features

Alessandro Vinciarelli Samy Bengio

December 2001

submitted for publication

Abstract. This work presents an Offline Cursive Word Recognition System dealing with single writer samples. The system is a continuous density hiddden Markov model trained using either the raw data, or data transformed using Principal Component Analysis or Independent Component Analysis. Both techniques significantly improved the recognition rate of the system.
Preprocessing, normalization and feature extraction are described in detail as well as the training technique adopted. Several experiments were performed using a publicly available database. The accuracy obtained is the highest presented in the literature over the same data.

1 Introduction

This work presents a system for offline recognition of single writer cursive words. The main approaches to the problem of Cursive Word Recognition (CWR) were developed using data produced by many persons [1, 2] and only recently some efforts were led towards the recognition of single writer samples [3, 4, 5]. This can be useful in tasks such as the transcriptions of forms filled by a single person or the automatic processing of personal notes.
Our approach is based on Continuous Density Hidden Markov Models. They are used to model the sequences of vectors obtained using a sliding window shifting column by column across the word images. The use of the sliding window allows one to avoid an independent segmentation which is a difficult and error prone process.
In all of the processing steps, the use of heuristics adapted to the specific handwriting style of the data was carefully avoided. This makes the system robust with respect to a change of writer.
We also investigated the use of Principal Component Analysis (PCA) and Independent Component Analysis (ICA) applied to the raw feature before feeding them to the HMM. Both techniques were able to improve the performance of the baseline system obtained without applying them.
The experiments were performed over a publicly available database used in several other works presented in the literature. The recognition rate we achieved is, to our knowledge, the highest among those obtained over the same data and in the same experimental conditions.
This paper is organized as follows: section 2 gives a general description of the system, section 3 shows in detail preprocessing and normalization, section 4 presents the feature extraction process, section 5 is dedicated to the Hidden Markov Models and section 6 introduces PCA and ICA. The final sections 7 and 8 show respectively the results we obtained and the conclusions we drew.

2 The Recognition System

The recognition system presented in this work is based on a sliding window approach. The general structure of the system is in figure 1.
The word is first preprocessed and normalized. At this stage, the aim is the removal of all the elements in the word image that are not relevant to the recognition. The preprocessing tries typically to eliminate backgrounds, rulers, or other elements that can be present in the word image, but do not give any useful information. The normalization eliminates the effects of the acquisition and of the handwriting style.
A sliding window shifts column by column from left to right across the resulting image. At each window position, a frame is isolated and a feature vector is extracted from it. This approach has the important advantage of avoiding the segmentation, a difficult and error prone process involving heuristics that often depend on the data.
The feature extraction is based on the distribution of the foreground pixels across the window. The features are robust with respect to the high variability of the patterns blindly isolated by the window.
The recognition is performed using Continuous Density Hidden Markov Models (HMMs). A model is available for each word in the lexicon and the word corresponding to the most likely model is selected as transcription of the handwritten data.
In order for HMMs to obtain better performance, further preprocessing using either Principal Component Analysis or Independent Component Analysis is proposed. Although very simple, the system is shown to have a good performance over data written by a single person. In the next sections, each single step of the processing is explained in more detail.

3 Preprocessing and Normalization

The preprocessing module takes as input the raw data and gives as output images containing the words to be recognized without any other element irrelevant to the recognition process. The tasks

performed at this level depend on the data: when the document from which the word is extracted presents a background pattern, the latter should be removed. In some other cases, the words are extracted from forms showing boxes and lines that should be eliminated. For our data, a binarization (performed with the Otsu algorithm [6]) is sufficient.
Once the image is preprocessed, it must be normalized. The normalization is supposed to eliminate some of the variability due to acquisition and handwriting style. Even if the system deals with single writer data, the removal of writer dependent features is an advantage because it allows one to change writer without changing the system. Without the normalization, different heuristics should be developed for each handwriting style resulting in a scarce flexibility with respect to a change of writer.
The normalization is achieved by removing *slant* and *slope* of the words. The first one is the angle between the vertical direction and the direction of the strokes supposed to be vertical in an ideal model of the handwriting. The second one is the angle between the horizontal direction and the direction of the line on which the word is implicitly aligned. The angles can change significantly even in samples produced by the same writer.
The normalization technique we used (for a full description see [7]) is completely adaptive and makes no use of parameters that must be set empirically. This is an important advantage because it avoids heavy experimental efforts to optimally tune the parameter set (operation that should be repeated for each writer). Moreover, the accuracy of the system is shown to be significantly improved [7] when using such technique rather than the widely applied normalization procedure described in [8].
In the next two subsections, the slope and slant removal techniques are described in more detail.

3.1 The Slope Removal Technique

The slope is removed by rotating the image so that the line on which the word is implicitly aligned (the so-called *baseline*) is horizontal. The baseline can be obtained by fitting the lower extrema of the character bodies. In order to identify such points, it is necessary to give a first rough estimate of the *core region*, i.e. the region enclosing the character bodies. The stroke extrema closest to the lower limit of the estimated core region can be used to fit the baseline. Once this is known, the word can be rotated until the baseline is horizontal.
The first estimate of the core region is then a fundamental step and the method we used to obtain it is based on the vertical projection of the word, the histogram of the values of the horizontal densities $H(i)$ (number of foreground pixels in row i. See figure 2 on the right side of the word image).
By using the histogram H it is possible to estimate the probability $p(h)$ of a line showing horizontal density h (see figure 2, lower plot). The probability distribution is obtained by simply counting the frequency of each value h in H. (The values of h range from 0 to the number of columns in the image). The number of lines available for the estimate is low (the average is around 150), but the experiments show that it is enough for slope removal purposes. The set $\pi = \{p(h) : h = 1, 2, \ldots, nCol\}$ so obtained is the probability density function of the horizontal density values.
The probability distribution (see lower plot in figure 2) is expected to be bimodal because, given a line, there are two possibilities: it belongs to the core region (expected to be dense) or it belongs to the other regions (expected to be sparse). The Otsu algorithm, based on the hypothesis that the two modes correspond to two Gaussian distributions, can estimate a density threshold allowing the distinction between lines of the core region and lines of the other regions.
The main advantage of the method is that local features, such as long horizontal strokes, that can be confused with the core region, have a negligible influence on the distribution. This allows a robust estimate of the threshold, as well as the core region.
Once the core region is estimated, the lower extrema of the character bodies can be found and used to estimate the baseline. Once this is obtained, the image is rotated until the line is horizontal and the word is desloped.

