Noname manuscript No.
(will be inserted by the editor)

Handwriting recognition using Cohort of LSTM and lexicon verification with extremely large lexicon

Bruno STUNER* and Clément Chatelain and Thierry Paquet

Received: date / Revised version: date

Abstract State-of-the-art methods for handwriting recognition are based on LSTM neural networks, which now provide very impressive character recognition performance. Character recognition is generally coupled with a lexicon-driven decoding process which integrates dictionaries. Unfortunately these dictionaries are limited to hundreds of thousands of words for the best systems, which prevents us from having a good language coverage, and therefore limits global recognition performance. In this article, we propose an alternative to the lexicon-driven decoding process based on a *lexicon verification* process and a new method to obtain hundreds of complementary LSTM RNN that are extracted from a single training, called *cohort*, coupled in different combination systems. Our first combination is a cascade made of a large number of complementary LSTM RNN for isolated handwritten word recognition. The proposed cascade achieves new state-of-the art performance on the Rimes and IAM datasets. The second contribution extends the idea of cohort and lexicon verification in a ROVER combination for handwriting line recognition and achieves state-of-the-art results on the Rimes dataset. Dealing with gigantic lexicon of 3 million words, the method also demonstrates interesting performance with a fast decision stage.

1 Introduction

Handwriting recognition is the numeric process of translating handwritten text images into strings of characters. The handwriting recognition process traditionally involves two steps [1]: optical character recognition and linguistic processing. Optical character recognition is a difficult task due to the variability of shapes in handwritten texts, since every human has their personal writing style. Therefore, even when using state-of-the-art classifiers like deep neural networks to recognize characters [2], a considerable amount of errors can occur by considering only the optical model. Linguistic processing aims at combining the characters hypotheses together so as to provide the most likely sequence of words in accordance with some high level linguistic rules. There are two types of linguistic knowledge: lexicons and language models. A language model is a probabilistic modelization of a language which generally provides word sequence likelihood, allowing to rank the recognition hypotheses provided by the optical model. Nowadays, the use of linguistic knowledge is an open problem. Lexicon-driven approaches aim at recognizing words thanks to the use of a lexicon. They search for the most likely word that belongs to the working lexicon, by concatenating the character hypotheses. There is currently no efficient alternative to the use of lexicon-driven recognition methods either for isolated words or for text recognition. The choices of which lexicon resource should be used, and which corpora should be selected for training the language model are still open questions. These choices directly affect recognition performance. In the case of lexicon-driven methods, where the characters are aligned on the lexicon words, too small lexicons fail to cover the test dataset, thus leading to misrecognition. However, using a large lexicon (1000 words and more) requires many computations and generally produces precision loss [3]. To the best of our knowledge, the largest lexicon used in literature was composed of 200K words [4] (60K words for [5]), and there could still remain out-of-vocabulary words (named entities, numbers, etc).

Normandie Univ, UNIROUEN, INSA Rouen, LITIS, 76000 Rouen, France
*Corresponding author tel : +33614213824

Over the last years, significant progress in handwriting recognition and especially in optical models have been made thanks to deep learning advances [6], namely with the Long Short Term Memories (LSTM) Recurrent Neural Networks (RNN) [7]. LSTM networks achieve state-of-the-art performance in various applications involving sequence recognition, such as speech recognition [8], machine translation [9], or optical character recognition [10,11]. Performance of complete systems including both LSTM networks and linguistic resources [12] is due to the very high raw performance of the optical model (i.e. without using additional linguistic resource). For example, the raw performance of the optical model is about 35% WER on the RIMES dataset when using a BLSTM optical model solely. The contribution of the language model is then to penalize the wrong hypotheses produced by the optical model, so as to favor the most likely word sequences from the language model point of view. We believe that the raw performance of the LSTM based optical models provide hints that such networks should be used in a more specific way, and not only as a character classifier using a lexicon directed recognition approach, as it is the case in most of today's studies reported in literature.

Breaking the standard use of LSTM RNN as a simple classifier introduced in a lexicon-driven decoding scheme, this work proposes a new recognition paradigm that improves handwriting recognition state-of-the-art performance. This new paradigm is based on classifiers combination using an efficient decision rule operating at word level, which consists in lexicon verification. Lexicon verification consists in accepting a word hypothesis if it belongs to the lexicon, and rejecting it otherwise. The underlying idea is that it is very unlikely that a wrong word hypothesis belongs to the lexicon. The major advantage of this strategy is that it constitutes an extremely fast decision process, especially when compared to the tedious lexicon-driven decoding process which generally consists in a Viterbi beam-search [13]. Classifier combination is introduced using a cascade framework to combine multiple word classifiers. It is based on two key points: i) a lexicon verification decision process ii) a pool of complementary recognizers. We introduce a very efficient way to produce hundreds of complementary word recognizers in a very reasonable training time. Following the recent theoretical results in deep learning[14], we observed that multiple complementary networks can be obtained during a single training stage. We exploit this theoretical result to produce hundreds of complementary LSTM networks using a single training. We call this ensemble of networks a cohort. We show that the proposed strategy reaches very high performance whatever the size of the lexicon. As a consequence, the approach has no limitation regarding lexicon size, as demonstrated by the results obtained using a gigantic lexicon of more than 3 million words. We further explore this recognition paradigm to recognize text lines. We show that by introducing a lexicon verification operator in a ROVER [15] combination scheme, hundreds of LSTM networks hypothesis can be combined efficiently and achieve state-of-the-art performance on the Rimes dataset without introducing any language model.

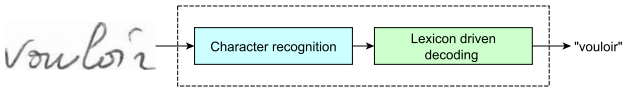

Fig. 1: The standard handwriting recognition paradigm.

This article is organized as follows : section 2 presents the state-of-the-art handwriting recognition review; section 3 is devoted to the lexicon verification and our approach made of a cascade of LSTM recurrent neural networks; section 4 describes the implementation of the method is described. Finally the results are presented and discussed on the Rimes and IAM datasets for isolated word recognition and text line recognition.

2 Related works

2.1 Handwriting recognition

Handwriting recognition models can be classified according to the character segmentation approach which can be either explicit (an algorithm specifically segments characters prior to their recognition), or implicit (characters are classified without prior segmentation)[1]. Handwriting models can also be classified according to the character recognition method (discriminant classifiers for hybrid approaches [16] or generative approaches for Hidden Markov Models (HMM) [17]). All these approaches rely on a lexicon-driven decoding stage. Indeed, since character recognition is not perfect, an efficient word recognition strategy is to postpone the character decision process until the end of the sequence recognition process, where the best character sequence hypothesis being a valid sequence is finally selected. This traditional scheme is represented in Figure 1.

