Improving Offline Handwritten Text Recognition with Hybrid HMM/ANN Models

S. España-Boquera, M.J. Castro-Bleda,
J. Gorbe-Moya, F. Zamora-Martínez

Departamento de Sistemas Informáticos y Computación,
Universidad Politécnica de Valencia, Valencia, Spain

sespana@dsic.upv.es, mcastro@dsic.upv.es,
jgorbe@dsic.upv.es, fzamora@dsic.upv.es

Abstract

This paper proposes the use of hybrid HMM (Hidden Markov Model)/ANN (Artificial Neural Network) models for recognizing unconstrained offline handwritten texts. The structural part of the optical models has been modeled with Markov chains, and a Multilayer Perceptron is used to estimate the emission probabilities. This paper also presents new techniques to remove slope and slant from handwritten text and to normalize the size of text images with supervised learning methods. Slope correction and size normalization are achieved by classifying local extrema of text contours with Multilayer Perceptrons. Slant is also removed in a non-uniform way by using Artificial Neural Networks. Experiments have been conducted on offline handwritten text lines from the IAM database, and the recognition rates achieved, in comparison to the ones reported in the literature, are among the best for the same task.

Keywords: Handwriting recognition, offline handwriting, hybrid HMM/ANN, HMM, neural networks, multilayer perceptron, image normalization.

1 Introduction and motivation

Offline handwritten text recognition is one of the most active areas of research in computer science and it is inherently difficult because of the high variability of writing styles. High recognition rates are achieved in character recognition and isolated word recognition, but we are still far from achieving high-performance recognition systems for unconstrained offline handwritten texts [48, 41, 1, 54, 12, 35, 23].

Automatic handwriting recognition systems normally include several preprocessing steps to reduce variation in the handwritten texts as much as possible and, at the same time, to preserve information that is relevant for recognition. There is no general solution to preprocessing of offline handwritten text lines, but it typically relies on slope and slant correction and normalization of the size of the characters. With the slope correction, the handwritten word is horizontally rotated such that the lower baseline is aligned to the horizontal axis of the image. Slant is the clockwise angle between the vertical direction and the direction of the vertical text strokes. Slant correction transforms the word into an upright position. Ideally, the removal of slope and slant results in a word image that is independent of these factors. Finally, size normalization tries to make the system invariant to the character size and to reduce the empty background areas caused by the ascenders and descenders of some letters.

This paper presents new techniques to remove the slope and the slant from handwritten text lines and to normalize the size of the text images by using Artificial Neural Networks (ANNs). Local extrema from a text image classified as belonging to the lower baseline by a Multilayer Perceptron (MLP) are used to accurately estimate the slope and the horizontal alignment. Slant is removed in a non-uniform way by also using ANNs. Finally, another MLP computes the reference lines of the slope and slant-corrected text in order to normalize its size.

Hidden Markov Models (HMMs) have been widely applied to offline handwriting recognition [19, 41, 54, 12, 35, 37, 51, 56], after their success in automatic speech recognition. The basic idea is that handwriting can be interpreted as a left-to-right sequence of ink signals, which is analogous to the temporal sequence of acoustic signals in speech. The motivation for the work on the hybrid HMM/ANN models presented here originates from

- a critical analysis of the state-of-the-art in offline handwritten text recognition [12, 51, 56],
- our earlier work on offline handwriting recognition using conventional HMMs [25],
- our own and others' experience in using hybrid HMM/ANN models for automatic speech recognition [3, 10, 17, 18, 24] and for online handwriting recognition [6, 45, 29, 39, 16, 26],
- and previous works which used hybrid HMM/ANN word models with remarkable success, although they were limited to digit recognition or small vocabulary tasks [33, 46, 32].

Hybrid HMM/ANN models compute the emission probabilities for the HMMs with a neural network instead of the commonly used Gaussian mixtures. This work is the first successful attempt, to the best of our knowledge, to use hybrid HMM/ANN models in unconstrained offline handwritten text recognition.

In many other works, artificial neural networks have been extensively applied to classify characters as part of isolated or continuous handwritten word recognizers [14, 13, 34, 50, 36].

In our experiments with hybrid HMM/ANN models, left-to-right Markov chains have been used to model graphemes, and a single neural network has been used to estimate the emission probabilities. The estimates of the posterior probabilities computed by the neural network are divided by the prior state probabilities resulting in scaled likelihoods which are used as emission probabilities in the HMMs.

We have conducted experiments with the "large writer-independent text line recognition task" of the IAM database [38, 27] using our preprocessing and conventional HMMs as optical models. This baseline experiment achieves comparative performance with state-of-the-art systems (38.8% of word error rate). Next, experiments with our hybrid HMM/ANN system were performed and excellent results were achieved improving our baseline by a relative word error rate reduction of more than 42% (from a word error rate of 38.8% to 22.4%).

Section 2 introduces our approach to handwritten text preprocessing, showing illustrative examples. Section 3 describes the proposed hybrid HMM/ANN recognition system. Experimental setup is detailed in Section 4 and recognition results for the HMM and HMM/ANN systems with the IAM line task are shown in Section 5. Analysis of the experiments and comparison with other approaches are presented in Section 6. Finally, the conclusions are drawn in Section 7.

2 Preprocessing and feature extraction

Handwritten image normalization from a scanned image includes several steps, which usually begin with image cleaning, page skew correction and line detection [37]. A database of skew-corrected lines has been used in all the experiments [38], thus page skew correction and line detection are skipped in this work. With the handwritten text line images, several preprocessing steps to reduce variations in writing style are usually performed: slope and slant removal and character size normalization. This paper presents new techniques to remove the slope and the slant from handwritten text lines, and to normalize the size of the text images with ANNs. Table 1 resumes the key ideas of the MLPs which are used for preprocessing. These MLPs are described in more detail in Section 4.

2.1 Image cleaning

Before any other preprocessing step, the text line scanned image is first cleaned and enhanced. Neural networks have been used in previous works for image restoration by learning the appropriate filters from examples [36]. Similarly, we have used a neural network filter to clean and enhance the handwritten text images by estimating the gray level of one pixel at a time [28]: an MLP (from now on called Enhancer-MLP) is fed with a square of pixels centered at the

Table 1: MLPs for preprocessing. MLPs are used for regression (Enhancer-MLP) and for classification (Slope-MLP, Normalize-MLP, and Slant-MLP).

Task	MLP	Input to MLP	Output to MLP	MLP topology
Image cleaning	Enhancer-MLP	Pixels in a window around current pixel	Restored value of current pixel	$(11\times11)\rightarrow32\rightarrow16\rightarrow1$
Slope removal	Slope-MLP	Pixels in a window around current pixel	Lower baseline/Not lower baseline pixel	$(50\times30)\rightarrow64\rightarrow16\rightarrow2$
Slant removal	Slant-MLP	Resized sheared image in a window around current column of pixels	Presence of slant angle	$(40\times40)\rightarrow64\rightarrow8\rightarrow1$
Size normalization	Normalize-MLP	Pixels in a window around current pixel	Ascender/Upper baseline/Lower baseline/ Descender/None pixel	$(50\times30)\rightarrow64\rightarrow16\rightarrow5$

pixel to be cleaned, and the output is the restored value of the pixel. The entire image is cleaned by scanning all the pixels in this way. A scheme of this process is illustrated in Figure 1. A real example of a cleaned image is shown in Figure 2(b).

2.2 Slope removal

With the skew-corrected lines, most handwriting recognition systems require the detection of the different areas of the cursive script: the main body area (area between the upper baseline and the lower baseline), the ascenders, and the descenders (see Figure 3 for an example). These areas can be detected by means of horizontal histogram projection [15, 11, 55] or also by obtaining the upper and lower contours of the image [57, 43]. Instead of relying on these geometric heuristics, our approach consists in automatically detecting a set of points from the image and classifying them by supervised machine learning techniques [47, 25]. The points to be classified are local vertical extrema of text contours which are used to determine the reference lines: line of ascenders, upper baseline, lower baseline, and line of descenders (see Figure 3). These reference lines provide an efficient way to perform slope removal and size normalization.

