Deep-Belief-Network based Rescoring for Handwritten Word Recognition

Partha Pratim Roy Youssouf Chherawala
Mohamed Cheriet
Synchromedia Laboratory, École de Technologie Supérieure, Montreal, Canada

Abstract

This paper presents a novel verification approach towards improvement of handwriting recognition systems using a word hypotheses rescoring scheme by Deep Belief Networks (DBNs). A recurrent neural network based sequential text recognition system is used at first to provide the N-best recognition hypotheses of word images. Word hypotheses are aligned with the word image to obtain the character boundaries. Then, a verification approach using a DBN classifier is performed for each character segments. DBNs are recently proved to be very effective for a variety of machine learning problems. The character probabilities obtained from DBNs are next combined with the base recognition system. Finally, the N-best recognition hypotheses list is reranked according to the new score. We have compared our proposed approach with an MLP based rescoring approach on the Rimes dataset. The results obtained show that the verification approach using DBNs outperforms that of MLP systems.

1. Introduction

Automatic handwriting recognition has been one of the most active areas of research in the last decades. The recognition task is challenging because of the large variability of writing styles, cursive nature, size of vocabularies, etc. [6, 7]. Though high recognition rates are achieved in character recognition, offline text recognition is not easy. Many pieces of text recognition work exist for segmenting the word into its character components by analyzing the character shapes. To avoid this task, stochastic approaches such as Hidden Markov Models (HMMs) have been applied to perform text recognition [15, 17, 19]. HMMs avoid explicit segmentation of cursive words into characters/sub-words by jointly performing segmentation and recognition [19]. Recently, Recurrent Neural Network (RNN) based sequence classifiers have shown superior performance over HMMs [9]. The bidirectional Long Short-Term Memory (BLSTM) architecture of RNN provides access to long range context along both input directions.

A further improvement in recognition performance can be achieved using a classifier combination approach. Combination of classifiers relies on the assumption that different classifiers have their own strengths and weaknesses which can compensate for each other through the combination. Work has been done to integrate a character recognition system into a word recognition module to rescore the word hypotheses [3]. Multi-Layer Perceptron (MLP) has been extensively applied to classify characters as part of isolated or continuous handwritten word recognizers [14]. Recently, new research on the training strategies of Deep Belief Networks (DBNs) [12] has allowed improved performance in machine learning and pattern recognition tasks. The deep networks learn a hierarchy of non-linear feature detectors that better capture the complex patterns in data. Due to its deep learning mechanism DBNs provide superior performance than MLPs. In this work, we use DBNs as a verification module to improve existing handwriting systems.

In this paper we propose a novel approach for improving the performance of existing handwritten recognition systems. An efficient verification method using DBNs is integrated in our system. This takes into account the strength and weakness of sequential recognition classifiers. Our proposed method is inspired from the work of Koerich et al. [14] where an HMM based recognition method is used for handwritten word recognition, and rescoring is performed by integrating a neural network based verification approach. We used a DBN classifier to verify the words recognized by a BLSTM neural network.

The main contributions of this paper are the following: 1) A framework combining the BLSTM neural network and DBNs for recognition system; 2) Integration of DBNs as a verification approach for word sequence recognition; and 3) A comparative analysis of our results with MLP-based verification system. The rest of this paper is organized as follows. Section 2
provides the details of our recognition system using a recurrent neural network with long-short term memory cells (LSTM). The rescoring methodology using DBNs is detailed in Section 3. Next, the experimental setup and results on the Rimes dataset are reported in Section 4. Finally, the conclusion is drawn in Section 5.

2 Word Recognition using BLSTM

Our word recognition system is based on a variant of the Recurrent Neural Network (RNN) known as the BLSTM neural network. In the following we discuss the feature extraction and recognition process.

2.1 Feature extraction

In the literature there exist a number of sliding window based features [15, 4] for sequence classification. In our earlier work [4] we evaluated some of these state-of-the-art features along with automatically learned features in the BLSTM framework. It was noted that the column-wise feature proposed by Marti and Bunke [15] outperformed other features. Hence, we employ the Marti-Bunke feature to represent the binary word images in this system.