2.2 The large vocabulary problem

When using lexicon-driven decoding, the character classification decisions are postponed until the end of the word so as to decide of the most probable word belonging to the working lexicon. On the one hand, lexicon

directed approaches allow to correct character recognition errors. On the other hand, they require using large lexicons in order to get high word coverage rates, and minimize Out Of Vocabulary (OOV) words. However, using very large lexicons requires pruning during the recognition phase, in order to get acceptable processing time. Results of the RIMES 2009 competition for isolated word recognition [10] show that most of the participants have implemented HMM based approaches for which we can observe a neat difference of the recognition rate between small, medium and large lexicon size with a maximum size of 5334 words. The lexicon's size has also a major role in the use of these methods in real applications where the lexicon is linked to a language that can contain hundreds of thousands of words. In specific cases, using such linguistic resources to improve recognition is not anymore compatible with real time constraints.

The large vocabulary problem is well known, as described in [3], where a vocabulary of 1000 words is considered to be a large lexicon. Today larger lexicons are considered. In [12] the authors used 50k, 12k and 95k word lexicons for the IAM, Rimes and OpenHaRT dataset respectively, but none of these lexicons are fully covering the evaluation set. To the best of our knowledge, the largest lexicon ever used is composed of 200 000 words [4] (60K words for [5]) using a n-gram model. But this is still generating problems as out of vocabulary words still remain. General purpose applications require unrestricted lexicons which may be composed of hundreds of thousands words (French Gutenberg dictionary has 336k words) to reach an acceptable coverage rate, but still without covering named entities, numbers, etc.

To overcome these limitations, solutions have been investigated: Pruning [18,19], lexicon free decoding using character language models as an alternate solution to using a lexicon [20,21,22,23,24], sub lexical units driven recognition [20,25,26]. Pruning methods exhibit problems such as excessive pruning, leading to increasing errors. Some methods[20,21] using hidden Markov models with other techniques like bag of symbols allow a lexicon free decoding , but they still have not reached the performance of the lexicon-driven approaches. Note that such approaches have also been used for numerical field recognition without lexicon [22,23,24], but digits are simpler to recognize than characters since they are generally isolated, and there are only 10 classes. Finally, in [20,25,27,26] the authors proposed a lexicon decomposition into prefix and suffix of the word's lexicon, modeled by n-grams. These methods are based on statistics extracted from a training corpus. The n-grams models reach state-of-the-art performance when dealing with out of lexicon word [25,27].

As reviewed in this paragraph, the very large vocabulary problem is still an open question. With the current fast progress in deep learning, many architectures are studied. Regarding handwriting and speech recognition, it is currently lead by the Long Short Term Memory recurrent neural networks.

2.3 Recurrent Neural Networks

Recurrent neural Networks (RNN), proposed more than 30 years ago with Hopfield networks [28], get their efficiency from their ability to process sequences, thanks to recurrent connections bringing information about the previous inputs or states in the sequence to the current position. However, for a long period of time, training recurrent neural networks suffered from the vanishing gradient problem [29]. As a consequence, long term dependencies can not be learned. Long Short Term Memory (LSTM) cells have thus been designed by Hochreiter et al. [7] in order to overtake this limitation. Many improvements still have been proposed on LSTM [30, 31] adding a gate and peepholes.Recently Gated Recurrent Units [32] have been proposed as an alternative to LSTM units, requiring less computation but with almost similar performance. In this paper we focus on LSTM cells.

LSTM cells enable to learn long or short term dependencies while processing sequences thanks to the introduction of gates (input, forget and output gates) followed by a sigmoid function, which controls the internal memory cell update (updating, resetting, expressing) by introducing a multiplier cell at each gate. RNN operate by processing the sequence in a particular direction, that is why bi-directionality has been introduced in RNN [33]. Then this idea has been extended to LSTM networks [8] to create bidirectional LSTM recurrent neural networks (BLSTM). However key applications like handwriting recognition are based on images which have two dimensions. In order to process images, multidimensional LSTM recurrent neural networks (MDLSTM) have been introduced [2].

LSTM networks only became popular once a new learning strategy was introduced, the Connectionist Temporal Classification (CTC)[34], which is a neural variant of the well known Forward Backward algorithm used for training HMM. CTC greatly helps training such networks by allowing embedded training of character models from the word or sentence label without the need for knowing the location of each character. This is working especially well thanks to the introduction of a "joker" class between characters, which allows the

network to postpone the decision until sufficient information is gathered along the sequence, so as to output the character hypothesis at one particular position in the input stream.

Today, LSTM recurrent neural networks are the state-of-the-art method for numerous sequence analysis applications, including optical character recognition [10, 35]. LSTM networks provide nearly binary posterior probabilities. One example of LSTM network outputs for the French word "demander" is given in Figure 2, for readability of the legend, only lowercase characters are shown, with the correct characters highlighted with colors. Such an output profile allows to apply a simple decoding scheme without lexicon, known as "Best path decoding" [36]. It takes the maximum a posteriori probability of the character class at each frame, and by removing every successive repetitions of each class (joker included), then the joker. The raw performance at the end of this lexicon free decoding scheme on a recognition task, although below state-of-the-art performance, are very high. For example, on the Rimes word isolated recognition task, standard MDLSTM-RNN [12] recognize 67% of the words, by using this simple best path decoding strategy.

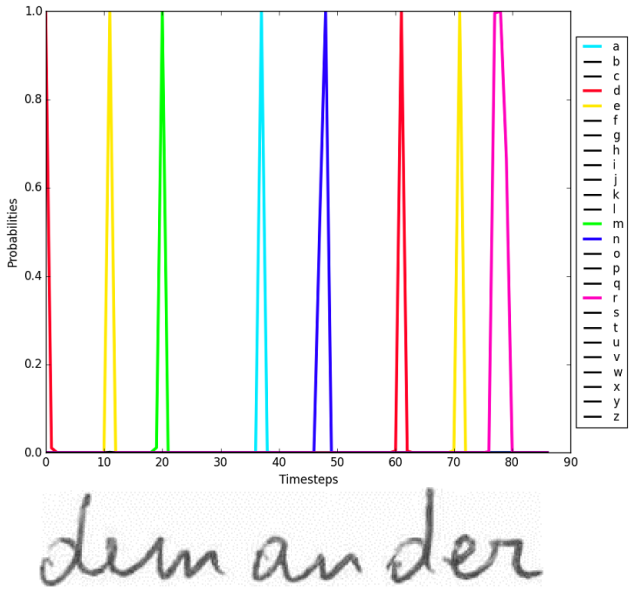

Fig. 2: LSTM network outputs at the end of a CTC training. The outputs form peaks for every character.

The LSTM RNN high performance shows that contribution to the recognition mostly comes from LSTM RNN (see Fig. 1). From these observations and considering the aforementioned problems inherent to the use of lexicon-driven recognition, in this paper we explore

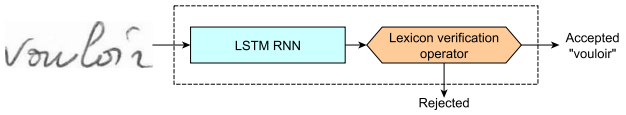

Fig. 3: Our new handwriting recognition paradigm.

a new paradigm for isolated handwritten word recognition. The following section is devoted to the presentation of this new strategy which is based on combining multiple character classifiers using a word lexicon verification rule of the sequence's hypotheses.