In a first step, and once the text line image has been cleaned, the local extrema are obtained. First, a vertical contour of the image is extracted (see Figure 2(c)). After that, the contour points are grouped into lines following a proximity criterion: two pixels on adjacent columns are considered to belong to the same line when the difference between their vertical coordinates is less than 3. Finally, the maxima of the upper contours and the minima of the lower contours are computed.

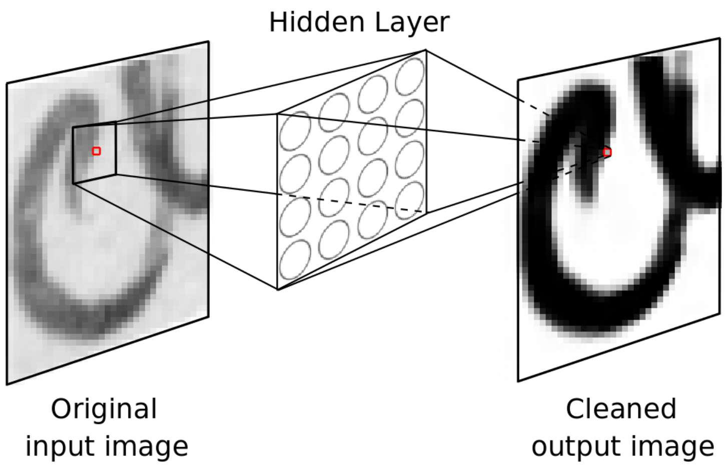

Figure 1: Enhancer-MLP: An MLP to enhance and clean images. The entire input image is cleaned by scanning it with the neural network.

Since the lower baseline suffices to correct the slope, an MLP that is trained to classify local extrema as belonging or not belonging to the lower baseline is used (this MLP will be called Slope-MLP). The input to this Slope-MLP is a window that is centered at the pixel to be classified.

Once the lower baseline points have been detected, the image is horizontally divided into segments in order to apply the slope correction to every segment. A vertical histogram projection is used to estimate the mean space width between ink regions, and this value is used as a threshold to split the image into segments. For each of these segments, the lower baseline is estimated by means of least squares fitting of the lower baseline points. An example of the splitting process of the estimated lower baseline is shown in Figure 2(d). These lower baselines are used to correct the slope and the vertical relative positions of the segments. Figure 2(e) illustrates an example of a slope-corrected image.

2.3 Slant removal

After the slope correction, slant is removed by means of a two-step method. In the first step, a global slant angle is estimated and removed by performing a shear operation to the image for every integer angle between an interval (in this case, [-50°,50°]) and choosing the sheared image whose vertical projection has the maximum variance [40]. In the second step, a novel non-uniform local slant correction method is applied: an MLP (from now on called Slant-MLP) is trained to estimate if a given column of the text line image is slanted. Using a sliding input window, the Slant-MLP is applied to every column of the image with some additional columns of context, for each integer angle in [-50°,50°]. This procedure generates a matrix which contains the estimated score of cor-

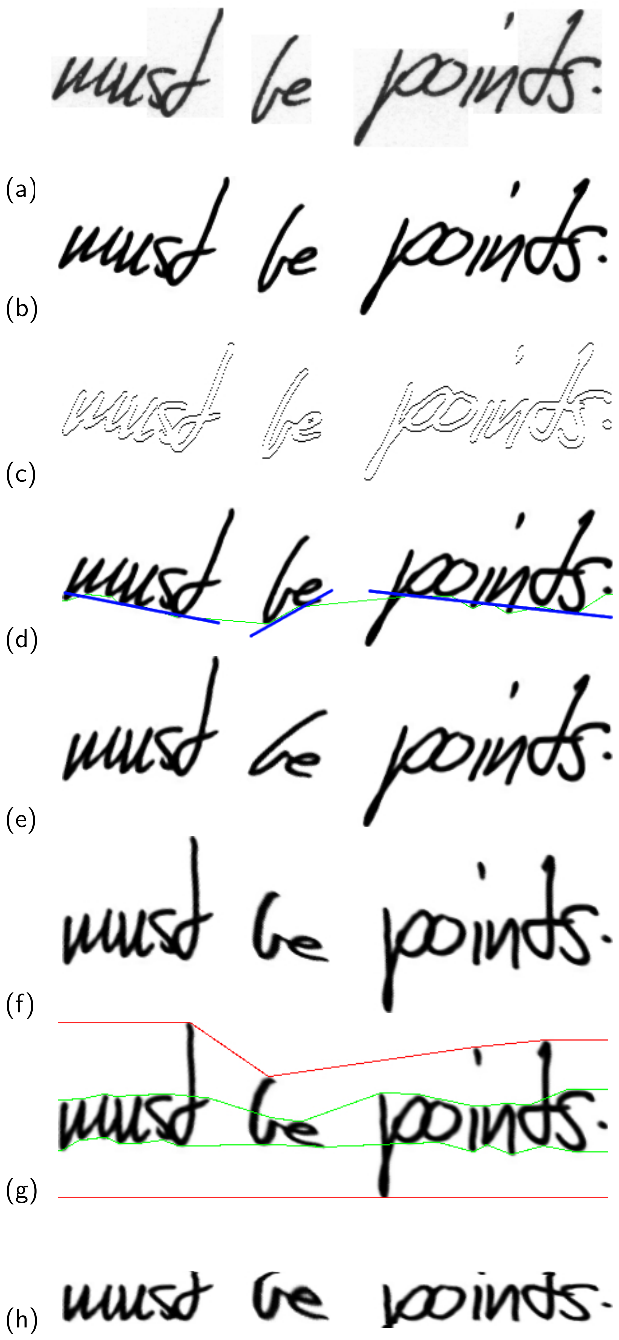

Figure 2: Preprocessing example. From top to bottom: (a) Original image from the IAM database. (b) Cleaned image. (c) Text contour extracted to obtain local extrema. (d) Local extrema classified by an MLP as lower baseline to be used for slope correction. (e) Slope-corrected image. (f) Slant-corrected image. (g) Local extrema labeled by an MLP as belonging to the four reference lines to be used for size normalization. (h) Normalized final image.

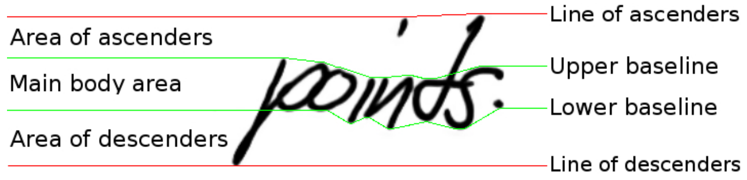

Figure 3: Example of text line image with the different areas (ascenders, descenders, and main body areas) and the reference baselines (upper and the lower baselines, and the lines of ascenders and descenders) of the script.

recting each column with each slant angle. Finally, a dynamic programming algorithm is applied over this matrix [52] to obtain the path with maximum score which also satisfies a smoothness criterion (the slant angle must not change more than $\pm 1°$ per column). This sequence of angles conforms the input to a non-uniform shear that generates the final slant corrected image. The total computational cost is linear with the size of the image and the number of shear angles considered. The whole slant removal process is illustrated in Figure 4 and an example of a slant-corrected image is shown in Figure 2(f).

2.4 Size normalization

When the image is slope and slant-corrected, the size of the text line is normalized in order to minimize the variations in size and position of its three zones (main body area, ascenders, and descenders). Furthermore, the normalized size of ascenders and descenders is reduced with respect to the body since they are not as informative (the presence or absence of ascenders and descenders is preserved, as well as the width, but not the actual height).

One approach to size normalization consists of detecting the reference baselines and to normalize the size according to them [6, 47, 25, 44]. Following this idea, our size normalization method detects and classifies the local extrema using the same method based on ANNs described in Section 2.2. This time, local extrema are classified into five classes (the four reference lines and points not belonging to any of these lines) by using another MLP (Normalize-MLP). Points belonging to the same class are used to obtain each reference line by linear interpolation (see Figure 2(g)). These lines comprise the three zones to be normalized. The normalization process is performed for each column of the image by linearly scaling the three zones to a fixed height. Ascenders are reduced to 20% of the final image height and descenders are reduced to 10%. See Figure 2(h) for an example of a normalized image.

