Cursive Word Recognition Using a Random Field Based Hidden Markov Model

George Saon and Abdel Belaïd

LORIA
Bât Loria, Campus scientifique, B.P. 239
54506 Vandœuvre-Lès-Nancy Cedex, France

Phone: (33) 03 83 59 30 80
Fax: (33) 03 83 41 30 79
E-mail: saon@loria.fr
Abstract

In this paper we present a two-dimensional stochastic method for the recognition of unconstrained handwritten words in a small lexicon. The method is based on an efficient combination of hidden Markov models (HMMs) and causal Markov random fields (MRFs). It operates in a holistic manner, at the pixel level, on scaled binary word images which are assumed to be random field realizations. The state-related random fields act as smooth local estimators of specific writing strokes by merging conditional pixel probabilities along the columns of the image. The HMM component of our model provides an optimal switching mechanism between sets of MRF distributions in order to dynamically adapt to the features encountered during the left-to-right image scan. Experiments performed on a French omni-scriptor, omni-bank database of handwritten legal check amounts provided by the A2iA company are described in great extent.

Keywords: off-line handwriting recognition, hidden Markov models, Markov random fields

1 Introduction

Recent works in off-line handwriting recognition (HWR) using stochastic models show an increasing interest in the extension of 1-D HMMs to two dimensions in order to fit more accurately the nature of the modeled data. However, it was proved by Levin in [12] that a direct extension of the dynamic time warping algorithm (DTW), which is the basic mechanism of these models, to the plane, results in an NP-complete problem. By applying a class of constraints to the matching, the complexity can be pulled down to polynomial. A type of models issued from such a simplification are the PHMMs (planar- or pseudo HMMs) [2, 11]. Although these models are easy to implement, the underlying statistical line-independency hypothesis does not always hold true in practice [2]. In order to cope with this shortcoming, some solutions consist of clustering homogeneous lines into classes using the $k$-means algorithm [11] or of creating super-state equivalence classes [6]. Yet these solutions, being based on classification algorithms, do not solve fine dependency cases between consecutive lines. Markov random fields naturally overcome this limitation.
given that the probabilities of pixels are conditioned by their direct 2-D neighbors.

Markov fields have been employed for a long time in statistical mechanics, the application of these models to images being more recent. They are used in image processing and artificial vision in tasks such as segmentation and restoration [7].

Among these models, the causal Markov random fields are very popular for two major reasons. First, as stated in [3], one cannot specify arbitrary conditioning neighborhoods for consistency issues (existence of the joint field probability), whereas there are several theoretical achievements on causal MRFs. On the other hand, recursive training and recognition procedures are more easily applicable on causal fields allowing a natural progression of the joint field mass probability calculus. The concept of causality may have different interpretations since the plane is not provided with a natural order.

Two types of causal MRFs are frequently encountered in image processing: the Markov random mesh (M RM) [1] and the unilateral Markov random field also called non-symmetric half-plane Markov chain (NSHP) [15]. Jeng in [8] noted that NSHPs are more appropriate than MRMs when an accurate model for representing two dimensional data is required (MRMs are conditionally independent on 45° diagonals which diminishes their capability to detect strokes having these orientations).

For handwriting recognition, two different approaches based on causal MRFs were simultaneously developed. Park and Lee introduced a third-order hidden Markov mesh random field (HMMRF) [13] and applied it to off-line handwritten character recognition [14]. By using a vector quantization technique, the cells resulting from a regular decomposition of the input image are encoded into a 2-D sequence of symbols from a finite alphabet. During the decoding phase, a fast look-ahead scheme [4] based on marginal MAP is used to find an efficient estimation of the hidden states. The state sequence found is a realization of an underlying stationary Markov random mesh process. In order to avoid the exponential complexity inherent to a complete state decoding, the authors use a simplified version of the 2-D EM algorithm (well explained in [5]) called decision-directed which was first proposed by Devijver [4]. This algorithm assumes that the lines and the columns are mutually independent which may decrease the modeling accuracy for specific applications as was noticed by the author.
Our approach [17, 18] consists of using NSHP Markov random fields at a pixel observation level without making any hypothesis on the dependency between lines and/or columns. In each state of the HMM, we observe the image columns in a left-to-right fashion. For the current column, the emission probability is computed using state-related NSHP-like conditional pixel distributions. A transition from a state to another implies an optimal change of these distributions in order to maximize the likelihood of the image. Training is based on the MLE optimization criterion and mainly consists in estimating the pixel distributions. The estimation is done by performing a maximum likelihood count of pixel configurations (value of the current pixel and of its neighbors) modulated by the probability of being in a given state which, at its turn, is expressed using modified Forward-Backward functions. The former functions serve also for computing the emission probability during recognition.