Using the sliding window techniques the text image is represented by a sequence of column-wise local feature vectors, \( O = o_1, ..., o_T \), where \( T \) is the width of the image. Here, the sliding window has a width of one pixel moving from left to right. The feature consists of a set of nine features including geometrical and contour-gradient information. Three global features capture the fraction of black pixels, the center of gravity, and the second order moment. The remaining six local features consist of the position of the upper and lower contour, the gradient of the upper and lower contour, the number of black-white transitions, and the fraction of black pixels between the upper and lower contours.

2.2 Recognition

The baseline recognition system is built on two distinct neural networks, the bidirectional long short-term memory blocks (BLSTM) as hidden layer and the connectionist temporal classification (CTC) for sequence decoding.

**Long Short-Term Memory (LSTM) layer:** The LSTM network nodes have a specific architecture referred to as memory block. Each memory block contains a memory cell and its interaction with the rest of the network is controlled by three gates, namely: an input gate, an output gate and a forget gate. The network architecture allows the memory cell to preserve its state over a long range of time. The 1D sequence recognition is improved by processing the input signal in both directions, i.e. one layer processes the signal in forward direction while another layer processes it in backward direction. The outputs of both layers are combined at the next layer as a feature map.

**Connectionist Temporal Classification (CTC) layer:** Usually most of the RNNs require pre-segmented training data or post-processing to transform its outputs into transcriptions. To avoid such process, the CTC output layer has been designed for sequence labeling. This layer is trained to predict the probability of an output label sequence given an input sequence. The output activation function provides the probability to observe each character for each sequence time. The objective function of CTC is defined as the negative log probability of the network correctly labeling the entire training set. Once the network is trained, the labeling of an unknown input sequence \( O \) is performed by choosing the label \( L^* \) with the highest conditional probability, i.e.

\[
L^* = \arg \max_L p(L|O) \quad (1)
\]

A token passing algorithm is used to compute the probability of each word from a given lexicon. For each word image it provides an N-best list of the ASCII character sequence of words and the corresponding log probability scores. The N-best list is reranked later with the verification score in the rescoring module.

3 Word Verification and Rescoring

3.1 Character alignment

One popular verification approach in text recognition system is to recognize individual characters in a word [14]. For the character recognition step, character segmentation is necessary. As mentioned earlier, character segmentation/boundary detection in cursive word image is difficult. It is noted that the character alignment using BLSTM is not easy because the CTC output layer produces a probability distribution over character transcription which does not directly match with characters in text image [18]. To fill this gap, we consider the HMM-based forced alignment approach [20] in this framework. This approach is widely used for providing consistent and accurate character segmentations. The forced alignment using the Viterbi algorithm finds the most probable boundaries for the given sequence of character units.

The BLSTM recognition system provides the N-best word hypotheses list for each word image. Then, the HMM is used in forced alignment mode to segment the word image into characters. The segmentation path is refined through iterative alignment and retraining,
called embedded training. Using the alignment algorithm we obtain the character segments \( S_1, S_2, \ldots S_n \) of a given word hypothesis (see Fig.1). Before alignment a slant correction is performed by applying a shear transform.

![Madame](image)

**Figure 1.** Character alignment in the word “Madame” using the Viterbi algorithm. Red lines indicate the segmentation zones of characters.

### 3.2 Deep Belief Network

Deep Belief Networks (DBNs) [12] have been successfully introduced in many machine learning tasks with competitive results. A DBN is created as a stack of its main building blocks called restricted Boltzmann machines (RBMs). RBM is a particular form of log-linear Markov Random Field that has a two-layer architecture, in which the visible stochastic units \( v \) are connected to the hidden stochastic units \( h \). Each layer of latent representation is learned by training an RBM to model the data distribution at the next lower layer, using the Contrastive Divergence (CD) algorithm. [12]. Given the model parameters \( \theta \), the weights of the connections and the biases of the individual units form a joint probability distribution \( P(v, h|\theta) \) over the visible units \( v \) and hidden units \( h \). For binary RBMs, this distribution is computed based on an energy function \( E(v, h|\theta) \):

\[
E(v, h|\theta) = -\sum_{i=1}^{V} \sum_{j=1}^{H} w_{ij} v_i h_j - \sum_{i=1}^{V} b_i v_i - \sum_{j=1}^{H} a_j h_j
\]