3.2 The Slant Removal Technique

Commonly applied deslanting techniques rely on the selection of long near vertical strokes. Their direction is used to estimate the slant, usually by averaging over the directions measured on all the selected strokes.
The selection of the strokes to be used in the slant estimation is a difficult task, often based on heuristics that are data dependent. Moreover, when few slant values are measured, the influence of a single measure, obtained from an erroneously selected stroke, can have a strong influence on the overall measure.
In the approach proposed here and in [7], we use a global measure of the *deslantedness* of the image that allows one to avoid the search for single strokes. A shear transform corresponding to each angle in a reasonable interval is performed over the original word image. For each shear transformed image, the value of the deslantedness is measured. The shear transformed image giving the highest deslantedness value is assumed to be the deslanted one (see figure 3).
The deslantedness measure is obtained as follows: first the horizontal projection of the word, i.e. the histogram V collecting the vertical densities $V(i)$ (number of foreground pixels in column i) is obtained. If $\Delta y(i)$ is the distance between the highest and the lowest foreground pixel in column i, then the following histogram can be obtained:

$$H_\alpha(i) = \frac{V(i)}{\Delta y(i)} \tag{1}$$

where α is the angle corresponding to the shear transform of the image being measured and i ranges from 1 to the number of columns $nCol$ of the image. The value of $H_\alpha(i)$ is 1 when the column contains a continuous stroke, and is between 0 and 1 in the other cases.
We hypothesize that, when the word is deslanted, the number of columns containing a continuous stroke is maximum, hence we use as a deslantedness measure the function:

$$S(\alpha) = \sum_{i:H_\alpha(i)=1}^{N} V(i)^2. \tag{2}$$

The value α^* of the slant is estimated as:

$$\alpha^* = \arg\max_\alpha S(\alpha). \tag{3}$$

The main disadvantage of this algorithm is that many shear transformed images must be computed and this requires a considerable computational effort.
This algorithm was compared, in terms of recognition rate, to the widely applied method presented in [8]. The results, presented in [7], show that the new deslanting algorithm improves the recognition rate by 3.6% over single writer data (the same used in the experiments described in this paper) and by 10.8% over multiple writer data (a collection of ~ 12000 words written by 200 different persons).

4 Feature Extraction

In most systems, the word image is segmented into small parts (called *primitives* or *fragments*) supposed to be the basic information units. A feature extraction process is applied to each fragment and, depending on the approach, the word is recognized with HMMs or Dynamic Programming.
The extraction of the fragments is referred to as *segmentation* and it is a difficult and error prone process. Many heuristics are needed and, in the case of single writer word recognition, ad hoc algorithms can be necessary, making the system not flexible with respect to a change of writer.
In order to avoid such problems, we applied a sliding window approach. A window shifts column by column from left to right and at each position isolates a frame. From each frame a feature vector is extracted.

The window is 16 pixels wide. At the resolution of our images (300 dpi) this corresponds more or less to 1.5 mm, a value comparable to the width of the strokes produced by a common pen. The feature extraction process performs a partition of the frame into cells regularly arranged in a 4×4 grid. In each cell i, the number n_i of foreground pixels is counted. The feature vector is then obtained as follows:

$$\mathbf{f} = (\frac{n_1}{N}, \frac{n_2}{N}, \ldots, \frac{n_{16}}{N}) \tag{4}$$

where $N = \sum_i n_i$ is the total number of foreground pixels in the frame.
Such a feature set, based on local averaging rather than on exact reconstruction of the pattern, is robust with respect to the high variability of the patterns contained in the frames [9, 10].
Long horizontal strokes belonging to ascenders and descenders often affect several frames containing letters (or parts of them) different than the one they belong to. This results in a noise that is eliminated through a simple heuristic: given a frame, the pixels that are in the ascender or descender region, but are not connected to the content of the core region, are eliminated. The feature extraction process is applied after the removal of such pixels.

5 Training and Recognition

During recognition, a matching score between the handwritten data and every entry of the lexicon is calculated. The best scoring word is assumed as transcription of the data.
The nature of the score depends on the approach used. In the Dynamic Programming based systems it is usually a cost (the best scoring word is then the one with the lowest score), while in the HMM based systems it is the probability of the data being generated by a certain word model (the best scoring word is the one with the highest probability).
In this paper, we used Continuous Density Hidden Markov Models. These are probability density functions over sequences of vectors and, thanks to their statistical nature, are a suitable tool to model the feature vector sequences obtained in the previous steps of the processing.
In the next three subsections, Hidden Markov Models, Continuous Density Hidden Markov Models and their training are considered in more detail.

5.1 Hidden Markov Models

Consider a system characterized by a state q belonging to a finite set of possible states $S = \{q = i : i = 1, 2, \ldots, N\}$. The system evolves by changing state at discrete time steps and its evolution from time t to time $t + \Delta t$ can be described by a sequence of state values $\{q_t, q_{t+1}, \ldots, q_{t+\Delta t}\}$.
The probability to be in a given state q at time t depends, in the most general case, on the state of the system in the d previous steps of the evolution (where d could be equal to t) and on the time t. Two assumptions are usually made for simplifying the problem: the first one, called *first order assumption*, states that this probability depends only on the state at time $t-1$ and is called a *transition* probability. The second one, called *stationarity assumption* is that this transition probability does not depend on t. Given these two assumptions, the transition probability assumes the form:

$$p(q_t|q_{t-1}) = a_{q_t q_{t-1}}. \tag{5}$$

This situation is shown in figure 4 (a). The transition probabilities are arranged in a matrix $A = \{a_{ij}\}$ with $a_{ij} = p(q_t = j|q_{t-1} = i)$ and $i = 1, 2, \ldots, N, j = 1, 2, \ldots, N$.
The state variable at time $t = 0$ cannot be expressed in terms of transition probabilities, hence an initial state probability distribution must be provided: $\pi = \{\pi_i = p(q_0 = i) : i = 1, 2, \ldots, N\}$.
The systems that can be described in such terms are governed by a *Markov Process* and the set $\lambda = \{A, \pi\}$ is called a *Markov Model* of order 1. A Markov Model allows the definition of a probability

density function over sequences of states:

$$p(\mathbf{Q} = q_1, q_2, \ldots, q_t) = \pi_{q_0} \prod_{i=0}^{t-1} a_{q_i q_{i+1}}. \tag{6}$$