The paper is organized as follows. In section 2, we define our proposed model and give the most important elements concerning training and recognition. Section 3 deals extensively with the experiments performed by describing the database in detail, the tests carried out and the results obtained. A discussion and concluding remarks are presented at the end of the paper.

2 Non-symmetric Half-plane Hidden Markov Models

Let $X = \{X_{ij}\}_{(i,j) \in L}$ be a random field defined over a $m \times n$ integer lattice $L$. We define $\Sigma_{ij} = \{(k, l) \in L \mid l < j \text{ or } (l = j, k < i)\}$ the non-symmetric half-plane and $\Theta_{ij} \subset \Sigma_{ij}$ the support of pixel $(i, j) \in L$ (see Figure 1).

$X$ is called a non-symmetric half-plane Markov chain if and only if:

$$P(X_{ij} | X_{\Sigma_{ij}}) = P(X_{ij} | X_{\Theta_{ij}}), \quad \forall (i, j) \in L$$ (1)

where by $P(X_{ij} | X_A)$, we generally mean $P(X_{ij} | X_{kl}, (k, l) \in A)$ for a subset $A \subset L$.

The joint field mass probability $P(X)$ may be computed following the chain decom-
Figure 1: Sets of pixels related to site \((i, j)\).

position rule of conditional probabilities:

\[
P(X) = \prod_{j=1}^{n} \prod_{i=1}^{m} P(X_{ij}|X_{\Theta ij}) = \prod_{j=1}^{n} \prod_{i=1}^{m} P(X_{ij}|X_{\Theta ij})
\]

As shown in equation (2), the joint probability can be computed column by column in a sequential manner. Then, we associate a stochastic state process to the field columns in order to tie the conditional probabilities relative to a column to a specific state. Moreover, the NSHP realization (pattern image) may be seen as an observation sequence of columns. With this in mind and assuming that the state process is a first-order Markov chain, the column products become observation probabilities which will be combined with the state transitions leading to a HMM-like decomposition scheme of the pattern likelihood with respect to a model. A transition from one state to another will result in changing the set of probability distributions and in dynamically modifying feature sensitivity during the left-to-right image scan.

Formally, let \(Q = q_1 \ldots q_n\) be a stochastic state process associated to the columns of \(X\). The random variables \(q_j\) take values in a finite set of states \(S = \{s_1, \ldots, s_N\}\). Using
equation (2), the pattern likelihood with respect to a model $\lambda$ may be written as:

$$P(X|\lambda) = \sum_{Q} P(X, Q|\lambda) = \sum_{Q} P(X|Q)P(Q|\lambda)$$

$$= \sum_{Q} \prod_{j=1}^{n} P(q_{j}|q_{j-1})P(X_{j}|X_{j-1}, \ldots, X_{1}, q_{j}, \lambda) \quad (*)$$

$$= \sum_{Q} \prod_{j=1}^{n} P(q_{j}|q_{j-1}) \prod_{i=1}^{m} P(X_{ij}|X_{\Sigma_{ij}}, q_{j}, \lambda)$$

$$= \sum_{Q} \prod_{j=1}^{n} P(q_{j}|q_{j-1}) \prod_{i=1}^{m} P(X_{ij}|X_{\Theta_{ij}}, q_{j}, \lambda)$$

where in $(*)$, we assume state independence at the column observation level.

![Diagram](image)

**Figure 2: Architecture of an NSHP-HMM model.**

Figure 2 shows an example of a model architecture indicating the pixel neighborhoods related to the states and the line indices. An NSHP-HMM model is defined by:

- $V = \{0, 1\}$ the vocabulary. We denote a pixel realization of $X_{ij}$ by $x_{ij} \in V$.
- $\Theta = \{\Theta_{ij}\}_{(i,j) \in L}, \Theta_{ij} = \{(i-j, j-j_k)|1 \leq k \leq P, j_k > 0 \text{ or } (j_k = 0, i_k > 0)\} \cap L,$
  where $P$ represents the number of neighboring pixels per site. $\Theta$ is called the neighborhood set and $P$ the order of the model.
- $S = \{s_1, \ldots, s_N\}$ the set of states. We denote by $q_j \in S$ the state for column $X_j$.
- $A = \{a_{kl}\}_{1 \leq k, l \leq N}, a_{kl} = P(q_{j+1} = s_k|q_j = s_k)$ the state transition matrix.
\( B = \{ b_{ik}(x, \mathbf{x}) \}_{1 \leq i \leq m, 1 \leq k \leq N}, \ x \in \{0, 1\}, \mathbf{x} \in \{0, 1\}^P, \ b_{ik}(x, \mathbf{x}) = P(X_{ij} = x | X_{\mathbf{e}_{kj}} = x, q_j = s_k), \) the conditional pixel observation probabilities.