(2)

where model parameters \( \theta = \{w, b, a\} \). \( w_{ij} \) is the weight between visible unit \( i \) and hidden unit \( j \). \( b_i \) and \( a_j \) are bias terms for visible unit \( i \) and hidden unit \( j \), respectively. \( V \) and \( H \) are the numbers of visible and hidden units. The marginal probability \( P(v|\theta) \) is computed as

\[
P(v|\theta) = \frac{\sum_{h} \exp(-E(v, h|\theta))}{Z(\theta)}
\]

(3)

where \( Z(\theta) \) is known as the partition function:

\[
Z(\theta) = \sum_{v} \sum_{h} \exp(-E(v, h|\theta)).
\]

(4)

For each data vector \( v \) we use Eq.4 to compute a vector of hidden unit activation probabilities \( h \). These hidden activation probabilities are used as training data for a new RBM. Once the training of RBMs is done, we initialize the weights of the hidden layers of a neural network with a number of hidden layers equal to the number of RBMs. After pre-training, we add a randomly initialized softmax output layer and use backpropagation to fine-tune the weights of the network in a discriminative fashion.

### 3.3 Score computation of word hypotheses

Using the Bayes rule, the output probability \( P(C|X) \) of a character \( (C) \) is obtained from the DBNs using Eq.5. \( P(C) \) is the a priori class probability. \( P(X|C) \) is the class-conditional feature probability and \( P(X) \) is input probability of character feature \( X \).

\[
P(C|X) = \frac{P(X|C)P(C)}{P(X)}
\]

(5)

The word hypothesis probability can be computed by combining the character probabilities of that word. Since we have assumed the words are having equal prior probabilities, we compute the probability score \( (H_d) \) of the \( d \)th hypothesis from the sequence of the \( L_d \) character segments \( (S_d) \) by taking the average character probabilities [13] as shown in Eq.6.

\[
H_d(S_d) = \frac{1}{L_d} \sum_{l=1}^{L_d} P(C_l|X_l)
\]

(6)

### 3.4 Reranking of hypotheses list

Our reranking approach is based on the algorithm proposed by [14]. The score obtained by the recognition and verification systems are normalized by Eq.7 so that the scores assigned to the N-best word hypotheses sum up to 1. \( C(S_d) \) is the modified score for each word hypothesis:

\[
C(S_d) = \frac{P(\cdot)}{\sum_{d=1}^{N} P(\cdot)}
\]

(7)

We use the confidence scores for each word hypothesis in the N-best word list. Two classifiers, the BLSTM and DBNs each produce \( N \) consistent measurements, one for each word hypothesis in the N-best word hypotheses list. To combine the output score of recognition \( (C_{RS}) \) and verification \( (C_{VS}) \) systems, it has been shown that the weighted sum rule maximizes the recognition accuracy [14]. The combination of confidence scores by the weighted sum rule is performed as follows:

\[
C_{new} = \alpha \times C_{RS} + (1-\alpha) \times C_{VS}
\]

(8)
where, $\alpha$ is the combination weight factor in weighted sum rule. The range of $\alpha$ lies between 0 and 1. It is set according to the validation data. Then the hypotheses list is reranked according to this new score.

4 Experimental Results

We have carried out experiments on word recognition tasks for Latin script. The Rimes dataset (Reconnaissance et Indexation de données Manuscrites et de facsimilés / Recognition and Indexing of handwritten documents and faxes) [1] has been used in ICFHR and ICDAR competitions to evaluate unconstrained handwriting recognition. There are more than 12,700 handwritten documents corresponding to 5,605 mails of two to three pages. These are mainly fictional French letters written by more than 1,300 writers. A total of 80 text-symbols are used in the dataset. The database consists of 59,203 word images divided into three subsets: 44,197 images for training, 7,542 for validation, and 7,464 for testing. In this work, we used the segmented word images and the reduced test dictionary of size 1,612 words. The word recognition is case and accent sensitive.