3 Proposed Approach

This paper proposes a new recognition paradigm which provides an efficient alternative to lexicon-driven decoding. This strategy is based on cascading complementary neural networks, combined with a reliable rejection stage based on lexicon verification (see Figure 3). This section describes the overall cascade architecture (Section 3.1), the proposed rejection stage (Section 3.2) and how complementary classifiers are generated (Section 3.3)

3.1 The cascade framework

The first element of our lexicon-driven decoding free approach is the cascade framework. Cascade of classifiers is a combination method that sequentially combines classifiers decisions by exploiting the complementary behavior of the classifiers, in order to progressively refine recognition decisions along the cascade. The most famous contribution on cascade of classifiers is from Viola and Jones [37], who applied their framework to face detection. The authors present a cascade based on a large ensemble of diverse and weak classifiers, which enable a fast decision process by sequentially introducing reliable decision stages. The performance of each classifier may be low but it must be associated to a high confidence decision stage that accepts or rejects the decision. In [38] and [39], an alternative cascade scheme is proposed by combining strong classifiers with different architecture and different input features. The strong classifier recognizes an important number of objects with a low error rate, while relying on a decision stage allowing to transfer rejects to the next classifiers.

This brief literature review shows that for both strategies, using either strong or weak classifiers, the strength of a cascade scheme comes from the reliability of the decision stage introduced after each classifier. In this paper we use the strong classifier approach, where we

combine hundreds of LSTM RNN. Classifiers are ordered by increasing error (i.e. by decreasing reliability) and a reliable decision stage is introduced based on a lexicon verification operator (see Figure 4). When the lexicon verification operator accepts an hypothesis, the classification process stops thus avoiding to use the whole set of classifiers most of the time. This decision mechanism is essential to control the performance of the cascade while enabling to significantly speed up the process, when many classifiers are involved.

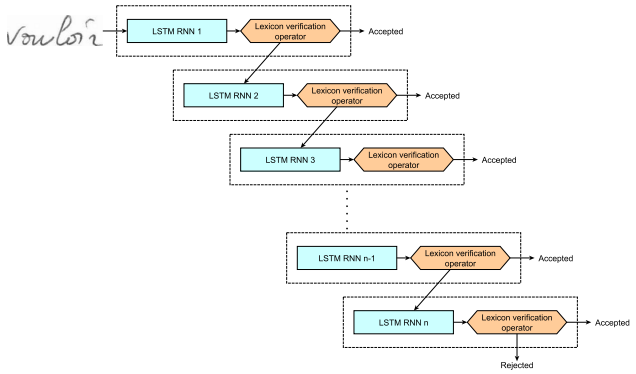

Fig. 4: The proposed BLSTM Cascade. Each classifier process the rejects from the previous layer.

The second important element of a cascade is the classifiers complementarity. Indeed, classifiers with similar behavior would not allow to refine the decision along the cascade. In this paper, we take benefit of the deep neural networks properties to get complementary classifiers during training of a single RNN, as discussed in section 3.3. We now discuss the decision reliability of our lexicon verification operator.

3.2 Lexicon verification operator reliability

The decision stage is made of a lexicon verification rule which consists in accepting a character string hypothesis if it belongs to the lexicon, or rejecting it otherwise. Such simple verification stage can serve as a decision in the cascade only if it provides reliable decisions and has a very low false acceptance rate.

A False Acceptance (FA) of our lexicon verification rule occurs when a wrong character string hypothesis produced by the recognizer matches one entry of the lexicon. The probability of false acceptance P_{FA} for a given recognizer can be expressed as described in equation (1) where W is a word hypothesis, L a lexicon, and *Reco* a recognizer. "W **is** erroneous" is *true* when the word hypothesis W is wrong, and "$W \in L$" is *true* when the word hypothesis W belongs to the lexicon.

$$P_{FA} = P(W \textbf{ is } \text{erroneous } \wedge W \in L \mid Reco) \quad (1)$$

Figure 5 shows the estimated probability P_{FA} of a single recognizer with respect to word length n. The probability was estimated by taking the results of one network trained on the Rimes dataset. One can observe that the probability P_{FA} (blue curve) is high for short words (less than 4 characters) and therefore that the verification rule is not reliable enough. That is why we introduce the Minimum Number of Decision Agreement (MNDA), that allows to drastically reduce the probability of false acceptance to a very acceptable level. This MNDA is the minimum number of classifiers that must take the same classification decision, among the classifiers that have been activated in the cascade. As can be observed on magenta and cyan curves from Figure 5, using a MNDA of 2 and 3 leads to a very low estimated probability (below half a percent), even for short words.

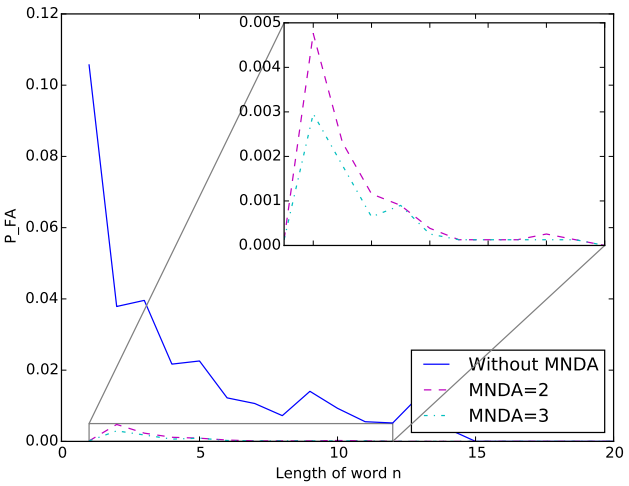

Fig. 5: Estimated P_{FA} over the length of word n for a given LSTM recognizer with and without Minimum Number of Decision Agreement.

Let us now analyze the effect of the lexicon verification solely (MNDA = 0) on a word recognition task. In this respect, Table 1 shows the performance obtained on the Rimes dataset using a lexicon free BLSTM recognizer, with and without the simple verification rule. We see that adding the verification rule to the BLSTM makes the error dramatically decreases by 93%, from 33.63% to only 2.25%. By looking at the remaining confusions, most of the errors are type case, accents and plural errors. This first experiment demonstrates the

strength of a verification strategy in combination with BLSTM classifiers.

Network	Accuracy	Error	Rejection
BLSTM	66.37	33.63	0
BLSTM + verif.	66.37	**2.25**	31.38

Table 1: Accuracy, error rate and rejection rate of a lexicon-free BLSTM recognizer on the Rimes dataset, with and without the lexicon verification strategy.