The statistical nature of Markov Models is very important because it allows them to model noisy and variable series such as those usually encountered in real data.
Consider now a system where the state is not observable directly, but through a stochastic process (see figure 4 (b)). In other words, when the system is in state i, an observation $\mathbf{o}$ is emitted with probability density function $b_i(\mathbf{o})$. At each time step, a different observation is emitted following the distribution related to the state assumed by the system.
Such a system can be described in terms of a *Hidden Markov Model* (for a good introduction see [11, 12, 13]), the word *Hidden* saying that the state variable cannot be observed directly. In this case also, we obtain a probability density function, but defined on the space of observation sequences rather than on state sequences. The probability of having an observation sequence $\mathbf{O} = \{\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_T\}$ can be written as follows:

$$p(\mathbf{O}) = \sum_{\{\mathbf{Q}=(q_0,q_1,\ldots,q_T)\}} \pi_{q_0} b_{q_0}(\mathbf{o}_1) \prod_{t=0}^{T-1} a_{q_t q_{t+1}} b_{q_{t+1}}(\mathbf{o}_{t+1}). \tag{7}$$

This equation uses the same elements described in the case of the Markov Models as well as the set $B = \{b_1(\mathbf{o}), b_2(\mathbf{o}), \ldots, b_N(\mathbf{o})\}$. An HMM is then defined by the set $\lambda = \{A, B, \pi\}$.
It is evident in equation 7 that, changing the elements of λ, the probability distribution changes, in other words, $p(\mathbf{O})$ is described by $p(\mathbf{O}|\lambda)$.
A fundamental problem in applying the HMMs is how to find, given a system to be modeled, the *good* HMM (we will discuss later what is meant by *good*), in other words, how to estimate correctly the parameters of λ. This central issue is discussed in subsection 5.3.

5.2 Continuous Density Hidden Markov Models

When the models deal with continuous observations, they are called Continuous Density HMMs. The emission probability function often used in this case is a Mixture of Gaussians, i.e. a weighted sum of Normal distributions:

$$b_i(\mathbf{o}) = \sum_j^G \omega_{ij} N(\mathbf{o}, \Sigma_{ij}, \mu_{ij}) = \sum_j^G \omega_{ij} \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma_{ij}|^{\frac{1}{2}}} exp(\frac{1}{2}(\mathbf{o} - \mu_{ij})' \Sigma_{ij}^{-1} (\mathbf{o} - \mu_{ij})) \tag{8}$$

where d is the dimension of the observations $\mathbf{o}$, μ_{ij} the mean of Gaussian j in state i, ω_{ij} the coefficients of the weighted sum and Σ_{ij} the covariance matrix of Gaussian j in state i.
Such a distribution function, when G tends to infinity, has universal approximation properties and is then suitable to model data with unknown distribution, such as the data extracted from handwritten words.
On the other hand, a mixture of Gaussians is characterized by many parameters (mixture coefficients, means, covariance matrices). The amount of training material necessary for reliable training is related to the number of parameters and, conversely, given a certain amount of training material, only mixtures with less than a corresponding number of parameters can be reliably trained.
In order to keep low the number of parameters, the data is usually assumed to be uncorrelated, i.e. the covariance of the observation components is null. This means that the covariance matrices Σ_i have non zero elements only on the principal diagonal and the number of parameters for each Σ_i is reduced from d^2 to d.
The assumption of uncorrelatedness is not realistic in most cases and this is a severe limit on the performance of the models. On the other hand, the training of models with Gaussians presenting full

covariance matrices requires too many training samples. The solution to the problem is the application of techniques (such as Principal Component Analysis and Independent Component Analysis) that, through a given transform, make the data as uncorrelated as possible.

5.3 HMM Training

The estimation of the parameters $\lambda = \{A, B, \pi\}$ is done through a *training* procedure. This consists of using known data (in the case of the handwriting this means to use observation sequences extracted from words of which the transcription is known) to estimate parameter values that make the models satisfy some criterion.
The HMMs are typically trained by maximizing the likelihood $p(\mathbf{O}_T|\theta)$, where $\mathbf{O}_T$ is the training set. Maximizing the likelihood corresponds to selecting the models that generate the data of the training set with the highest probability:

$$\lambda^* = \arg\max_{\lambda} \prod_i p(\mathbf{O}_i|\lambda). \tag{9}$$

This is what we mean by *good* model when training with the Maximum Likelihood criterion.
HMM training is usually performed with the Baum-Welch algorithm [14, 15, 16, 17, 18], an example of the Expectation-Maximization technique. Once the model is trained, the probability of an unknown sequence of observations being generated by the model can be estimated. In handwriting recognition, a different model for each word in a list of allowed transcriptions (called *lexicon*) is trained. During recognition, the word corresponding to the most likely model (the one giving the highest probability) is selected as transcription of the handwritten data.

