A Novel Word Spotting Method Based on Recurrent Neural Networks

Volkmar Frinken*, Andreas Fischer*, R. Manmatha†, and Horst Bunke*
*Institute of Computer Science and Applied Mathematics, University of Bern,
Neubrückstrasse 10, CH-3012 Bern, Switzerland
Email:{frinken, afischer, bunke}@iam.unibe.ch
†Department of Computer Science, University of Massachusetts Amherst
140 Governors Drive, Amherst, MA 01003-9264, USA
Email: manmatha@cs.umass.edu

***Abstract*—Keyword spotting refers to the process of retrieving all instances of a given keyword from a document. In the present paper, a novel keyword spotting method for handwritten documents is described. It is derived from a neural network based system for unconstrained handwriting recognition. As such it performs template-free spotting, i.e. it is not necessary for a keyword to appear in the training set. The keyword spotting is done using a modification of the CTC Token Passing algorithm in conjunction with a recurrent neural network. We demonstrate that the proposed systems outperforms not only a classical dynamic time warping based approach but also a modern keyword spotting system, based on hidden Markov models. Furthermore, we analyze the performance of the underlying neural networks when using them in a recognition task followed by keyword spotting on the produced transcription. We point out the advantages of keyword spotting when compared to classic text line recognition.**

I. Introduction

The automatic recognition of handwritten text – such as letters, manuscripts or entire books – has been a focus of intensive research for several decades [1], [2]. Yet the problem is far from being solved. Particularly in the field of unconstrained handwriting recognition where the writing styles of various writers must be dealt with, severe difficulties are encountered.

Making handwritten texts available for searching and browsing is of tremendous value. For example, one might be interested in finding all occurrences of the word "complain" in the letters sent to a company [3]. As another example, libraries all over the world store huge numbers of handwritten books that are of crucial importance for preserving the world's cultural heritage. Making these books available for searching and browsing would greatly help researchers and the public alike. Certain efforts have already been put into word spotting for historical data [4], [5]. Another related application is the segmentation of images of historical documents into meaningful regions, which can be improved with keyword spotting. In [6] the keyword "Fig." is spotted in the images to help identifying figures and their corresponding captions. Finally, it is worth mentioning that Google and Yahoo have announced their intention to make handwritten books accessible through their search engines [7]. In this context, keyword spotting will be a valuable tool for users browsing the contents of these books.

Transcribing the entire text of a handwritten document for searching is not only inefficient as far as computational costs are concerned, but it may also result in poor performance, since mis-recognized words cannot be found. Therefore, techniques especially designed for the task of keyword spotting have been developed. Next, we review related work from this area.

A. Related Work

1) Word based Keyword Spotting: The task of keyword spotting as detecting a word or a phrase in an image has been initially proposed in [8] for printed text and a few years later in [9] for handwritten text. The first methods consider single word images and adopted approaches common in optical character recognition (OCR). They make use of pixel wise comparison of the query and the test image (or selected parts of it, called zones of interest (ZOI)) or evaluate a global distance value between the two pixel sets. Notable works in this domain include XOR comparison [10], Euclidean distance [11], Scott and Longuet-Higgins distance [9], Hausdorff distance of connected components [12] and the sum of Euclidean distances of corresponding key points (corner features) [13].

More complex, holistic features are the moments of the black pixels, investigated in [14]. In [15] and [16], several binary 'Gradient, Structural and Convexity' (GSC) features are explored. In [17] different pixel-wise gradient matchings are compared and the authors propose an elastic matching procedure. The authors in [18] discuss discrete cosine transformation of the contour to obtain a feature vector, while the use of Gabor features is investigated in [19]. Holistic word features in conjunction with a probabilistic annotation model are proposed in [20]. This system allows one to spot arbitrary words. However, problems are reported for keywords not occurring in the training set.

The most common local approach is to represent a word as a sequence of features, extracted via a sliding window. Comparing such sequences using dynamic time warping (DTW) is one of the most commonly used word spotting methods [21], [22] and still widely used [4]. A comparison of DTW with pixel-wise comparisons is given in [11] and a comparison of (GSC) based spotting with DTW in [15], [23]. A proposal of

using sequential data in conjunction with a holistic approach for keyword spotting in the speech recognition domain is made in [24] where a sequence is transformed into a vector space and then classified using kernel machines.

2) Line based Keyword Spotting: All former approaches require the text in an image to be segmented into individual words before keyword spotting takes place. A different scenario is given if the document is segmented into lines only. A DTW based system that automatically selects keyword candidates in a handwritten text line is described in [25]. For general systems that rely on an automatic segmentation, a method is proposed in [26] that also takes the probability of a correct segmentation into account.

Using a handwritten text line recognition system for keyword spotting circumvents the segmentation problem. Techniques based on handwriting recognition (HWR) have become fairly popular recently, especially using Hidden Markov Models (HMM) [27], [28], [29], [30]. In [8] pseudo-2D HMMs have been investigated and [31] proposes generalized HMMs where more than one emission in each hidden state is allowed. Unsupervised adaptation of whole word HMMs to a specific writer is proposed in [32] and [33] discusses the usage of the Fisher Kernel of the HMM to estimate a good confidence measure.