We have shown that Lexicon Verification can significantly reduce recognition errors, and that the number of False Acceptance can be reduced by adding a Minimum Number of Decision Agreement. In the design of a cascade we introduce a rejection rule by combining a lexicon verification operator with Minimum Number of Decision Agreement. Such an operator will serve as an efficient rule to control the decisions at each stage of the cascade. In the next section, we investigate how to easily generate complementary LSTM RNN.

3.3 Generation of complementary LSTM RNN

There are many ways to generate complementary LSTM neural networks. The first one is by using different architectures (BLSTM vs MDLSTM, changing the number of layers or neurons, etc.) or using different input features (pixels, Histogram of Gradients, etc.). However these modifications are costly in terms of design, training time, and limited in number.

Some previous experiments have shown that similar LSTM recurrent neural networks trained with different initial weights can be combined with success [11]. These networks have similar recognition rates, but the connection weights are different and therefore they have some complementarity properties which allow to combine them with success. However training many networks starting with different initializations is slow as it takes up to a week to train a single network.

In order to get as much complementary networks as possible with a limited effort considering both time in design and training, the idea lies in controlling the training phase of deep neural networks, and is inspired by the work of Choromanska et al. in [14]. This work makes a parallel between fully-connected neural networks loss function and high degree random polynomials which have a huge number of local minima of same order of magnitude. The authors conclude that when training a large neural network, reaching a local minimum is nearly similar to reaching the global optimum. As a consequence, exploring these local minima may be a simple but relevant strategy to obtain a large amount of complementary networks that perform equally well, but with different local properties. We call the ensemble of networks obtained by this strategy a cohort, and we use the obtained cohort to feed a cascade.

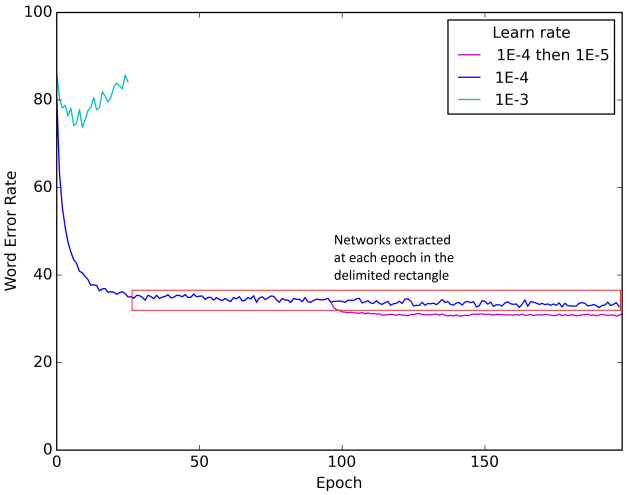

Fig. 6: Word Error Rate over epoch of a network trained on the Rimes dataset with learning rate: fixed at 10^{-4} (blue), at 10^{-4} then 10^{-5} at epoch 96 (magenta) and at 10^{-3} (cyan).

However, this easy and fast strategy to get many complementary networks requires some attention on the training parameters in order to get the desired property. Indeed training must avoid to be trapped in one local minimum in order to get complementary classifiers. Training a neural network using steepest gradient descent is controlled by three important parameters which are the learning rate, the momentum value and dataset shuffling. Momentum is the factor that control the weights update and thus the convergence of the training procedure. We use a fixed momentum of 0.9, which is commonly used in the literature, in order to help escaping local minima. When training networks, the examples are shuffled between each epoch in order to get a better convergence by preventing cycles. As training examples are randomly shuffled between epochs, we can consider that the state of the network (the weights' value) is randomly reached from one epoch to another. The last and maybe most important parameter is the learning rate. It is the weight associated to the value of the gradient that serve as the update rule of the network weights. Figure 6 shows the impact of the learning rate on training a BLSTM network defined in 4.2 on the Rimes dataset (see 4.1). A large learning rate does not allow convergence of the network, as shown on

Figure 6 (cyan curve) where the network does not converge properly with a learning rate of 10^{-3}. A too small learning rate leads to very small changes of the weights values and does not allow reaching another local minimum, and as a consequence gives no complementary networks between epochs. As shown on Figure 6 on the magenta curves decreasing the learning rate to 10^{-5} at the epoch 96 (taken arbitrarily but any other epoch at this stage would have been suited) produces less fluctuation of the WER leading to less variability and then nearly no complementarity. The desired behavior is obtained for a learning rate set to 10^{-4}.

By choosing these parameters, the networks selected at each epoch of one single complete training phase can constitute a cohort. The red rectangle on Figure 6 shows the region where the networks can be selected, where training has converged to reach a region with no overall significant decrease of the WER, but with small fluctuations that highlight the various local minima reached, thus providing networks with different local properties. The proposed training strategy although simple is very effective to get complementary networks.

At this stage, we have proposed a lexicon verification rule that can serve as an effective reliable rejection rule, and an easy mean to get hundreds of complementary LSTMM RNN during a single training session. Let us now describe the implementation details of the cascade.

4 Cascade implementation and results

Based on the methodological principles established in the previous section, we now detail our implementation and present the results achieving state-of-the-art performance. We also analyze the method regarding both the complementarity between networks and the lexicon verification stage in terms of performance.

4.1 Dataset and Evaluation

The experiments have been carried out on the Rimes[10] and IAM [40] isolated words datasets and using the official splits and lexicons provided with these datasets. The character sets contain upper and lower case characters of the Latin alphabet, symbols that may occur in a word like "'" or "-", and many others. For the Rimes dataset there is also accented characters, and the evaluation is made on lower case as in [11] for comparison purpose, and also because of some ground truth ambiguity [1]. The lexicons for both datasets are composed of all the words of the training (Rimes 51738 running words, Iam 66153 running words), validation (Rimes 7464 running words, Iam 7350 running words) and test (Rimes 7776 running words, Iam 17586 running words) sets, as used in the competitions and related papers, giving lexicons size of 5744 and 12202 words for Rimes and IAM respectively. For the Rimes dataset, some additional lexicons are used in order to evaluate lexicon sensitivity. They are the test dataset lexicon (1692 words) and the training lexicon (5334 words). Notice that except for the training lexicon, the lexicons have a 100% coverage rate as it is the case for every evaluation conducted for related isolated word recognition tasks in the literature.

We also evaluate the performance of our approach on very large lexicons that better represent what can be found in real world industrial applications (e.g. There exist millions of named entities for addresses : countries, regions, towns, street names, names,...). In this respect, we have collected by ourselves two extremely large lexicons that are available on request:

- The first one is the union of the lexicons from the french Wikipedia, Wiktionnaire, French dictionnary Gutemberg and the Rimes dataset, leading to a French lexicon of **3,276,994** words;
- The second one is the union of the lexicons found in the one billion word data [41] and the IAM dataset, leading to an English lexicon of **2,439,432** words.

For all the experiments, we measure the recognition performance with the Character Error Rate (CER), calculated with the Levenshtein distance. We also measure the accuracy, the error rate and the rejection rate for isolated word recognition. We also measure Word Error Rate for text line recognition.

