Pattern Recognition 47 (2014) 1642–1652

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Neural network language models for off-line handwriting recognition

F. Zamora-Martínez [a,b,*], V. Frinken [c], S. España-Boquera [b], M.J. Castro-Bleda [b], A. Fischer [d], H. Bunke [d]

[a] Escuela de Enseñanzas Técnicas, Departamento de Ciencias Físicas, Matemáticas y de la Computación, Universidad CEU Cardenal Herrera, C/San Bartolomé 55, 46155 Alfara del Patriarca, Valencia, Spain
[b] Departamento de Sistemas Informáticos y Computación, Universitat Politècnica de València, Valencia, Spain
[c] Faculty of Information Science and Electrical Engineering, Kyushu University, Japan
[d] Institute for Computer Science and Applied Mathematics, University of Bern, Bern, Switzerland

ARTICLE INFO

Article history:
Received 5 November 2012
Received in revised form
17 August 2013
Accepted 22 October 2013
Available online 4 November 2013

Keywords:
Handwritten text recognition (HTR)
Language models (LMs)
Neural networks (NNs)
Neural network language model (NN LM)
Bidirectional long short-term memory neural networks (BLSTM)
Hybrid HMM/ANN models
ROVER combination

ABSTRACT

Unconstrained off-line continuous handwritten text recognition is a very challenging task which has been recently addressed by different promising techniques. This work presents our latest contribution to this task, integrating neural network language models in the decoding process of three state-of-the-art systems: one based on bidirectional recurrent neural networks, another based on hybrid hidden Markov models and, finally, a combination of both. Experimental results obtained on the IAM off-line database demonstrate that consistent word error rate reductions can be achieved with neural network language models when compared with statistical *N*-gram language models on the three tested systems. The best word error rate, 16.1%, reported with ROVER combination of systems using neural network language models significantly outperforms current benchmark results for the IAM database.

1. Introduction

Off-line handwritten text recognition (HTR) is the task of transforming an image containing handwritten text into a machine-readable representation of that text. Since handwritten text is one of the most frequently used forms of communication, there has been a substantial interest in the last decades for pattern recognition methods that allow an automatic transcription [1,2]. This research has led to a number of industrial applications with near-perfect recognition accuracy for small vocabulary or single writer tasks such as bank check reading [3] and address reading [4]. However, general, large vocabulary unconstrained handwriting recognition continues to be a difficult problem for which only research prototypes exist. Impressive progress has been recently made with the development of BLSTM neural networks and hybrid HMM/ANN models, with a reported recognition accuracy of 75.1% on writer-independent, unconstrained English texts [5,6].

Less attention, however, has been paid to the language models that are used in the recognition process. Current best practice is still to use back-off *N*-gram language models estimated from large corpora as it was done more than twenty years ago [7,8] and used in HTR for more than ten years [9]. In this paper we present an alternative way of using the *N*-gram statistics by training and using neural network language models (NN LMs) [10]. This kind of language model, where a neural network estimates word probabilities depending upon the previously recognized words, takes profit of a continuous space representation of words to perform an automatic smoothing based on learned features from data. Since they have already led to very interesting results in other machine learning tasks, such as Automatic Speech Recognition [10–14], or Statistical Machine Translation [15–19], it seems worth studying their performance on HTR.

We demonstrate in this paper the applicability of NN LMs for two complete state-of-the-art recognition systems, which are independent of each other as far as pre-processing, features, and underlying recognition methodology are concerned, i.e., BLSTM NN and hybrid HMM/ANN models. In addition, we show that the systems can be combined to get even further improvement. We report in this paper the current best recognition results for the

* Corresponding author at: Escuela de Enseñanzas Técnicas, Departamento de Ciencias Físicas, Matemáticas y de la Computación, Universidad CEU Cardenal Herrera, C/San Bartolomé 55, 46155 Alfara del Patriarca, Valencia, Spain.
Tel.: +34 96 136 90 00x2361; fax: +34 96 130 09 77.
E-mail addresses: francisco.zamora@uch.ceu.es, fzamora@dsic.upv.es (F. Zamora-Martínez), vfrinken@ait.kyushu-u.ac.jp (V. Frinken), sespana@dsic.upv.es (S. España-Boquera), mcastro@dsic.upv.es (M.J. Castro-Bleda), afischer@iam.unibe.ch (A. Fischer), bunke@iam.unibe.ch (H. Bunke).

0031-3203/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.patcog.2013.10.020

IAM off-line database, a widely known benchmark database for unconstrained English handwriting.

The rest of this paper is structured as follows. Section 2 is devoted to describe the IAM off-line handwriting database and the corpora used for language model training. Next, Section 3 introduces the neural network language model. The recognition systems tested in this work are described in Section 4. Sections 5 and 6 present the way NN LMs are trained and coupled in the decoding process. An experimental evaluation of the NN LM with the proposed recognition systems as well as their combination is given in Section 7, and conclusions are drawn in Section 8.

2. Handwriting and LM databases

This section presents a brief description of the handwritten data used for the recognition experiments, and also the corpora used for language model training.

2.1. IAM off-line handwriting database

All experiments reported in this work have been conducted on a subset of lines from the IAM Handwriting Database version 3.0 [20].[1] This database consists of transcriptions of 5685 sentences from the LOB corpus [21] written by 657 different writers, with no restrictions on style or writing tool. The text is written on white paper and scanned at a resolution of 300 dpi. In order to focus on the recognition task, the text line segmentation was manually checked to produce perfectly segmented text lines (some examples are shown in Fig. 1).

The subset used in our experiments is the same as in [5,6]: a training set of 6161 lines, a validation set of 920 lines, and a test set of 2781 lines. These three sets are writer disjoint, thus any writer who contributed text to any of these sets did not contribute to one of the other sets. Statistics of this database can be seen in Table 1. A total of 87,967 instances of 11,320 distinct words occur in the union of the training, validation, and test sets.

The ground truth is given in form of the transcription of each text line, without any further information about where in the text line a character occurs. A total of 79 distinct graphemes appear in the database, which are all English letters, digits, a special symbol for crossed out words, and a variety of punctuation marks (see Table 2).

2.2. Corpora for LM training and dictionaries

The word language models are trained with three different text corpora: the LOB corpus [21] (excluding those sentences that contain some line from the test set or the validation set of the IAM task), the Brown corpus [22], and the Wellington corpus [23]. Some statistics of the corpora are shown in Table 3. The resulting lexicon has approximately 103 K different words.

Two different dictionaries have been used in the experiments presented in this paper:

- a 103 K dictionary, which corresponds to the full training lexicon,
- a 55 K dictionary, obtained after removing the singletons (that is, words that appear only once in the entire text corpus) from the one with 103 K.

Table 4 shows the out-of-vocabulary (OOV) words rate for each dictionary size, along with the OOV words rate for the dictionary of the 10 K most frequent words used as the shortlist for the NN LMs (see Section 5.2).

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

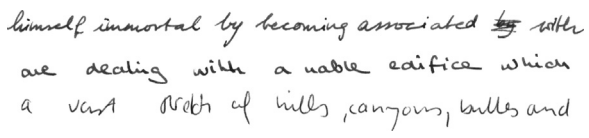

Fig. 1. Examples of line images from the IAM-DB.

Table 1
Off-line IAM-DB statistics.

Dataset	# Lines	# Words	# Lexicon
Train	6161	53,871	7750
Val	920	8672	2425
Test	2781	25,424	5314
Total	9862	87,967	11,320

Table 2
Graphemes for the IAM-DB.

Lower case letters	a b c d e f g h i j k l m n o p q r s t u v w x y z
Upper case letters	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
Digits	0 1 2 3 4 5 6 7 8 9
Punctuation marks	< space > - , ; : ! ? / . ' '' () * & +
Garbage symbol	#

Table 3
Training corpora statistics for the language models.