Not only HMMs but also Neural Networks (NN) have found their way into keyword spotting with so called bidirectional long short-term memory (BLSTM) NN [34], [35], similarly to the system we propose in this paper. The mentioned work, however, deals only with keyword spotting in speech. Furthermore, one node in the output layer of the neural network symbolizes one keyword and is triggered when the word occurs in the input data. Therefore, the number of keywords to be spotted are limited, the word has to be known beforehand, and the keyword must occur in the training set.

3) Document based Keyword Spotting: To work on completely unsegmented pages of text, a system can either include a segmentation step [14] or take a segmentation-free approach. In [36] a codebook of shapes is used to create a compressed version of each document. A keyword search is then done using the stored shape codebook entries. Finally, a common approach to segmentation-free word spotting is to consider the task as an image retrieval tasks for an input shape representing the word image [37], [38], [39].

B. Contribution

In this paper we present a keyword spotting method for handwritten text based on BLSTM Neural Networks. The application of these networks in conjunction with the so-called *CTC Token Passing* algorithm to produce a transcription of handwritten text was presented in [40]. In this paper, we propose a new version of the CTC Token Passing algorithm and apply it to a different task, namely keyword spotting. To the knowledge of the authors, this is the first time that neural networks and CTC Token Passing algorithm are used for this task. With our system, fast and reliable keyword spotting can be performed without the need of neither transcribing the text line nor segmenting it into individual words.

The imposed changes and the different underlying task have further implications on selecting single neural networks. The network having the lowest word error rate when performing recognition is not necessarily the best network for keyword spotting. A system which optimizes word error rates has to do well at recognizing the most frequent words but may do poorly at recognizing function words which are less frequent. According to Zipf's Law, the capability of recognizing stop words has a huge impact on the word error rate, while in keyword spotting stop words do not matter. In fact, they are excluded in most experiments. A keyword spotting system must perform equally well on every search term, even rare words and names that might or might not occur in a dictionary or language model. Thus, a system optimized on the word error rate may not be as good for search as one build directly for this task [41]. In text retrieval it has been shown that training a model by maximizing the likelihood of the training data according to the model does not lead to the best results [42].

A preliminary version of the system described in this paper has been presented in [43], [44]. The current paper provides significant extensions with respect to the underlying methodology and the experimental evaluation. Firstly, we demonstrate the system's applicability to historical data as well as modern handwriting. We use two different historical data sets. One data set consists of letters written by George Washington associates, a well known database for the task of keyword spotting [10], [13], [17], [18], [22], [38]. Since the writing is done in cursive early modern English, we investigated the performance of the proposed system when it is trained on modern handwriting. The second historical data set is an epic poem in middle high German, written in the 13th century [45].

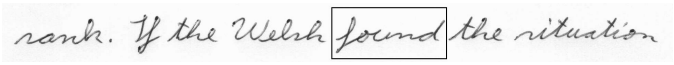
As the second extension over [43], [44], an extensive comparison with several reference systems is presented. On the one hand, a common DTW algorithm as well as a modern HMM-based algorithm is used for comparison. On the other hand, a handwriting recognition system is used that produces an ASCII output on which the keyword search is done.

Finally, a brief discussion is given about keyword spotting as a research area independent of handwriting recognition, backed up with empirical arguments about the correlation of the recognition performance versus the keyword spotting performance of individual systems. The question whether keyword spotting is easier and faster is answered in Section IV-E in the affirmative.

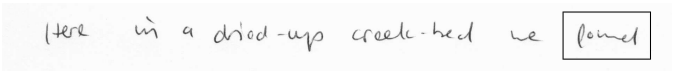
The rest of the paper is structured as follows. In Section II, the proposed keyword spotting system is introduced in detail. The reference systems are presented in Section III. The experimental evaluation is described in Section IV and conclusions are drawn in Section V.

II. Word Spotting using BLSTM

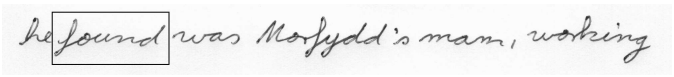
Keyword spotting refers to the process of retrieving all instances of a given word from a document. In this paper, we focus on handwritten documents, such as letters, memorandums, or manuscripts. Without transcribing the data, a user should still be able to search for any possible word, just like using a search engine. How the result of such a search may



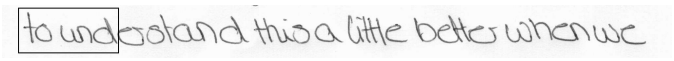
(a) Returned log Likelihood: -1.7125



(b) Returned log Likelihood: -8.4097



(c) Returned log Likelihood: -9.0727



(d) Returned log Likelihood: -11.0900

(e) Returned log Likelihood: -11.6082

Fig. 1. Search results for the word "found".

look like can be seen in Fig. 1. Note that the base system just returns a likelihood of the word being found. If the likelihood of a keyword occurring in the text line is above a given threshold, the text line is returned as a positive match along with the position of the keyword.