6 Data transform

The use of Continuous Density HMMs involves the estimation of many parameters during training. This is an important limit because it requires huge amounts of training data that can be difficult to obtain. An important reduction of the number of parameters can be achieved by using Gaussians with diagonal covariance matrices in the mixtures used as emission probabilities (see section 5.2). The use of diagonal covariance matrices corresponds to the hypothesis of *uncorrelated* data. The data is said uncorrelated when the covariance of components i and j ($i \neq j$) is zero:

$$E\{(x_i - \bar{x}_i)(x_j - \bar{x}_j)\} = 0 \tag{10}$$

where x_i and x_j are components i and j of vector $\mathbf{x}$ and $\bar{x}_i$ and $\bar{x}_j$ their respective means.
The condition in equation 10 is generally not true for real data, but this can be made as uncorrelated as possible through apposite transforms. In the simplest case, the transform is linear:

$$\mathbf{y} = A\mathbf{x} \tag{11}$$

where $\mathbf{x}$ is the original vector, $\mathbf{y}$ the transformed vector and A a $k \times d$ matrix (d is the dimension of the original vector).
When $k = d$, the transform simply corresponds to a change of reference frame, when $k < d$, the transform corresponds to a projection of the original data onto a subspace of the original d-dimensional space. This determines a dimensionality reduction that further decreases the number of parameters in the HMMs. While in the case of $k = d$ the data are simply represented in a different way, when a reduction of the dimensionality is performed, there is a possible information loss. The value of k must be selected as a trade-off between the advantages due to dimensionality reduction and the disadvantages due to information loss.
The elements of A are chosen in order to give a representation of the data that is more suitable for the processing. This means, in our case, to have data as uncorrelated and low dimensional as possible. In the next subsections two transforms that can be used for such purpose will be examined in detail: Principal Component Analysis (PCA) and Independent Component Analysis (ICA).

6.1 Principal Component Analysis

Principal Component Analysis (PCA) is one of the so-called *second order methods*. This means that they use only the information contained in the covariance matrix of the data vectors. This corresponds to assume that the data follows a distribution completely determined by second order information (variance) and any further information is unuseful. The purpose of the second order methods is typically to find a representation of the data minimizing the reconstruction (mean-square) error [19, 20]. The PCA can be defined by using a recursive formulation. The direction of the first principal component is defined as:

$$\mathbf{w}_1 = \arg \max_{||\mathbf{w}||=1} E\{(\mathbf{w}^T\mathbf{x})^2\} \tag{12}$$

where $\mathbf{w}_1$ has the same dimension as the data vector and $\mathbf{x}$ has zero mean (this condition can always be easily achieved by substracting the mean vector estimated over the same set used to perform the PCA). The above equation corresponds to finding the direction that accounts for the highest amount of variance when the data is projected onto it.
Once $k-1$ vectors $\mathbf{w}_i$ have been determined, the k^{th} one can be obtained as follows:

$$\mathbf{w}_k = \arg \max_{||\mathbf{w}||=1} E\{[\mathbf{w}^T(\mathbf{x} - \sum_{i=1}^{k-1} \mathbf{w}_i\mathbf{w}_i^T\mathbf{x})]^2\} \tag{13}$$

Once all the $\mathbf{w}_i$ are found, the principal component j of a data vector $\mathbf{x}$ is given by $z_j = \mathbf{w}_j^T\mathbf{x}$.
In practice, the set of $\mathbf{w}_i$ corresponds to the set of eigenvectors of the covariance matrix $E\{\mathbf{x}\mathbf{x}^T\}$ and the corresponding eigenvalues σ_i^2 are the values of the variances of the data projections along their directions. It can be demonstrated [19] that, by approximating the vectors $\mathbf{x}$ with the first $k < d$ principal components, the average mean squared error is given by:

$$\Delta_k = \sum_{i=k+1}^{d} \sigma_i^2. \tag{14}$$

The value of Δ_k can be used to decide whether the projection of the original data onto a subspace of dimension k gives raise to an acceptable error or not (although this is not the criterion we used in the experiemnts described in this paper).
The use of PCA in the context of handwriting recognition based on Continuous Density Hidden Markov Models has several advantages. The first one is the dimension reduction of the data. In the hypothesis that the variance is essentially due to information contained in the data, it is possible to discard the components corresponding to the smallest eigenvalues. The reduction of the data dimension allows to use models with less parameters that are then easier to train.
The second one, specifically related to the use of Gaussian Mixtures as emission probability functions, is that the data is decorrelated, i.e. the elements of their covariance matrix are significantly different from zero only on the principal diagonal. This allows to use Gaussians with diagonal covariance matrix in the mixtures resulting in a much smaller number of parameters in the HMMs.

6.2 Independent Component Analysis

Independent Component Analysis is a statistical method for transforming an observed mutidimensional random vector into components that are statistically as independent from each other as possible [21, 20, 22]. Two variables x and y are said statistically independent when $p(y_i, y_j) = p(y_i)p(y_j)$. The transform leading to the independent components is modeled for simplicity as a linear transform, as in equation (11). The elements of A can be estimated by simply assuming that the components of $\mathbf{y}$ are *statistically independent* and *nongaussian* [21]. This last hypothesis is necessary because, when the y_i are gaussians and statistically independent, the transform A can be identified only up to an orthogonal transform and this ambiguity must be avoided. Moreover, a measure of nongaussianity,

the *negentropy*, is related to the statistical independence of the variables and this is the basis of the practical criterion used to find the independent components.
The negentropy of a variable $\mathbf{y}$ is defined as follows:

$$J(\mathbf{y}) = H(\mathbf{y}_{gauss}) - H(\mathbf{y}) \tag{15}$$

where $\mathbf{y}_{gauss}$ is a gaussian variable with the same covariance matrix and mean as $\mathbf{y}$ and H is the entropy:

$$H(\mathbf{y}) = -\int p(\mathbf{y}) \log p(\mathbf{y}) d\mathbf{y}. \tag{16}$$

An important advantage of $J(\mathbf{y})$ is that, if the y_i are forced to be uncorrelated (which in our case is a desirable condition), it is related to their mutual information:

$$I(y_1, y_2, \ldots, y_d) = J(\mathbf{y}) - \sum_i J(y_i). \tag{17}$$

The mutual information is the information theoretic measure of the independence of the variables. The independent components can then be obtained by finding the directions of maximum negentropy. This transforms the ICA problem into a numerical optimization problem. Since the negentropy is very heavy to compute, several approximations of it were proposed allowing a fast and reliable estimation of the independent components [23, 24].
The advantages in applying ICA to the handwriting data is two-fold when performing the recognition with HMMs. First the data is decorrelated, second a dimension reduction can be obtained by reducing the number of independent components used (the dimension of $\mathbf{y}$ in equation (11) can be smaller than the dimension of $\mathbf{x}$).