Corpora	# Sentences (K)	# Words (M)	# Lexicon (K)
LOB (excluding IAM Val & Test)	47.7	1.06	57.7
Brown	56.6	1.14	55.5
Wellington	58.3	1.14	54.4
Total	162.6	3.34	103.4

Table 4
Dictionaries and out-of-vocabulary (OOV) words rate in the IAM validation and test sets for each dictionary size.

Dictionary	$\|\Omega\|$ (K)	OOV rate (%) Val	Test
NN LMs shortlist	10	9.8	10.1
Dictionary removing singletons	55	4.1	3.7
Dictionary with the full vocabulary	103	3.2	2.9

3. Language modeling with neural networks

For handwriting recognition tasks, the text image is converted into a sequence of feature vectors X which is used by the underlying recognition system to find the $W^\star$ transcription with the highest likelihood:

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

which can be reformulated, using the Bayes rule, as

$$W^{\star} = \arg\max_{W \in \Omega^{+}} P(X|W)P(W) \tag{2}$$

In this decoding process, the statistical language model $P(W)$ plays a key role. Statistical N-gram LMs [24,8,25] only consider the $N-1$ previous words to estimate the LM probability for a sequence of words of length $|W|$:

$$p(w_1 \ldots w_{|W|}) \approx \prod_{i=1}^{|W|} p(w_i|w_{i-n+1} \ldots w_{i-1}) \tag{3}$$

Neural Network Language Models (NN LMs) are statistical N-gram LMs that use the capability of NNs to estimate probabilities in order to satisfy expression (3). Contrarily to conventional statistical N-gram LMs, the generalization ability of NNs and the continuous space representation of words allow the NN LMs to perform an implicit smoothing.

A scheme of a NN LM is illustrated in Fig. 2. Each output neuron estimates the conditional probability $p(w_i|w_{i-n+1} \ldots w_{i-1})$, for a given word $w_i \in \Omega$. The input of the NN LM is composed of the sequence $w_{i-n+1}, \ldots, w_{i-1}$. A "1-of-$|\Omega|$" encoding scheme would be a natural representation of the input words, but it is not suitable for large vocabulary tasks due to the huge size of the resulting NN. To overcome this problem, a *distributed representation* for each word has been used in this work as follows:

- Input neurons are divided into groups $L_1, \ldots, L_{N-1}$, as depicted in Fig. 2. Each input word is associated to a different L_j and is encoded following a "1-of-$|\Omega|$" scheme.
- Every L_j is projected onto a much smaller set of projection neurons P_j. Thus, P_j represents the distributed encoding of input neurons at L_j. Each projection P_j could be computed efficiently taken the column of weights which corresponds to the input neuron activated with 1.
- The weights of all projection layers are shared, that is, the weights from each local encoding of input L_j to the corresponding subset of projection units P_j are the same for all input words during training. The NN LM is able to learn both the distributed representation of words onto a continuous space and the conditional probability estimates of Eq. (3), as introduced in [26,10].

After training, the projection layer can be removed from the network since it is much more efficient to replace it by a pre-computed table of size $|\Omega|$ (see Fig. 3) which stores the distributed encoding of each word and is computed as

$$P_j = L_j^T \cdot W_{l,p} + B_p \tag{4}$$

where L_j^T is the transpose of L_j, $W_{l,p}$ is the matrix of NN weights from each input word to the corresponding projection units subset (shared among all input words), and B_p is the vector of biases of each projection units subset (which is also shared). The product becomes the selection of one column of weights, as before, since only one neuron of each L_j is activated with 1.

The hidden layer of the NN LM, denoted as H, computes

$$H = \tanh(P^T \cdot W_{P,H} + B_h) \tag{5}$$

where P^T is the transposed of the projection layer vector, $W_{P,H}$ is the matrix of NN weights from the projection layer to the hidden layer, B_h is the vector of the hidden layer biases, and $\tanh(\cdot)$ is the hyperbolic tangent activation function.[2]

[2] In this work, $\tanh(\cdot)$, as well as $\exp(\cdot)$ and $\log(\cdot)$ are applied element-wise to each component of the vector.

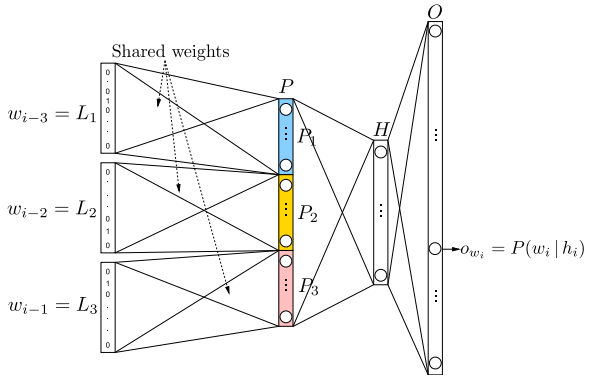

Fig. 2. The architecture of a 4-gram NN LM during training. In this example, the history of word w_i is represented by $h_i = w_{i-1}, w_{i-2}, w_{i-3}$.

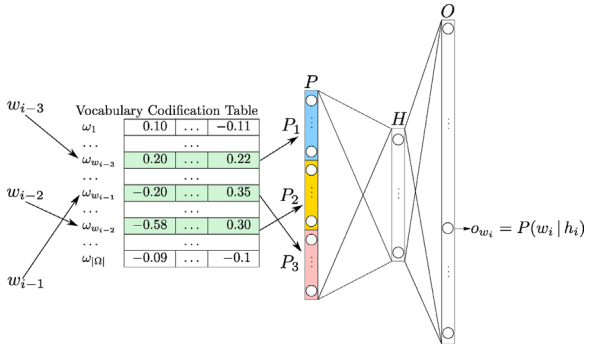

Fig. 3. The architecture of a NN LM after training, where the first layer is substituted by a look-up table with the distributed encoding for each word $\omega \in \Omega$.

The output layer O has $|\Omega|$ units, one for each word of the vocabulary. The computation of O is as follows:

$$A = H^T \cdot W^{H,O} + B^O \tag{6}$$

$$O = \frac{\exp(A)}{\sum_{j=1}^{|\Omega|} \exp(a_j)} \tag{7}$$

where $W^{H,O}$ is the matrix of weights from the hidden layer to the output layer, B^O is the vector of output layer biases, A is the vector of activation values computed before applying the softmax normalization, and a_j is the j-th component of A.

4. Recognition systems

This section presents a brief description of the recognition systems which are used in this work to evaluate the performance improvement reached with neural network language models in unconstrained off-line handwriting recognition. The first two systems have been developed independent of each other, with different text line normalization and features extraction methods following previous works [9,6]. Despite that, handwritten lines go through a sequence of similar preprocessing steps in order to reduce variations in the writing style as much as possible while preserving information that is relevant for recognition. The steps that are applied comprise image cleaning, skew correction, slant removal, and size normalization. Both recognition approaches work on sequential data and employ a sliding window, moving from the left to the right over the normalized text line, in order to extract a sequence of feature vectors that serve as input into the underlying system. The third system is a combination of the others. The specifics of each approach are described below.

4.1. BLSTM neural network recognizer

The first recognition approach is based on recurrent neural networks and performs the recognition based on a sequential representation of a normalized text line using 9 geometric features.

The individual steps in this system are as follows. First, the skew angle is determined by a regression analysis based on the bottom-most black pixel of each pixel column. Then, the skew of the text line is removed by rotation. Afterwards the slant is corrected in order to normalize the directions of long vertical strokes found in characters like 't' or 'l'. After estimating the slant angle based on a histogram analysis, a shear transformation is applied to the image. Next, a vertical scaling is applied to obtain three writing zones of the same height, i.e., lower, middle, and upper zone, separated by the lower and upper baseline. To determine the lower baseline, the regression result from skew correction is used, and the upper baseline is found by vertical histogram analysis. Finally the width of the word is normalized. For this purpose, the average distance of black/white transitions along a horizontal straight line through the middle zone is determined and adjusted by horizontal scaling.