\( \pi = \{ \pi_i \}_{1 \leq i \leq N} \) with \( \pi_i = p(q_1 = s_i) \), the initial state probabilities.

This definition is worth some explanations. On the first hand, the resulting model acts like an HMM with multiple and non-correlated observation sources. Indeed, for a given state, the observation is an \( m \)-dimensional vector whose component probabilities are multiplied suggesting a statistical independence hypothesis at this level. Therefore, the model benefits of all the properties (dynamic warping, training optimality, etc.) which are the strengths of HMMs in pattern recognition.

On the other hand, from the viewpoint of the random field component, we have specified the local states of a non-isotropic and non-homogeneous NSHP by means of explicit distributions (MRF interpretation). These local states depend simultaneously upon the line index and the observation state. The field anisotropy allows a two-directional variable modeling of the patterns. Moreover, on the horizontal axis, the change of the NSHP distributions is guided by the hidden warping structure and is done in an optimal manner in order to maximize the image likelihood.

In the following, we show how to estimate the emission probability of a pattern (the image likelihood), and we give some elements concerning training and recognition.

An optimal evaluation of the likelihood \( P(X|\lambda) \) is obtained using modified Forward-Backward functions. We will define the Forward function \( \alpha \) (Backward function \( \beta \) following a dual definition) as being the accumulated field probability until column \( X_j \) of \( X \) when ending in state \( s_i \), \( \alpha_j(i) = P(X_1X_2 \ldots X_j, q_j = s_j|\lambda) \):

\[
\begin{align*}
\alpha_1(i) &= \pi_i \prod_{k=1}^{m} b_{ki}(X_{k1}, X_{\mathbf{e}_{ki}}), \quad 1 \leq i \leq N \\
\alpha_j(i) &= \sum_{l=1}^{N} a_{j-1}(l) a_l \prod_{k=1}^{m} b_{ki}(X_{kj}, X_{\mathbf{e}_{kj}}), \quad j = 2 \ldots n \\
P(X|\lambda) &= \sum_{i=1}^{N} \alpha_n(i)
\end{align*}
\] (4)
2.1 Training

During training, the goal is to determine the parameters \((A, B, \pi)\) of the model which maximize the product \(\prod_{r=1}^{R} P(X^{(r)}|\lambda)\), where \(X^{(r)}\) are sample images used to train the model \(\lambda\). We use the maximum likelihood criterion (MLE) by performing Baum-Welch re-estimation. We will only detail the conditional pixel probability re-estimation:

\[
\bar{b}_{il}(x, \mathbf{x}) = \begin{cases} 
\sum_{r=1}^{R} \frac{1}{P_r} \sum_{j=1}^{n_r} \frac{\alpha_j^{(r)}(l) \beta_j^{(r)}(l)}{\text{den} \neq 0} & \text{if } X_{ij}^{(r)} = x \text{ and } X_{\Theta_{ij}}^{(r)} = \mathbf{x} \\
\frac{1}{P_r} \sum_{j=1}^{n_r} \frac{\alpha_j^{(r)}(l) \beta_j^{(r)}(l)}{\text{den} \neq 0} & \text{if } X_{ij}^{(r)} = \mathbf{x} \text{ and } X_{\Theta_{ij}}^{(r)} = \mathbf{x} \\
\bar{b}_{il}(x, \mathbf{x}), & \text{otherwise}
\end{cases}
\]

\(x \in \{0, 1\}, \quad \mathbf{x} \in \{0, 1\}^p, \quad 1 \leq i \leq m, \quad 1 \leq l \leq N\)

where by \(P_r = P(X^{(r)}|\lambda)\), we mean the emission probability of sample \(X^{(r)}\) and by \(n_r\) its length. Let us take a closer look to equation (5). In fact, pixel probability re-estimation is done by performing an ML count of the number of times that a given pixel configuration is encountered. Note that all samples are supposed to have the same number of lines \(m\) which necessitates a height normalization procedure prior to training or recognition.

2.2 Recognition

We chose a model discriminant approach by constructing an NSHP-HMM model for each different word class. Recognition is performed simply by calculating the pattern likelihood for all models (using equation (4)) and by labeling the image according to the model which produces the maximum a posteriori probability via Bayes decision rule. Another way to do this is to compute Viterbi-like distortion measures of the input pattern for all models and to label the image according to a marginal MAP criterion. As will be seen in the next section, the latter is less efficient in terms of the recognition score than using the
emission probability (computed in the sense of equation (4)) since it disregards the MLE optimization criterion assumed during training.