A. Preprocessing

In the database used for the experiments described in Section IV, all documents are already segmented into individual text lines. From each line, a sequence of feature vectors is extracted, which is then submitted to the neural network.

For algorithmic processing, a normalized text line image is represented by a sequence of N feature vectors $x_1, \ldots, x_N$ with $x_i \in \mathbb{R}^n$. This sequence is extracted by a sliding window moving from the left to the right over the image. At each of the N positions of the sliding window, n features are extracted. The sliding window has a width of one pixel. It is moved in steps of one pixel, i.e., N equals the width of the text line. From each window $n = 9$ geometric features are extracted, three global and six local ones. The global features are the 0^{th}, 1^{st} and 2^{nd} moment of the black pixels' distribution within the window. The local features are the position of the top-most and that of the bottom-most black pixel, the inclination of the top and bottom contour of the word at the actual window position, the number of vertical black/white transitions, and the average gray scale value between the top-most and bottom-most black pixel. To compute the inclination of the top and bottom contour, the sliding window to the left of the actual one is considered. For further details on the feature extraction step, we refer to [46]. The extracted features are local and each feature vector represents the data of the text line image at one position only. There is, however, an indirect influence between all words of a text line. The preprocessing steps which are explained in IV-A are globally applied to the entire text line. Therefore, the way a word is written influences the modifications applied to the entire text line prior to the feature extraction.

B. The Proposed System

The keyword spotting system proposed in this paper is based upon previous work where BLSTM Neural Networks have been used for the handwriting recognition task [40]. Applying BLSTM NN to handwriting recognition consists of two parts. The first part is a preprocessing phase, performed by the neural network. It maps each position of an input sequence to a vector, indicating the probability of each character possibly being written at that position. The second part, called CTC Token Passing algorithm, takes this sequence of letter probabilities as well as a dictionary and a language model as its input and computes a likely sequence of words. For the keyword spotting task, we leave the first part unchanged but developed a different postprocessing algorithm specifically for keyword spotting.

Since the BLSTM NN preprocessing is already explained in [40] we treat it as a black box in the current paper. We only give a brief explanation in Section II-C below and refer to [40] for further details. Understanding the CTC Token Passing postprocessing upon which our algorithm is founded, however, is essential. Therefore, we describe the algorithm in full length in Section II-D.

C. BLSTM Neural Networks

The underlying neural network is a recurrent neural network with a special architecture. To overcome the vanishing gradient problem that describes the exponential increase or decay of information in recurrent connections in a neural network, the nodes in the hidden layer are replaced by *long short-term memory* (LSTM) cells, displayed in Fig. 2. The gates of these cell are normal nodes and control the flow of information into and out of each cell. When the input gate is open, the central node's value is replaced by the output activation of the net input node. When the output gate is open, information flows out into the network and when the forget gate is open, the cell's memory is reset to zero.

The network is *bidirectional*, meaning the text line is processed from both left-to-right and right-to-left. This is done because context from both sides of a character is useful to improve the recognition. The information from two separate input layers is collected in two separate LSTM layers, respectively, and finally joined in the output layer. This is illustrated in Fig. 3. The output layer contains one node for each possible character as well as one additional node, called ε node, which is activated when no evidence about the presence of any character can be inferred. The normalization of the output activations to sum up to 1 results in a vector that can be interpreted as a character probability vector (Fig. 4).

D. CTC Token Passing Algorithm

The CTC Token Passing algorithm for single words expects a sequence of letter probabilities of length t as input from the neural network, together with a word w as a sequence

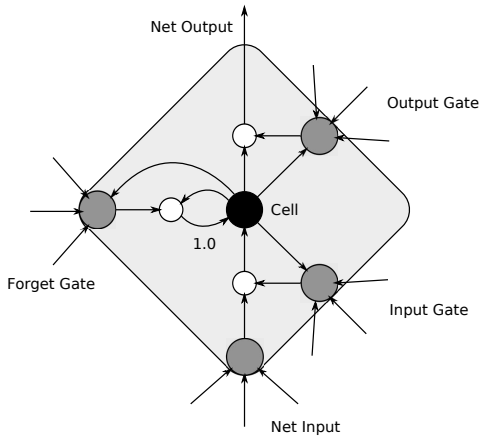


Fig. 2. Gates control the information flow into and out of each LSTM Node.

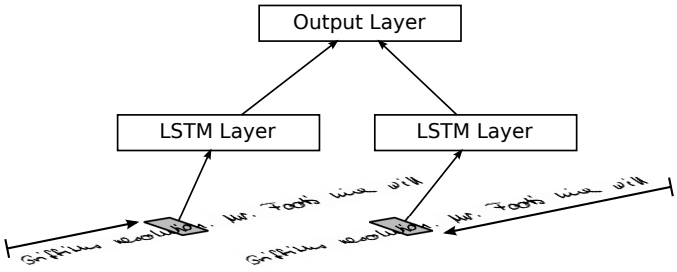


Fig. 3. An illustration of the mode of operation of the BLSTM Neural Network. For each position, the output layer sums up the values of the two hidden LSTM layers.

of ASCII characters. In a dynamic programming fashion, the best path through the letter probability sequence is computed that correspond with the letters from the input word w. The value of that path is then returned as a matching score, i.e. the probability that the input to the neural network was indeed the given word.