4.2 Architecture

Two well established LSTM RNN architectures have been used for the experimentations: The first is a BLSTM architecture inspired from [42]: it is a two layers network made respectively of 70 and 120 LSTM blocks, separated by a subsampling layer of 100 hidden neurons without bias. The network also has two layers to reduce the sequence length. The first one concatenates the input vectors in pairs, while the second one concatenates the output vectors of the first layer in pairs. As input features, we use histogram of oriented gradients (HOG) [43] that have demonstrated their efficiency for

[1] Namely on certain upper case characters, especially for letter "**j**" in the word "**je**" or "**j'**" where many ground truth errors occur in the dataset.

handwriting recognition [44]. Images are normalized to 64 pixels height. A sliding window of 8 pixel width extracts the HOG features at a 1 pixel pace. The second architecture is a MDLSTM architecture similar to the reference in [2]. This architecture is composed of three layers of respectively 2, 10 and 50 LSTM cells, and two subsampling layers of 6 and 20 neurons. Raw images are normalized and directly fed to the network. The decoding time is 10 milliseconds per word on average (on an intel i7-3740QM CPU).

For both architectures we use a lexicon free "Best path decoding" algorithm [36] described in (2.3) to retrieve the characters string. Training and decoding have been performed with RNNLIB [45].

4.3 Generating a cohort of LSTM RNN

To get a cohort of complementary LSTM RNN, the three following strategies are followed: (i) Our training trick described in section 3.3, for which we get one network per epoch ; (ii) Different starting initialization ; (iii) Two different architectures BLSTM and MDLSTM.

Moreover, we use a common trick to improve performance of neural networks by performing transformations over the training set, in order to increase the size of the training set. By using rotations and warping, we multiply the size of the training set by 3, and as shown in Figure 7 the word error rate improves. Beside providing a lower error, we observe that the transformations also bring more fluctuations during training by adding more training samples between two epoch, thus increasing the number of weight updates and the potential local differences between two network between two epochs.

2100 different networks are collected from only ten different trainings on the Rimes dataset with different random initializations:

- 2 BLSTM trainings;
- 1 MDLSTM training;
- 4 BLSTM trainings with transformations;
- 3 MDLSTM trainings with transformations.

Training these 10 networks with 3 computers on CPU only took 10 days, while training 2100 networks with different initialization would have taken 2 years and 9 months. Regarding the IAM dataset, we also trained two networks with transformations (tripling the size of train set), two BLSTM and two MDLSTM, leading to a total of 1039 networks for this task. The purpose of doing several training is threefold, it allows us to: i) get more networks with training run in parallel, (ii) check successfully that the method doesn't depend on the initial weights or architecture selected, and (iii) improve performance thanks to the two architectures and using different features.

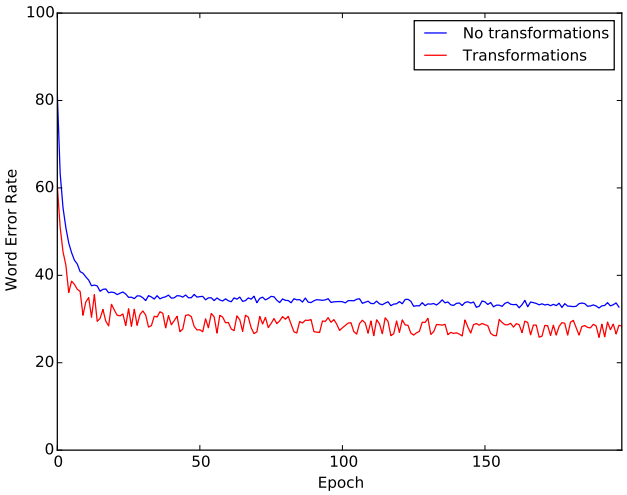

Fig. 7: Comparisons of Word Error Rate over of two networks with and without transformations (higher fluctuations amplitude) with learn rate fixed at 10^{-4}.

4.4 Results

4.4.1 Performance of cascades built from a single cohort

In this section, cascades are built using networks from a single cohort of RNN. Cascade from single cohort of BLSTM, BLSTM with transformations, and MDLSTM are considered. We select 100 networks to form each cohort and combine each cohort in a cascade, using the lexicon verification operator.

When combining hundreds of networks into a cascade, the performance depends on both the classifier sorting, and the value of MNDA in the decision stage. Ordering the classifiers in the cascade is made according to the deletion error criterion (rate of omitted characters) on the validation set. Tuning the MNDA parameter was carried out by considering long and short words.

As previously seen in Figure 5, we observe that short words (3 and less characters) are more prone to mistakes, justifying the fact that short and long words will be considered differently in the rejection stage of the cascade. In these experiments we have chosen a MNDA of 3 for long words and 10 for short words, however, small deviations from the values do not affect the results much. For example when selecting MNDA in the range 2 to 10 for long words, and MNDA in the range

5 to 40, we observed no modification of the CER, while the WER is slightly modified by 0.25% on the validation set. It would also possible to optimize these parameters on the validation data set.

In Table 2, the first line is a reminder of the performance obtained with a single BLSTM during the preliminary experiments (See section 3.2). A Word Recognition Rate of 66% was obtained. When using transformations, the performance improves to 71%. Combining 100 networks highly improves the performance up to 84.5%, while combining transformations with 100 BLSTM increase the performance to 89.56%. Finally, it appears that improvement does not depend on the type of network nor on the input features used. For example we observe a similar significant improvement of the performance when using MDLSTM, as reported on the last line of Table 2.

The study of the mis-accepted words shows that the majority of the errors comes from accented letters ("a" and "à" or "demande" and "demandé"), case sensitivity ("je" and "Je"), plurals ("conseillers" and "conseiller") and close words ("tiers" and "tiens"). We now analyze the results for cascades composed of classifiers from multiple cohorts, in the hope to increase further the complementarity within the cascade and then reducing the errors.

4.4.2 Performance of cascades with multiple cohorts

We evaluate the cascade of cohorts of LSTM for the Rimes and IAM datasets. First we report the results on the Rimes dataset. By gathering the 10 cohorts described in section 4.3, 2100 networks are combined in the cascade.

Table 3 presents the results for different lexicon sizes. The performance gap between small (1692 words) and large (5744 words) lexicons is very low ($\simeq$ 0.48 points) compared to the traditional performance drop observed. The performance gap between the large (5744 words) and the gigantic ($>$ 3M words) lexicon is more important, however it remains rather low with a gap of 5.71%. Both results prove the low sensitivity of the approach to the lexicon size. Moreover, it is to be noticed that the cascade built using the training lexicon only obtains good results due to the capacity of the approach to generate more rejects. Finally we observe that when using the French dictionary in addition to the training lexicon similar performance are obtained than when using the test lexicon in addition to the training lexicon and French dictionary.

Regarding the gigantic lexicon, the system provides very interesting accuracy (90.14%) as the lexicon is nearly 600 times larger than the Rimes lexicon. Such lexicon size opens new perspectives for processing named entities (person, street and city names, numbers, etc.) or multilingual documents for example. To the best of our knowledge, there is no other reference in the literature using such lexicon size.

