Handwriting recognition by using deep learning to extract meaningful features

Joan Pastor-Pellicer, María José Castro-Bleda, Salvador España-Boquera [a] Francisco Zamora-Martínez [b]
[a] *Universitat Politècnica de València, Camino Vera s/n, Valencia 46021, Spain*
[b] *R&D Department, Veridas S.L., Pol. Ind. Talluntxe II, Tajonar 31192, Spain*

Recent improvements in deep learning techniques show that deep models can extract more meaningful data directly from raw signals than conventional parametrization techniques, making it possible to avoid specific feature extraction in the area of pattern recognition, especially for Computer Vision or Speech tasks. In this work, we directly use raw text line images by feeding them to Convolutional Neural Networks and deep Multilayer Perceptrons for feature extraction in a Handwriting Recognition system. The proposed recognition system, based on Hidden Markov Models that are hybridized with Neural Networks, has been tested with the IAM Database, achieving a considerable improvement.

Keywords: handwriting recognition, deep learning, convolutional neural networks

1. Introduction

The field of Handwriting Recognition (HWR) has been a topic of intensive research for a long time (see some surveys in [13,43,64,14,28]). However, recognizing unconstrained handwritten text remains a challenging task. HWR has two main modalities: the online case, where the trajectories of strokes are recorded while the user is writing, and the offline modality, where only the text image is available (e.g. scanned document). The offline case is more challenging due to the lack of temporal relations between strokes.

Connectionist methods and, especially, deep neural networks [3,52,4] are able to extract meaningful features from raw values (in offline HWR, the scanned text image) as in [25,17,8,10].

Along these lines, this work proposes the use of deep neural networks (and, more specifically, deep Multilayer Perceptrons (MLPs) and Convolutional Neural Networks (CNNs)) to extract meaningful features for unconstrained offline HWR. CNNs have already been used in our research group for several related applications [39,41]. In previous works, we also developed a HWR engine based on Hidden Markov Models (HMMs) that was hybridized with Artificial Neural Networks (ANNs) which gave the best results in a fair comparison at that time [19]. Thus, the idea of using raw pixels from the text line images emerged naturally. The performance of the HWR engine using deep MLPs and CNNs to extract meaningful features has been very advantageous, as the experiments will show.

The remainder of this paper is organized as follows: Section 2 provides a short introduction to the state of the art. Our new proposals are presented in detail in Section 3. The experimental setup and results are described and analyzed in Sections 4 and 5, and we present our conclusions in Section 6.

2. State of the Art

Offline HWR involves several stages from the image acquisition of the documents (e.g. scanning an ancient book) to the final result which can be in the form of a text transcription, a graph of words (in order to model recognition ambiguities) or even an index for keyword spotting applications.

In this work, we will center our attention on the transcription of text line images, hence skipping some steps such as image cleaning/enhancing, text detection or text line segmentation.

A HWR system receives a text line image which is generally converted to a sequence $X = (x_1 \dots x_m)$ of feature vectors or frames. Although text line images are inherently bidimensional, this can be done because writing in a particular order makes it possible to consider the image as a sequence. The main goal is to find the likeliest word sequence $W^\star = (w_1 \dots w_n)$ that maximizes the posterior probability:

$$W^{\star} = \underset{W \in \Omega^{+}}{\arg\max}\, P(W|X) \quad (1)$$

The sequence with the maximum probability, given the input X, is searched in every possible sequence of words of a given vocabulary Ω.

From this point of view, the recognition of handwritten text lines images shares many characteristics with Large Vocabulary Continuous Speech Recognition (LVCSR): a joint segmentation and classification task is required in both cases for decoding since we cannot split the image or the audio into words or even graphemes/phonemes in order to classify them afterwards. To overcome this cyclic dependency (known as *Sayre's paradox* [50]), HMMs have been used for decades for these and for many other sequence labeling problems [47]. For HMMs, the previous Formula (1) is decomposed, by using Bayes' theorem, as the product of the optical model $P(X|W)$ and the statistical Language Model (LM), $P(W)$, which can be simplified as follows:

$$W^{\star} = \underset{W \in \Omega^{+}}{\arg\max}\, P(X|W)P(W) \quad (2)$$

The optical modeling $P(X|W)$ of the baseline system is estimated by a HMM over the sequence of features. HMMs have two kinds of parameters: the *emission probabilities* $p(x_n|q_i)$ for the frame x_n and the state q_i, and the *transition probabilities* $p(q_j|q_i)$ from state q_i to q_j.

Emission probabilities $p(x_n|q_i)$ have been classically estimated by using Gaussian Mixture Models (GMMs) [47]. The use of connectionist techniques (and, particularly, deep learning techniques) in this context usually comes in two flavours: in tandem systems the output of the connectionist system is fed to GMMs, whereas in hybrid HMM/ANNs the output of the connectionist system is directly used to estimate emission probabilities.

Scaled emission probabilities can be estimated with discriminative models (e.g., Neural Networks (NNs)) that approximate the posterior probabilities of each state $P(x|q) \propto P(q|x)/P(q)$. The classifier receives as input the information of the frame x along with the values of neighboring frames. The output corresponds to the number of possible states in the HMM (unless some of them were tied). Thus, for a repertoire of n different characters, each of which being modeled with s states, the network has $n \cdot s$ softmax output units, which approximate posterior probabilities [12,7].

Several ANNs types have been used for this purpose: MLPs in [54], CNNs in [8], Recurrent Neural Networks (RNNs) in [37], and combinations of them. Additionally, we can find other related models: Radial Basis Functions in [56], Support Vector Machines (SVMs) in [57], or time-delay networks in [15,29,51].