7 Experiments and Results

The experiments were performed over a publicly available database used in several works that appeared in the literature [3, 5, 25, 26]. The database is the transcription of a text extracted from the LOB corpus, a collection of texts supposed to represent the average english. The dataset[1] is composed of 4053 words and its lexicon size is 1374. The length distribution of the samples is shown in figure 5. The peak for words of length 3 is due to the high frequency of words like *the* or *and*.
In the next subsections, the training technique adopted and the results obtained are presented.

7.1 HMM Training

The dataset was first split into three subsets with a random process: training set (2362 words), validation set (675 words) and test set (1016 set). This reproduces the experimental conditions in [3].
A different HMM is created for each letter. This makes the system flexible with respect to a change of lexicon because it allows to build the word models as concatenations of letter models. In this way, it is sufficient to have in the training set samples of the letters composing the words to be modeled rather than samples of the words themselves. Moreover, the number of parameters is kept lower because the word models share the parameters belonging to the same letters. This allows a better training given the same amount of training data.
The training is not performed directly on the letter models, but on their concatenations corresponding to the words in the training set. This technique is called *embedded training* and has two important advantages: the first one is that the letters are modeled when being part of a word (that is the actual situation of the letters in the cursive handwriting), the second one is that it is not necessary to segment the words into letters to perform the training.
The topology of the models is strictly left-to-right and the number of states S and Gaussians G in

[1] The data can be downloaded at the following ftp address: `ftp://ftp.eng.cam.ac.uk/pub/data`.

the mixtures is the same for every letter model. The optimal values of S and G are selected through cross-validation [27]: all the systems corresponding to couples (S, G) falling in a range determined by the amount of training data are trained and tested. The system giving the highest recognition rate on the validation set is retained as optimal. Such system is retrained over the union of training and validation set and is tested over the test set giving a final measure of the recognition rate.
This procedure was applied using as data either the raw feature vectors, the Principal Components or the Independent Components extracted from them with PCA and ICA respectively. In these last cases, the number of components retained had also to be set through cross-validation. The results obtained are shown in the next section.

7.2 Recognition Results

By using the Expectation-Maximization algorithm, it is possible to calculate the probability of an observation sequence being generated by a word model. For each entry of the lexicon, a different HMM is built by concatenating the models related to its letters. The word corresponding to the model with the highest likelihood is selected as transcription of the data. We approximated the likelihood using the Viterbi algorithm [28, 29] which computes the likelihood of the optimal alignment between the observed sequence and the states of the model. The first system was obtained using the raw feature vectors. Systems with $5 \leq S \leq 15$ and $1 \leq G \leq 14$ were trained over the training set and tested over the validation set. The best accuracy was obtained by the system with $S = 11$ and $G = 12$. It was trained again over the set composed of the training and of the validation set and tested over the test set. The recognition rate obtained (the lexicon size is 1374) is **92.4**%.
A second system was obtained using the Principal Components extracted from the data. Different systems can be obtained not only by varying S and G, but also by changing the number P of Principal Components retained. The amount of variance as a function of the number of Principal Components retained is shown in figure 6. Since the last components account for a small amount of variance, we can expect that the loss of information when eliminating them is negligible.
Systems with $5 \leq S \leq 11$, $6 \leq G \leq 15$ and $13 \leq P \leq 16$ were trained and tested. The best result was obtained by a system with $S = 9$, $G = 13$ and $P = 16$ that gives an accuracy of **94.7**% (corresponding to a reduction of 30.3% of the error rate with respect to the system using the raw data).
We retrained (over validation and training set) and tested (over the test set) also the best system obtained for $P = \{13, 14, 15\}$. The systems using less Principal Components have a performance very close to the system with $P = 16$ (see table 1) as could be expected from the plot in figure 6. Note that the dimensionality reduction, in this case, does not give any advantage.
A third system was obtained using the Independent Components extracted through ICA. In this case the parameters to be set by cross-validation are S, G and I, the number of independent components extracted from the data. The explored range of parameters was as follows: $7 \leq S \leq 13$, $5 \leq G \leq 15$ and $13 \leq I \leq 16$. The best system ($S = 11$, $G = 14$, $I = 14$) was trained again over the union of the training and validation set and tested over the test set giving an accuracy of **93.6**% (corresponding to a reduction of 15.8% of the error rate with respect to the system using the raw data).
The same procedure was followed for the optimal systems obtained with $I = \{13, 15, 16\}$ (see table 2). Unlike using PCA, the reduction of the number of components using ICA determined an improvement of the performance.
The best system was obtained by training over the Principal Components. Its performance over the data set used (94.7%) is significantly higher than the accuracies claimed (over the same data) in [3] (the number of words in training, validation and test set is the same, but the lexicon size is 1334; and the best performance obtained is 92.8%), and slightly better than the recognition rate of the system presented in [5, 25] (using the same subsets and lexicon we used the system achieved a recognition rate of 94.6%). On the other hand, this last system needs a word segmentation (including some heuristics to find the points where to cut), a feature vector of dimension 140 (the feature extraction process estimates complex features such as profiles in 8 directions), the training of a Multi Layer Perceptron, the training of HMMs with a topology allowing transitions not only to neighboring states (like in a

strictly left-right topology), but also to other states (resulting in an bigger number of parameters).
Figure 7 shows the recognition rate on the test set as a function of the word lenght. The system achieves a 100% accuracy for samples longer than six letters. The low recognition rate for short words has several causes. First, when there are few letters, the misalignemnt of a single letter model with the corresponding letter in the word can have a strong influence. When there are more letters, an eventual misalignment can be equilibrated by other letters. A second problem is that the normalization scheme is conceived to work on long words (more than 3 letters) and produces sometimes bad results over short samples. For this reason, the performance of the system can be better on datasets having a length distribution with more weigth on the longer samples.