Given the image of a single word, a horizontal sliding window with a width of 1 pixel is used to extract nine geometric features at each position from left to right, three global and six local ones. The global features are the 0th, 1st and 2nd moment of the black pixels' distribution within the window. The local features are the position of the top-most and bottom-most black pixel, the inclination of the top and bottom contour of the word at that position, the number of vertical black/white transitions, and the average gray scale value between the top-most and bottom-most black pixel. For details on the normalization operations and feature extraction steps, we refer to [9].

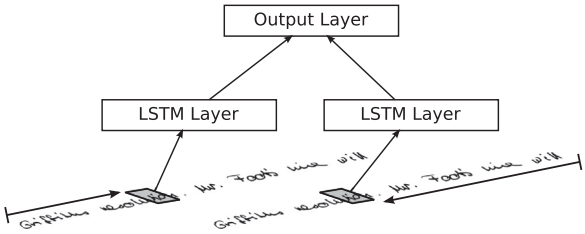

Fig. 4. An illustration of the mode of operation of the BLSTM neural network. For each position, the output layer sums up the values of the two hidden LSTM layers.

The recognizer used in this paper is a recently developed recurrent neural network, termed *bidirectional long short-term memory* (BLSTM) neural network [5]. A hidden layer is made up of so called *long short-term memory* blocks instead of simple nodes to circumvent the exponential increase or decay of information that is encountered in common recurrent neural networks.

The network is *bidirectional*, i.e., a sequence is fed into the network in both the forward and the backward mode using two separate input and hidden layers, joined in one output layer. The output layer contains one node for each possible character in the sequence plus a special ε node, to indicate "no character". At each position, the output activations of the nodes are normalized so that they sum up to 1, and can hence be treated as posterior probabilities of the characters' occurrences at each position.

The output of a sequence is therefore a matrix of probabilities and a path through the matrix represents a recognition result. The probability of such recognition result is given as the product of each element along the path and the optimal path can be found efficiently using dynamic programming. For more details about BLSTM networks, we refer to [5,27]. A BLSTM NN with one hidden LSTM layer processing a line of handwritten text is depicted in Fig. 4.

4.2. Hybrid HMM and neural network recognizer

This system comprises a sophisticated text line preprocessing procedure using neural networks to clean the handwritten text images, to correct the slant, and to classify points extracted from the image which are used to estimate the slope and to find the main body area of the text line in order to perform size normalization. The preprocessed image is then transformed into a sequence of feature vectors following the approach described in [28]. A sliding window of square cells is applied on the image, and three values are extracted from each cell: normalized gray level, horizontal derivative of the gray level, and vertical derivative of the gray level. A window comprising 20 cells moving in steps of 2 pixels has been used, leading to sequences of 60-dimensional feature vectors.

The underlying recognition engine is based on the widely known Hidden Markov Model paradigm which has been hybridized with a Multilayer Perceptron (MLP) to estimate the emission probabilities instead of using Gaussian Mixtures, as described in [6]. A number of HMM and MLP topologies were empirically tested on the validation set. The best one consisted of left-to-right HMMs with loops but without skips, and with 7 states. Regarding the MLP, two hidden

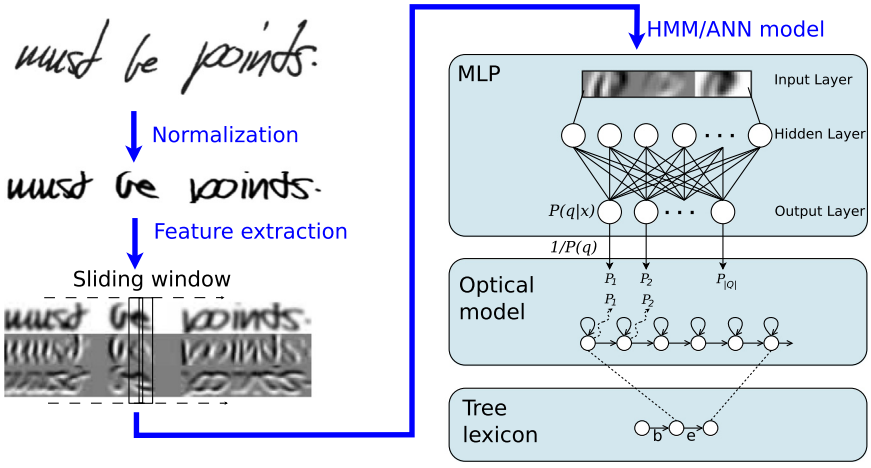

Fig. 5. Scheme of the HMM/ANN recognition system.

layers of 192 and 128 units, respectively, using the softmax activation function, was the best featured network. The MLP received, in addition to the current feature vector, a context of 4 vectors on the left and 4 on the right. Each output unit of the MLP estimated a posterior probability [29] which, after dividing by the priors following Bayes rule, was used as the emission of the corresponding HMM state. A scheme of the preprocessing, feature extraction, and the recognition system is given in Fig. 5. For more details about this system, we refer to [6].

4.3. ROVER recognizer

Another aim of this work was to investigate whether or not a further improvement can be obtained even when the underlying recognizers are sophisticated and highly tuned to achieve low error rates. For this reason, the two recognition engines described above were combined. This was done using a generalized version of the Recognizer Output Voting Error Reduction (ROVER) system [30,31].

First of all, both recognition engines were modified in order to output N-best lists instead of just the best hypothesis. The original recognition algorithm for BLSTM neural networks [5] was extended with a modified token passing algorithm [32] to record n-best word endings at each sliding window position. From the resulting word lattice, we extracted N-best hypotheses with the SRILM toolkit [33] based on the $A^\star$ algorithm [34]. In this paper, we have used $n=10$ best word endings and $N=2000$ best hypotheses. For the HMM/ANN recognition system, N-best hypotheses were generated following a lazy version of Eppstein's algorithm [35], obtaining $N=2000$ best hypotheses.

After that, the N-best output word strings from each recognition system were first aligned and then combined in a weighted voting scheme. The weights of the word hypotheses in the combination were based on their posterior probabilities which were estimated from the N-best lists of each recognition engine. The combination was done using the SRILM toolkit [33], selecting the 1-best hypothesis to measure the ROVER engine. The toolkit allows the use of different weight parameters, which were optimized on the validation set, and measured on the test set afterwards. A scheme of the ROVER system is illustrated in Fig. 6.

5. Training the neural networks for language modeling

Neural networks required for language modeling have some peculiarities such as large input and output layer sizes related to the vocabulary or the tremendous amount of available data for training. These characteristics have implications for the training of the NN LMs, which are addressed in this section.

5.1. Training algorithm

The training of the neural networks used for language modeling is performed using the stochastic version of the error back-propagation algorithm [36,29], where the weight decay regularization term is added to all the weights of the NN, excepting the biases. The cross-entropy error function has been used:

$$E = D^T \cdot \log\ O \tag{8}$$

where D is the vector of desired outputs, and E is the computed error.

5.2. Dealing with the output size of the NN LM

The size of the NN LM is dominated by the output layer which should correspond to the size of the vocabulary $|\Omega|$. The larger the lexicon size, the more expensive the training of the neural network is. For this reason, it is usual to reduce the size of the output layer by taking into account a subset of the vocabulary, known as *shortlist*, comprising the most frequent words.