On the other hand, tandem systems can make use of ANNs similar to hybrid HMM/ANN but posteriors are fed to GMMs. Other tandem approaches are possible and, for example, in [26], an MLP bottleneck is used to extract features for the tandem. This approach was later improved by Deep Belief Networks (DBN) as seen in [49].

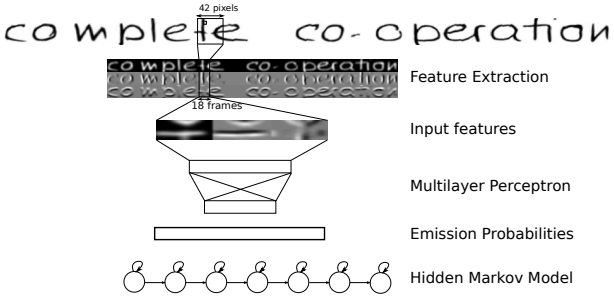
Other connectionist approaches get rid off the decomposition of $P(W|X)$ into optical model and LM and try to directly predict a sequence of labels. This is the case of the well-known Connectionist Temporal Classification (CTC) loss function, which was introduced in [21]. CTC aggregates the contribution of every alignment and assumes a new *blank* grapheme/phoneme output in the net to allow the model not to emit a label at each time step. This technique, meant for RNNs, has generated many successful works, particularly in HWR and LVCSR, among others [35,23,22]. In particular, when used with Long Short-Term Memories (LSTMs) [27], which have shown to successfully tackle the vanishing gradient problem derived from long time recurrences. In addition, Bidirectional Long Short-Term Memories (BLSTM) [53,1,24] allow the sequence to be scanned by two RNNs, one from left-to-right and another from right-to-left.

LSTMs have been extended to the multidimensional case [25] and 2D-LSTMs has lead to impressive results in HWR [65] tasks at the expense of a high computational cost. In this regard, [46] proposes the use of CNNs combined with LSTMs as a cost-efficient alternative to Multidimensional Recurrent Layers.

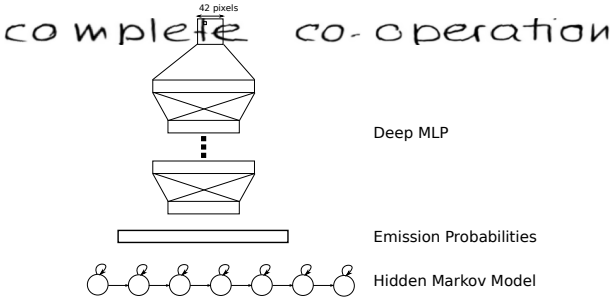
3. Proposed approaches

Our baseline system [19,41] is based on Hidden Markov Models that are hybridized with Neural Networks where emission probabilities are estimated by an MLP. The input is a sequence of feature vectors extracted following the approach presented in [60]. An illustration of the baseline system is depicted in Figure 1a.

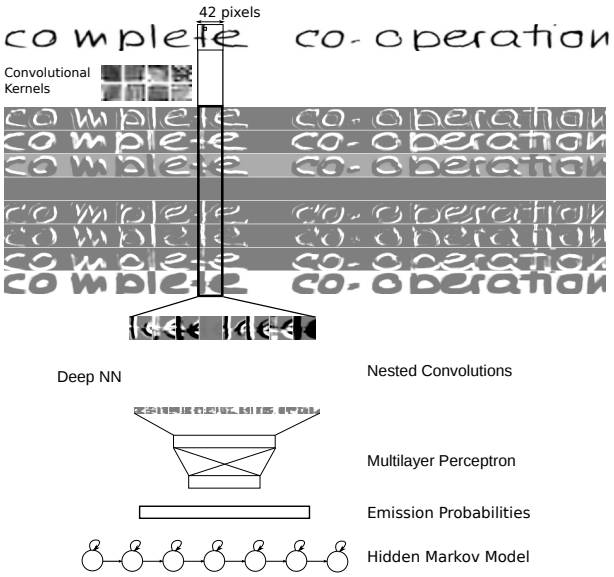
In the proposed approaches, instead of relying on a previous feature extraction process, we rely on the raw image input, by using a deep neural network to directly extract meaningful features (see Figure 1b) or by using CNNs (see Figure 1c).



(a) Baseline HMM/ANN recognition system using a sequence of feature vectors as input.



(b) HMM/ANN recognition system using the raw image as input.



(c) HMM/CNN recognition system.

Fig. 1. Handwriting Recognition systems.

3.1. Deep MLPs

The primary goal of this work is to extract meaningful features for HWR using deep learning techniques. When using the baseline system, the input of the MLP is a set of feature frames that are centered at the current frame (see Figure 1a). However, in the first proposed approach, the sliding window receives a patch of raw pixels that are directly fed to the NN as illustrated in Figure 1b. The choice of a squared window has given good results in preliminary experiments [41,40], leading to a window of 42×42 pixels (this size being the median of the height of input line images from training data). The window also advances two pixels at a time as in [41,40].

When dealing with raw images, there are several issues to keep in mind to improve the performance and generalization. Several standard regularization methods such as weight decay or max weight penalty have been employed. Regularization techniques such as dropout have helped to improve results. We have also used a layer-wise pretraining with Stacked Denoising Autoencoders (SDAE) [63] in order to train deeper nets.

3.2. CNNs with 2D convolutions

In the classical HMM/ANN architecture, the use of a sliding window (where the same NN is applied to classify each frame) can be seen as a 1D convolution on the X axis. Now, we would like to explore the use of 2D convolutions combined with pooling layers and higher level convolutions that will hopefully be able to extract more useful features.