2.5 Feature extraction

After preprocessing, a feature extraction method is applied to capture the most relevant characteristics of the character to recognize. In our systems, a hand-

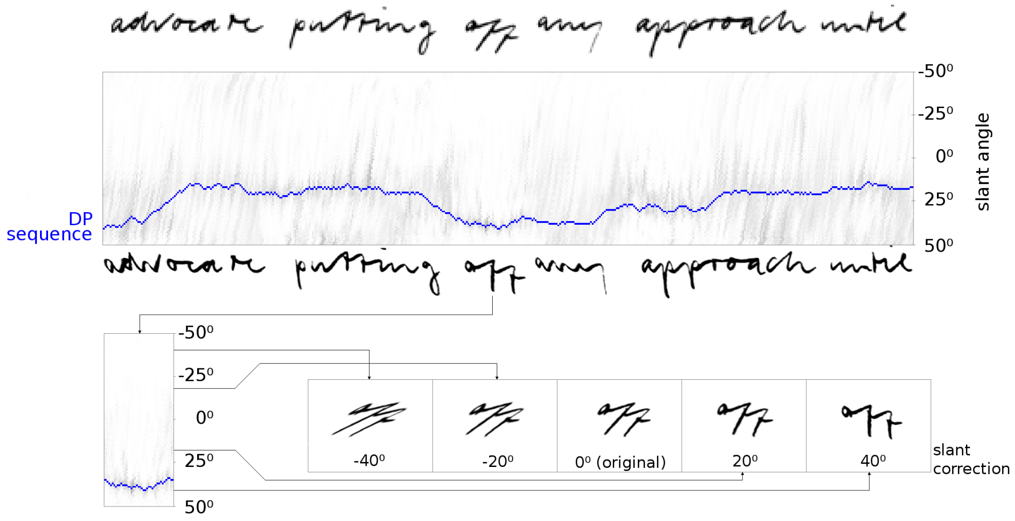

Figure 4: An example of slant removal: the original text line image (top) and the slant-corrected text line image (bottom). The Slant-MLP estimates a measure of the slant angle of each pixel column, shown as a grey level matrix. A dynamic programming (DP) algorithm obtains the optimal sequence of slant angles. Beneath the matrix is a detail of it for a segment of the text line image.

written text line image is converted into a sequence of fixed-dimension feature vectors. Following [51], features are extracted by applying a grid to the image and computing three values for each cell of the grid: the normalized gray level and the horizontal and vertical gray level derivatives. A grid of square cells with 20 rows has been used so every feature vector is composed of 60 values. An example of a graphical representation of the features obtained in this way is shown at the top of Figure 5.

3 Hybrid HMM/ANN modeling

For small vocabulary handwriting recognition tasks (for example, check amounts or postal addresses), it is possible to model words individually. But for large vocabulary or even unconstrained tasks, the only feasible approach is to recognize individual graphemes and map them onto complete words belonging to a fixed vocabulary Ω. The same problem has to be addressed for automatic speech recognition, and HMMs have been accepted as the standard solution [42]. For offline handwritten text recognition, the image is converted into a sequence $X = (x_1 \dots x_m)$ of feature vectors and, under the statistical approach to pattern recognition [42, 30], the goal of general handwritten text recognition is to find the likeliest word sequence $W^\star = (w_1 \dots w_n)$ maximizing the a-posteriori probability:

$$W^\star = \underset{W \in \Omega^+}{\operatorname{argmax}}\ P(W|X) \ . \tag{1}$$

The application of the Bayes rule leads to a decomposition of $P(W|X)$ into the optical model $P(X|W)$ and the statistical language model $P(W)$. The problem can then be reformulated as:

$$W^\star = \underset{W\in\Omega^+}{\operatorname{argmax}}\ P(X|W)P(W) \ . \tag{2}$$

In state-of-the-art handwritten text recognition systems, $P(X|W)$ is usually estimated by a HMM-based recognizer and a word n-gram language model is usually used to approximate $P(W)$. Typically, each grapheme is modeled by a left-to-right HMM and the number of states is chosen globally or individually for each grapheme. Gaussian mixtures are used to model the output distributions in each state q given the feature vector x, $P(x|q)$. The Baum-Welch algorithm is used for training the HMMs, whereas the Viterbi algorithm is used for recognition.

3.1 The hybrid HMM/ANN approach

In the hidden Markov modeling approach, the emission probability density $P(x|q)$ must be estimated for each state q of the Markov chains, that is, the probability of the observed feature vector x given the hypothesized state q of the model. In the proposed hybrid HMM/ANN approach, the emission probabilities are provided with a neural network since ANNs can be trained to estimate probabilities that are related to these emission probabilities. In particular, an MLP can be trained to approximate the a-posteriori probabilities of states, $P(q|x)$, if each MLP output unit is associated with a specific state of the model and if it is trained as a classifier [10, 9]. In order to obtain such distribution for every state q from the set Q of Markov chain states, the softmax activation function has been chosen at the output layer:

$$f(y_q) = \frac{\exp{(y_q)}}{\sum_{i\in Q}\exp{(y_i)}} \tag{3}$$

where y_q is the q-th output value before applying the softmax function. This activation function enables the estimation of valid probability values, i.e., to lie between zero and one and to sum to one.

The a-posteriori probability estimates from the MLP outputs, $P(q|x)$, can be converted to emission probabilities $P(x|q)$ by applying Bayes rule:

$$P(x|q) = \frac{P(q|x)P(x)}{P(q)} \ . \tag{4}$$

The class priors $P(q)$ can be estimated from the relative frequencies of each state from the information produced by a forced Viterbi alignment of the training data. Thus, the scaled likelihoods $P(x|q)/P(q)$ can be used as emission probabilities in the proposed system, since, during recognition, the scaling factor $P(x)$ is a constant for all classes [10]. This allows MLPs to be integrated into hybrid structural-connectionist models via a statistical framework.

The advantages of this approach are the discriminate training criterion (all MLP parameters are updated in response to every input feature vector) and the fact that it is no longer necessary to assume an a-priori distribution of the data. Furthermore, if left and right contexts are used at the input of the MLP, important contextual information can be incorporated into the probability estimation process.

Another strength of this approach is that computing emission probabilities with hybrid HMM/ANN models is usually faster than conventional HMMs with Gaussian emissions since it only requires a forward-pass of the MLP for all states of the Markov chains.

On the other hand, one of the weaknesses of this hybrid HMM/ANN approach is that every feature vector must be labeled to train the MLP. However, this is not a serious drawback since these labels can be generated by running a previous trained handwriting recognition system in a forced alignment mode in order to initialize these labels.

In our experiments, we modeled graphemes with left-to-right Markov chains and a single neural network with one output unit for each state of the Markov chains was used to estimate the distribution probabilities. An MLP with sigmoid hidden units and softmax output units is used. The estimates of the posterior probabilities computed by the MLP are divided by the prior state probabilities resulting in scaled likelihoods which are used as emission probabilities in the HMMs. A scheme of the proposed hybrid HMM/ANN recognition system is shown in Figure 5.

3.2 Training the HMM/ANN models

The training of the MLP is discriminant at the state level of Markov chains, since each output is optimized during training by samples of its own class as well as by samples of the other classes. Training of the whole hybrid HMM/ANN system is done by an iterative Expectation-Maximization algorithm where the training of the supervised ANN and either Baum-Welch or Viterbi alignment of the training corpus are alternated. We have opted for Viterbi alignment and the training procedure proceeds as follows:

1. Assign an initial labeling of desired MLP outputs to every feature vector of the training and validation datasets. This labeling can be computed by dividing the image into equal parts or by using a previously trained handwritten recognition system in a forced alignment mode.

2. Assign an initial non-zero value to transition probabilities of the Markov chains.

3. Train the supervised ANN with the training pairs, using the mean square error on the validation dataset as the stopping criterion.