Since the output layer of the NN is restricted to the words in the shortlist, it is necessary to properly estimate the probability of words out of this subset (known as *out-of-shortlist* (OOS) words). This issue has been tackled in several ways in the literature [10,12,37], but the main idea consists of using a special output neuron to estimate the joint probability of all OOS words. To this end, Eq. (3) is evaluated as follows:

$$p(w_i|w_{i-n+1}\ldots w_{i-1}) = \begin{cases} o_{w_i} & \text{if} \quad w_i \in \Omega - OOS; \\ o_{OOS} \cdot p(w_i|OOS) & \text{if} \quad w_i \in OOS. \end{cases} \tag{9}$$

where

- o_{w_i} is the activation of the output neuron corresponding to word w_i when $w_i \in \Omega - OOS$,
- o_{OOS} is the activation of the output neuron corresponding to all OOS words. This value is an estimate of $\sum_{w \in OOS} p(w|w_{i-n+1}\ldots w_{i-1})$, and
- $p(w_i|OOS)$ is a standard unigram probability computed over the set of OOS words.[3]

In this work, a shortlist comprising the 10 K most frequent words has been used in the NN LMs for both the 55 K and the 103 K dictionaries.

5.3. Dealing with the input size of the NN LM

Some NN LM approaches use the entire vocabulary Ω at the input. In the work described in this paper, we have compared this method with a much simpler approach which consists of using an input layer for the words belonging to the shortlist plus one input neuron representing all OOS words. We obtained indistinguishable results in the previous experiments for both approaches. For that reason, we decided to use the restricted vocabulary (shortlist words plus a neuron associated to the OOS words) at both the input and output layers of the neural network.

Therefore, an additional neuron, called OOS neuron, is added at the input and at the output of the neural network to take into account the OOS words. During training, every OOS word is replaced by the OOS identifier. During evaluation, when the computation of the probability of an OOS word is needed, the activation of the OOS neuron is combined with a simple standard unigram model computed over the set of OOS words, as described in Eq. (9).

Thus, as the input to the NN LM is also truncated to the shortlist words, the same neural network is used for both 55 K and 103 K language models (10 K shortlist words at the input and output layer), although the estimation of $p(w_i|OOS)$ is different for each dictionary size: a unigram of 45 K words, for the 55 K dictionary, and a unigram of 93 K, for the 103 K dictionary.

5.4. Ensembles of NN LMs and standard N-grams

Using ensembles rather than single neural nets was successfully applied before as described, for example, in [38,39]. In order to achieve a better generalization [10], an ensemble of neural networks has been used: four neural networks were trained and

[3] There are other alternatives in the literature. For instance, [10] estimates $p(w_i|OOS, w_{i-n+1}\ldots w_{i-1})$, instead of $p(w_i|OOS)$, with a standard statistical N-gram. [12] proposes to approximate the value of $p(OOS|w_{i-n+1}\ldots w_{i-1})$ to 0, obtaining indistinguishable results compared to the previous approach, mostly due to the linear combination of NN LMs with a standard N-gram LM.

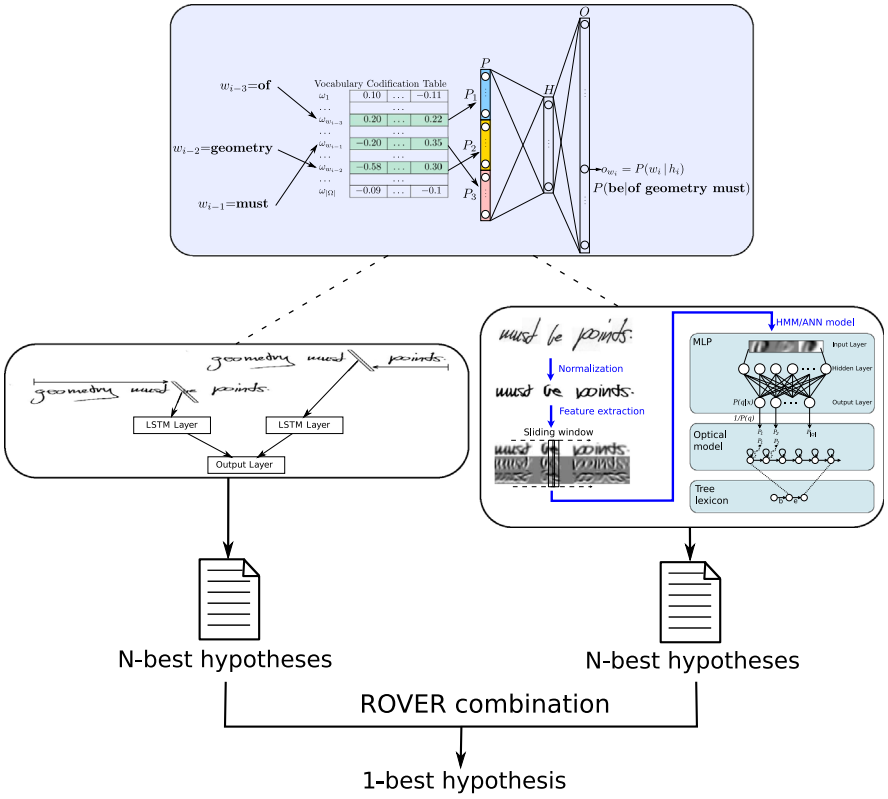

Fig. 6. Scheme of the ROVER recognition system.

their results were linearly combined to build the final NN LM. Each neural network has a different projection layer size (160, 192, 224, 256), but the same hidden layer size (200 neurons). Similar configurations were proven to work well in previous works [40,41].

Simultaneously, we linearly combined the probability computed by the neural networks with a standard statistical *N*-gram LM. Regardless of the strategy chosen, the combination of both models is a key requirement for improving results [17].

6. Coupling recognizers and NNLMs

The integration of the optical models and the language model during decoding is of fundamental importance. Usually, the combination of HMM-based optical models and *N*-gram-based language models is performed via finite state automata combination referred as trellis or search space. Although this basic idea remains unchanged when using NN LMs, some special considerations have to be taken to deal with the higher computational cost required to evaluate these models. To this end, a special preprocessing which includes the pre-computation of softmax normalization constants is described in this section, along with other peculiarities of the decoding algorithm.

6.1. Fast evaluation of NN LMs

Since NN LMs are quite expensive to evaluate and this cost is dominated by the computation of the softmax normalization factor (denominator of Eq. (7)), we have used an adapted version of [42] which consists of pre-computing the *M* softmax normalization factors most probably needed during decoding. The cost of retrieving these values is negligible compared with the cost of computing them. A space/time trade-off has to be considered in the sense that the more the space is dedicated to store pre-computed softmax normalization factors, the more time reduction can be obtained. When a given normalization factor is not found, it can be computed on-the-fly or some kind of smoothing must be applied. We have followed the latter idea: when a softmax normalization factor is not found, another simpler model (in this work, a lower order NN LM) is used. The design of the NN LM requires the following steps:

1. Set the order n of the *N*-gram NN LMs.
2. For all $n>2$, use the $(n-1)$-grams from the training set as input into the corresponding NN LM and store the softmax normalization factors in a table.
3. Pre-compute every softmax normalization factor for every shortlist word using the bigram NN LM, and store them in a table.

During decoding, the softmax pre-computed factor associated to the $(n-1)$-length prefix of the current token is searched in the corresponding table. If the normalization factor is found, the corresponding *N*-gram NN LM computes the probability; otherwise, we have to switch to the next lower order NN LM. Note that normalization factors associated to bigrams will always be found, since the bigram NN LM only requires a table for shortlist words, which can be efficiently represented with a conventional array.

Using this decoding algorithm, the coupled systems require between 4 and 6 s, in average, to decode a text line on a Linux system using an Intel® Core™ i5 750 CPU or an AMD® Opteron™ 2354 CPU. The use of NN LMs leads to only an additional cost of about 0.4 s, mostly due to the smoothing approach just explained.

6.2. Decoding lines with sentence based LMs

There is a mismatch between the LMs, which are trained with sentences, and the IAM task, which is composed of lines that do not correspond to whole sentences (see the examples of Fig. 1).