Figure 1c illustrates the CNN for feature extraction and conditional probability computation in our setup. In the proposed settings, several parameters must be chosen for the CNN, such as the number of convolutions, pooling layers, activation functions, number, and size of the convolutional kernels as well as the classifier, which is usually an MLP.

It is quite important, in practice, and a challenging task to obtain an architecture with a good cost-efficiency trade-off. Thus, the computational restrictions that are essential for finding an appropriate but efficient architecture must not be forgotten. We have explored three alternatives with all of those limitations in mind, namely: *1)* using well known CNN architectures, *2)* using a specific network for the mentioned task, and *3)* using a model inspired by a well established feature extraction technique.

3.2.1. Using well known architectures

Our first attempt using CNNs for feature extraction imitates some of the previous architectures that have achieved good results in similar tasks. This is the case of the convolutional net LeNet CNN [33], which obtained good results on the MNIST database [34]. In addition, the increase in computational resources (especially advances in GPU computing and distributed systems) has allowed the use of deeper and more complex models. In recent years, these issues, combined with an appropriate parameter tuning, have led to remarkable improvements in performance, especially in image vision tasks. This is the case of nets like AlexNet [32], GoogleNet [59], and Very Deep Convolutional Networks [55], which have reported excellent results in other tasks such as the ImageNet Large Scale Visual Recognition Challenge contest [48].

3.2.2. Adhoc networks dealing with HWR

Most of the bibliography architectures are designed for tasks like MNIST, which consists of 28×28 pixel images corresponding to the 10 digits, or, for instance, the ImageNet database, which has larger inputs and more than a thousand classes. However, in our case study, the net input consists of 42×42 pixels, and there is an output for each different HMM state, which corresponds to 553 neurons (7 states $\times$ 79 graphemes).

When tuning a NN model we would have to explore, in an ideal case, every possible parameter and hyperparameter in order to obtain the most successful configuration. The use of CNNs and deeper nets makes this tuning process worse since more parameters are added, most of which are related to the new topology and layer configurations. Thus, in order to guide our exploration, we should be concerned about the kernel sizes to extract useful features, the number of kernels to cover the variability of the text, and deeper layers of the model to properly represent the characteristics of the problem.

In the feature extraction process proposed in [60], the frames are computed using 5×5 cells. Coincidentally, *LeNet-5* uses 5×5 convolutional kernels in both convolution layers. We will, therefore, explore kernel sizes between 5 and 8 pixels per side allowing the model to consider slightly bigger window sizes.

When analyzing the kernels trained in some preliminary experiments, we could conclude that there is a tendency to extract redundant information from 16 kernels in the first convolution. Some of the learned kernels detect edges in several orientations, others estimate the ink text zones, and some of them model the background. It turns out that all of these features can be extracted with no more than 5 to 10 kernels. Due to the above-mentioned computational constraints, we will avoid large number of kernels, at least, in the first convolutions.

3.2.3. Cell feature extraction by kernels

Our baseline HWR system used the parametrization described in [60]. In this work, we will design CNNs that are powerful enough to mimic this feature extraction process. However, it is important to note that the convolution kernels are not limited to extract these features, since they will learn on their own.

The original feature extraction divides the input into cell regions. For each region, three values are extracted: one value with the proportion of gray level and two values for the vertical and horizontal derivatives. A linear regression model is performed to find the optimal derivative directions.

The minimal requirements for a CNN to model these features are that one convolution could compute the vertical derivatives from the differences between the upper and lower cell values. Similarly, another convolution can compute the horizontal derivatives, whereas a third convolution would be enough to estimate the smoothed gray level. This leads to a CNN with only one convolution layer of three maps of 5×5.

3.3. CNNs with 1D convolutions

We have also explored a CNN that convolves the text line images in only one direction. The convolutional kernels would have the height of the image, and they would advance from left to right. Therefore, each kernel extracts only one feature per column. We explored two different approaches:

- Applying the vertical kernels directly into the raw image (Figure 2).
- Applying the vertical kernels after a 2D convolved map (Figure 3). In this case, the first set of 2D convolutions is obtained followed by the application of 1D kernels to these previous maps.

In the first case, the vertical kernels are applied to the input window, and we have a set of $K \times F$ features (with K being the number of kernels, and $F = C - w + 1$ where C is the number of columns of the sliding window and w is the width of the kernel since padding is not applied). In the second example, a 2D convolution is performed using the same parameters as in previous models. The vertical kernels are applied to the extracted maps afterward. A larger number of kernels is used to overcome the restriction of having one feature per kernel and column.

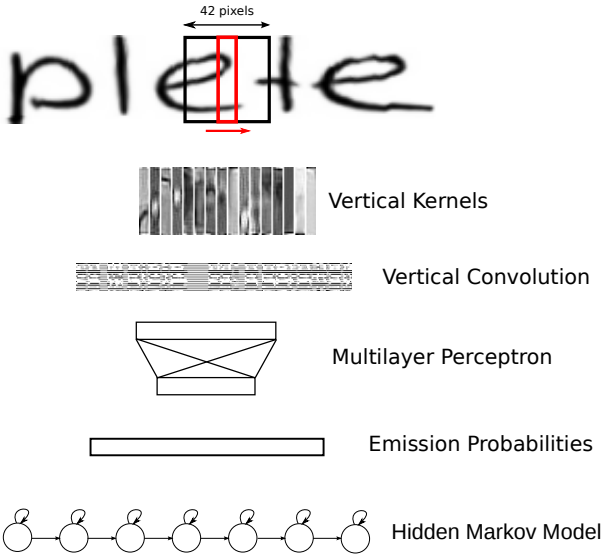


Fig. 2. Vertical model (I). The kernels run over the input window and only in the horizontal direction.