4. Use the partially trained hybrid ANN/HMM models to perform a forced Viterbi alignment of the training data. This Viterbi procedure uses the

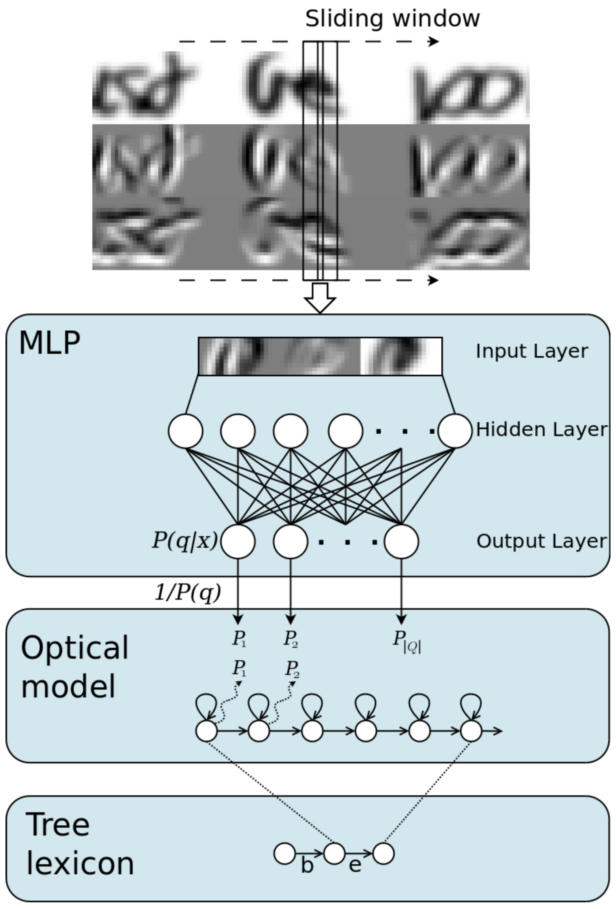

Figure 5: A scheme of the proposed hybrid HMM/ANN recognition system. First, the image is preprocessed (see Figure 2) and the resulting feature vector, plus a left and right context, is processed by an MLP. The $|Q|$ outputs of the MLP (after dividing by the prior state probabilities) are used as emission probabilities in the HMMs.

class priors estimated from the relative frequencies of each class in the training data. This Viterbi alignment is used for both obtaining a new segmentation or labeling of the training and validation sets and also for counting the number of times each HMM transition has been used. These counts are used to reestimate the transition probabilities.

5. Go to step 3 until convergence, that is, until the difference between two consecutive iterations is below a threshold.

4 Experimental setup

4.1 The IAM database

All experiments reported in this paper are conducted on handwritten text lines from the IAM database [38]. The version 3.0 of this database includes over 1 500 scanned forms of handwritten text from more than 650 different writers, for a total of more than 13 000 fully transcribed handwritten lines, without restrictions on the writing style or the writing instrument used. The sentences have been extracted from the Lancaster-Oslo/Bergen (LOB) text corpus [31].

A writer-independent text line recognition task has been considered. The subset of the IAM database used in this work consists of 6 161 training lines (from 283 writers), 920 validation lines (56 writers) and 2 781 test lines (161 writers). All these data sets are disjoint, and no writer has contributed to more than one set. These partitions are the same as those used in several works by Bunke et al [27, 7, 8].

A total of 87 967 instances of 11 320 distinct words occur in the union of the training, validation, and test sets. Lexicon is modeled with 78 characters: 26 lowercase letters, 26 uppercase letters, 10 digits, 14 punctuation marks, the space, and a character for garbage symbols.

4.2 MLP for image cleaning

As described in Section 2.1, an MLP has been used for image cleaning by learning the appropriate filter from examples.

Training data. Original noisy images from the IAM database and the same images that were cleaned by hand formed the training pairs. Additionally, artificially noised images (created by following the ideas presented in [28]) were also used as training data.

Enhancer-MLP. In this case, the MLP is used for regression: the input is a fixed-sized moving window of 11×11 pixels centered at the pixel to be cleaned, and the output is the restored value of the current pixel (see Figure 1). The Enhancer-MLP has two hidden layers of 32 and 16 sigmoid units and one output linear unit. Training was performed using the stochastic version of the backpropagation algorithm with momentum term [9], using the mean square error (MSE) function. Last column of Table 1 shows the topology of the MLPs which are used for preprocessing.

4.3 MLPs for slope removal and size normalization

As pointed out in Section 2.2 and 2.4, two MLPs to classify local extrema as belonging to one of the reference lines (lower line, upper line, line of descenders, or line of ascenders) are needed as part of the slope removal and image size normalization processes.

Training data. We needed supervised training patterns to train MLPs to classify interest points as belonging to the reference lines. A subset of 1 000 images from the IAM training set has been used. Local extrema of the 1 000 images were semi-automatically labeled using an active learning approach: first, a horizontal projection algorithm was used to classify the points belonging to each reference line of a subset of the 1 000 images; second, the subset of images was manually corrected using a graphical tool designed to this purpose [25]; third, these images were used to train an MLP to classify interest points. With this "pretrained" MLP, interest points of the 1 000 images were automatically classified, and, afterwards, all the images were manually supervised. At the end of this process, we had a training set composed of the interest points of the 1 000 images: 800 lines were used as training data and the remaining 200 lines were used as validation data.

Slope-MLP. The Slope-MLP was trained to classify local extrema as belonging to or not belonging to the lower baseline. The Slope-MLP input is a moving window around the current pixel, being the choice of an appropriate window size a trade-off between context and input size. To partially overcome this problem, we have opted to use a fisheye distorsion centered at the pixel to classify (see Figure 6 for an example) [25]. The fisheye distortion maintains a very accurate resolution near the center and, at the same time, has a much smaller size than using the original image. In this way, a detailed image near the interest point and a coarse representation of the relative position of the surrounding text is obtained: the input to this Slope-MLP is a window of 500×250 pixels centered at the point to be classified, downsampled to 50×30 values using the fisheye distortion. Two output units with a softmax activation function were used to determine whether or not the current pixel belonged to the lower baseline. After doing a scanning of topologies, two hidden layers of 64 and 16 sigmoid units were used.

Normalize-MLP. Size normalization was achieved by using a second MLP, which classifies the local extrema into five classes (the four reference lines and points not belonging to any of these lines). The input to this Normalize-MLP is the same as the Slope-MLP input and the output corresponded to five output units with softmax activation function. We used two hidden layers of 64 and 128 sigmoid units.

Both MLPs, Slope-MLP and Normalize-MLP, were trained using the stochastic version of the backpropagation algorithm with momentum term and the cross entropy error function.

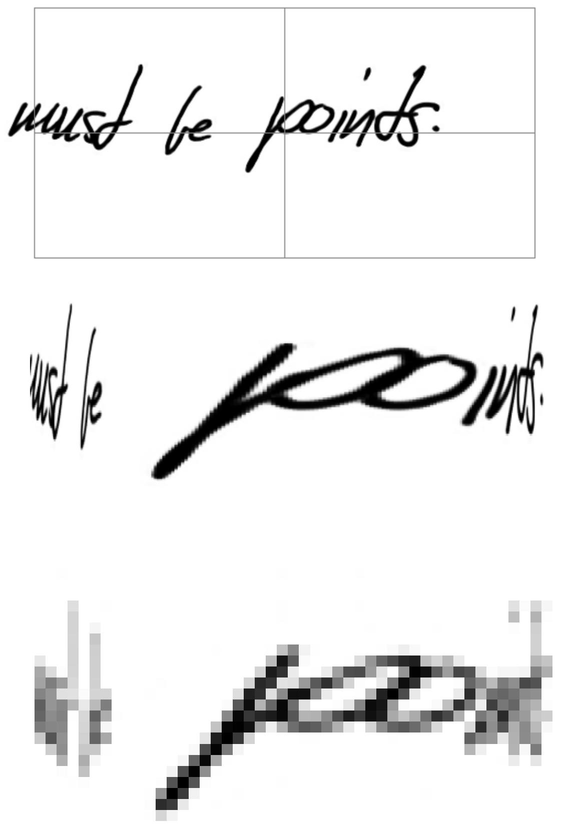

Figure 6: Fisheye lens example (from up-to-down): original image of 500×250 pixels centered at the pixel to be classified; the same image with a fisheye lens distortion; the image downsampled to 50×30 values used by the neural network classifier.

4.4 MLP for slant removal

As described in Section 2.3, part of the process of slant removal needs an MLP to determine whether or not an image has slant.

Training data. The same set of 1 000 images was manually slant-corrected in a non-uniform way by using a graphical tool. The user specifies a series of slant angles which are interpolated for every image column. This information is used to train the Slant-MLP. As before, 200 images were used for validation.