3 Experiments and Results

Experiments were carried out on a database provided by the A2iA company which is commonly referred as 0503. The database consists of over 23,000 images of handwritten legal bank check amounts (more than 100,000 words) issued from various French banks and often of very poor quality. Each image was scanned at 240dpi and the legal amount zone was previously located and isolated. Horizontal and diagonal bars have been removed and the amounts were extracted from their backgrounds. Finally, the database consists of cleaned, binary legal amount images. Each amount is accompanied by a description indicating among other things the word labels (in French, there are 26 in all for writing checks) and the horizontal segmentation limits. For words which present insertions, deletions or which are abbreviations, the labels start respectively, with ‘+’, ‘-’ and ‘/’. A more detailed description of this database can be found in [10]. Several examples of word images are shown at the end of the paper. Next, we will indicate the main preprocessing steps performed by our system.

3.1 Preprocessing

3.1.1 Slant correction

In order to reduce the variability of the writing slant of the words, a simple horizontal shearing transform of the image is performed. The average slant angle of the characters is computed using a fast one-scan technique inspired by Kimura’s algorithm [9]. This technique consists in marking for each line of the image the black pixels which have a left or right white neighbor. For such a pixel, say \((i, j)\), we check if one of the 5 pixels above \(((i - 1, j - 2), (i - 1, j - 1), (i - 1, j), (i - 1, j + 1)\) and \((i - 1, j + 2)\)) is marked and we increment a global counter among \(n_{-2}, n_{-1}, n_0, n_1, n_2\) accordingly. Finally, the average
slant angle \( \theta \) is found as:
\[
\theta = \arctan \left( \frac{n_1 - n_{-1} + 2(n_2 - n_{-2})}{n_0 + n_1 - n_{-1} + 2(n_2 - n_{-2})} \right)
\] (6)

Intuitively, the right term represents a weighted number of changes in a given direction with respect to the vertical one. Horizontal changes of a range greater than 2 pixels are neglected since they are less frequent.

3.1.2 Coarse band extraction and natural length estimation

We define the natural length of a word as the number of horizontal black-white transitions averaged over the lower case zone. Thus, band extraction is performed only in order to detect this zone and to estimate the natural length. Since the band information is not to be used during recognition (we do not consider it sufficiently reliable), it is not necessary to detect it with a high accuracy. We used a rather simple algorithm which consists of calculating the product between the horizontal black pixel histogram and the histogram of horizontal black-white transitions. The baseline and the upper line are detected when this product falls below a given fraction of its maximum.

3.1.3 Height image scaling and smoothing

The recognition models require that all samples have the same number of lines. It is therefore necessary to apply a size reduction in order to meet this requirement. In practice, we reduce the images to a constant height of 20 lines. Scaling is done uniformly in the sense that the new width is proportional to this height, therefore avoiding the problems related to non-linear distortions of the word image. This procedure implicitly performs a smoothing of the initial image which is necessary since the shear transform of the slant correction algorithm produces some irregularities on the word contours. Some precautions are taken for words which have excessively long upstrokes and/or downstrokes. For such words, we prefer to cut non-uniformly the initial image to a multiple of the new height.
3.1.4 Data mirroring

We observed that the order of the image scan during training and recognition is relevant. The previously detailed models operate in a left-to-right and top-to-bottom manner on the input images. It is quite intuitive to expect that a different kind of traversal may model differently specific features in the images. This comes from the causal nature of the random fields and of the HMM. As seen before, due to the structure of the neighborhoods, the information coming from the pixels located right or below the current one is neglected when processing its conditional observation probability. One way to take into account the various scan order-dependent modeling possibilities is to mirror the image horizontally, vertically, horizontally and vertically while keeping the left-to-right and top-to-bottom processing order of the NSHP-HMM models. In this way, from the initial image, we create 4 mirrored images which will be processed by specific sub-models. The pre-processing steps are illustrated in Figure 3.

![Diagram of preprocessing steps](image)

Figure 3: The preprocessing steps performed.

3.2 Word Model Training

For reasonable computational trainability reasons, we limited the maximum number of samples for a given word to 2000. For each class, 9/10 randomly chosen samples were used for training the models, that is on average 1800 samples, except for some words which are less frequent (such as "un", "onze", "douze", etc), where the number of samples ranges from 400 to 1000. The total number of samples used for training the sub-models related to a given scan is 36829. Next, we show how we chose the initial model parameters.
We will limit ourselves to the description of the word sub-models corresponding to the left-to-right and top-to-bottom scan of the images since it remains identical for the other sub-models.