4. Experimental setup

4.1. Evaluation corpus: IAM database

The IAM offline dataset [36] is composed of forms containing handwritten English sentences that are extracted from the LOB corpus [30]. The version 3.0 of the IAM Dataset was used.[1] This version collects 5685 sentences from 657 different writers, with a total of 115000 word instances comprising 78 different graphemes. The forms are divided into lines, which are the input for the experimentation of this work. The standard partitions of this version were used: 6,161 training lines (from 283 writers), 920 validation lines (56 writers), and 2,781 test lines (161 writers).

4.2. The baseline recognition engine

The recognition engine is based on a hybridized HMM with an MLP to model graphemes, which was presented in [19,68,39,41]. Each grapheme is modeled with a 7-state left-to-right HMM topology with loops and without skips. The connectionist model used to estimate the emission probabilities of the HMM states was an MLP with 2 hidden layers of 512 and 256 units, respectively, using the softmax activation at the output layer. The HMM/ANN system is trained by means of an Expectation-Maximization procedure with a forced

[1] http://www.iam.unibe.ch/fki/databases/iam-handwriting-database

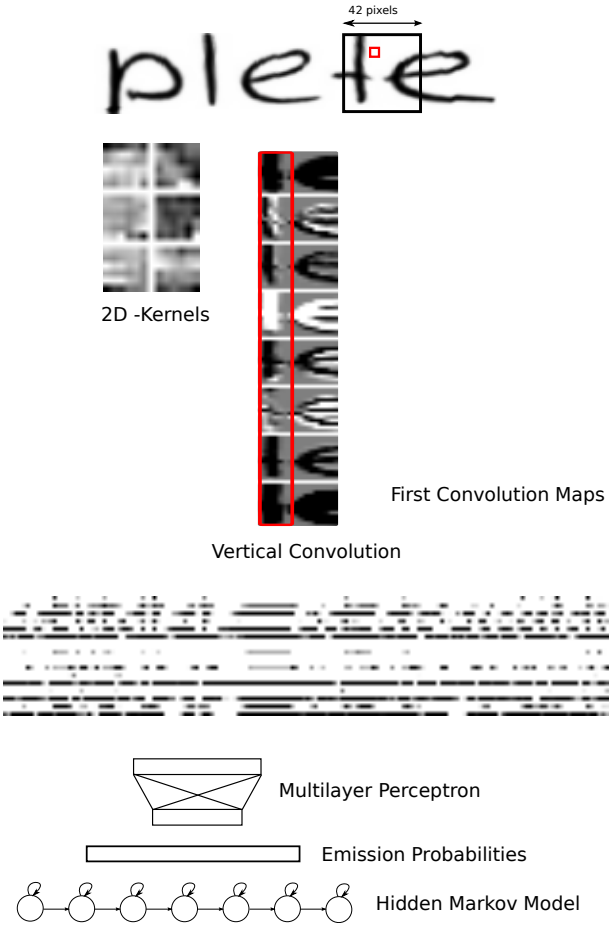


Fig. 3. Vertical models (II). Vertical kernels run over the maps generated by the first 2D convolutions.

Viterbi alignment by using the April-ANN toolkit [67]. This toolkit has also been used to perform the experiments based on deep MLPs and CNNs described in the following sections.

The images received by the recognition engine were preprocessed following the skew and slant correction presented in [19] and following the height normalization proposed in [39]. The emissions computed for each HMM state were calculated taking into account the current frame and the surrounding ones, for a total of 11 frames. Each frame was extracted from a window of 28×42 pixels following the approach presented in [60].

For the LM, a 4-gram with a Witten-Bell smoothing that was trained with the SRILM toolkit [58] was used. The text corpora used to train the n-gram LM were: the LOB corpus [30] (excluding those sentences that contain lines from the test set or the validation set of the IAM task), the Brown corpus [20], and the Wellington

Table 1

Deep MLPs fed directly with a raw image input from a window size of 42×42 (1764 pixels).

Hidden Layers	Dropout
2 hidden layers (2048, 512)	0, 0.2, 0.5
3×512 + SDAE	0, 0.2
5×512 + SDAE	0, 0.2
7×512 + SDAE	0, 0.2

corpus [2]. The lexicon of the LM had approximately 103K different words. This LM is the same as the one in [68], whose larger vocabulary differs from the previous work of [19]. Word insertion penalty and grammar scale factor parameters were optimized on the validation set by means of the Minimum Error Rate Training procedure [38].

4.3. Using raw input and deep MLPs

Our first goal is to compare the baseline system with a new one, avoiding an explicit handcrafted feature extraction. Table 1 shows the configuration used for the deep MLP-based systems with a receptive field (42×42 pixels). We also trained deeper MLPs up to 7 hidden layers of 512 Rectified Linear Unit (ReLU) neurons that were pretrained with SDAE in order to obtain faster convergence. The use of dropout helped significantly in these configurations.

4.4. Using CNNs for preprocessing

Table 2 summarizes the explored CNN topologies by enumerating, for each one, the sequence of kernels, pooling layers, flatten procedures and fully connected layers applied in each case along with their parameters and the size (number of maps and their dimensions) of the corresponding outputs. First, a topology based on *LeNet* (*LeNet-5*) was tested. For the second alternative, after several trials, we could highlight one special configuration, called *Adhoc CNN*, which led us to the best results. We also decided to apply max pooling layers to not only speed up the computations but also to make our model more robust to translations. We tried increasing max-pooling layers of 3×3 and 4×4. Since the suitability of the max-pooling is very task dependent, we also performed experiments removing them.