The pseudo code of the CTC Token Passing algorithm for single word recognition is given in Algorithm 1. To introduce the formal notation, let the sequence of letter probabilities be n and let $n(l, k)$ denote the probability of the letter l to occur at position k. Furthermore let the word w to be matched be a sequence of letter $w = l_1 l_2 \dots l_n$.

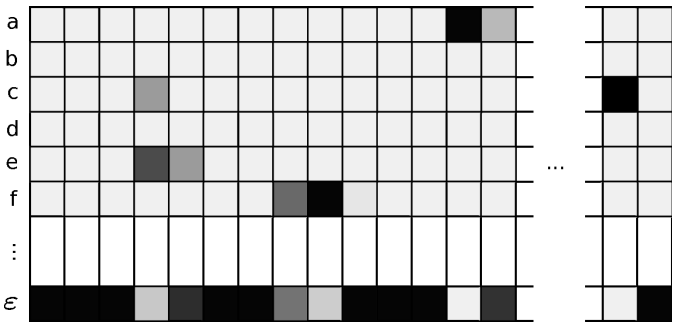


Fig. 4. The activation level for all nodes in the output layer. The activation is close to 0 most of the time for normal letters and peaks only at distinct position. In contrast, the activation level of the ε node is nearly constantly 1.

In a first step, the word w is expanded into a sequence

$$w' = \varepsilon l_1 \varepsilon l_2 \dots \varepsilon l_n \varepsilon = c_1 c_2 c_3 \dots c_{2n+1} \ .$$

Additionally, for every character c_i $(i = 1, \dots, 2n+1)$ and every position $j = 1, \dots, t$ in the text line, a token $\vartheta(i, j)$ is created. This token holds the probability for character c_i to be present at position j together with the probability of the best path from the beginning to position j. All tokens are initialized to 0 except for the tokens for c_1 and c_2, which correspond to the first ε symbol and the first character of the word l_1. These are initialized to the value of ε respectively c_1 at the first position of the sequence (Lines 3–4).

During the following loop over all input sequence positions j, the tokens $\vartheta(\cdot, j)$ are updated, so that a) the token's corresponding letter l occurs at position j, b) in the best path, all letters of the word occur in the given order, c) between two subsequent letters of the word, only ε-node activations are considered and d) if two subsequent letters of a given word are the same (e.g. positions 3 and 4 in "Hello"), at least one ε node must lie between them. To compute the value of the token $\vartheta(i, j)$, a set $\mathbb{T}_{best}$ is created in which all valid tokens are stored that can act as predecessor to the token $\vartheta(i, j)$ according to the constraints mentioned above. If at sequence position j the letter c_i is considered (which might be a real letter or ε), the token corresponding to the same letter c_i at sequence position $j-1$ is valid (Line 8). The token corresponding to letter c_{i-1} (ε if c_i is a real letter and a real letter if $c_i = \varepsilon$) at sequence position $j-1$ is valid for each but the first letter (Line 9 and 10). Since two different letters might follow each other without an ε-node activation, the token corresponding to letter c_{i-2} is valid for these cases, too (Line 11 to 15). Afterwards, the probability of the best token in $\mathbb{T}_{best}$ is multiplied with $n(i, j)$ to obtain the probability of $\vartheta(i, j)$. Algorithm 1 is a slightly simplification of the one given in [40] which was designed for full sequence transcription, but it is sufficient for our task of keyword spotting.

E. Modification to perform Keyword Spotting

Algorithm 1 can now be adopted to spot any given word in a text line s of arbitrary length. The idea is to consider only the product of the output probabilities of the keyword letters at the positions where they would fit best. Therefore, a virtual node is added to the output nodes, called the *any*- or $*$-node. The keyword to be spotted is then preceded and succeeded by $*$ to symbolize the *any*-node. A path through the output activation matrix of a text line will be on the *any*-node until the most likely position of the keyword in the text line, then traverse through the letters of the expanded word and eventually finish on the *any*-node again. The value of the *any*-node is $n(*, j) = 1$ for all values of j so that the $*$-segments of the path do not influence the product.

In order to find entire words but no sub-words contained within longer words, we add a $'\_'$ (white space) character to the front and the end of the keyword:

$$w' = *\_ l_1 l_2 \dots l_n \_*$$

This, however, might lead to problems since keywords occur-

Algorithm 1 The CTC Token Passing Algorithm for single word recognition

Require: input word $w = l_1 l_2 \dots l_n$
Require: sequence of letter probabilities, accessible via $n(\cdot, \cdot)$
Initialization:
expand w to $w' = \varepsilon l_1 \varepsilon l_2 \varepsilon \dots \varepsilon l_n \varepsilon = c_1 c_2 \dots c_{2n+1}$
$\vartheta(1,1) = n(\varepsilon, 1)$
$\vartheta(2,1) = n(l_1, 1)$
Main Loop:
for all sequence positions $2 \leq j \leq t$ **do**
  for all positions i of the extended word $1 \leq i \leq 2n+1$ **do**
    $\mathbb{T}_{best} = \{\vartheta(i, j-1)\}$
    if $i > 1$ **then**
      $\mathbb{T}_{best} = \mathbb{T}_{best} \cup \vartheta(i-1, j-1)$
      if $i > 2$ **then**
        if $c_i \neq \varepsilon$ and $c_i \neq c_{i-2}$ **then**
          $\mathbb{T}_{best} = \mathbb{T}_{best} \cup \vartheta(i-2, j-1)$
        end if
      end if
    end if
    $\vartheta(i,j) = \max(\mathbb{T}_{best}) \cdot n(i,j)$ ▷ multiply the best token's probability with the letter probability
  end for
end for
return $\max\{\vartheta(2n+1, t),\ \vartheta(2n, t)\}$ ▷ The word can either end on the last ε (c_{2n+1}) or on the last regular letter (c_{2n})

ring at the beginning of a text line do not necessarily have a white space preceding them. Similarly, a text line image can end with the last pixels of the last word. Therefore, we add sequence elements to the beginning and the end of each text line that represent extra white space.

If we now use the CTC-Algorithm for single word recognition to compute the probability of the word being w', we compute in fact the probability that the text line starts with any possible character but at some point in the text line the first letter of the word w occurs. It is followed by the second letter, and so on until the word's last letter, followed by a whitespace and then, again, by the *any* character. Obviously, the size and content of the text before and after the keyword w' is irrelevant, since $n(*, j) = 1$. Yet, the returned probability of a word still depends upon the word's length. To receive a normalized value which can then be thresholded, we take the logarithm of the probability $p_{CTC}(w|s)$ and divide it by the search word's length

$$f_{CTC}(w|s) = \frac{\log(p_{CTC}(w|s))}{|w|} \ .$$

An approximation of the keyword's length that works very well is to use the number of characters of the word. This value is constant throughout the test set for each keyword. A more refined procedure is to use the length of the part of the text line that is assumed to be the keyword. In the rest of the paper, however, we focus on using the number of letters in the word for the purpose of normalization because it returned better results.

III. Reference Systems

In this section, we will describe the reference systems to which we compared the approach proposed in this paper. The first one is a Dynamic Time Warping (DTW) based keyword spotting system, while the second one is a recently proposed learning-based keyword spotting system using Hidden Markov Models (HMM). Finally, we also use a state-of-the-art HWR system to transcribe the text. On this result, a simple ASCII search is performed.

Note that the DTW reference system is based on a prior word segmentation of the text line image, much like other popular word spotting techniques, e.g. holistic approaches that model word images with HMMs [47]. For comparison with our NN-based word spotting system, we have applied DTW to perfectly segmented word images, i.e. we do not take segmentation errors into account.

A. DTW Reference System

DTW is a dynamic programming approach that finds an optimal alignment between two sequences by a pairwise comparison of elements of the first sequence to elements of the second sequence. Each element in the one sequence can be assigned to several consecutive elements in the other sequence. In [48], DTW was proposed for word spotting in speech recognition, and also the first approaches to word spotting for handwritten text used DTW representing text as a sequence of features vectors (see Section II). While various features have been proposed in conjunction with DTW [22], [29], [49], we use the same set of features that is used for NN-based word spotting to ensure an objective comparison.

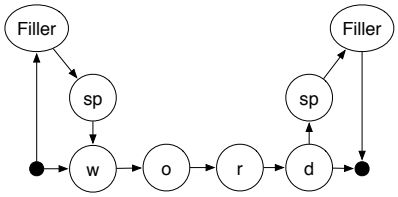
Our DTW implementation, similarly to the one described in [4], makes use of a Sakoe-Chiba band [50] to speed-up the computation. The only pruning criterion we used was the length of the word, i.e. one word image must not be more than twice as long as the other.

In order to spot a certain keyword, all instances of that word occurring in the training set are compared to all words in each text line. In this paper, we consider a perfect, manually corrected word segmentation in order to rule out the influence of segmentation errors on the word spotting performance. This results in a bias of the system evaluation in Section IV in favor of the reference system. The minimum of all these DTW distances serves as a distance function of the keyword's word class to the text line. If the DTW distance of a keyword to the text line is below a given threshold, the text line and the word having the minimum distance is returned as a positive match.

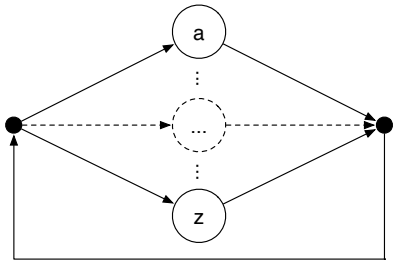
B. HMM Reference System

The second reference system was recently presented in [30]. It is based on Hidden Markov Models (HMMs). HMMs are state-of-the-art for modeling handwritten text [51] and have been widely used for keyword spotting [3], [28], [31], [33], [47], [52], [53]. In [30], trained character models are used to spot arbitrary keywords in complete text line images using an efficient lexicon-free approach.