We have addressed this issue by ignoring the context cues[4] during decoding. This solution leads to an undefined input for NN LMs when the beginning of a line is decoded.[5] Using the technique just described in the previous subsection, this problem is limited only to the first decoded word. The proposed solution consists of using a statistical unigram to deal with the first word, and to switch to the general method afterwards.

6.3. Final language models

In order to summarize, let us enumerate the language models estimated for each dictionary used in this work (the 55 K and 103 K dictionaries):

- A 4-gram NN LM, a 3-gram NN LM and a bigram NN LM, all of them sharing the following features:
  - all neural networks use the same 10 K input/output shortlist of words;
  - all neural networks add an OOS neuron at the input and output layer;
  - they all use the formula from Eq. (9) to compute the OOS word probabilities; and, finally,
  - each *N*-gram NN LM is an ensemble composed of four linearly combined NNs.
- A standard statistical 4-gram LM, 3-gram LM and bigram LM, all of them sharing the following features:
  - all LMs were estimated using the same standard smoothing technique [8];
  - for each considered *N*-gram, three different LMs were estimated, one for each corpus in Table 3 (LOB, Brown, and Wellington corpora). Then, the three *N*-gram LMs were linearly combined to minimize the perplexity (PPL) on the validation dataset. The weights of these linear combinations were computed by means of the `compute-best-mix tool` from SRILM toolkit [33]. This procedure was followed for the final bigram, 3-gram and 4-gram LMs. This approach performs slightly better than training a model over the concatenation of the three corpora [5].
- A statistical unigram to compute the probability of the first word of each line.

Tables 5 and 6 show PPL using different smoothing techniques and different combination of LMs. Note that the PPL is computed ignoring the probability assigned to OOV words, but taking into account OOV words at the context of the *N*-grams. This issue is important to perform a fair comparison between standard *N*-grams and NN LMs.

Table 5 shows the PPL of the *N*-gram LMs for the validation set, using different standard smoothing techniques [8], and with and without using interpolated models of different *N*-gram models. The models were trained both for the 55 K and 103 K vocabularies. With natural or Witten Bell (WB) smoothing, the perplexity measure decreases with increased modeling length. Nevertheless, this is not the case when using Kneser–Ney techniques. In summary, best PPL is obtained when using Witten Bell smoothed 4-gram LMs (which are pure back-off, without interpolation, smoothing).

Table 6 shows the PPL obtained for the validation dataset for different combinations of NN LMs with WB smoothed LMs, for different *N* orders, and for the two considered vocabulary sizes. In all cases increasing *N* up to 4-grams leads to better results, as well as the combination of NN LMs with WB LMs. Notice that NN LMs alone (not combined with WB LMs) achieve better perplexities than WB LMs. The addition of 4-gram NN LMs improves by a 18% the 4-gram WB LM baseline, in the case of 55 K vocabulary, and improves up to a 20% in the case of 103 K vocabulary.

Table 5
Validation PPL for *N*-gram LMs using different standard smoothing techniques, being *int.* if the model is an interpolation of different *N* orders. OOV words were considered at *N*-gram context, but ignored in PPL computation.

N-gram LM	Int.	$\|\Omega\|$ (K)	Bigram	3-gram	4-gram
Modified Kneser–Ney	Yes	55	455	460	466
Modified Kneser–Ney	No	55	480	474	477
Kneser–Ney	Yes	55	450	453	458
Kneser–Ney	No	55	457	448	449
Natural smoothing	No	55	460	428	424
Witten–Bell	Yes	55	**434**	408	406
Witten–Bell	No	55	441	**399**	**393**
Modified Kneser–Ney	Yes	103	500	506	512
Modified Kneser–Ney	No	103	520	518	522
Kneser–Ney	Yes	103	496	500	505
Kneser–Ney	No	103	507	498	500
Natural smoothing	No	103	509	476	471
Witten–Bell	Yes	103	**480**	452	449
Witten–Bell	No	103	488	**442**	**437**

Table 6
Validation PPL for different combinations of Witten–Bell smoothed *N*-gram LMs and NN LMs. OOV words were considered at *N*-gram context, but ignored in PPL computation.

NN LM	$\|\Omega\|$ (K)	Witten–Bell			
		None	Bigram	3-gram	4-gram
None	55	–	441	399	393
Bigram	55	374	363	343	340
3-gram	55	358	339	331	328
4-gram	55	**348**	**330**	**324**	**321**
None	103	–	488	442	437
Bigram	103	407	395	375	372
3-gram	103	390	369	361	358
4-gram	103	**378**	**359**	**353**	**351**

6.4. Decoding with NN LMs

Note that only one language model is used in each individual LM look-up when decoding a line: a unigram is used for the first word, a bigram NN LM for second word, and so on until reaching the highest order (in this case, up to 4-gram NN LM). But note also that when a given softmax normalization factor is not found, a lower order *N*-gram NN LM look-up is performed.

As stated in Section 5.4, in order to obtain a better generalization, NN LM use network ensembles composed of a linear combination of four neural networks. Besides, a further improvement is obtained by combining the NNLM with a standard statistical *N*-gram. The weights of this linear combinations were computed by means of the `compute-best-mix+ tool` from SRILM toolkit [33], minimizing the PPL on the validation set. Again, let us remember that OOV words were ignored for the PPL computation as for the combination of the models, which is very important to ensure better results on PPL, and to obtain comparable numbers. However, OOV words were taken into account in *N*-gram contexts.

[4] Context cues are special symbols which mark the beginning and the end of each sentence.

[5] A NN LM representing an *N*-gram of order *n* requires $n-1$ words at the input to compute the output. This poses a problem if we ignore the context cue at the beginning of the sentence, because the context cue symbol is usually used to complete the initial input.

6.5. Optimizing the combination of likelihoods and LM probabilities

In the decoding process expressed in Eq. (2), the language model $P(W)$ plays a key role. That equation can be generalized as a log-linear combination of three information sources: the optical model probability, the language model probability, and a hypothesis length penalization. Therefore, three weights λ are needed. The optical model probability is always fixed to one ($\lambda_0=1$), the language model weight (λ_1) is known as Grammar Scale Factor (GSF), and the hypothesis length penalization weight (λ_2) is known as Word Insertion Penalty (WIP). So, Eq. (2) for sequence recognition can be rewritten as

$$W^{\star} = \arg\max_{W \in \Omega^{+}} P(X|W)P(W)^{\lambda_1} \exp(|W|)^{\lambda_2} \tag{10}$$

Word insertion penalty and grammar scale factor parameters have been optimized on the validation set, for both HMM/ANN and BLSTM NN systems, by means of the Minimum Error Rate Training (MERT) procedure [43] using the Simplex Downhill algorithm [44].

7. Experimental evaluation

In this section, the recognition engines presented in Section 4 are independently evaluated using both standard N-gram language models and the NN LMs listed in Section 6.3. The more promising individual configurations obtained for the validation set are evaluated on the test set. The obtained results are compared with other approaches found in the literature. In a third experiment, we take profit of the dissimilarity of both recognition systems by combining the N-best outputs, obtaining a further improvement by using NN LMs.

We investigated the performance of the BLSTM NN and HMM/ANN systems. Our main goal is to study the effect of the dictionary size (55 K and 103 K) and the influence of NN LMs in the overall system. To this end, we compared the Word Error Rate (WER) and Character Error Rate (CER), which are percentages obtained from the Levenshtein distance between the recognized sequence and the corresponding ground truth. They are calculated as

$$100\frac{D+I+S}{L}$$

being D the number of deleted units, I the number of inserted units, S the number of substituted units, and L the total number of units in the ground truth transcriptions. (A unit is a word or a character depending on what is being measured.) Note that, since we are working in an unconstrained task using open dictionaries, there are OOV words in the validation and test sets of the IAM-DB (see Table 4). This imposes a lower bound to the WER. For example, a lower bound of 3.2% and 2.9% WER for the validation and test sets, respectively, with the 103 K dictionary. Note also that CER is affected by OOV words as well, but with a weaker effect.