Slant-MLP. Each image is sheared for different integer angles from -50° to +50° and resized to 40 pixels height, preserving the aspect ratio. The input to this Slant-MLP is a square of 40×40 pixels centered at the column to be evaluated, and the output is a measure of the local slant presence (shown as gray levels in Figure 4). After doing a parameter and topology scanning, two hidden layers of 64 and 8 units were used. Training was performed using the stochastic version of the backpropagation algorithm with momentum term, using the mean square error function.

5 Experiments

5.1 Dictionary and language model

A word bigram language model was trained with three different text corpora: the LOB corpus [31] (excluding those sentences that contain lines from the test set of the IAM database), the Brown corpus [22], and the Wellington corpus [2]. In order to cope with the fact that lines are fragments of sentences, we have randomly broken each sentence from the corpus into fragments to resemble lines. All this text is supplemented with the training lines from the IAM database. Then, the final training material comprises:

- Sentences: 51 560 LOB sentences (2 134 sentences which contained IAM test lines were eliminated), 51 763 Brown sentences, and 20 592 Wellington sentences.
- Fragments of sentences to resemble lines: more than 400 000 lines randomly obtained from the above set of sentences.
- Lines: finally, the 6 161 IAM training lines were also added.

The bigram language model used in the recognition systems was generated using the SRI Language Modeling Toolkit [49] with the modified Kneser-Ney back-off discounting.

To achieve unconstrained handwriting recognition, an open dictionary composed of the 20 000 most frequently occurring (case insensitive) words in the training material was used to test our recognition systems.

Table 2: Tuning the number of states of the HMMs: Word Error Rate of the HMMs on the validation set.

Model	WER on Validation (%)
6-state HMMs	35.1 ±1.6
7-state HMMs	34.5 ±1.6
8-state HMMs	32.9 ±1.5
9-state HMMs	33.3 ±1.5
10-state HMMs	35.0 ±1.5
11-state HMMs	37.3 ±1.5
12-state HMMs	40.0 ±1.6

5.2 Measuring recognition performance

The recognition performance was measured in terms of the Word Error Rate (WER), which is computed by comparing the output of the recognizer with the reference transcription. WER is defined as the number of word errors (insertions, substitutions and deletions) summed over the whole test set and divided by the total number of words in the transcriptions of the reference set:

$$\mathrm{WER} = 100 \times \frac{\text{insertions} + \text{substitutions} + \text{deletions}}{\text{total number of words}}. \quad (5)$$

A null WER is only reached if the recognizer output matches the reference transcription exactly.

The Character Error Rate (CER) was also measured for the final test experiments. CER is defined as expression (5), but with characters instead of words.

In order to properly compare different systems, it is highly desirable to provide not only the value of the WER (or CER) but also a confidence interval for it. In [53], the author proposes a method for computing these intervals without simulations. Following his work, we have computed the confidence interval for every experiment with the IAM validation and test sets, which are composed of 920 and 2 781 lines, respectively. In every experiment, the computed intervals correspond to a 95% confidence level.

5.3 Baseline experiments: HMMs

Baseline recognition HMM experiments were conducted using continuous density HMMs with diagonal covariance matrices of 64 Gaussians in each state, and with a left-to-right topology without skips. The 78 optical models were trained and tested with the HTK toolkit [58].

The validation set of the IAM database was used to optimize the number of states of the optical HMMs and the integration of the statistical language model. A bigram language model and an open dictionary were used as explained in

Table 3: Tuning the GSF: Word Error Rate of the best HMMs on the validation set for different Grammar Scale Factors.

Model	GSF	WER on Validation (%)
8-state HMMs	30	35.6 ±1.7
8-state HMMs	35	33.9 ±1.6
8-state HMMs	40	32.9 ±1.5
8-state HMMs	45	32.9 ±1.5
8-state HMMs	50	32.8 ±1.4
8-state HMMs	55	33.3 ±1.4
8-state HMMs	60	33.8 ±1.4

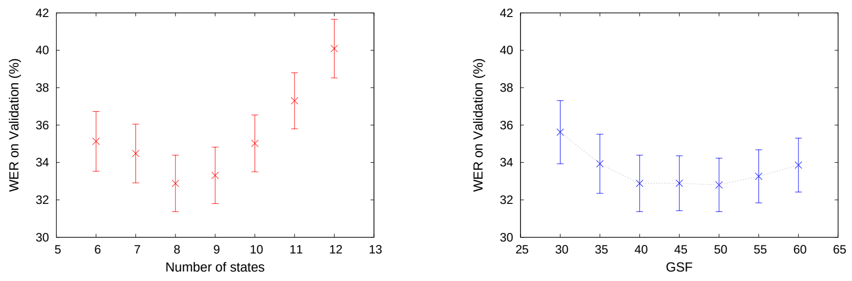

Figure 7: Tuning HMMs: Word Error Rate of the HMMs on the validation set varying the number of states (left) and for different Grammar Scale Factors (right). WER is given with a 95% confidence interval.

Section 5.1.[1] Table 2 summarizes the experiments varying the number of states of the HMMs and Figure 7 plots the word error rate with a 95% confidence level interval. From these figures, it can be observed that several configurations achieved equivalent statistical performance, the 8-state HMMs being the best topology.

To compensate for scale differences between the likelihood values $P(X|W)$ from the HMMs and the probabilities $P(W)$ provided by the language model in equation (2), a grammar scale factor is used to weight the influence of the bigram language model against the optical model. The grammar scale factor used for these experiments was fixed to 40. Afterwards, the grammar scale factor was optimized by systematically testing values from 20 to 60 on the 8-state HMMs. Table 3 shows the performance of this experiment on the validation set using bigrams. It can be observed that performance is almost identical between a

[1]Those LOB sentences which contained IAM validation lines were also excluded to estimate the bigram language model for the tuning experiments.

grammar scale factor of 40 and 50, with the lowest word error rate of 32.8% being achieved with a grammar scale factor of 50. All these results are also plotted in Figure 7 with a 95% confidence level interval.

5.4 Experiments with hybrid HMM/ANN models

Hybrid HMM/ANN models with a different number of states and different topologies and parameters of MLP were tested. In all cases, the MLP input consisted of 9 consecutive feature vectors (the central feature vector and a context of four vectors at each side). The softmax outputs (after being divided by the prior state probabilities) were used as emission probabilities of the states of the 78 optical models. Thus, we trained fully connected MLPs of 540 input units (the 60-dimensional nine feature vectors). The number of output units is determined by the total number of states of the 78 optical models (from 78×6 output units for 6-state HMMs to 78×9 output units for 9-state HMMs), since each output unit of the MLP is related to one state of the HMMs. The number of hidden units was determined empirically by measuring the MSE on the validation set. Other parameters such as the learning rate and the momentum term were also empirically tuned with the validation data.

Training was performed using stochastic backpropagation with momentum and the mean square error function. In order to monitor the generalization performance during learning and to stop training when there was no longer an improvement, the validation set was used. More than five million training patterns (corresponding to the 6 161 training lines) and close to 800 000 validation patterns (from 920 lines) composed the training and validation datasets, respectively. Due to the large time requirements to train the MLPs, we used a resampling algorithm: only 300 000 training patterns and 200 000 validation patterns were used in each training epoch. These subsets were randomly selected in each run.

The emission probabilities are obtained by dividing the a-posteriori probability estimates from the MLP outputs by the class priors. The a-priori probabilities of the states are estimated from the relative frequencies of each state, which are computed from the segmentation given by a forced Viterbi alignment of the training data.

The initial segmentation at state level that is required to train an MLP at the first step of the EM algorithm (see Section 3.2) was generated by running a "pretrained" hybrid HMM/ANN handwriting recognition system in a forced alignment mode. The word error rate on the validation set, using a closed dictionary from the IAM task composed of 10 353 words and a bigram language model estimated with the LOB corpus, was used as the stopping criterion. Around ten iterations of the EM training algorithm were enough for all the configurations. Figure 8 illustrates a typical evolution of the EM training, showing the evolution of the mean square error on the training and validation set. The evolution of the WER on validation is also shown in the graphic. The hybrid HMM/ANN models were trained and tested with the APRIL toolkit [21], which was developed for neural networks and pattern recognition tasks in our research group.