The same image preprocessing and feature extraction methods are applied as for the proposed system (see Section II). In the training phase, character HMMs are trained based on transcribed text line images. At the recognition stage, the score of an unknown text line image is given by the likelihood ratio $R = \frac{L_K}{L_F}$ between a keyword text line model K and a filler text line model F. The keyword model K is shown in



(a) Keyword HMM



(b) Filler HMM

Fig. 5. Hidden Markov Models

Figure 5(a) for the keyword "word". It is constrained to contain the exact keyword character sequence at the beginning, in the middle, or at the end of the text line, respectively, separated with the space character "sp". The general filler model F is given by an arbitrary sequence of characters as shown in Figure 5(b). It is used to normalize the likelihood score L_K obtained by the keyword model. In a final step, the likelihood ratio R is normalized with respect to the length of the keyword character sequence L and is compared to a threshold T for keyword spotting $\frac{R}{L} > T$. For more details on the HMM-based reference system, we refer to [30].

C. Transcription and ASCII Search

Most papers about word spotting claim that the task of word spotting should not be done using a HWR system to transcribe the text and search on the ASCII output. It is argued that spotting single words is fundamentally easier and should perform substantially better at a lower computational cost. However, the authors of the current paper have not found any formal proof nor any empirical evidence reported in the literature to substantiate this claim. We are therefore interested in such a transcription and a subsequent ASCII search (TAS).

The neural networks used for the word spotting approach proposed in this paper can easily be used for handwriting recognition as mentioned in Subsection II-D. The CTC token passing algorithm described in [40] takes as its input the output activations of the neural network and statistical information about all recognizable words, which implies that a dictionary dictates which words can be recognized at all. As a result of the recognition process, we get the transcription of the given text line, i.e. a likely sequence of words. The CTC Token Passing algorithm, however, is unable to return any form of lattice or *n-best* list. Furthermore, using the returned word probabilities do not perform very well. Hence, we only use one transcription for each text line with the binary information whether the keyword occurs in that transcription or not.

As has been shown in the HWR domain [46], language information can have a positive effect on the recognition rate. Therefore we used two different reference systems in our experiments. The first HWR-based reference system makes use of additional language information, while the second one can only access data that is available in the training and validation set. The language information we use is given in form of a bi-gram language model. Ideally, such a model contains, for each pair of words (w_1, w_2), the probability $p(w_1|w_2)$ that word w_1 is followed by word w_2 in a text. Obviously, these bi-gram probabilities are not known, but can be estimated from a sufficiently large text collection. In our experiments, the following two reference systems were used.

1) TAS with language model: This reference system makes use of the London/Oslo/Bergen (LOB) corpus [54] as an external sources to estimate the bi-gram probabilities of the words. The LOB corpus is a large collection of more than a million words (newspapers, etc.) and resembles a cross-section of the English language at the time of its publication (1961).

2) TAS without language model: In case no additional information is available, we set the list of words that can possibly be recognized equal to all the words in the training set. The bi-gram probabilities are estimated on the training set using *modified Kneser-Ney* smoothing [55], [56].

IV. Experimental Evaluation

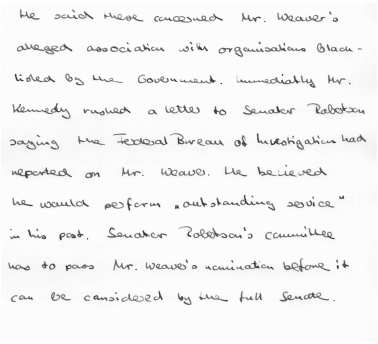
A. The Data Sets

For testing the proposed keyword spotting method, we used three different data sets, the IAM off-line database (IAM DB)[1] [57], the George Washington database[2] (GW DB), and medieval manuscripts of an epic poem (PARZIVAL DB) [45]. See Fig. 6 for samples of the data. The pages of all data sets were scanned and interactively separated into individual text lines.

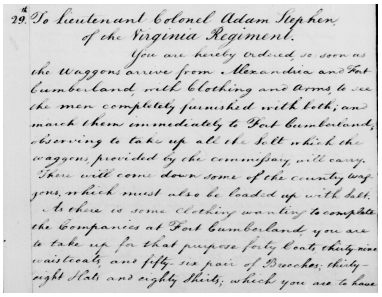
The segmented text lines are normalized prior to recognition in order to cope with different writing styles. First, the skew angle is determined by a regression analysis based on the bottom-most black pixel of each pixel column. Then, the skew of the text line is removed by rotation. Afterwards the slant 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 the skew correction is used, and the upper baseline is found by vertical histogram analysis. For more details on the text line normalization operations, we refer to [46]. Finally the width of the text is normalized. For this purpose, the average distance of black/white transitions along a horizontal straight line through the middle zone is