4.1 Performance evaluation

The raw data from the Rimes dataset was used for performance evaluation of sequential feature based text recognition. Since the images in the Rimes dataset are in gray tone, we used the Otsu binarization algorithm to convert them to binary.

4.1.1 Recognition result

As discussed earlier, the 9 dimensional Marti-Bunke feature extracted from the word image is used as the sequential features in RNN system for baseline recognition accuracy. The neural network architecture is made of a hierarchy of three LSTM layers. The layers of the last level are connected to the CTC network. For our experiments, we used the RNNLIB implementation of recurrent neural networks [8]. In the network, the learning rate has been set to $10^{-4}$ and a momentum of 0.9 has been used. The experiment is reproduced several times because of the random initialization of the neural network during the training phase. To evaluate our present approach we have considered the system that produced the best performance. The best word recognition result is noted as 90.73%.

For comparison of the word recognition system we have considered the popular HMM framework. We make a comparative study of these systems with Marti-Bunke feature in the Rimes dataset. In the basic HMM framework, the word is modeled by the concatenation of the character models. These models contain a fixed number of hidden states arranged in left-to-right Bakis topology without skips. Finally, the trained models are used to decode the test images by Viterbi algorithm. We use the Hidden Markov Model Toolkit (HTK) [20] for training and recognition. In our approach we use the same 9 dimensional Marti-Bunke feature (it has been used in BLSTM) along with its dynamic feature: delta and acceleration in frame-wise feature and obtain feature vector of dimension 27. These derivative features capture a wider temporal context at the frame level and represent the dynamics of features around the current window. The addition of dynamic features in the feature vector improves handwriting recognition [2]. Additional features are used in HMM to capture wider context for fair comparison with BLSTM. We tried the dynamic feature in BLSTM but the improvement was negligible. We consider continuous density HMMs with diagonal covariance matrices of GMMs in each state. During HMM training, 128 Gaussian mixtures and 7-state left-to-right HMM are used. The validation data is used to learn these parameters.

Table 1. Word Recognition Result by BLSTM and HMM.

<table>
<thead>
<tr>
<th>Choices</th>
<th>BLSTM</th>
<th>HMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top 1</td>
<td>90.73%</td>
<td>75.27%</td>
</tr>
<tr>
<td>Top 2</td>
<td>95.32%</td>
<td>84.46%</td>
</tr>
<tr>
<td>Top 3</td>
<td>96.40%</td>
<td>88.07%</td>
</tr>
<tr>
<td>Top 4</td>
<td>96.95%</td>
<td>90.94%</td>
</tr>
<tr>
<td>Top 5</td>
<td>97.41%</td>
<td>91.62%</td>
</tr>
</tbody>
</table>

Table 1 shows the comparison of BLSTM and HMM based recognition results. Here, Top N denotes that the true word is present among the N-best word hypotheses. As it is seen from the table BLSTM outperforms the recognition performance compared to HMM system. With Top 5 choices the accuracy with BLSTM reaches up to 97.41%. The Top N results obtained from BLSTM are used in the following stages to evaluate the verification and rescoring performances.

4.1.2 Verification result

For verification purpose, we use the HMM models (described in previous sub-section) to align the characters in the word image. The hypotheses list of ASCII transcription and boundary segmentation of each character are next passed to DBN-based verification process. The verification stage produces recognition confidence for each character segments. Finally, the score of word recognition is computed from the confidence scores of the characters.
For character verification purpose, the character-segment images are resized into 28 x 28 pixels. The pixel intensity values are next fed into DBNs for recognition. In the experiment, all DBNs were pre-trained in an unsupervised manner using RBMs. The RBMs were trained using stochastic gradient descent with a mini-batch size of 100 training cases in the pre-training process. The weights in RBMs were initialized randomly with a normal distribution with mean 0 and standard deviation 0.1. All RBMs were trained for 100 epochs. The hidden layers used logistic sigmoid non-linearities and a softmax layer was used for the output layer to provide posterior probability estimates for each output class. For fine-tuning, we used the same mini-batch size as in pre-training. We studied the performance of DBNs for word recognition when the number of hidden units increases from 128 to 1536. The best result is obtained when the size of hidden unit is 1024. We have performed experiments up to 3 layers for training. The number of nodes in each hidden layer is kept fixed at 1024. The word score obtained by DBNs are shown in Table 2. By Top 1 result we obtained 64.01% accuracy.