8 Conclusions

This work presented a system for the offline recognition of cursive words written by a single person. All the processing steps were described in detail. Moreover, the application of Principal Component Analysis and Independent Component Analysis was investigated.
Several experiments were performed on a publicly available database. The recognition accuracy achieved with the approach proposed here is, to our knowledge, the highest among the results over the same data presented in the literature. The analysis of the recognition as a function of the word length shows that the system achieves a 100% recognition rate for samples longer than six letters. This suggests that the performance of our system in tasks involving words with high average length can be very good.
Both PCA and ICA had a positive effect on the recognition rate, PCA in particular reduced the error rate, with respect to the use of raw data, by 30.3%. A further improvement can probably be obtained by using *nonlinear* or *kernel* PCA [30]. Such techniques often work better than the linear transform we used to perform PCA.
The use of data dependent heuristics was avoided in order to make the system flexible with respect to a change of writer. Any ad-hoc algorithm for the specific style of the writer was avoided.
The prior information about the word frequency and distribution can be useful to improve the recognition of short words. These are typically articles, conjunctions and propositions that appear often in written sentences. For this reason, a possible future direction to follow is the application of language models that take into account this kind of information.

References

[1] T. Steinherz, E. Rivlin, N. Intrator, Off-line cursive script word recognition - a survey, Intl. J. of Document Analysis and Recognition 2 (2) (1999) 1–33.

[2] R. Plamondon, S. Srihari, On line and off-line handwriting recognition: A comprehensive survey, IEEE Trans. on Patt. An. and Mach. Int. 22 (1) (2000) 63–84.

[3] A. W. Senior, A. J. Robinson, An off-line cursive handwriting recognition system, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (3) (1998) 309–321.

[4] U. Marti, H. Bunke, Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition system, International Journal of Pattern Recognition and Artificial Intelligence 15 (1) (2001) 65–90.

[5] Y. Tay, P. Lallican, M. Khalid, C. Viard-Gaudin, S. Knerr, Offline handwritten word recognition using a hybrid neural network and Hidden Markov Model, in: Proceedings of 6^{th} International Symposium on Signal Processing and its Applications, 2001.

[6] R. M. Haralick, L. G. Shapiro, Computer and Robot Vision, Addison Wesley, USA, 1992.

[7] A. Vinciarelli, J. Luettin, A new normalization technique for cursive handwritten words, Pattern Recognition Letters 22 (9) (2001) 1043–1050.

[8] R. M. Bozinovic, S. N. Srihari, Off-line cursive script word recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 11 (1) (1989) 69–83.

[9] O. Trier, A. Jain, T. Taxt, Feature extraction methods for character recognition - a survey, Pattern Recognition 29 (4) (1996) 641–662.

[10] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceeedings of the IEEE 86 (11) (1998) 2278–2324.

[11] L. Rabiner, A tutorial on Hidden Markov Models and selected applications in speech recognition, in: A. Waibel, L. Kai-Fu (Eds.), Readings in Speech Recognition, Morgan Kaufmann, Palo Alto, CA, 1989, pp. 267–296.

[12] F. Jelinek, Statistical Methods for Speech Recognition, MIT Press, 1997.

[13] Y. Bengio, Markovian models for sequential data, Neural Computing Surveys 2 (1999) 129–162.

[14] L. Baum, T. Petrie, Statistical inference for probabilistic functions of finite state markov chains, Annals of Mathematical Statistics 37 (1966) 1554–1563.

[15] L. Baum, J. Egon, An inequality with applications to statistical estimation for probabilistic function of a markov process and to a model for ecology, Bulletin of American Metheorological Society 73 (1967) 360–363.

[16] L. Baum, G. Sell, Growth functions for transformation on manifolds, Pac. J. Math. 27 (2) (1968) 211–227.

[17] L. Baum, T. Petrie, G. Soules, N. Weiss, A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains, Annals of Mathematical Statistics 41 (1) (1970) 164–171.

[18] L. Baum, An inequality and associated maximization technique in statistical estimation for probabilistic functions of markov processes, Inequalities 3 (1972) 1–8.

[19] C. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1995, Ch. 8, pp. 310–319.

[20] A. Hyvärinen, Survey on Independent Component Analysis, Neural Computing Surveys 2 (1999) 94–128.

[21] P. Comon, Independent Component Analysis, a new concept?, Signal Processing 36 (3) (1994) 287–314.

[22] A. Hyvärinen, E. Oja, Independent Component Analysis: A tutorial, Neural Networks 13 (4-5) (2000) 411–430.

[23] A. Hyvärinen, Fast and robust fixed-point algorithms for Independent Component Analysis, IEEE Transactions on Neural Networks 10 (3) (1999) 626–634.

[24] A. Hyvärinen, A fast fixed-point algorithm for Independent Component Analysis, Neural Computation 9 (7) (1997) 1483–1492.

[25] Y. Tay, P. Lallican, M. Khalid, C. Viard-Gaudin, S. Knerr, An offline cursive handwritten word recognition system, in: Proceedings of IEEE Region 10 Conference, 2001.

[26] U. Marti, H. Bunke, Towards general cursive script recognition, in: Proc. of Intl. workshop on Frontiers in Handwriting Recognition, Korea, 1998, pp. 379–388.

[27] M. Stone, Cross-validatory choice and assessment of statistical prediction, Journal of the Royal Statistical Society 36 (1) (1974) 111–147.

[28] A. Viterbi, Error bounds for convolutional codes and an asimptotically optimal decoding algorithm, IEEE Transactions on Information Theory IT-13 (2) (1967) 260–269.

[29] G. Forney, The Viterbi algorithm, Proceedings of IEEE 61 (3) (1973) 268–278.

[30] B. Schölkopf, A. Smola, K. Müller, Kernel Principal Component Analysis, in: B. Schölkopf, C. Burges, A. Smola (Eds.), Advances in Kernel Methods, MIT Press, 1998, pp. 327–352.