The models with the minimal configuration able to imitate the cell feature extraction were tagged as *Cell/Kernel 1 and 2 Conv.* As can be observed, the size of the kernels increased up to 6×6 and a stride of 3 was applied in each direction. Finally, two different topologies with one and two convolution-activation-pooling layers were tried (*Vertical 1 and 2*).

Table 2

CNN topologies for the recognition system.

	Operation Type	**Parameters**	**(Output) Size**
LeNet-5	input		1×42×42
	convolution	16 kernels 5×5	16×38×38
	Max-pool (ReLU)	2×2	16×19×19
	convolution	32 kernels 5×5	32×15×15
	Max-pool (ReLU)	2×2	32×8×8
	flatten		2048
	fully-conn. (ReLU)		500
	fully-conn. (softmax)		553
Adhoc CNN	input		1×42×42
	convolution	8 kernels 7×7	8×36×36
	max-pool (ReLU)	3×3	8×12×12
	convolution	16 kernels 3×3	16×10×10
	max-pool (ReLU)	2×2	16×5×5
	flatten		400
	fully-conn. (ReLU)		128
	fully-conn. (softmax)		553
Cell/Kernel 1 Conv.	input		1 × 42 × 42
	convolution	8 kernels 6×6+3+3	6×13×13
	flatten		1014
	fully-conn.		512
	fully-conn.		128
	fully-conn. (softmax)		553
Cell/Kernel 2 Conv.	input		1 × 42 × 42
	convolution	8 kernels 6×6+3+3	6×13×13
	convolution	16 kernels 4×4+2+2	16×6×6
	flatten		576
	fully-conn. (ReLU)		256
	fully-conn. (ReLU)		64
	fully-conn. (softmax)		553
Vertical 1	input		1×42×42
	convolution	16 kernels 42×6+1+3	16×1×13
	flatten		208
	fully-conn. (ReLU)		256
	fully-conn. (ReLU)		64
	fully-conn. (softmax)		553
Vertical 2	input		1×42×42
	convolution	8 kernels 6×6+3+3	6×13×13
	convolution	16 kernels 13×4+1+2	16×1×5
	flatten		80
	fully-conn. (ReLU)		256
	fully-conn. (ReLU)		64
	fully-conn. (softmax)		553

Table 3

Overall performance of the proposed systems on the Development set (configurations with the § mark make use of SDAE, the configuration with the † mark is the *Approach 1* of Table 4, while the configuration with the ‡ mark is the *Approach 2* of the same table).

				Dev.	
				WER	**CER**
	Baseline [41]			**15.6 ± 1.1**	**5.6± 0.5**
HMM/ANN		**Dropout**			
	Raw input	2048−512	0	16.1 ± 1.1	5.8 ± 0.5
	+ Deep MLPs		0.2	14.6 ± 1.1	5.1 ± 0.5
		3×512§	0	15.4 ± 1.0	4.9 ± 0.4
			0.2†	**13.7 ± 1**	**4.7 ± 0.4**
		5×512§	0	14.1 ± 1.1	4.6 ± 0.4
			0.2	14.2 ± 1.0	4.9 ± 0.5
		7×512§	0	15.2 ± 1.1	4.9 ± 0.5
			0.2	14.5 ± 1 .0	5.1 ± 0.4
HMM/CNN	LeNet-5			14.6 ± 1.1	4.6 ± 0.4
	Adhoc			14.4 ± 1.1	4.8 ± 0.4
	Cell-kernel 1 ‡			**13.9 ± 1.1**	**4.4 ± 0.4**
	Cell-kernel 2			14.3 ± 1.1	4.9 ± 0.4
	Vertical 1			15.5 ± 1.1	5.2 ± 0.4
	Vertical 2			15.3 ± 1.1	5.4 ± 0.5

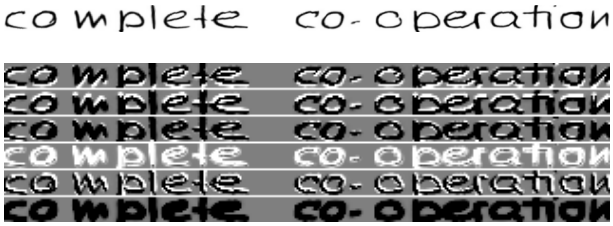
5. Results

Our baseline HWR system, based on hybrid HMMs with ANNs using handcrafted features was presented in [19]. With some slight modifications, our best results were reported in [41], obtaining a 15.6% and 19.0% Word Error Rate (WER) for validation and test sets, respectively. Table 3 shows the overall performance of the proposed systems, with a confidence interval of 95% [62]. First, it can be observed that all the deep models with more than two layers using raw inputs improved the baseline version. Indeed, when using two hidden layers in the raw setup, the results were worse than the baseline, unless dropout was added, where the results were similar. Dropout significantly helped in the deep model modality, reaching the best performance with three hidden layers and a drop rate equal to 0.2, obtaining a WER of 13.7 for the development set (this configuration is called *Approach 1* in Table 4). We tried drop rates that were larger than 0.2 but the performance did not improve. As a matter of fact, although some results with deep models were better than others, there was no statistically significant difference among them. For Character Error Rate (CER), deep models statistically improved the baseline system.