- **State number**: it is proportional to the average word length in pixel columns, $\pi$, after height normalization. In practice, a number of states equal to $\pi/2$ (ranging from 14 for model "et" to 45 for "soixante" for $m = 20$ lines) gave the best recognition results.

- **State transitions**: we allow only transitions to the current or to the next state (strict left-to-right architecture). Initially, transition probabilities are equiprobable, that is $a_{ii} = a_{ii+1} = 0.5$, $1 \leq i < N - 1$.

- **Number of lines**: for computational trainability reasons, we limited this number to $m = 20$. Experiments were carried out with $m = 10$, $m = 15$ and $m = 20$ lines.

- **Model order (number of neighborhood pixels)**: we experimented models of order $P = 0 \ldots 4$ corresponding to the neighborhoods depicted in Figure 4. The impact of the model order on the average word recognition score is illustrated in Figure 7.

- **Conditional pixel observation probabilities**:

\[
    b_{\theta}(x, x) = \begin{cases} 
        \frac{1}{\text{den}} \sum_{k=1}^{K} \left\{ j \mid 1 + \frac{(l - 1)n_k}{N} \leq j \leq \frac{ln_k}{N}, X_{ij}^{(k)} = x, X_{\theta ij}^{(k)} = x \right\}, & \text{if } \text{den} \neq 0 \\
        0.5, & \text{otherwise}
    \end{cases}
\]

(7)

All samples were divided in $N$ vertical bands of equal width. In (7), a normalized count of the number of pixel configurations $X_{ij}^{(k)} = x$ and $X_{\theta ij}^{(k)} = x$, within band $l$ is performed over all samples $X^{(k)}$ and this for all possible pixel configurations $(x, x)$.

In Figure 5, we show the average sample log-likelihood for some word models as a function of the number of training steps. Note the smoothness of the log-likelihood due to the large number of samples used and to the parameter initialization method.
Figure 4: Various neighborhoods and orders considered during testing.

Figure 5: Evolution of the average sample log-likelihood.
3.2.1 Natural length estimation

For each word class, we compute during training the distribution of the natural sample lengths. The estimation of the natural length probability is performed by a simple frequency count without making any assumptions on the distribution laws. This quantity will be used during recognition as an extra feature of the system. Its role is to limit some confusions which appeared in the previous versions of our system between words of different writing complexities which present common parts (for example, "cinq" and "cinquante" or "cent" and "centimes"). This probability term will be used with a certain weight in the final class a posteriori calculus.

3.2.2 Pixel observation probability post-processing

During recognition, some "unseen" pixel configurations may appear within the input pattern. Even if the number of samples used for training is very high, such configurations are frequent because of the huge variability of the patterns at the pixel level. The probability of such a local configuration is 0 and will heavily penalize the emission probability of the pattern. The aim of the post-processing is to increase the probabilities which fall below a given threshold after training (set to 0.001 in practice) while decreasing the dual ones by the same quantity in order to respect the Markovian constraints at this level.

3.2.3 Word prototype synthesis

It is always interesting to have visual feedback on the real learning capabilities of the word models. The Figure 6 shows the prototypes generated for the entire lexicon. The grey levels code the conditional probability of black pixels, and depend upon the state and the line index of the NSHP-HMM. These prototypes were obtained in the following way. After training, a Viterbi state-decoding is performed for each sample. Since the model architecture is strictly left-to-right (see Figure 2), it is possible thereafter to estimate the frequency of staying in a given state. In order to generate a prototype of a given length, we proceed column by column and we change states according to their frequencies. For creating the current pixel neighborhood, we threshold the previously generated probabilities.
As we can see, despite the enormous variability of the patterns, the models are able to focus on pixel distributions which characterize specific writing strokes. For example, note the presence of the upstrokes and the downstrokes for the words which are supposed to have one, the presence of holes in ”deux”, ”dix”, ”onze”, the presence of i-dots in ”huit”, ”dix”, ”quinze”, the presence of t-bans in ”cent”, ”centimes” and so on. For several words, the beginning of their corresponding prototypes is very blurred because of the samples which are written with capital initials as in ”huit”, ”cent”, ”mille”, etc.

As illustrated in [16] for a handwritten digit recognition task, the NSHP-HMM models are application independent since their inputs may be any kind of binary images and there is no need to provide special domain knowledge except the choice of the initial model parameters.