At the end of the cascade there are some remaining rejected words. These rejects can be processed with a lexicon-driven decoding method. Notice that generally, for practical applications like automated postal sorting, it is interesting not to process rejects because the cost of a wrong decision is higher than processing it by a human operator. To compare the results with other state-of-the-art methods, which have no rejection stage, we use a Viterbi lexicon decoding on the rejected words at the end of the cascade. As shown in Table 4, we achieve state-of-the-art performance for a 5744 words lexicon on the Rimes dataset. For doing so, character posterior probabilities are estimated by the average class posterior probabilities of ten LSTM networks randomly picked among the cohort, before running the Viterbi decoder. Averaging the output probabilities from different networks improves the performance (in comparison to taking the output of only one network). Notice that the number of networks randomly selected has a very low impact on the performance above a certain threshold. Selecting 5, 10, 20 or 40 networks yields nearly similar results. To draw this conclusion we have computed the CER and error rate on the validation set for randomly selected sets of 5, 10, 20, 40 networks among the cohorts and get a standard deviation below 0.03 on the CER and error rate. The gap between this study and the results presented in [26], both in terms of CER and error rate is significant with an absolute decrease of 0.56 (29%) of the CER and 0.42 (11%) of the error rate.

Table 4 also stress the low performance obtained by a standard lexicon directed Viterbi decoding using one single BLSTM and using a very large lexicon. This demonstrates the weakness of the traditional recognition paradigm. One can observe the large increase of the CER (80% increase) and the accuracy decrease (4.6% decrease) when decoding with a lexicon of 342275 words compared to decoding with a lexicon of 5744 words. As a comparison we can notice that our approach performs almost similarly for the two lexicon sizes with only a 2.6% decrease of the accuracy. Table 5 presents results on the IAM dataset. Both CER and error rate are better than state-of-the-art by respectively 0.66 (19% decrease) and 0.52 (8% decrease). The result for the gigantic lexicon is also impressive with 85% of words recognized with a lexicon 200 times larger than the initial one.

System	Accuracy	Error	Rejection	CER
Single BLSTM	66.37	33.63	0	11.24
Single BLSTM (+transformations)	71.78	28.22	-	8.89
Cascade of 100 BLSTM	84.53	3.13	12.34	0.99
Cascade of 100 BLSTM (+transformations)	89.56	2.74	7.70	0.82
Cascade of 100 MDLSTM	80.27	4.09	15.64	1.40

Table 2: Accuracy, error rate, rejection rate and Character Error Rate of different Systems on the Rimes dataset with the 5744 words lexicon (training and test). Networks from the cascade are extracted from a single training procedure (i.e. cohort).

Verification lexicon	Lexicon size	Accuracy	Error	Rejection	CER
Test	*1692*	96.08	2.32	1.60	0.76
Training + Test	*5744*	95.60	**3.00**	1.40	0.99
Training	*5 334*	91.32	3.68	5.00	1.26
Training + Test + Dictionary	*342 275*	93.41	5.47	1.12	1.72
Training + Dictionary	*341 865*	92.01	5.77	2.21	1.85
All lexicons + wikipedia	*3 276 994*	90.14	9.18	0.68	2.67

Table 3: Results of the 2100 networks cascade on the Rimes dataset for various lexicon sizes.

System	Lexicon size	Accuracy	Error rate	CER
One BLSTM + Viterbi	5744	91.48	8.52	2.77
One BLSTM + Viterbi	342 275	87.24	12.76	5.00
Cascade 2100 + Viterbi	5744	**96.52**	3.48	**1.34**
Cascade 2100 + Viterbi	342 275	93.96	6.04	2.11
Menasri et al. [11]	5744	95.25	4.75	-
Poznanski et al. [26]	5744	96.10	3.90	1.90

Table 4: Results of the 2100 networks cascade on Rimes dataset with Viterbi compared to state-of-the-art methods.

System	Lexicon size	Accuracy	Error rate	Rejection rate	CER
This work	*12202*	92.46	**5.04**	2.50	2.08
This work	*2 439 432*	85.51	13.30	1.19	4.77
This work + Viterbi	*12202*	**94.07**	5.93	-	**2.78**
Poznanski et al. [26]	*12202*	93.55	6.45	-	3.44

Table 5: Results of the 1039 networks cascade on the IAM dataset.

4.5 Processing time

We analyze now one potential drawback of the approach, which requires many networks and thus a large amount of processing time. However, the cascade architecture is very effective regarding computation cost, because once a candidate is accepted by the verification stage, it does not pass through the rest of the cascade. Regarding the lexicon verification processing time, it is below one microsecond, even considering the gigantic lexicons. Finally, when considering our system with 2100 networks, the mean processing time per word is 729 milliseconds. 80% of the words are processed in less than 0.175s, after 14 networks or less, and 90% in less than half a second (after 41 networks or less)[2]. As our code has not been optimized nor parallelized, we believe that there is room for strong improvements. Considering memory issues, the networks involved in these experiments do not require a lot of memory space, as the whole set 2100 networks fit into 6GB, even with double precision encoding (64 bits).

Above timing consideration, one would ask whether the whole cohort of networks is necessary, or if a subset of them would perform similarly, which in this case would also alleviate the processing time. This point is addressed in the next section.

4.6 Pruning classifiers from the cohort

The aim of the selection is to decimate the cohort of classifiers so as to keep the most efficient reduced set of classifiers. This has been achieved by simply removing the networks providing the poorest performance on the

[2] evaluated on an Intel CPU i7-3740QM.

validation dataset. Networks which don't recognize new words, or which yield too much false acceptance are removed. By doing so, we manage to reduce the number of networks from 2100 to 118. We observed a very acceptable performance drop compared to the cohort of 2100 networks with an accuracy of 92.96%. Nearly similar results are obtained when using Viterbi decoding of the rejects: the WER is **3.64%** (previously 3.48%) and the CER is **1.49%** (previously 1.34%). Note that this reduced architecture still improves state-of-the-art performance. The mean processing time of this pruned cohort is decreased to 197 milliseconds (previously 729ms).

5 Extending the cohort principle and lexicon verification to handwritten lines recognition

The previous results are very interesting and we wonder how to extend the cohort principle and the lexicon verification to operate at line level. Text lines are more difficult to recognize because the system has to cope with line segmentation into words in addition to the recognition. The word segmentation is provided implicitly by optical character recognizers and explicitly by lexicon-driven decoding methods. However the word segmentation may vary between different recognizers like LSTM RNN, or different lexicon-driven methods. In order to combine multiple recognition systems, the state-of-the-art method is the Recognizer Output Voting Error Reduction (ROVER) [15]. ROVER proceeds the outputs of multiple classifiers with two modules: alignment and voting. ROVER combines multiple recognizers by aligning the multiple recognition outputs with dynamic programming. Then, the ROVER voting module selects the best solution according to the frequencies of word hypothesis and their recognition score. The ROVER voting module is closely related to the voting module that we have introduced in our cascade combination scheme. The main difference being the lexicon verification operator which was very relevant in our case for combining many recognizers. This is why we slightly modified the ROVER voting module so as to introduce the lexicon verification operator in the ROVER voting module. This can be done simply by giving a score to each word hypothesis depending whether it belongs to the lexicon or not. This score is substituted in place of the classifier score that is associate to each word hypothesis.