Tables 7 and 8 show the WER and CER of IAM-DB validation set for different LM configurations of BLSTM and HMM/ANN systems. In all cases, NN LMs perform better, even when these models are not combined with WB LMs. The benefits of combining WB LMs in the NN LM systems is not totally clear when measured on validation set, being less clear for BLSTM system. Two possible explanations are the fact that this set is substantially smaller than the test set and the fact that it has already been used to optimize the systems. The increase of the dictionary size leads to a better performance for both recognizers, although this effect seems more pronounced for the HMM/ANN engine, whose performance reaches the best figures when using NN LMs with the 103 K dictionary.

In summary, each recognition system takes advantage of NN LMs, with better recognition for the 103 K dictionary. The best statistical N-grams are 4-grams LMs, for both 55 K and 103 K dictionaries.

Table 7
Validation % WER and % CER results for the BLSTM system.

NN LM	$\|\Omega\|$ (K)	Witten–Bell None		Bigram		3-gram		4-gram	
		WER	CER	WER	CER	WER	CER	WER	CER
None	55	–	–	18.2	7.8	18.1	7.9	18.0	7.9
Bigram	55	17.6	7.7	17.6	7.7	17.6	7.7	17.6	7.7
3-gram	55	17.5	7.7	17.6	7.7	17.6	7.7	17.6	7.7
4-gram	55	17.4	7.7	17.6	7.7	17.7	7.8	17.6	7.6
None	103	–	–	17.6	7.8	17.5	7.7	17.4	7.7
Bigram	103	17.0	7.6	17.0	7.5	17.1	7.6	17.0	7.5
3-gram	103	16.9	7.5	17.0	7.5	17.0	7.5	16.9	7.5
4-gram	103	16.7	7.5	17.0	7.5	17.0	7.5	17.0	7.5

Table 8
Validation % WER and % CER results for the HMM/ANN system.

NN LM	$\|\Omega\|$(K)	Witten–Bell None		Bigram		3-gram		4-gram	
		WER	CER	WER	CER	WER	CER	WER	CER
None	55	–	–	18.4	6.7	17.9	6.5	17.8	6.5
Bigram	55	17.2	6.2	17.1	6.3	17.1	6.3	17.0	6.3
3-gram	55	17.1	6.2	17.0	6.2	16.8	6.1	16.8	6.2
4-gram	55	17.0	6.1	17.0	6.1	16.8	6.1	16.8	6.1
None	103	–	–	17.4	6.3	16.9	6.1	16.8	6.1
Bigram	103	16.1	5.8	16.1	5.8	15.9	5.7	16.0	5.8
3-gram	103	15.9	5.7	15.9	5.8	15.7	5.7	15.7	5.7
4-gram	103	16.0	5.7	15.8	5.7	15.7	5.7	15.7	5.7

The performance for the test set has been measured only for the more promising recognition configurations on validation (a trade-off between WER, CER and PPL). The results are shown with a 95% confidence interval in Fig. 7. In all cases, systems which use NN LMs perform consistently better than systems using only WB LM. The relative improvement is up to 2.7% of WER for the BLSTM system, and up to 5.2% for the HMM/ANN system. The improvements are statistically significant for a confidence of 59% and 87% for BLSTM and HMM/ANN, respectively. However, under a system comparison test [45], the improvements have more than a 99.9% probability of being statistically significant. The system comparison tests are also shown in Fig. 7. Confidence intervals and system comparison tests were computed following the bootstrap technique described in [45].

Thus, every system improves from WB LMs to NN LMs and the combined ROVER system outperforms the independent recognizers. A final WER 16.1% is achieved for the test set with the ROVER system, which corresponds to a further statistical significant improvement over the two best individual systems (25% relative over the best individual BLSTM system, and 20% relative over the best HMM/ANN system). Similarly, the final CER of 7.6% corresponds to a relative improvement over the individual systems of 26% and 8%, respectively. The improvement of WER from ROVER WB to ROVER NN LMs+WB is statistically significant with a 75% of confidence, but the system comparison test shows a statistical significance of more than 99.9%.

Finally, the WER and CER performance on the test set of various recognition systems reported in the literature has been collected in Table 9 in order to better analyze the obtained results with respect to other systems. The table includes the baseline results obtained in the previous versions of the two recognition systems used in this paper, as presented in [5,6], both of which use the same 20 K bigram language model.

System	LM	$\|\Omega\|$	WER (%)	CER (%)	ΔWER
BLSTM	WB 4-grams	103K	22.2 ± 0.7	10.5 ± 0.4	–
BLSTM	NN LMs 4-grams	103K	21.8 ± 0.7	10.3 ± 0.4	0.4 ± 0.4
BLSTM	+ WB 4-grams	103K	21.6 ± 0.7	10.3 ± 0.4	0.6 ± 0.3
HMM/ANN	WB 4-grams	103K	21.1 ± 0.7	8.6 ± 0.4	–
HMM/ANN	NN LMs 4-grams	103K	20.5 ± 0.7	8.4 ± 0.4	0.6 ± 0.6
HMM/ANN	+ WB 4-grams	103K	20.0 ± 0.7	8.3 ± 0.4	1.1 ± 0.5
ROVER	WB 4-grams	103K	16.8 ± 0.6	8.0 ± 0.4	–
ROVER	NN LMs 4-grams	103K	16.4 ± 0.6	7.7 ± 0.4	0.4 ± 0.5
ROVER	+ WB 4-grams	103K	16.1 ± 0.6	7.6 ± 0.4	0.7 ± 0.4

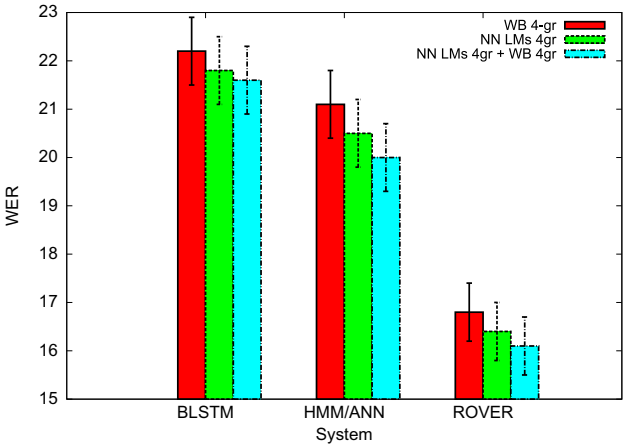

Fig. 7. Test results with 95% confidence intervals. Δ WER is the WER difference with 99.9% confidence interval, comparing every NN LM system with its corresponding WB baseline.

Table 9
Reference systems.

System	$\|\Omega\|$ (K)	WER (%)	CER (%)
Natarajan et al. (HMM)[a] [46]	–	40.1	–
Bertolami et al. (GHMM) [47]	20	35.5	–
Bertolami et al. (HMMs) [47]	20	32.8	–
Drew et al. (GHMM) [48]	50	29.2	10.3
TU Dortmund (HMM) [49]	–	28.9	–
Drew et al. (MLP-GHMM + M-MPE) [50]	50	28.8	10.1
Graves et al. (BLSTM NN/CTC) [5]	20	25.9	–
España-Boquera et al. (HMM/ANN) [6]	20	25.9	10.5
BLSTM + NN LMs + WB	103	21.6	10.3
HMM/ANN + NN LMs + WB	103	20.0	8.3
ROVER + NN LMs + WB	103	16.1	7.6

[a] Experiments using different subsets of the IAM-DB for training, validation, and testing.