3.3 Recognition

Recognition was performed on 4098 word images (corresponding to 1/10 of the database) that is, on average, on 200 samples for the most frequent words. In order to detail our recognition method, let us denote by $X^i$, $i = 1 \ldots 4$, the mirrored images of the height-scaled input word. Moreover, let $l$ be its natural length (average number of horizontal black-white transitions in the lower case band). The a posteriori probability of a word class $\omega$ will be given by:

$$
P(\omega|X^1, \ldots, X^4, l) = \frac{\left[\prod_{i=1}^{4} P(X^i|\lambda^i_\omega)\right] P(l|\omega) P(\omega)}{P(X^1, \ldots, X^4, l)}$$

(8)

where by $P(X^i|\lambda^i_\omega)$, we mean the emission probability of $X^i$ in the NSHP-HMM sub-model $\lambda^i_\omega$, by $P(l|\omega)$ the natural length probability and by $P(\omega)$ the a priori probability of the label $\omega$. Since $P(X^1, \ldots, X^4, l)$ remains constant during recognition and following equation (8), the MAP word label $\omega^*$ can be found as:

$$
\omega^* = \arg\max_{\omega \in \Omega} \left[\prod_{i=1}^{4} P(X^i|\lambda^i_\omega)\right] P^\alpha (l|\omega) P^\beta (\omega)
$$

(9)

with $\Omega$ representing the vocabulary. The weights $\alpha$ and $\beta$ are introduced for practical
<table>
<thead>
<tr>
<th>French Word</th>
<th>English Translation</th>
</tr>
</thead>
<tbody>
<tr>
<td>un</td>
<td>quatorze</td>
</tr>
<tr>
<td>deux</td>
<td>quinze</td>
</tr>
<tr>
<td>trois</td>
<td>seize</td>
</tr>
<tr>
<td>quatre</td>
<td>vingt</td>
</tr>
<tr>
<td>cinq</td>
<td>trente</td>
</tr>
<tr>
<td>six</td>
<td>quarante</td>
</tr>
<tr>
<td>sept</td>
<td>cinquante</td>
</tr>
<tr>
<td>huit</td>
<td>soixante</td>
</tr>
<tr>
<td>neuf</td>
<td>cent</td>
</tr>
<tr>
<td>dix</td>
<td>mille</td>
</tr>
<tr>
<td>onze</td>
<td>francs</td>
</tr>
<tr>
<td>douze</td>
<td>et</td>
</tr>
<tr>
<td>treize</td>
<td>centimes</td>
</tr>
</tbody>
</table>

Figure 6: Prototype synthesis by the word models.
reasons. In Figure 7 we show the evolution of the recognition results as a function of the number of training steps for models having the orders and the neighborhoods depicted in Figure 4. As we can see, the 4th-order NSHP-HMMs gave the best results and generally, the recognition rate increases with the order of the models. However, above order 4, presently we were not able to improve the score significantly.

![Plot](image)

**Figure 7:** Evolution of the recognition score as a function of the number of training steps.

In Table 1, we study the influence of various parameters on the average recognition score. Top i means that the input word was found among the first i candidates. Finally, we obtain a 80.48% top 1 and a 85.77% top 3 average word recognition rate using weighted probabilities (model a priori and natural length).

The following results (Table 2 and confusion matrix 3) were obtained by applying 3 corrective training steps on the misrecognized samples (during training) to all word models. This step increases the average score of about 2% as shown in Table 2. Figure 8 illustrates some examples of misrecognized words. For each image, we indicate the output of the system and the true label.

Comparatively, on the same database, Knerr et al. in [10] reports the following results using two independent word recognition chains (Table 4).
<table>
<thead>
<tr>
<th>Method</th>
<th>Top 1</th>
<th>Top 2</th>
<th>Top 3</th>
<th>Ni/Naot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1S+EP+WAP+WNL</td>
<td>71.38%</td>
<td>78.14%</td>
<td>79.87%</td>
<td>2925/4098</td>
</tr>
<tr>
<td>1S+VP+WAP+WNL</td>
<td>72.06%</td>
<td>78.75%</td>
<td>80.09%</td>
<td>2953/4098</td>
</tr>
<tr>
<td>4S+VP</td>
<td>75.99%</td>
<td>80.36%</td>
<td>81.70%</td>
<td>3114/4098</td>
</tr>
<tr>
<td>4S+VP+WAP</td>
<td>75.89%</td>
<td>80.28%</td>
<td>81.58%</td>
<td>3110/4098</td>
</tr>
<tr>
<td>4S+VP+WNL</td>
<td>77.89%</td>
<td>82.24%</td>
<td>83.46%</td>
<td>3192/4098</td>
</tr>
<tr>
<td>4S+VP+WAP+WNL</td>
<td>77.77%</td>
<td>82.14%</td>
<td>83.38%</td>
<td>3187/4098</td>
</tr>
<tr>
<td>4S+EP</td>
<td>77.99%</td>
<td>82.33%</td>
<td>83.70%</td>
<td>3196/4098</td>
</tr>
<tr>
<td>4S+EP+WAP</td>
<td>77.87%</td>
<td>82.28%</td>
<td>83.58%</td>
<td>3191/4098</td>
</tr>
<tr>
<td>4S+EP+WNL</td>
<td>80.43%</td>
<td>84.48%</td>
<td>85.82%</td>
<td>3296/4098</td>
</tr>
<tr>
<td>4S+EP+WAP+WNL</td>
<td>80.48%</td>
<td>84.46%</td>
<td>85.77%</td>
<td>3298/4098</td>
</tr>
</tbody>
</table>