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

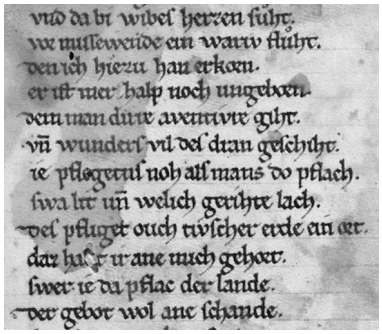
[2] George Washington Papers at the Library of Congress, 1741-1799: Series 2, Letterbook 1, pages 270-279 & 300-309, http://memory.loc.gov/ammem/gwhtml/gwseries2.html



(a) IAM database

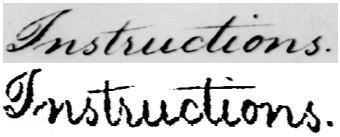


(b) GW database

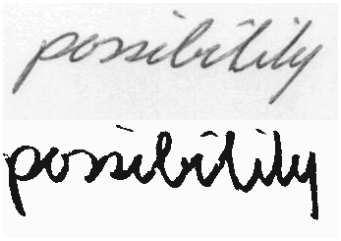


(c) PARZIVAL database

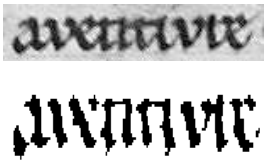
Fig. 6. Samples from the databases used in the experiments.



(a) A word from the GW database.



(b) A word from the IAM database.



(c) A word from the PARZIVAL database.

Fig. 7. A visualization of the effects of the preprocessing steps.

determined and adjusted by horizontal scaling. The result of the preprocessing steps can be seen in Fig. 7.

Additionally, the lines were separated into single words as well. Note that only the DTW reference system requires the words to be separated. By contrast, our approach works with entire text lines. The two databases can be characterized as follows.

IAM off-line DB: This database consists of 1,539 pages of handwritten English text, written by 657 writers. It is split up into a training set of 6161 text lines, a validation set of 920 text lines and a writer independent test set of 929 text lines.

GW DB: The GW Dataset consists of 20 pages of letters, orders and instructions of George Washington from 1755. The pages originate from a large collection with a variety of images, the quality of which ranges from clean to very difficult to read. The selected pages we use are relatively clean. The text is part of a larger corpus, written not only by George Washington but also by some of his associates. It inhibits some variations in writing style. However, the writing on the pages we consider is fairly similar. The considered pages include 4,894 words on 675 text lines. The GW DB contains the same pages as the one in [27], but we found the automatically segmented and extracted words to be too erroneous. Focusing on keyword spotting rather than document image preprocessing in this paper, we manually segmented the data set into individual words. Hence, there is a slight difference in the number of words and word classes.

PARZIVAL DB: We also use the PARZIVAL database presented in [45] for our experimental evaluation. This database contains digital images of medieval manuscripts originating in the 13th century. Arranged in 16 books, the epic poem *Parzival* by Wolfram von Eschenbach was written down in Middle High German with ink on parchment. There exist multiple manuscripts of the poem that differ in writing style and dialect of the language. The manuscript used for experimental evaluation is *St. Gall, collegiate library, cod. 857* that is written by multiple authors. Figure 6(c) shows an example page.

B. Experimental Setup

Using the training set, we trained 50 randomly initialized neural networks and used the validation set to stop the back propagation iterations in the training process. See [40] for details on the neural network training algorithm.

Part of the LOB corpus has been used to create a set of bi-gram probabilities. Since the text written in the IAM database is also a subset of the LOB corpus, the calculated bi-gram probabilities are well suited for that task. For the GW database, the situation is slightly different. Although the text is written is English, the form of English, with respect to both spelling and grammar, has changed since. Therefore, we mixed the bi-gram probabilities computed on the LOB corpus with bi-gram probabilities computed on the training and validation set of the GW DB in order to create a language model that is both general and similar to the language at hand. Mixing has been done using the SRILM toolkit [56]. The same toolkit has been used to create the language model for the PARZIVAL database. The ground truth of both sets, training and validation, have

been used to estimate the bi-gram probabilities and to create the dictionary.

Each systems returns a value for every tested word w. This value is a logarithmic probability $f_{CTC}(w|s)$ in case of our proposed system, the nearest distance to the prototypes $\min_i(DTW(w_i, \cdot))$ when using the DTW approach, or just a binary information whether the word has been found in the transcription using an ASCII search. Note that our DTW implementation returns a DTW distance of infinity if the keyword does not occur in the training set, since no candidate can be found to match the word. We perform a line retrieval task to compare the proposed system with DTW. A text line is considered a positive match if the keyword occurs in the training set. Hence, the DTW system returns the smallest distance of all prototype words to any word in the text line.

The word spotting algorithm compares this value against a global threshold to decide whether or not it is a match. In order to make the results as precise as possible, we used all returned values $f_{CTC}(w|s)$ for the proposed approach and all returned DTW distances for the DTW approach as global thresholds. Each threshold produces one point in the *recall-precision* plot which merge into a continuous curve for many different thresholds. To compare different systems, we chose to consider the average precision over all recall values since it includes information about the entire curve.

IAM DB

We performed slightly different experiments on each of the three datasets. The first experiment was done using the IAM database. We tested 2,807 different keywords (all non stop words among the 4,000 most frequent words[3] that also occur in the training set) in the IAM dataset. The average number of occurrences in the training set of each keyword is 5.26 and in the test set 0.53.