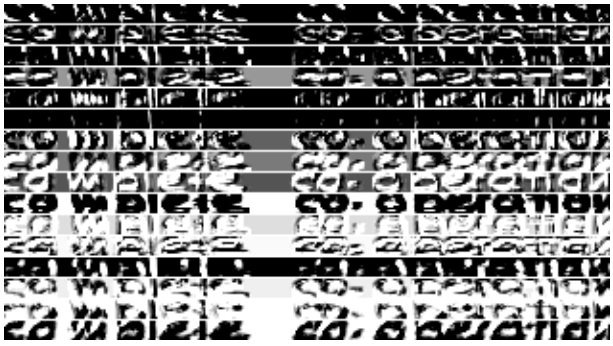
The HMM/CNN showed better performances with respect to the baseline system. When compared with the deep MLPs using raw inputs, the results were similar when dropout was used. When exploring the different nets, good performances in the *Adhoc CNN* net or even *LeNet-5* could be expected. Even though the performance in these cases is quite good, the best result achieved so far has been with a simple net (*Cell-kernel 1*, corresponding to *Approach 2* in Table 4), using one convolution with a stride of three in each direction and only six kernels. We presume that the simplicity of the model eased the training, and with six kernels the model covers most of the variability of the handwritten (as illustrated in Figure 4a). In this particular case, the net extracts 1014 features from the convolution process, which are conveniently combined with two fully connected layers of 512×512, respectively. The same model with two convolutions (*Cell-kernel 2*) had a fine performance not far from the best models.

Finally, vertical models showed a more modest performance, which did not improve the traditional 2D convolution models, but they were still better than the baseline. As before, CER was significantly better with the HMM/CNN than with the baseline system.

Regarding the execution time, both the baseline experiments and the proposed architectures have been trained and evaluated by means of the same April-ANN toolkit [67]. This toolkit performs all the computation on CPU, but it makes a heavy use of linear algebra optimized libraries (in particular, the Intel MKL library [66]). Experiments have been conducted on computers with different specifications, so a fair comparison of the execution time is limited to those that have been performed on the same type of machine. Most experiments have been performed on an Intel Core i7-3770 CPU at 3.40GHz with 32Gb of RAM: the "Raw input + Deep MLPs" 2048-512 0 and 2048-512 0.2 versions of Table 3 have required an execution time, for decoding, of 5.37 and 5.07 seconds/sentence on average, respectively. On the same machine, Lenet-5 only required 3.34 seconds/sentence, while Adhoc, Vertical 1 and Vertical 2 required 4.72, 4.17 and 5.19 seconds/sentence, respectively. Although other experiments (Baseline, Cell-kernels, $[3,5,7] \times 512$) have been conducted on different computers, the average execution time per sentence is rougly similar with times between 4 and 6 seconds/sentence. On comparison, the baseline system required around 4.8 seconds/sentence on a slightly slower machine (Intel Core i5-750 at 2.67GHz). Obviously, for production use the system should make use of GPU to drastically reduce these times.



(a) Six maps extracted by the first convolution of the *Cell/Kernel*. A free interpretation of the features learned is: 1) lower text contours, 2) upper contours, 3) borders, 4) strokes 5) right contours, 6) background model (background pixels got higher activation than text).



(b) Maps extracted by the Adhoc CNN. For instance, one kernel generates lower/right contours (fifth kernel), and another learns the upper/left edges (fifth from the tail).

(c) Generated maps by LeNet-5.

Fig. 4. Generated maps from several convolution nets.

Table 4

Performance for the IAM database. *Approach 1* uses deep MLPs and raw input (the configuration with the † mark in Table 3), and *Approach 2* uses the HMM/CNN system (the configuration with the ‡ mark in Table 3). The results of the table are divided into isolated words, line and paragraph recognition.

		Test Set	
System	$\|V\|$	**WER**	**CER**
Isolated word recognition			
Bianne-Bernard et al. [6]	10K	32.7	-
Bluche et al. [9]	10K	20.5	-
Poznanski and Wolf [45]	No-OOV	6.29	3.37
Line recognition			
Bertolami and Bunke [5]	20K	32.8	-
Plötz and Fink [44]	-	28.9	-
Graves et al. [23]	20K	25.9	18.2
Toselli et al. [61]	9K	25.8	-
Dreuw et al. [18]	50K	28.8	10.1
Pastor et al. [41] (baseline)	103K	19.0	7.5
Approach 1	103K	17.5	6.6
Approach 2	103K	17.2	6.3
Paragraph recognition			
Kozielski et al. [31]	50K	13.3	5.1
Doetsch et al. [16]	50K	12.2	4.7
Bluche et al. [10] (ROVER)	50K	11.9	4.9
Pham et al. [42]	50K	13.6	5.1
Bluche [11]	-	10.9	4.4
Puigcerver [46]	-	12.2	4.4

Table 4 shows the best results of our contributions together with the results of other works reported in the literature using the same database. The table is divided among isolated word recognition, line recognition and paragrah recognition. As mentioned above, we have used lines for training and evaluation. Although all the results cannot be directly compared, it can be observed that the use of CNN models to extract features from raw input for HWR consistently improve recognition rates. This has been proved in our HWR system (*Approach 2* statistically improves the baseline) and, regarding the results summarized in Table 4, we can observe that some of them have also relied on CNNs [11,46].

6. Conclusions

We have presented several improvements to our HWR engine by removing handcrafted feature extraction from the text images and using deep learning techniques directly on the raw input. Deep MLPs and CNNs have been analyzed for the current HWR task. The results presented for the IAM Database validate this approach, consistently with other authors' works.

Although several CNN topologies are explored, one of the configurations that led to good results is comprised of a single convolution layer without pooling, achieving a WER of 17.2. If we compare this result with the baseline (HMM/ANN with features system which has a WER of 19.0), a considerable step forward in the recognition performance has been achieved.

Despite the use of CNNs is not novel in HWR, the value of the experiments reported here lies in the fact that it provides a fair comparison between handcrafted and machine learned features by the virtue of using a baseline whose only difference with the proposed approaches is the replacement of the feature extraction stage, hence isolating the effect of this single stage from the whole HWR pipeline. Besides that, several different CNN topologies have been compared. We can also mention that the topologies proposed here only require a modest computational cost compared with the alternatives that can be found elsewhere.