1s, 4s = number of scans (sub-models)
EP, VP = emission and Viterbi probability
WAP, WNL = weighted a priori and natural length probability

Table 1: Influence of various parameters on the average recognition rate.
<table>
<thead>
<tr>
<th>Word</th>
<th>Top 1</th>
<th>Top 2</th>
<th>Top 3</th>
<th>Top 4</th>
<th>Top 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>un</td>
<td>68.35%</td>
<td>74.68%</td>
<td>79.75%</td>
<td>81.01%</td>
<td>84.81%</td>
</tr>
<tr>
<td>deux</td>
<td>83.41%</td>
<td>89.18%</td>
<td>92.31%</td>
<td>94.23%</td>
<td>95.67%</td>
</tr>
<tr>
<td>trois</td>
<td>81.50%</td>
<td>88.11%</td>
<td>92.51%</td>
<td>94.71%</td>
<td>95.15%</td>
</tr>
<tr>
<td>quatre</td>
<td>86.30%</td>
<td>91.78%</td>
<td>94.32%</td>
<td>95.69%</td>
<td>96.67%</td>
</tr>
<tr>
<td>cinq</td>
<td>85.37%</td>
<td>89.76%</td>
<td>91.71%</td>
<td>92.20%</td>
<td>92.68%</td>
</tr>
<tr>
<td>six</td>
<td>79.50%</td>
<td>92.50%</td>
<td>94.50%</td>
<td>96.50%</td>
<td>98.00%</td>
</tr>
<tr>
<td>sept</td>
<td>75.46%</td>
<td>86.50%</td>
<td>91.41%</td>
<td>95.09%</td>
<td>96.32%</td>
</tr>
<tr>
<td>huit</td>
<td>74.00%</td>
<td>82.00%</td>
<td>88.00%</td>
<td>91.50%</td>
<td>92.00%</td>
</tr>
<tr>
<td>neuf</td>
<td>74.35%</td>
<td>81.15%</td>
<td>84.82%</td>
<td>86.91%</td>
<td>89.53%</td>
</tr>
<tr>
<td>dix</td>
<td>75.16%</td>
<td>86.34%</td>
<td>90.06%</td>
<td>93.79%</td>
<td>96.27%</td>
</tr>
<tr>
<td>onze</td>
<td>44.74%</td>
<td>55.26%</td>
<td>57.89%</td>
<td>68.42%</td>
<td>73.68%</td>
</tr>
<tr>
<td>douze</td>
<td>36.59%</td>
<td>46.34%</td>
<td>48.78%</td>
<td>56.10%</td>
<td>60.98%</td>
</tr>
<tr>
<td>treize</td>
<td>26.32%</td>
<td>39.47%</td>
<td>42.11%</td>
<td>52.63%</td>
<td>55.26%</td>
</tr>
<tr>
<td>quatorze</td>
<td>57.50%</td>
<td>77.50%</td>
<td>87.50%</td>
<td>90.00%</td>
<td>92.50%</td>
</tr>
<tr>
<td>quinze</td>
<td>81.48%</td>
<td>92.59%</td>
<td>94.44%</td>
<td>96.30%</td>
<td>96.30%</td>
</tr>
<tr>
<td>seize</td>
<td>44.00%</td>
<td>60.00%</td>
<td>68.00%</td>
<td>70.00%</td>
<td>72.00%</td>
</tr>
<tr>
<td>vingt</td>
<td>82.93%</td>
<td>91.06%</td>
<td>93.50%</td>
<td>94.85%</td>
<td>95.66%</td>
</tr>
<tr>
<td>trente</td>
<td>66.90%</td>
<td>76.06%</td>
<td>83.10%</td>
<td>88.03%</td>
<td>91.55%</td>
</tr>
<tr>
<td>quarante</td>
<td>72.60%</td>
<td>83.56%</td>
<td>85.62%</td>
<td>86.99%</td>
<td>89.73%</td>
</tr>
<tr>
<td>cinquante</td>
<td>85.52%</td>
<td>89.66%</td>
<td>91.03%</td>
<td>93.79%</td>
<td>95.86%</td>
</tr>
<tr>
<td>soixante</td>
<td>85.47%</td>
<td>91.03%</td>
<td>93.16%</td>
<td>94.02%</td>
<td>95.73%</td>
</tr>
<tr>
<td>cent</td>
<td>90.11%</td>
<td>93.46%</td>
<td>95.41%</td>
<td>96.29%</td>
<td>96.55%</td>
</tr>
<tr>
<td>mille</td>
<td>88.55%</td>
<td>92.93%</td>
<td>94.28%</td>
<td>95.62%</td>
<td>96.97%</td>
</tr>
<tr>
<td>francs</td>
<td>81.10%</td>
<td>92.66%</td>
<td>97.06%</td>
<td>98.72%</td>
<td>99.17%</td>
</tr>
<tr>
<td>et</td>
<td>95.97%</td>
<td>98.17%</td>
<td>99.63%</td>
<td>100.00%</td>
<td>100.00%</td>
</tr>
<tr>
<td>centimes</td>
<td>83.70%</td>
<td>88.59%</td>
<td>91.03%</td>
<td>92.12%</td>
<td>95.11%</td>
</tr>
<tr>
<td>Average</td>
<td>82.50%</td>
<td>89.56%</td>
<td>92.72%</td>
<td>94.57%</td>
<td>95.74%</td>
</tr>
</tbody>
</table>