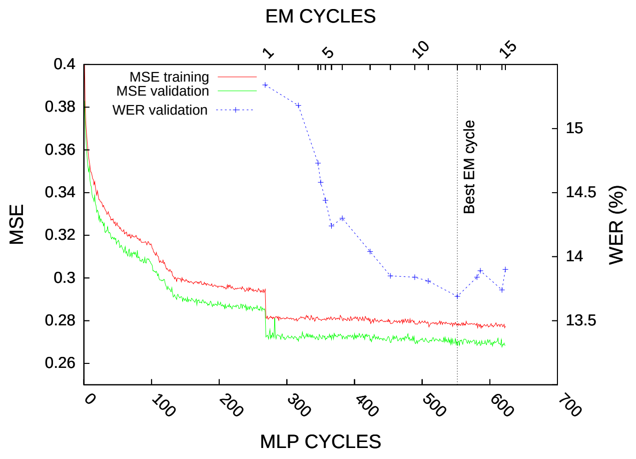

Figure 8: Evolution of the EM training algorithm for the 7-state HMMs and an MLP of two hidden layers of 192 and 128 units each. The mean square error (MSE) is shown for the training and validation data, along with the WER on validation of each iteration of the EM algorithm.

Table 4 shows the performance achieved for the different configurations of the hybrid HMM/ANN systems on the validation dataset using the bigram language model and the open dictionary as explained in Section 5.1. The lowest WER was obtained by using 7-state Markov chains and an MLP topology of two hidden layers of 192 and 128 units, respectively. Further experiments were conducted with the optimized configuration: Table 5 shows the performance of this hybrid HMM/ANN system on the validation dataset using the bigram language model with different grammar scale factors. The grammar scale factor has been optimized by systematically testing values from 6 to 16 on the validation text lines. Small changes in word error rate are observed, and the best performance, 19.0%, was achieved with a grammar scale factor of 10 or 12. All these word error rate results are plotted in Figure 9 with a 95% confidence level interval.

6 Discussion and Comparison

This section describes the performance of the optimized systems on the test set, and a comparison is made with the best published results. Experiments to study the influence of the dictionary on the recognizer were also carried out.

Table 4: Tuning the topology of the hybrid HMM/ANN models: Word Error Rate of the hybrid HMM/ANN models on the validation set.

Model	WER on Validation (%)
6-state HMMs, MLP 192-128	19.5 ±1.3
7-state HMMs, MLP 192-128	19.0 ±1.2
8-state HMMs, MLP 384-128	19.1 ±1.2
9-state HMMs, MLP 384-128	19.5 ±1.2

Table 5: Tuning the GSF: Word Error Rate of the best hybrid HMM/ANN models on the validation set for different Grammar Scale Factors.

Model	GSF	WER on Validation (%)
7-state HMMs, MLP 192-128	6	21.3 ±1.4
7-state HMMs, MLP 192-128	8	19.6 ±1.3
7-state HMMs, MLP 192-128	10	19.0 ±1.2
7-state HMMs, MLP 192-128	12	19.0 ±1.2
7-state HMMs, MLP 192-128	14	19.3 ±1.2
7-state HMMs, MLP 192-128	16	20.1 ±1.1

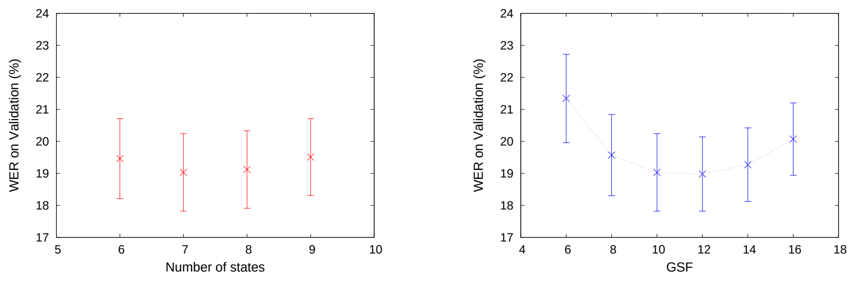

Figure 9: Tuning HMM/ANN models: Word Error Rate of the HMM/ANN models on the validation set varying the topology (left) and for different Grammar Scale Factors (right). WER is given with a 95% confidence interval.

Table 6: Testing the systems: Error Rate of the HMMs and the hybrid HMM/ANN models on the test set.

	Results of Test (%)			
Best model	WER		CER	
8-state HMMs	38.8	±1.0	18.6	±0.6
7-state HMMs, MLP 192-128	22.4	±0.8	9.8	±0.4

6.1 HMM vs. hybrid HMM/ANN models

Table 6 shows the error rate of the recognized test lines of the IAM task using the best HMM and the best hybrid HMM/ANN systems. We tested each system with the open dictionary of 20 000 words and a bigram language model, as explained in Section 5.1. For each recognition experiment, two performance figures were obtained: the word error rate and the character error rate. No parameters were optimized on the test set.

Our baseline experiment achieved comparative performances with state-of-the-art HMM systems: a WER of 38.8% ±1.0 with a 95% confidence interval and a CER of 18.6% in the interval (18.0, 19.2). Our final hybrid HMM/ANN system achieved excellent results: a WER of 22.4% in the interval (21.6, 23.2) and a CER of 9.8% in the interval (9.4, 10.2). The hybrid system outperforms our baseline in 16 points in WER, which represents a relative error rate reduction of 42%. Similarly, the character error rate improved nearly 9 points, which represents a relative percentage of improvement that is greater than 47%.

Besides the word and character error rates, another measure to consider when evaluating a recognition engine is the decoding time, since a high value might diminish the usability of the recognition system in practical applications. Unfortunately, this time is not reported by most authors. Our hybrid HMM/ANN prototype required an average time of 0.76 seconds per word for preprocessing and 0.65 seconds per word on decoding [20]. This CPU time was measured in a single core of an Intel® Core™2 Quad CPU Q6600 @ 2.40GHz using DDR2-800Mhz memory. These times could be reduced in the production stage of the recognition engine, and the latency could also be reduced by using several cores since many steps are highly parallelizable.

6.2 Influence of the dictionary

A series of experiments to study the influence of the dictionary size were carried out. Open dictionaries with between 10 000 and 30 000 words were generated by taking the N most frequently occurring words in the training material for the language model.

Table 7 shows the word error rate and the character error rate of the test set when the size of the open dictionary was increased. The second column of this table shows the test set coverage, and the last two columns show the word and

Table 7: Influence of the dictionary size (with open dictionaries): Word Error Rate of hybrid HMM/ANN models on the test set.

		Results of Test (%)			
Dictionary size	Coverage (%)	WER		CER	
10 000	66.57	26.9	±0.9	11.7	±0.5
15 000	74.28	24.2	±0.8	10.5	±0.4
20 000	78.86	22.4	±0.8	9.8	±0.4
25 000	81.97	21.9	±0.8	9.4	±0.4
30 000	84.39	21.2	±0.8	9.1	±0.4

Table 8: Influence of using closed dictionaries: Word Error Rate of hybrid HMM/ANN models on the test set.

	Results of Test (%)			
Dictionary size	WER		CER	
4 953	15.5	±0.7	6.9	±0.4
20 000	16.8	±0.7	7.5	±0.4

the character error rate of the test set using a bigram language model estimated for each lexicon. To give an idea of the meaning of coverage in the IAM line task, consider, for example, that with the open dictionary of 20 000 words, a coverage of 78.86% was achieved, that is, 21.14% of the words in the test set were out-of-vocabulary words. However, if we measure the out-of-vocabulary running-words, this figure falls to 4.99%. (The out-of-vocabulary running-words were 1 268, that is, 1 268 running-words from the 25 424 running-words in the test set were not in the lexicon.) As expected, performance increased with lexicon size and test coverage. Figure 10 plots this experiment against test set coverage.

Another experiment was also carried out to study the influence of using closed dictionaries. Two closed dictionaries were generated: one containing only the 4 953 words in the IAM test line set, and another one padding the first one up to 20 000 words (the most frequently occurring words in the training material for the language model). The influence of using the closed dictionaries is shown in Table 8 using a 95% confidence interval for WER. In this case, a bigram language model estimated for each lexicon was also used. Not surprisingly, the closed dictionary containing only the test set words achieved the best performance. The score with the 20 000 word closed dictionary was still better than those reached with open dictionaries.