Then we perform this new Lexicon Verification ROVER scheme by combining multiple LSTM RNN cohorts. For that, we generate four LSTM RNN cohorts on the Rimes text line dataset, as we did with the cascade. We use the Rimes text line training dataset lexicon only, as a consequence there are OOV words on the test dataset. The evaluation uses the same methods as in [12,46] to compute the Word Error Rate (WER) and the CER. 454 networks of the 4 cohorts have been extracted to perform the ROVER combination. Results are presented in Table 6. We compare our results to two state-of-the-art methods [46,12] which are classic recognition systems with a LSTM RNN followed by a lexicon-driven decoding and a n-gram language model train on the Rimes corpus. We achieve state-of-the-art performance with the proposed ROVER combination with the the cohort principle and the lexicon verification operator. These results are very interesting as they outperform standard methods that use lexicon-driven decoding with language models.

System	CER	WER
ROVER + cohort + LV (this work)	**2.6**	**8.4**
Voigtlaender et al. [46]	2.8	9.6
Pham et al. [12]	3.3	12.3

Table 6: Results of a ROVER combination with lexicon verification of 454 networks on the Rimes dataset.

To validate these results we carried out the same experiment on the IAM text line dataset, on which we extracted 377 networks from four cohorts. The modified ROVER combination results are presented in table 7. Contrary to other methods in the literature [12,46,47], we only use the lexicon of the LOB corpus. Whereas methods like [12,46,47] rely on many English corpus (LOB, Brown, Wellington) to learn powerful language models. These corpora have syntactical properties that fit the IAM dataset very well. As one can see in table 7, our results are comparable to [12] on the validation set, which contains easier examples on which the LSTM RNN performs better, but these results are worse on the test set. The difficulty that our method faces in this situation is that it relies mostly on the performance of the LSTM RNN. The CER of a LSTM RNN on the IAM dataset is about 3 points higher than on the Rimes dataset. We deduce that the performance of the LSTM RNN is the key point to perform better than systems based on language models in a context where many linguistic resources are available. In [47], the authors built a ROVER combination of BLSTM with neural network language models (NN LMs) and ANN/HMM with the same NN LMs. In this study, we obtain better results than this previous state-of-the-art method without using any language model.

Since the performance of the LSTM RNN model is the key issue, we have also applied the cohort combination scheme using identical convolutionnal and recurrent neural networks architectures (denoted CNN

LSTM) as presented in [48]. In this experiment we have trained two identical CNN LSTM from which we extracted a cohort of 164 CNN LSTM following our methodology. The results are shown in table 7. By combining the CNN LSTM cohort with ROVER we significantly improved (2.3 decrease of CER on eval) the performance obtained during previous experiments, and we now reach nearly the state of the art results. This experiment with CNN LSTM shows that the cohort principle can be extended to other neural network architectures than BLSTM and MDLSTM. That is to say that other tasks where neural network are used like document segmentation, object detection or speech recognition may benefit from the cohort principle to improve performance.

System	WER dev	WER eval	CER dev	CER eval
ROVER + cohort (BLSTM) + LV (this work)	10.9	16.1	3.9	6.7
ROVER + cohort (CNN LSTM) + LV (this work)	7.3	11.5	**2.4**	4.4
Voigtlaender et al. [46]	**7.1**	**9.3**	**2.4**	**3.5**
Pham et al. [12]	11.2	13.6	3.7	5.1
Zamora-Martinez et al. [47]	-	16.1	-	7.6

Table 7: Results of a ROVER combination with lexicon verification of 377 networks on the IAM dataset.

These results show that the concept of cohort controlled by a lexicon verification operator can be implemented easily for text line recognition. The results outperform the state of the art on the Rimes text line dataset, whereas good results are obtained on the IAM text line dataset. This demonstrates the interest of our proposition when language models may be difficult to estimate, in the presence of few training data.

6 Conclusion

This works presents a new recognition paradigm that substitute the traditional lexicon directed recognition by a lexicon verification procedure. It is exploited in a cascade framework involving hundreds of LSTM recurrent neural networks. The networks are obtained during a single training procedure, therefore requiring no hyper-parameter optimization and a limited training duration. As we have seen, the combination of hundreds of networks called a cohort lead to a great improvement of the performance both for isolated word recognition and text line recognition.

Another important part of the success of the cascade combination is due to the lexicon verification operator.The combination shows a low probability of false acceptance, and has also a very low sensitivity to lexicon size. Besides, our method runs with gigantic lexicons without extra processing time, which has never been done before. We successfully achieve state-of-the-art results for both Rimes and IAM datasets of isolated words.

We also proposed a modified ROVER combination scheme that can benefit from combining a cohort of classifiers controlled by a lexicon verification voting scheme. We achieved state-of-the-art result on the Rimes dataset, although the IAM results are less convincing when using the same feature set due to the weaker performance of the BLSTM network, the results have significantly improved when using a CNN LSTM architecture, reaching near state of the art results. Combining the cohort principle with the lexicon verification strategy performs very well in situations were resources, like training corpora, are rare. As the lexicon size is not an important issue anymore, it also opens new perspectives for large lexicon applications such as processing of multilingual documents, or named entity recognition. Moreover the cohort principle could be extended to other network architectures like pure CNN in the aim to get performance gains in other computer vision tasks.

References

1. R. Plamondon, S. N. Srihari, Online and off-line handwriting recognition: a comprehensive survey, IEEE PAMI 22 (1) (2000) 63–84.
2. A. Graves, J. Schmidhuber, Offline handwriting recognition with multidimensional recurrent neural networks, in: NIPS, 2009, pp. 545–552.
3. A. L. Koerich, R. Sabourin, C. Y. Suen, Large vocabulary off-line handwriting recognition: A survey, Pattern Analysis & Applications 6 (2) (2003) 97–121.
4. M. Hamdani, P. Doetsch, M. Kozielski, A. E.-D. Mousa, H. Ney, The rwth large vocabulary arabic handwriting recognition system, in: IAPR International Workshop on Document Analysis Systems, 2014, pp. 111–115.
5. T. Bluche, J. Louradour, M. Knibbe, B. Moysset, M. F. Benzeghiba, C. Kermorvant, The a2ia arabic handwritten text recognition system at the open hart2013 evaluation, in: Document Analysis Systems, 2014, pp. 161–165.
6. Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.
7. S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780.
8. A. Graves, J. Schmidhuber, Framewise phoneme classification with bidirectional lstm and other neural network architectures, Neural Networks 18 (5) (2005) 602–610.
9. I. Sutskever, O. Vinyals, Q. V. Le, Sequence to sequence learning with neural networks, in: NIPS, 2014, pp. 3104–3112.
10. E. Grosicki, H. El Abed, Icdar 2009 handwriting recognition competition, in: ICDAR, 2009, pp. 1398–1402.