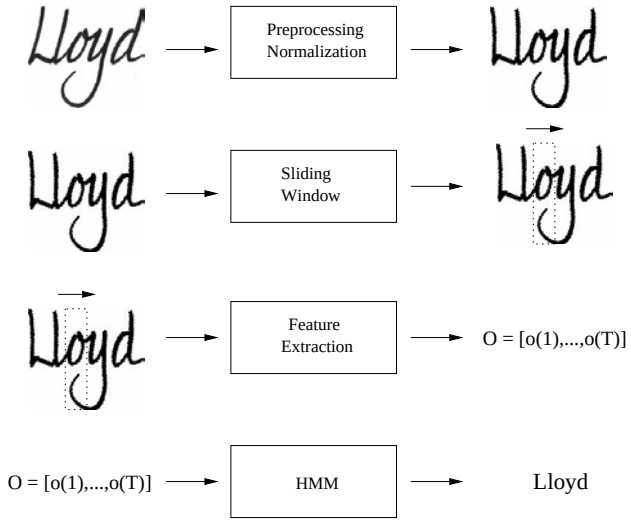


Figure 1: Architecture of the system.

P	Acc.(%)	S	G
16	94.7	9	13
15	94.2	8	10
14	94.6	7	13
13	94.3	8	13

Table 1: Accuracy on the test set as a function of P. Given a number of P, the optimal values of G and S are set through cross-validation. The system giving the highest result is trained again over the union of training and validation set and tested over the test set.

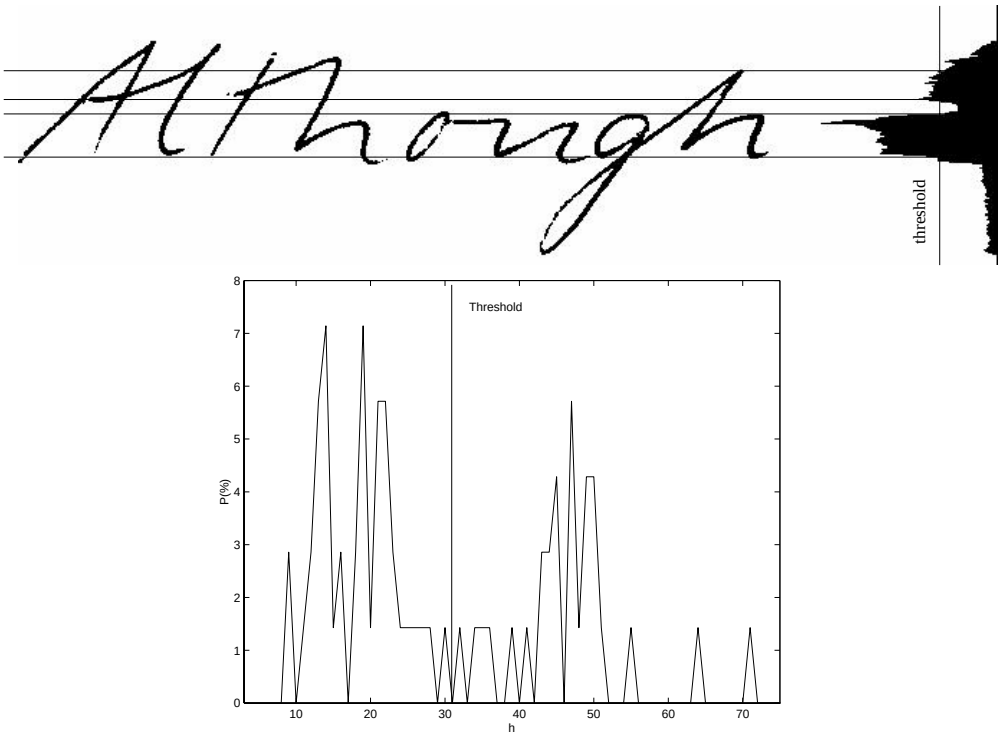


Figure 2: Slope removal technique. The vertical projection of the image (right side of the upper figure) is converted into a probability density function (lower plot). The distribution (although roughly estimated) shows clearly the two modes related to dense and sparse areas. The Otsu thresholding algorithm finds the density threshold that separates the two modes. In order to distinguish between the actual core region and the false core region candidate determined by the ascender region, the following simple heuristic is used: the number of foreground pixels in the two candidate regions is counted, the region containing the biggest number of pixels is selected as the actual core region.

I	Acc.(%)	S	G
16	92.8	12	12
15	93.5	11	7
14	93.6	11	14
13	88.9	12	11

Table 2: Accuracy on the test set as a function of I. Given a number of I, the optimal values of G and S are set through cross-validation. The system giving the highest result is trained again over the union of training and validation set and tested over the test set.

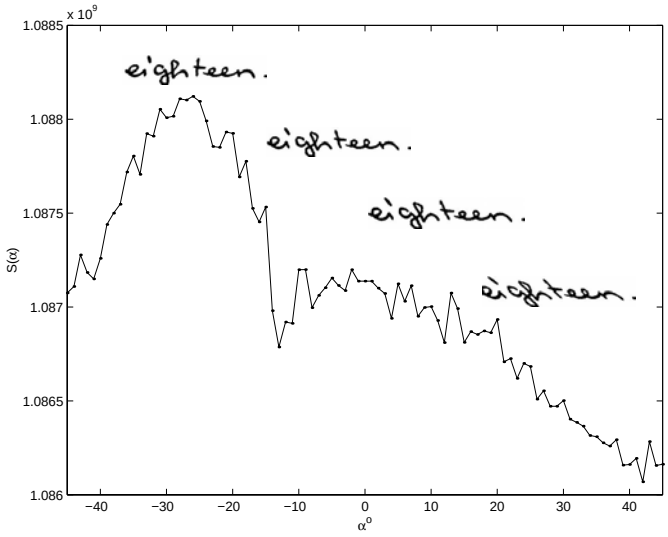


Figure 3: Slant removal technique. The value of $S(\alpha)$ is maximum for the deslanted image. The shear transformed images corresponding to some values of α are shown.