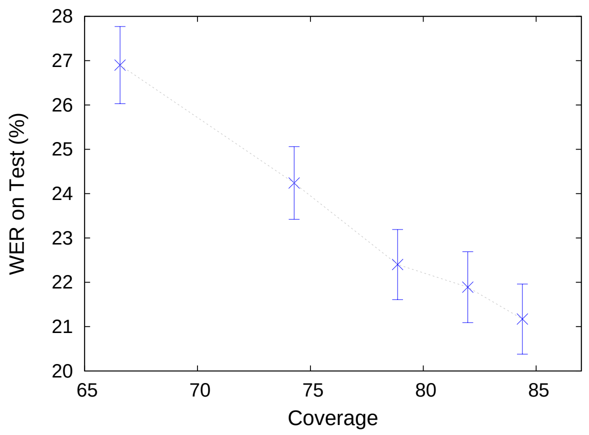

Figure 10: HMM/ANN performance with open dictionaries plotted against test set coverage. WER on test is given with a 95% confidence interval.

6.3 Comparison with other systems

Comparisons to other recognition systems in the literature are difficult due to the lack of availability of common databases. With regard to the publications using the IAM database, a more detailed comparison is possible. In a very recent work, Graves et al [27] presented a novel handwriting recognition system based on recurrent neural networks, which achieved the best published recognition rates to date, a WER of 25.9% with bigrams. In order to compare both systems, we contacted the authors to exactly reproduce the same experimental conditions. They provided us with their dictionary and language model and we retrained our optical models using the same grapheme set as them and normalizing each input feature to zero mean and unit variance. Also, case differences in words were taken into account during recognition and in the WER computation. Surprisingly, we obtained the very same 25.9% of WER for the test set, despite the fact that recurrent neural networks and HMM/ANN are very different approaches.

7 Conclusions

In this paper, we have presented a hybrid HMM/ANN system for recognizing unconstrained offline handwritten text lines. The key features of the recognition system are the novel approach to preprocessing and recognition, which are both based on ANNs. The preprocessing is based on using MLPs:

- to clean and enhance the images,
- to automatically classify local extrema in order to correct the slope and to normalize the size of the text lines images, and
- to perform a non-uniform slant correction.

The recognition is based on hybrid optical HMM/ANN models where an MLP is used to estimate the emission probabilities.

The main property of ANNs which is useful for preprocessing tasks is their ability to learn complex nonlinear input-output relationships from examples. Used for regression, an MLP can learn the appropriate filter from examples. We have exploited this property to clean and enhance the text images. Used for classification, MLPs can be used to determine the membership of interest points from the image to the reference lines, which is useful for slope correction and size normalization, and to locally detect slant in a text image. This preprocessing favouribly behaved when compared to other preprocessing techniques. We tested our HMM and HMM/ANN systems, performing the same experiments that those presented here but by using more classical techniques to correct slope, slant and size normalization [51]. We obtained a 54.3% and 29.8% test WER, respectively, which represent a percentual decrease of 29% and 25% when compared to the test results from Table 6.

The proposed hybrid HMM/ANN recognition system outperformed our baseline experiment, which is a state-of-the-art HMM-based system that includes our preprocessing. The novel hybrid HMM/ANN approach obtained an impressive 42% relative improvement in WER over our baseline. We compared our system with the recurrent neural network approach presented in [27] under the same experimental conditions and we obtained the same results.

Our next goal is to upgrade our recognition engine by using ensembles of MLPs [9, 18], by combining several recognizers [8, 59], and by using deep connectionist architectures [5, 4]. The first very basic idea is to use several MLPs rather than just a single one, to solve a given pattern classification or regression task [9, 18]. This idea can be directly applied to the optical hybrid HMM/ANN models, using an ensemble of MLPs to estimate the emission probabilities of the Markov chains, as well as using ensembles of MLPs in every preprocessing step. Another idea is to combine several individual recognition systems (based on HMMs, HMM/ANN models or recurrent ANNs) and specialized classifiers [8, 59]. Finally, as pointed out in [5, 4], using deep learning methods would lead us to better trained ANNs which could improve every step of our recognition engine.

Acknowledgements

This work has been partially supported by the Spanish Ministerio de Educación y Ciencia (TIN2006-12767) and by the BPFI 06/250 scholarship from the Conselleria d'Empresa, Universitat i Ciencia, Generalitat Valenciana.

The authors would like to acknowledge the valuable help provided by Moisés Pastor, Juan Miguel Vilar, Alex Graves, and Marcus Liwicki.

References

[1] N. Arica and F.T. Yarman-Vural. An overview of character recognition focused on off-line handwriting. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 31(2):216–233, 2001.

[2] L. Bauer. Manual of Information to Accompany The Wellington Corpus of Written New Zealand English. Technical report, Department of Linguistics, Victoria University, Wellington, New Zealand, 1993.

[3] Yoshua Bengio. A connectionist approach to speech recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(4):647–667, 1993.

[4] Yoshua Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1), 2009.

[5] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Neural Information Processing Systems*, pages 153–160, 2006.

[6] Yoshua Bengio, Yann LeCun, Craig Nohl, and Chris Burges. LeRec: A NN/HMM Hybrid for On-Line Handwriting Recognition. *Neural Computation*, 7(6):1289–1303, 1995.

[7] R. Bertolami and H. Bunke. Ensemble Methods to Improve the Performance of an English Handwritten Text Line Recognizer. In *Arabic and Chinese Handwriting Recognition*, volume 4768 of *Lecture Notes in Computer Science*, pages 265–277. Springer, 2008.

[8] R. Bertolami and H. Bunke. Hidden Markov models-based ensemble methods for offline handwritten text line recognition. *Pattern Recognition*, 41(11):3452–3460, 2008.

[9] C. M. Bishop. *Neural networks for pattern recognition.* Oxford University Press, 1995.

[10] H. Bourlard and N. Morgan. *Connectionist speech recognition—A hybrid approach*, volume 247 of *Engineering and computer science.* Kluwer Academic, 1994.

[11] Radmilo M. Bozinovic and Sargur N. Srihari. Off-line cursive script word recognition. 11(1):68–83, 1989.

[12] Horst Bunke. Recognition of Cursive Roman Handwriting – Past, Present and Future. In *Proc. 7th International Conference on Document Analysis and Recognition*, volume 1, pages 448–459, Edinburgh, Scotland, 2003.

[13] C.J.C. Burges, J.I. Ben, J.S. Denker, Y. LeCun, and R. Nohl. Off-Line recognition of handwritten postal words using neural networks. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(4):689–704, 1993.

[14] C.J.C. Burges, O. Matan, Y. LeCun, J.S. Denker, L.D. Jackel, C.E. Stenard, C.R. Nohl, and J.I. Ben. Shortest Path Segmentation: A Method for Training a Neural Network to Recognize Character Strings. *International Joint Conference on Neural Networks*, 3:165–172, 1992.

[15] D. J. Burr. A normalizing transform for cursive script recognition. In *Proc. 6th International Conference on Pattern Recognition*, pages 1027–1030, Munich, Germany, 1982.

[16] Émilie Caillault and Christian Viard-Gaudin. Mixed Discriminant Training of Hybrid ANN/HMM Systems for Online Handwritten Word Recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 21(1):117–134, 2007.

[17] M. J. Castro and F. Casacuberta. Hybrid connectionist-structural acoustical modeling in the ATROS system. In *Proc. 6th European Conference on Speech Communications and Technology*, volume 3, pages 1299–1302, Budapest, Hungary, 1999.

[18] M. J. Castro and F. Casacuberta. Committees of MLPs for Acoustic Modeling. In *Proc. 5th Iberoamerican Symposium on Pattern Recognition*, pages 797–807, Lisboa, Portugal, 2000.

[19] A. El-Yacoubi, M. Gilloux, R. Sabourin, and C. Y. Suen. An HMM-Based approach for off-line unconstrained handwritten word modeling and recognition. 21(8):752–760, 1999.