Careful attention has to be paid when comparing different systems reported in the literature, even if they use the same corpus, since they may use different language models, dictionaries, and set of parameters. Nevertheless, there is a clear trend towards lower error rates. Yet, there is still room left for improvement, since the lowest possible word error rate using the 103 k dictionary is, as far as the OOV words rate, only 2.86%.

8. Conclusions

The applicability of neural network language models in unconstrained off-line handwriting recognition has been studied in this work. Unlike most current approaches based on statistical *N*-grams, NN LMs rely on the generalization capability of NNs to perform automatic smoothing based on the continuous space projection of input words (distributed encoding).

Two very different recognition systems, one based on recurrent neural networks and the other based on hybrid HMM/ANN models, are used to empirically demonstrate the suitability of NN LM in HTR tasks such as the one presented in this paper. In a set of experiments, the value of using NN LMs for language modeling has been shown for both recognition systems under different vocabulary sizes.

A comparison of the individual systems shows that the hybrid HMM/ANN system is better suited to deal with large vocabularies than the BLSTM NN system. Also, hybrid HMM/ANN models seem to take more advantage of NN LMs, yet both systems achieve a remarkably low error rate.

In a final experiment the dissimilarity of both approaches has been exploited by successfully combining the *N*-best outputs. The obtained results are, to our knowledge, the lowest word and character error rates reported to date on the challenging off-line IAM task. The final word error rate of 16.1% means a relative reduction by 20% over the best individual system and a relative reduction of 38% over the best previously published reference system. The character error rate is reduced by 8% over the best individual system and by 25% over the best result published so far.

9. Future work

In the near future, more tightly coupled combination techniques might be investigated, such as a tandem system, where the posteriors computed by the HMM/ANN engine serve as an additional input into the BLSTM neural network. Also, the inclusion of different recognition systems and language models may be a promising future line of research.

Conflict of interest

None declared.

Acknowledgments

The authors wish to acknowledge the anonymous reviewers for their detailed and helpful comments to the paper. We also thank Alex Graves for kindly providing us with the BLSTM Neural Network source code. This work has been supported by the European project FP7-PEOPLE-2008-IAPP: 230653, the Spanish Government under project TIN2010-18958, as well as by the Swiss National Science Foundation (Project CRSI22_125220).

References

[1] A. Vinciarelli, A survey on off-line cursive word recognition, Pattern Recognition 35 (7) (2002) 1433–1446.
[2] R. Plamondon, S.N. Srihari, On-line and off-line handwriting recognition: a comprehensive survey, IEEE Trans. Pattern Anal. Mach. Intell. 22 (1) (2000) 63–84.
[3] S. Impedovo, P.S. pei Wang, H. Bunke (Eds.), Automatic Bankcheck Processing, World Scientific, 1997.
[4] A. Brakensiek, G. Rigoll, Handwritten Address Recognition Using Hidden Markov Models, in: Reading and Learning, Vol. 2956 of Lecture Notes in Computer Science, Springer Berlin/Heidelberg, 2004, pp. 103–122.
[5] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, J. Schmidhuber, A novel connectionist system for unconstrained handwriting recognition, IEEE Trans. Pattern Anal. Mach. Intell. 31 (5) (2009) 855–868.
[6] S. Espa na-Boquera, M.J. Castro-Bleda, J. Gorbe-Moya, F. Zamora-Martínez, Improving offline handwritten text recognition with hybrid HMM/ANN models, IEEE Trans. Pattern Anal. Mach. Intell. 33 (4) (2011) 767–779.
[7] S. Katz, Estimation of probabilities from sparse data for the language model component of a speech recognizer, IEEE Trans. Acoust., Speech, Signal Process. 34 (3) (1987) 400–401.
[8] S. Chen, J.T. Goodman, An empirical study of smoothing techniques for language modeling, Comput. Speech Lang. 13 (4) (1999) 359–393.
[9] U.-V. Marti, H. Bunke, Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition system, Int. J. Pattern Recognit. Artif. Intell. 15 (2001) 65–90.
[10] H. Schwenk, Continuous space language models, Comput. Speech Lang. 21 (3) (2007) 492–518.
[11] H. Schwenk, J.-L. Gauvain, Connectionist language modeling for large vocabulary continuous speech recognition, in: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2002, pp. 765–768.
[12] A. Emami, L. Mangu, Empirical study of neural network language models for arabic speech recognition, in: Proceedings of the IEEE Workshop on Automatic Speech Recognition Understanding (ASRU), 2007, pp. 147–152.
[13] T. Mikolov, S. Kombrink, L. Burget, J. Cernocký, S. Khudanpur, Extensions of recurrent neural network language model, in: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2011, pp. 5528–5531.
[14] S. Le-Hai, I. Oparin, A. Alexandre, J.-L. Gauvaing, Y. François, Structured output layer neural network language model, in: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), vol. 11, 2011, pp. 5524–5527.
[15] H. Schwenk, D. Déchelotte, J.-L. Gauvain, Continuous space language models for statistical machine translation, in: Proceedings of the COLING/ACL, 2006, pp. 723–730.
[16] H. Schwenk, M.R. Costa-jussà, J.A.R. Fonollosa, Smooth bilingual *N*-gram translation, in: Proceedings of Empirical Methods in Natural Language Processing (EMNLP), 2007, pp. 430–438.
[17] H. Schwenk, P. Koehn, Large and diverse language models for statistical machine translation, in: Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP), 2008, pp. 661–666.
[18] L.H. Son, A. Alluzen, G. Wisniewski, F. Yvon, Training continuous space language models: some practical issues, in: Proceedings of Empirical Methods in Natural Language Processing (EMNLP), 2010, pp. 778–788.
[19] F. Zamora-Martinez, M.J. Castro-Bleda, H. Schwenk, *N*-gram-based machine translation enhanced with neural networks for the French–English BTEC-IWSLT'10 task, in: Proceedings of the Seventh International Workshop on Spoken Language Translation (IWSLT), 2010, pp. 45–52.
[20] U.V. Marti, H. Bunke, The IAM-database: an English sentence database for offline handwriting recognition, Int. J. Doc. Anal. Recognit. 5 (2002) 39–46.
[21] S. Johansson, E. Atwell, R. Garside, G. Leech, The tagged LOB corpus: user's manual (Technical Report), Norwegian Computing Centre for the Humanities, Bergen, Norway, 1986.
[22] W. Francis, H. Kucera, Brown corpus manual, manual of information to accompany a standard corpus of present-day edited American English (Technical Report), Department of Linguistics, Brown University, Providence, Rhode Island, US, 1979.
[23] 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.
[24] F. Jelinek, Statistical Methods for Speech Recognition, Language, Speech, and Communication, The MIT Press, 1997.
[25] M. Sundermeyer, R. Schlüter, H. Ney, On the estimation of discount parameters for language model smoothing, in: Interspeech, Florence, Italy, 2011, pp. 1433–1436.
[26] Y. Bengio, R. Ducharme, P. Vincent, C. Jauvin, A neural probabilistic language model, J. Mach. Learn. Res. 3 (2) (2003) 1137–1155.
[27] A. Graves, S. Fernández, F. Gomez, J. Schmidhuber, Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks, in: Proceedings of the 23rd International Conference on Machine learning, ICML'06, ACM, 2006, pp. 369–376.
[28] A.H. Toselli, A. Juan, J. González, I. Salvador, E. Vidal, F. Casacuberta, D. Keysers, H. Ney, Integrated handwriting recognition and interpretation using finite-state models, Int. J. Pattern Recognit. Artif. Intell. 18 (4) (2004) 519–539.
[29] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1995.
[30] J. Fiscus, A post-processing system to yield reduced word error rates: recognizer output voting error reduction (ROVER), in: Proceedings of the IEEE Workshop on Automatic Speech Recognition Understanding (ASRU), 1997, pp. 347–354.
[31] A. Stolcke, H. Bratt, J. Butzberger, H. Franco, V.R. Rao Gadde, M. Plauché, C. Richey, E. Shriberg, M.K. Sönmez, F. Weng, J. Zheng, The SRI March 2000 Hub-5 conversational speech transcription system, in: Proceedings of the NIST Speech Transcription Workshop, 2000.
[32] S.J. Young, N.H. Russell, J.H.S. Thornton, Token passing: a simple conceptual model for connected speech recognition systems (Technical Report CUED/F-INFENG/TR38), University of Cambridge, 1989.
[33] A. Stolcke, SRILM: an extensible language modeling toolkit, in: Proceedings of the International Conference on Spoken Language Processing (ICSLP), 2002, pp. 901–904.
[34] P.E. Hart, N.J. Nilsson, B. Raphael, A formal basis for the heuristic determination of minimum cost paths, IEEE Trans. Syst., Sci., Cybern. 4 (2) (1968) 100–107.
[35] V. Jimenez, A. Marzal, A lazy version of Eppsteins K shortest paths algorithm, in: Proceedings of the Second International Workshop on Experimental and Efficient Algorithms, WEA 2003, Ascona, Switzerland, May 26–28, 2003, vol. 2, Springer-Verlag, 2003, p. 179.
[36] D.E. Rumelhart, G.E. Hinton, R.J. Williams, PDP: computational models of cognition and perception, I, MIT Press, 1986, Ch. Learning Internal Representations by Error Propagation, pp. 319–362.
[37] J. Park, X. Liu, M. Gales, P. Woodland, Improved neural network based language modelling and adaptation, in: Proceedings of Interspeech, 2010, pp. 26–30.
[38] L. Hansen, Salamon, Neural network ensembles, IEEE Trans. Pattern Anal. Mach. Intell. 12 (10) (1990) 993–1001.
[39] L.I. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, Wiley, New York, NY, USA, 2004.
[40] F. Zamora-Martinez, M.J. Castro-Bleda, Ceu-upv english-spanish system for wmt11, in: Proceedings of the Sixth Workshop on Statistical Machine Translation, Association for Computational Linguistics, Edinburgh, Scotland, 2011, pp. 490-495. ⟨http://www.aclweb.org/anthology/W11-2162⟩.
[41] F. Zamora-Martínez, S. España-Boquera, M. Castro-Bleda, R. de Mori, Cache neural network language models based on long-distance dependencies for a spoken dialog system, in: ICASSP, 2012, pp. 4993–4996.
[42] F. Zamora-Martínez, M. Castro-Bleda, S. España-Boquera, Fast evaluation of connectionist language models, in: International Work-Conference on Artificial Neural Networks, Vol. 5517 of LNCS, Springer, 2009, pp. 33–40.
[43] F. Och, Minimum error rate training in statistical machine translation, in: Proceedings of the ACL, 2003, pp. 160–167.
[44] J. Nelder, R. Mead, A simplex method for function minimization, Comput. Organ. 7 (1965) 308–313.
[45] M. Bisani, H. Ney, Bootstrap estimates for confidence intervals in ASR performance evaluation, in: IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 1, 2004, pp. I-409-12.
[46] P. Natarajan, S. Saleem, R. Prasad, E. MacRostie, K. Subramanian, Multi-lingual offline handwriting recognition using hidden Markov models: a script-independent approach, in: Arabic and Chinese Handwriting Recognition, Springer Berlin/Heidelberg, 2008, pp. 231–250.
[47] R. Bertolami, H. Bunke, Hidden Markov model-based ensemble methods for offline handwritten text line recognition, Pattern Recognition 41 (11) (2008) 3452–3460.
[48] P. Dreuw, G. Heigold, H. Ney, Confidence and margin-based MMI/MPE discriminative training for online handwriting recognition, Int. J. Doc. Anal. Recognit. 14 (3) (2011) 273–288.
[49] T. Plötz, G. Fink, Markov models for offline handwriting recognition: a survey, Int. J. Doc. Anal. Recognit. 12 (2009) 269–298.
[50] P. Dreuw, P. Doetsch, C. Plahl, H. Ney, Hierarchical hybrid MLP/HMM or rather MLP features for a discriminatively trained Gaussian HMM: a comparison for offline handwriting recognition, in: Proceedings of the IEEE International Conference on Image Processing, 2011, pp. 3541–3544.