Table 2: Detailed recognition results after corrective training.
<table>
<thead>
<tr>
<th>Method</th>
<th>top 1</th>
<th>top 2</th>
<th>top 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analytic chain</td>
<td>78.4%</td>
<td>87.6%</td>
<td>93.6%</td>
</tr>
<tr>
<td>Holistic chain</td>
<td>77.1%</td>
<td>88.0%</td>
<td>95.0%</td>
</tr>
<tr>
<td>Combination (neural net)</td>
<td>87.1%</td>
<td>94.5%</td>
<td>97.8%</td>
</tr>
</tbody>
</table>

Table 4: Recognition results obtained by the A2iA company.

Figure 8: Examples of misrecognized words.
3.4 Complexity and Implementation Issues

Next, we will be concerned with the estimation of the complexity during training and recognition. For the first phase, it is obvious that the pixel probability re-estimation step is the most time consuming. We will focus on the complexity of this step for a given word sub-model corresponding to a particular image scan. Let us denote by $T$ the number of training steps, by $K$ the number of samples and by $\pi$ the average sample length. Following equation (5), $B$ re-estimation requires $O(T \times K \times \pi \times m \times N)$ multiplications and $O(T \times K \times \pi \times m \times N \times 2^{P+1})$ additions ($m$ is the number of lines and $N$ the number of states). During recognition, one needs to compute the pattern likelihood using equation (4) which can be done in $O(n \times m \times N)$ multiplications if $n$ represents the length of the input pattern. Note the linearity in the number of states because of the particular left-to-right architecture. Besides, it does not depend on the order of the model since the operations performed at pixel level consist only of table lookups.

All the algorithms are written in parallel C and run on a Silicon Graphics Origin2000 computing server. This server presents a distributed shared memory architecture (8 Gb main memory) and has 64 MIPS R10000 processors, each of them being clocked at 195 MHz and providing about 250 MFlops. We choose to parallelize the training and the testing (computing of the emission probability) at a sub-model level meaning that the above mentioned operations are performed on a single processor. A master process distributes the tasks associated to the sub-models across the processors and calculates at the end the recognition results. The average recognition speed of our system is estimated at 680.72 msec per word image on a single processor, that is approximately 1.5 words per second.

4 Conclusion

In this paper we have described a two-dimensional approach to off-line handwritten word recognition in a small lexicon. The key point of this approach is the combination of HMMs and causal Markov random fields in order to efficiently model the words at a pixel level. We have seen throughout the article how the estimation of NSHP-like conditional pixel probabilities is performed by keeping a close relation with the major benefits of HMM
formalism (dynamic warping, Baum-Welch re-estimation algorithm, MLE optimization criterion, etc.). The application of these models to word recognition shows encouraging results and leaves our system open to further improvements.

References