[20] S. España-Boquera, M.J. Castro-Bleda, F. Zamora-Martínez, and J. Gorbe-Moya. Efficient Viterbi algorithms for lexical tree based models. In *Advances in Nonlinear Speech Processing, International Conference on Non-Linear Speech Processing*, volume 4885 of *Lecture Notes in Computer Science*, pages 179–187. Springer, 2007.

[21] S. España-Boquera, F. Zamora-Martínez, M. J. Castro-Bleda, and J. Gorbe-Moya. Efficient BP Algorithms for General Feedforward Neural Networks. In *Bio-inspired Modeling of Cognitive Tasks*, volume 4527 of *Lecture Notes in Computer Science*, pages 327–336. Springer, 2007.

[22] W.N. Francis and H. Kucera. Brown Corpus Manual, Manual of Information to accompany A Standard Corpus of Present-Day Edited American English. Technical report, Department of Linguistics, Brown University, Providence, Rhode Island, US, 1979.

[23] Hiromichi Fujisawa. Forty years of research in character and document recognition-an industrial perspective. *Pattern Recognition*, 41(8):2435–2446, 2008.

[24] R. Gemellovo, F. Mana, and D. Albesano. Hybrid HMM/Neural Network based Speech Recognition in Loquendo ASR. 2008.

[25] J. Gorbe-Moya, S. España-Boquera, F. Zamora-Martínez, and M. J. Castro-Bleda. Handwritten Text Normalization by using Local Extrema Classification. In *Proc. 8th International Workshop on Pattern Recognition in Information Systems*, pages 164–172, Barcelona, Spain, 2008.

[26] A. Graves, S. Fernandez, M. Liwicki, H. Bunke, and J. Schmidhuber. Unconstrained online handwriting recognition with recurrent neural networks. In *Advances in Neural Information Processing Systems 20*, pages 577–584. MIT Press, 2008.

[27] Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. A Novel Connectionist System for Unconstrained Handwriting Recognition. 31(5):855–868, 2009.

[28] J. L. Hidalgo, S. España, M. J. Castro, and J. A. Pérez. Enhancement and cleaning of handwritten data by using neural networks. In *Pattern Recognition and Image Analysis*, volume 3522 of *Lecture Notes in Computer Science*, pages 376–383. Springer-Verlag, 2005.

[29] S. Jaeger, S. Manke, and A. Waibel. Npen++: An On-Line Handwriting Recognition System. In *Proc. 7th International Workshop on Frontiers in Handwriting Recognition*, pages 249–260, Amsterdam, 2000.

[30] F. Jelinek. *Statistical Methods for Speech Recognition.* Language, Speech, and Communication. The MIT Press, 1997.

[31] S. Johansson, E. Atwell, R. Garside, and G. Leech. *The Tagged LOB Corpus: User's Manual.* Norwegian Computing Centre for the Humanities, Bergen, Norway, 1986.

[32] Jin Ho Kim, Kye Kyung Kim, and Ching Y. Suen. An HMM-MLP Hybrid Model for Cursive Script Recognition. *Pattern Analysis & Applications*, 3:314–324, 2000.

[33] S. Knerr and E. Augustin. A Neural Network-Hidden Markov Model Hybrid for Cursive Word Recognition. In *Proc. 14th International Conference on Pattern Recognition*, volume 2, pages 1518–1520, Washington, DC, USA, 1998.

[34] A.L. Koerich, Y. Leydier, R. Sabourin, and C.Y. Suen. A Hybrid Large Vocabulary Handwritten Word Recognition System Using Neural Networks with Hidden Markov Models. *Eighth International Workshop on Frontiers in Handwriting Recognition*, pages 99–104, 2002.

[35] A.L. Koerich, R. Sabourin, and C.Y. Suen. Large vocabulary off-line handwriting recognition: A survey. *Pattern Analysis & Applications*, 6(2):97–121, 2003.

[36] Simone Marinai, Marco Gori, and Giovanni Soda. Artificial Neural Networks for Document Analysis and Recognition. 27(1):23–35, 2005.

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

[38] U.-V. Marti and H. Bunke. The IAM-database: an English sentence database for offline handwriting recognition. *International Journal of Document Analysis and Recognition*, 5(1):39–46, 2002.

[39] S. Marukatat, T. Artières, B. Dorizzi, and P. Gallinari. Sentence Recognition through hybrid neuro-markovian modelling. In *International Conference on Document Analysis and Recognition*, pages 731–735, Seattle, Washington, USA, 2001.

[40] M. Pastor, A. Toselli, and E. Vidal. Projection profile based algorithm for slant removal. In *Proc. International Conference on Image Analysis and Recognition*, volume 3212 of *Lecture Notes in Computer Science*, pages 183–190, Porto, Portugal, 2004.

[41] R. Plamondon and S. N. Srihari. On-line and off-line handwritting recognition: a comprehensive survey. 22(1):63–84, 2000.

[42] L. Rabiner and B. H. Huang. *Fundamentals of Speech Recognition.* Prentice-Hall, 1993.

[43] V. Romero, M. Pastor, A. H. Toselli, and E. Vidal. Improving handwritten off-line text slant correction. In *Proc. 6th IASTED International Conference on Visualization, Imaging, and Image Processing*, pages 389–394, Palma de Mallorca, Spain, 2006.

[44] Joachim Schenk, Johannes Lenz, and Gerhard Rigoll. On-Line Recognition of Handwritten Whiteboard Notes: A Novel Approach for Script Line Identification And Normalization. In *Proc. 11th International Workshop on Frontiers in Handwriting Recognition*, pages 540–543, Montréal, Québec, Canada, 2008.

[45] M. Schenkel, I. Guyon, and D. Henderson. On-line cursive script recognition using time delay neural networks and hidden Markov models. *Machine Vision and Applications*, 8(4):215–223, 1995.

[46] A. W. Senior and A. J. Robinson. An off-line cursive handwritten recognition system. 20(3):309–321, 1998.

[47] P.Y. Simard, D. Steinkraus, and M. Agrawala. Ink normalization and beautification. In *Proc. 8th International Conference on Document Analysis and Recognition*, pages 1182–1187, 2005.

[48] Tal Steinherz, Ehud Rivlin, and Nathan Intrator. Offline cursive script word recognition - a survey. *International Journal of Document Analysis and Recognition*, 2(2):90–110, 1999.

[49] A. Stolcke. SRILM: an extensible language modeling toolkit. In *Proc. International Conference on Spoken Language Processing*, pages 901–904, Denver, Colorado, 2002.

[50] Y.H. Tay, M. Khalid, R. Yusof, and C. Viard-Gaudin. Offline cursive handwriting recognition system based on hybrid Markov model and neural networks. In *Proceedings of the 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pages 1190–1195, 2003.

[51] A. H. Toselli, A. Juan, J. González, I. Salvador, E. Vidal, F. Casacuberta, D. Keysers, and H. Ney. Integrated Handwriting Recognition and Interpretation using Finite-State Models. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(4):519–539, 2004.

[52] Seiichi Uchida, Eiji Taira, and Hiroaki Sakoe. Nonuniform slant correction using dynamic programming. In *Proc. 6th International Conference on Document Analysis and Recognition*, volume 1, pages 434–438, Seattle, USA, 2001.

[53] J. M. Vilar. Efficient computation of confidence intervals for word error rates. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5101–5104, Las Vegas, Nevada, 2008.

[54] A. Vinciarelli. A survey on off-line cursive word recognition. *Pattern Recognition*, 35(7):1433–1446, 2002.

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

[56] Alessandro Vinciarelli, Samy Bengio, and Horst Bunke. Offline Recognition of Unconstrained Handwritten Texts Using HMMs and Statistical Language Models. 26(6):709–720, 2004.

[57] K. Y. Wong, R. G. Casey, and F. M. Wahl. Document Analysis system. *IBM Journal of Research and Developement*, 26(6):647–655, 1982.

[58] S. J. Young, P. C. Woodland, and W. J. Byrne. HTK: Hidden Markov Model Toolkit V1.5. Technical report, Cambridge University Engineering Department Speech Group and Entropic Research Laboratories Inc., 1993.

[59] Francisco Zamora-Martínez, María José Castro-Bleda, Salvador España-Boquera, and Jorge Gorbe-Moya. Improving isolated handwritten word recognition using a specialized classifier for short words. Lecture Notes in Artificial Intelligence. Springer-Verlag, 2010. To appear.