F. Zamora-Martínez received the Ph.D. degree in computer science from the Universitat Politècnica de València, Spain, in 2012. He is teaching at the Universidad CEU-Cardenal Herrera in Valencia from 2008. He has work during his PhD in the integration of neural network language models for handwritten text recognition, automatic speech recognition and machine translation. His current research interests are deep learning, neural networks and pattern recognition applications.

V. Frinken received his Ph.D. from the University of Bern, Switzerland in 2011. Since then he has been a post-doc researcher at the Computer Vision Center of the Autonomous University of Barcelona, Spain. From 2011–2012 he was a fellow of the Marie Curie Host Fellowship for the Transfer of Knowledge, issued under the European Commission's Framework Program for Research and Technological Development. His research interests are handwriting recognition and machine learning techniques, especially semi-supervised learning, keyword spotting, neural networks, language models, and historical document processing.

S. España-Boquera received the Licenciado degree in computer science in 1998 from the Universitat Politècnica de València, Spain, and the Licenciado degree in mathematics in 2004 from the Universitat de València, Spain. He joined the Departamento de Sistemas Informáticos y Computación of the Universitat Politècnica de València in 1999 where he is currently teaching and finishing his Ph.D. degree. His current research interests include handwriting and speech recognition, image processing, sequence learning, language technologies, and algorithmics.

M.J. Castro-Bleda is currently an Associate Professor of the Departamento de Sistemas Informáticos y Computación at the Universitat Politècnica de València, Spain, where she has taught since 1993. She received her Ph.D. degree in computer science from this same University, in 1998. Her main research interests include machine learning, speech and handwritten text recognition, and language technologies.

A. Fischer received the Ph.D. degree in computer science from the University of Bern, Switzerland, in 2012. Since then he has been a post-doctoral researcher, first at the University of Fribourg, Switzerland, and later at the Centre for Pattern Recognition and Machine Intelligence (CENPARMI) of Concordia University in Montreal, Canada, under a fellowship from the Swiss National Science Foundation. His research interests include handwriting recognition, especially in historical papers, interactive systems, hidden Markov models, neural networks, and structural pattern recognition.

H. Bunke joined the University of Bern as a Professor of Computer Science in 1984. He is now a Professor Emeritus. Horst Bunke served as 1st Vice-President and Acting President of the International Association for Pattern Recognition (IAPR). He also is a former Editor-in-Charge of the International Journal of Pattern Recognition and Artificial Intelligence, and former member of the editorial board of various journals. He is the recipient of the 2010 KS Fu Prize, awarded by the IAPR. Moreover, he received the IAPR/ICDAR Outstanding Achievements Award in 2009 and an honorary doctor degree from the University of Szeged, Hungary, in 2007. He held visiting positions at the University of Notre Dame (Melchor Visiting Professor 2012), Chinese Academy of Science, Beijing (1987, 2012), the IBM Los Angeles Scientific Center (1989), the University of Szeged, Hungary (1991), the University of South Florida at Tampa (1991, 1996, 1998–2007 yearly), the University of Nevada at Las Vegas (1994), Kagawa University, Takamatsu, Japan (1995), Curtin University, Perth, Australia (1999), Australian National University, Canberra (2005), Autonomous University, Barcelona (2005, 2012), and NICTA, Brisbane (2009) and Melbourne (2011). He has about 700 publications, including over 40 authored, co-authored, edited or co-edited books and special editions of journals. His *h*-index is 57, as determined by Google Scholar and harzing.com software. In the DBLP Computer Science Bibliography, which captures more than 2 million papers from the whole discipline of Computer Science, Horst Bunke ranks among the 100 most prolific authors.