11. F. Menasri, J. Louradour, A.-L. Bianne-Bernard, C. Kermorvant, The a2ia french handwriting recognition system at the rimes-icdar2011 competition, in: Document Recognition and Retrieval XIX, 2012, pp. 82970Y–82970Y.
12. V. Pham, T. Bluche, C. Kermorvant, J. Louradour, Dropout improves recurrent neural networks for handwriting recognition, in: ICFHR, 2014, pp. 285–290.
13. L. Fissore, G. Micca, R. Pieraccini, P. Palace, Strategies for lexical access to very large vocabularies, Speech Communication 7 (4) (1988) 355–366.
14. A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, Y. LeCun, The loss surfaces of multilayer networks., in: AISTATS, 2015.
15. J. G. Fiscus, A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover), in: Automatic Speech Recognition and Understanding, 1997, pp. 347–354.
16. A. Senior, T. Robinson, Forward-backward retraining of recurrent neural networks., NIPS (1996) 743–749.
17. A. El-Yacoubi, M. Gilloux, R. Sabourin, C. Y. Suen, An hmm-based approach for off-line unconstrained handwritten word modeling and recognition, IEEE PAMI 21 (8) (1999) 752–760.
18. S. Madhvanath, V. Govindaraju, Holistic lexicon reduction for handwritten word recognition, in: Document Recognition III, 1996, pp. 224–234.
19. S. Madhvanath, V. Krpasundar, V. Govindaraju, Syntactic methodology of pruning large lexicons in cursive script recognition, Pattern Recognition 34 (1) (2001) 37–46.
20. A. Brakensiek, J. Rottland, G. Rigoll, Handwritten address recognition with open vocabulary using character n-grams, in: IWFHR, 2002, pp. 357–362.
21. A. Bharath, S. Madhvanath, Hmm-based lexicon-driven and lexicon-free word recognition for online handwritten indic scripts, IEEE PAMI 34 (4) (2012) 670–682.
22. M. Shridhar, G. Houle, F. Kimura, Handwritten word recognition using lexicon free and lexicon directed word recognition algorithms, in: ICDAR, Vol. 2, 1997, pp. 861–865.
23. C. Chatelain, L. Heutte, T. Paquet, A two-stage outlier rejection strategy for numerical field extraction in handwritten documents, in: ICPR, Vol. 3, 2006, pp. 224–227.
24. C. Chatelain, L. Heutte, T. Paquet, Segmentation-driven recognition applied to numerical field extraction from handwritten incoming mail documents, in: Document Analysis System, 2006, pp. 564–575.
25. M. Hamdani, A. E.-D. Mousa, H. Ney, Open vocabulary arabic handwriting recognition using morphological decomposition, in: ICDAR, IEEE, 2013, pp. 280–284.
26. A. Poznanski, L. Wolf, Cnn-n-gram for handwriting word recognition, in: CVPR, 2016, pp. 2305–2314.
27. M. Kozielski, D. Rybach, S. Hahn, R. Schlüter, H. Ney, Open vocabulary handwriting recognition using combined word-level and character-level language models, in: ICASSP, pages=8257–8261, year=2013,.
28. J. J. Hopfield, Neural networks and physical systems with emergent collective computational abilities, Proceedings of the national academy of sciences 79 (8) (1982) 2554–2558.
29. S. Hochreiter, The vanishing gradient problem during learning recurrent neural nets and problem solutions, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 6 (02) (1998) 107–116.
30. F. A. Gers, J. Schmidhuber, F. Cummins, Learning to forget: Continual prediction with lstm, Neural computation 12 (10) (2000) 2451–2471.
31. F. A. Gers, N. N. Schraudolph, J. Schmidhuber, Learning precise timing with lstm recurrent networks, Journal of Machine Learning Research 3 (2003) 115–143.
32. J. Chung, C. Gulcehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, arXiv preprint arXiv:1412.3555.
33. M. Schuster, K. K. Paliwal, Bidirectional recurrent neural networks, Signal Processing, IEEE Transactions on 45 (11) (1997) 2673–2681.
34. A. Graves, S. Fernández, F. J. Gomez, J. Schmidhuber, Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks, in: ICML, 2006, pp. 369–376.
35. H. El Abed, V. Margner, M. Kherallah, A. M. Alimi, Icdar 2009 online arabic handwriting recognition competition, in: ICDAR, 2009, pp. 1388–1392.
36. A. Graves, Supervised sequence labelling with recurrent neural networks, Vol. 385, Springer, 2012.
37. P. Viola, M. Jones, Rapid object detection using a boosted cascade of simple features, in: CVPR, Vol. 1, 2001, pp. I–511.
38. B. Zhang, Reliable classification of vehicle types based on cascade classifier ensembles, Intelligent Transportation Systems 14 (1) (2013) 322–332.
39. P. Zhang, T. D. Bui, C. Y. Suen, A novel cascade ensemble classifier system with a high recognition performance on handwritten digits, Pattern Recognition 40 (12) (2007) 3415–3429.
40. U.-V. Marti, H. Bunke, The iam-database: an english sentence database for offline handwriting recognition, IJDAR 5 (1) (2002) 39–46.
41. C. Chelba, T. Mikolov, M. Schuster, Q. Ge, T. Brants, P. Koehn, T. Robinson, One billion word benchmark for measuring progress in statistical language modeling, arXiv preprint arXiv:1312.3005.
42. L. Mioulet, G. Bideault, C. Chatelain, T. Paquet, S. Brunessaux, Exploring multiple feature combination strategies with a recurrent neural network architecture for off-line handwriting recognition, in: Document Recognition and Retrieval, 2015, pp. 94020F–94020F.
43. N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: CVPR, Vol. 1, 2005, pp. 886–893.
44. G. Bideault, L. Mioulet, C. Chatelain, T. Paquet, Spotting handwritten words and regex using a two stage blstm-hmm architecture, in: Document Recognition and Retrieval, 2015.
45. A. Graves, Rnnlib: A recurrent neural network library for sequence learning problems, `https://sourceforge.net/projects/rnnl`.
46. P. Voigtlaender, P. Doetsch, H. Ney, Handwriting recognition with large multidimensional long short-term memory recurrent neural networks, in: ICFHR, 2016, pp. 228–233.
47. F. Zamora-Martinez, V. Frinken, S. España-Boquera, M. J. Castro-Bleda, A. Fischer, H. Bunke, Neural network language models for off-line handwriting recognition, Pattern Recognition 47 (4) (2014) 1642–1652.
48. B. Shi, X. Bai, C. Yao, An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition, IEEE transactions on pattern analysis and machine intelligence 39 (11) (2017) 2298–2304.