We compare the performance of DBN-based verification approach with canonical MLP based [11] system. An MLP is composed of an input layer, an output layer, and hidden layers to solve problems that are not linearly separable. We used the standard back-propagation algorithm with weights initialized randomly to train the network. The input data of the MLP was 28 x 28 input data. We tested MLP with varying hidden units from 64 to 512 and set it to 128 according to the experiment result on the validation set. The number of hidden layer was set to two according to the experiment results. It has been observed that increasing the number of hidden units did not improve the word recognition performance. The comparative results for the test set of DBNs and MLPs are shown in Table 2. It is clear that DBNs outperform MLP based verification system.

Table 2. Word Verification Result by DBN and MLP.

<table>
<thead>
<tr>
<th>Choices</th>
<th>DBN</th>
<th>MLP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top 1</td>
<td>64.01%</td>
<td>57.38%</td>
</tr>
<tr>
<td>Top 2</td>
<td>81.63%</td>
<td>73.92%</td>
</tr>
<tr>
<td>Top 3</td>
<td>92.38%</td>
<td>85.37%</td>
</tr>
<tr>
<td>Top 4</td>
<td>96.37%</td>
<td>92.96%</td>
</tr>
<tr>
<td>Top 5</td>
<td>97.41%</td>
<td>97.41%</td>
</tr>
</tbody>
</table>

4.1.3 Reranking result

The verification score of DBNs are combined with the BLSTM score to rerank the hypotheses list. In our combination approach with the weighted sum rule, the weight (\( \alpha \)) was adjusted using the validation data. We found the best rescoring result in validation data with \( \alpha = 0.9 \). We show the influence of weight in the combination on test-set in Fig. 2. It is noticed that when the value of \( \alpha \) is more than 0.8, we get better accuracy than using only recognition system. We obtain the best accuracy as 91.84% by integrating DBN-based verification approach. The gain is 1.12% from using only BLSTM. Using MLP based verification scheme the reranking score obtained is 91.03%. These results are noted with the best value of \( \alpha \) on the validation set.

![Figure 2. The influence of \( \alpha \) in reranking the hypotheses list for recognition.](image)

4.2 Comparison with other systems

In literature we find a number of systems with their own features and pre-processing approaches. Hence, the comparison of our approach with other systems will not be uniform. To have an idea, we report here some recent performances on the Rimes dataset. The TUM MDLSTM [10] reported the performance as 93.2% with hyper-parameters optimization. The system from [16] is based on a weighted combination of 7 systems: one hybrid MLP-HMM, two tandem GMM-HMMs and four MDLSTM. With the combination they obtained 95.2% accuracy. Bianne-Bernard et al. [2] reported 89.1% by combining HMMs using context-independent and context-dependent systems. The combination was performed by a neural network. The feature extraction was done based on [17]. Doetsch et al. [5] used a tandem LSTM-HMM system. The accuracy was reported as 90.3%.

In our approach after integrating DBN-based verification the best accuracy is achieved as 91.84%. We have demonstrated that the proposed rescoring approach can be combined with other recognition system to improve further. Hence, recognition accuracy of
other existing isolated or combination approaches can be improved with this scheme.

5 Conclusion

We have presented a verification based re-ranking approach using Deep Belief Networks (DBNs) for handwritten word recognition. The efficient discriminative feature of DBNs is combined with the recurrent neural network based sequence classifier to improve the recognition performance further. The N-best recognition hypotheses of word image obtained from RNNs are processed by DBNs for verification. The character probability score computed from DBNs are combined with the base recognition system. A comparative evaluation of DBNs and MLPs on the Rimes (Latin) word recognition dataset has been conducted in our experiments. The results demonstrate that substantial gains in performance are obtained by a DBN-based rescoring scheme.

References