There are many lines of future work to be pursued. First, we propose to apply other normalization techniques to speed up the training in order to improve results. We also need to do a more exhaustive error analysis to determine which steps of the whole transcription pipeline we should focus on to assure new improvements. Many novel CNN architectures are recently appearing in the general field of Computer Vision and, although many of them are not originally intended for HWR, some of them can also be adapted to this particular subfield. Finally, we plan to try the use of CTC decoding and to adopt training procedures for HMM/ANN in the line of [69].

Acknowledgments

Work partially supported by the Spanish MINECO and FEDER founds under project TIN2017-85854-C4-2-R.

References

[1] P Baldi, S Brunak, P Frasconi, G Soda, and G Pollastri. Exploiting the past and the future in protein secondary structure prediction. *Bioinformatics (Oxford, England)*, 15(11):937–946, 1999.

[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, Aaron C. Courville, and Pascal Vincent. Unsupervised Feature Learning and Deep Learning: A Review and New Perspectives. *CoRR*, abs/1206.5538, 2014.

[4] Yoshua Bengio, Yann LeCun, and Geoffrey Hinton. Deep learning. *Nature*, 521:436444, May 2015.

[5] Roman Bertolami and Horst Bunke. Hidden Markov model-based ensemble methods for offline handwritten text line recognition. *Pattern Recognition*, 41(11):3452 – 3460, 2008.

[6] Anne Laure Bianne-Bernard, Fares Menasri, Rami Al-Hajj Mohamad, Chafic Mokbel, Christopher Kermorvant, and Laurence Likforman-Sulem. Dynamic and contextual information in HMM modeling for handwritten word recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(10):2066–2080, 2011.

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

[8] Théodore Bluche, Hermann Ney, and Christopher Kermorvant. Feature extraction with convolutional neural networks for handwritten word recognition. In *12th International Conference on Document Analysis and Recognition (ICDAR)*, pages 285–289, 2013.

[9] Théodore Bluche, Hermann Ney, and Christopher Kermorvant. Tandem HMM with convolutional neural network for handwritten word recognition. In *38th International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pages 2390–2394, 2013.

[10] Theodore Bluche, Hermann Ney, and Christopher Kermorvant. A Comparison of Sequence-Trained Deep Neural Networks and Recurrent Neural Networks Optical Modeling for Handwriting Recognition. *Slsp-2014*, pages 1–12, 2014.

[11] Thodore Bluche. *Deep Neural Networks for Large Vocabulary Handwritten Text Recognition*. PhD thesis, Universit Paris Sud - Paris XI, 2015.

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

[13] R. M. Bozinovic and S. N. Srihari. Off-Line Cursive Script Word Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(1):68–83, 1989.

[14] Horst Bunke. Recognition of Cursive Roman Handwriting – Past, Present and Future. In *International Conference on Document Analysis and Recognition*, volume 1, pages 448–459, August 2003.

[15] Emilie Caillault, Christian Viard-Gaudin, and Abdul Rahim Ahmad. MS-TDNN with global discriminant trainings. In *International Conference on Document Analysis and Recognition (ICDAR)*, pages 856–860, 2005.

[16] P. Doetsch, M. Kozielski, and H. Ney. Fast and Robust Training of Recurrent Neural Networks for Offline Handwriting Recognition. In *14th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 279–284, 2014.

[17] Philippe Dreuw, Patrick Doetsch, Christian Plahl, and Hermann Ney. Hierarchical hybrid MLP/HMM or rather MLP features for a discriminatively trained Gaussian HMM: A comparison for offline handwriting recognition. In *International Conference on Image Processing (ICIP)*, pages 3541–3544, 2011.

[18] Philippe Dreuw, Georg Heigold, and Hermann Ney. Confidence and Margin-Based MMI/MPE Discriminative Training for Online Handwriting Recognition. *International Journal of Document Analysis and Recognition*, 14(3):273–288, 2011.

[19] S. España-Boquera, M. J. Castro-Bleda, J. Gorbe-Moya, and F. Zamora-Martinez. Improving Offline Handwritten Text Recognition with Hybrid HMM/ANN Models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 33(4):767–779, 2011.

[20] 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.

[21] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *23rd International Conference on Machine Learning (ICML)*, pages 369–376. ACM, 2006.

[22] Alex Graves and Navdeep Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *31st International Conference on Machine Learning (ICML)*, pages 1764–1772, 2014.

[23] Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5):855–868, 2009.

[24] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional LSTM networks. In *International Joint Conference on Neural Networks (IJCNN)*, volume 4, pages 2047–2052, 2005.

[25] Alex Graves and Jürgen Schmidhuber. Offline handwriting recognition with multidimensional recurrent neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 545–552, 2009.

[26] František Grézl, Martin Karafiát, Stanislav Kontár, and Jan Černocký. Probabilistic and bottle-neck features for LVCSR of meetings. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 4, 2007.

[27] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[28] Sebastiano Impedovo. More than twenty years of advancements on frontiers in handwriting recognition. *Pattern Recognition*, 47(3):916–928, 2014.

[29] S. Jaeger, S. Manke, J. Reichert, and A. Waibel. Online handwriting recognition: The NPen++ recognizer. *International Journal on Document Analysis and Recognition*, 3(3):169–180, 2001.

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

[31] Michal Kozielski, Patrick Doetsch, and Hermann Ney. Improvements in RWTH's system for off-line handwriting recognition. In *12th International Conference on Document Analysis and Recognition (ICDAR)*, pages 935–939. IEEE, 2013.