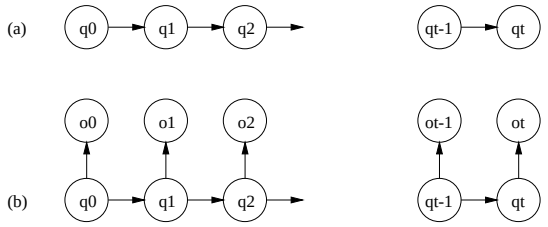


Figure 4: Markov (a) and Hidden Markov Models (b). In the Markov Model, the state variable can be observed directly. The arrow indicates that the state variable at a certain time step influences the distribution of the variable at the following time step. In the Hidden Markov Model, the state variable cannot be observed directly, but only through an observation. The arrow shows that the distribution of this observation is influenced by the state variable. As in the case of the Markov Model, the state at a certain time step influences the distribution of the state variable at the following step.

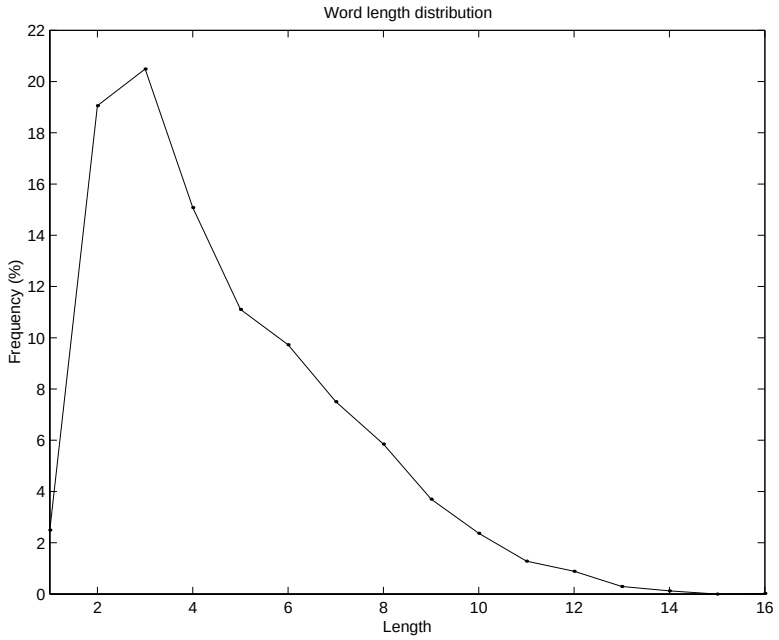


Figure 5: Word length distribution of the samples. The high peak around 3 is due to the high frequency of words like *the* or *and*.

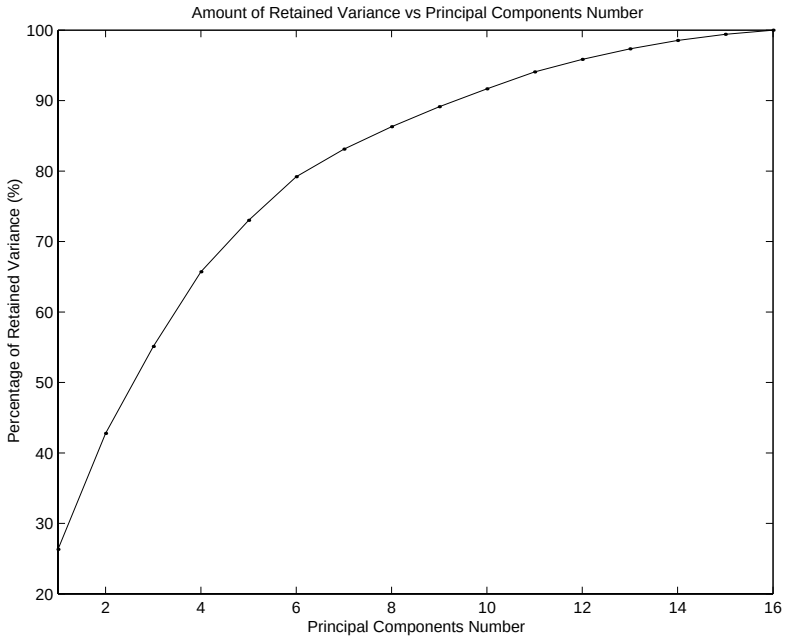


Figure 6: Variance retained as a function of the number of Principal Components. The last 4 components account for less than the 5% of the total variance of the data.

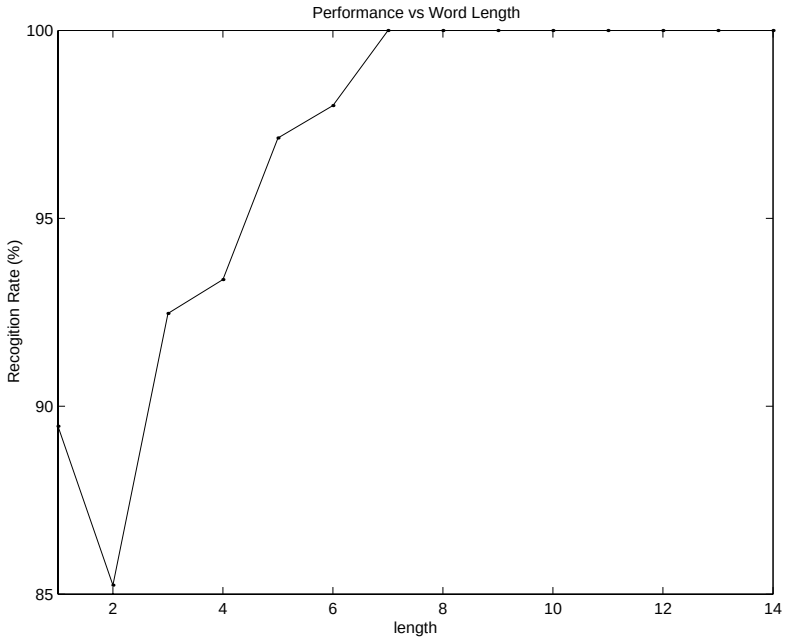


Figure 7: Accuracy on the test set with respect to word length. The system achieves 100% recognition rate for samples longer than 6 letters.