[32] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In F Pereira, C J C Burges, L Bottou, and K Q Weinberger, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 25, pages 1097–1105. Curran Associates, Inc., 2012.

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

[34] Yann LeCun, Corinna Cortes, and Chris Burges. MNIST handwritten digit database. `http://yann.lecun.com/exdb/mnist/`.

[35] Marcus Liwicki, Alex Graves, Horst Bunke, and Jürgen Schmidhuber. A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks. In *9th International Conference on Document Analysis and Recognition (ICDAR)*, pages 367–371, 2007.

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

[37] S. Marukatat, T. Artieres, R. Gallinari, and B. Dorizzi. Sentence recognition through hybrid neuro-Markovian modeling. In *6th International Conference on Document Analysis and Recognition (ICDAR)*, pages 731–735, 2001.

[38] F J Och. Minimum Error Rate Training in Statistical Machine Translation. In *41st Annual Meeting on Association for Computational Linguistics*, volume 1 of *ACL '03*, pages 160–167, 2003.

[39] J. Pastor-Pellicer, S. España-Boquera, F. Zamora-Martínez, M. Zeshan Afzal, and Maria Jose Castro-Bleda. Insights on the use of convolutional neural networks for document image binarization. In *The International Work-Conference on Artificial Neural Networks*, volume 9095, pages 115–126, 2015.

[40] Joan Pastor Pellicer. *Neural Networks for Document Image and Text Processing*. PhD thesis, Universitat Politcnica de Valncia. Departamento de Sistemas Informticos y Computacin, Valencia, Spain, 2017.

[41] Joan Pastor-Pellicer, Salvador España-Boquera, María José Castro-Bleda, and Francisco Zamora-Martínez. A combined Convolutional Neural Network and Dynamic Programming approach for text line normalization. In *13th International Conference on Document Analysis and Recognition (ICDAR)*, 2015.

[42] Vu Pham, Theodore Bluche, Christopher Kermorvant, and Jerome Louradour. Dropout Improves Recurrent Neural Networks for Handwriting Recognition. In *International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 285–290, 2014.

[43] R. Plamondon and Sargur N. Srihari. On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(1):63–84, 2000.

[44] T. Plötz and G.A. Fink. Markov models for offline handwriting recognition: a survey. *International Journal of Document Analysis and Recognition*, 12:269–298, 2009.

[45] A. Poznanski and L. Wolf. CNN-N-Gram for Handwriting-Word Recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2305–2314, 2016.

[46] Joan Puigcerver. Are multidimensional recurrent layers really necessary for handwritten text recognition? In *Document Analysis and Recognition (ICDAR), 2017 14th IAPR International Conference on*, volume 1, pages 67–72. IEEE, 2017.

[47] L R Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, 1989.

[48] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[49] Tara N. Sainath, Brian Kingsbury, and Bhuvana Ramabhadran. Auto-encoder bottleneck features using deep belief networks. *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 4153–4156, 2012.

[50] Kenneth M. Sayre. Machine recognition of handwritten words: A project report. *Pattern Recognition*, 5(3):213–228, 1973.

[51] 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.

[52] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *CoRR*, abs/1404.7828, 2014.

[53] M. Schuster and K. K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

[54] Andrew W Senior and Anthony J Robinson. An off-line cursive handwriting recognition system. *IEEE transactions on pattern analysis and machine intelligence*, 20(3):309–321, 1998.

[55] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *ArXiv*, pages 1–14, 2014.

[56] Elliot Singer and R. P. Lippman. A speech recognizer using radial basis function neural networks in an HMM framework. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 629–632. IEEE, 1992.

[57] J. Stadermann. A hybrid SVM/HMM acoustic modeling approach to automatic speech recognition. *International Conference on Spoken Language Processing (ICSLP)*, 2004.

[58] A. Stolcke. SRILM: an extensible language modeling toolkit. In *International Conference on Spoken Language Processing (ICSLP)*, pages 901–904, 2002.

[59] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going Deeper with Convolutions. *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–12, 2015.

[60] 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.

[61] Alejandro H. Toselli, Verónica Romero, Moisés Pastor, and Enrique Vidal. Multimodal interactive transcription of text images. *Pattern Recognition*, 43(5):1814–1825, 2010.

[62] Juan Miguel Vilar. Efficient computation of confidence intervals for word error rates. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5101–5104, 2008.

[63] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *Journal of Machine Learning Research*, 11(3):3371–3408, 2010.

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

[65] P. Voigtlaender, P. Doetsch, and H. Ney. Handwriting Recognition with Large Multidimensional Long Short-Term Memory Recurrent Neural Networks. In *15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 228–233, 2016.

[66] Endong Wang, Qing Zhang, Bo Shen, Guangyong Zhang, Xiaowei Lu, Qing Wu, and Yajuan Wang. Intel math kernel library. In *High-Performance Computing on the Intel® Xeon Phi*, pages 167–188. Springer, 2014.

[67] Francisco Zamora-Martínez et al. April-ANN toolkit, A Pattern Recognizer In Lua, Artificial Neural Networks module, 2013. https://github.com/pakozm/ [github.com]april-ann.

[68] F. Zamora-Martnez, V. Frinken, S. Espaa-Boquera, M.J. Castro-Bleda, A. Fischer, and H. Bunke. Neural network language models for off-line handwriting recognition. *Pattern Recognition*, 47(4):1642–1652, 2014.

[69] Albert Zeyer, Eugen Beck, Ralf Schlüter, and Hermann Ney. Ctc in the context of generalized full-sum hmm training. In *Interspeech*, pages 944–948, 2017.