GW DB

The next two experiments were done using the GW database. Due to the relatively small size of this database, we performed a 4-fold cross validation. The 20 pages are split up into four blocks, consisting of five pages each. Two blocks are used for training, one block for validation and one for testing. We selected all words occurring in the training set to perform the word spotting. The average number of occurrences in the training set of each keyword is 2.02 and in the test set 0.74. Note that we also include stop words in this setup to make the results more comparable to the existing literature [41], [59].

With the experiments on the GW data set we addressed also another issue. Because DTW and other QBE systems do not need to be trained on the data set, they can be used for any new script or writing style. An interesting experiment would therefore be to compare them to neural networks that have not been trained on the actual data base. Consequently, we reused the 50 neural networks from the experiments on the IAM DB and evaluated their applicability on the GW data set.

[3]We used the stop word list from the SMART project [58], which can be found at http://jmlr.csail.mit.edu/papers/volume5/lewis04a/a11-smart-stop-list/english.stop

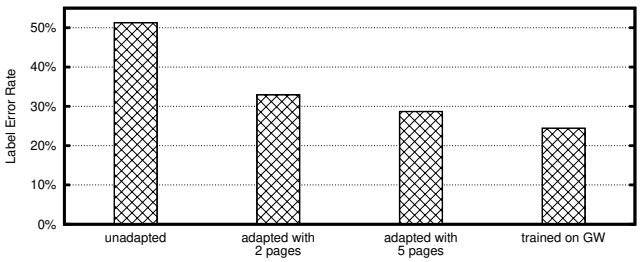


Fig. 8. The average label error rate of the adapted networks when used for transcription.

Assuming a user is willing to transcribe a few pages of text, the network can be retrained on the new data for adaptation. In Fig. 8 the effect of the adaptation to the GW data set can be seen as a reduction of the label error rate. This approach with similar experiments has also been proposed in [44].

Two different adaptation experiments were conducted using again a 4-fold cross validation and the same splitting of the 20 pages into 4 blocks as above. In the first adaptation experiment, we used one page of one block for training and one page of the same block for validation. One block was used for testing, for the purpose of better comparability to the other GW experiments. In the other adaptation experiment, we used two pages of one block for training and the other three pages for validation. Again, one block was used for testing.

In addition, we used the neural networks also for TAS word spotting. Since we assume that we have none or only very little transcribed text of the GW DB when adopting the networks, we used the language model created on the LOB corpus only. However, we added the keywords with a fixed uni-gram probability (the average of all existing uni-gram probabilities).

PARZIVAL DB

The last experiments were done using the PARZIVAL data set. We trained 10 neural networks on 2,237 lines of transcribed text, and used 912 additional lines as a validation set. The test set contained 1,329 lines. The text is written in middle high German and the meaning of most words is unknown to the authors of this paper. Hence we did not try to distinguish between stop words and non-stop word. To avoid an unfair comparison with the DTW reference system, we used all 3,220 words occurring in the training set as possible keywords. The average number of occurrences in the training set of each keyword is 3.53 and in the test set 1.79. We also performed TAS keyword spotting. As mentioned above, the language model was created using the training and validation set. This limits the TAS approach, since words not occurring in the training or validation set cannot be recognized because they are not contained in the language model.

C. Results

In the experiments, the single best neural network on the validation set was chosen to spot the keywords on the test set. Its performance was compared to the performance using DTW, HMM, and TAS for all three databases. The proposed system and the HMM and DTW reference systems use an adjustable

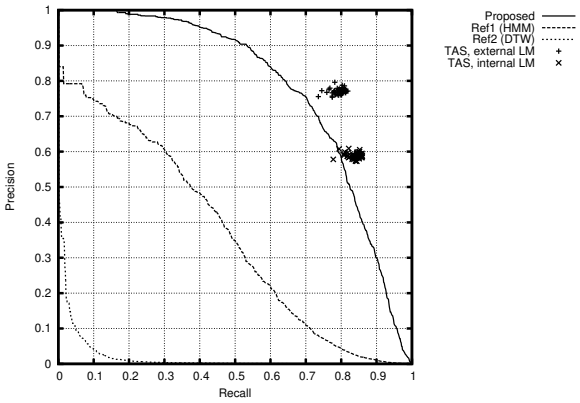


Fig. 9. Recall-precision plots of all systems on the IAM database.

threshold which makes it possible to plot a continuous curve. TAS, on the other hand, works with binary information of a word being successfully spotted or not. It has therefore a fixed precision and a fixed recall characteristic that corresponds to a single point in the plot. Thus we did not select a single best system on the validation set but present the performance of all systems on the IAM and PARZIVAL database results. For the GW database, we differentiate between comparing our system to DTW and HMM, on the one hand, and analyzing the effect of adapting the neural networks to the GW database, on the other hand. For the sake of readability, we do not plot the results of the TAS systems.

In this first experiment, we used the IAM DB to compare DTW, HMM, TAS and the proposed BLSTM keyword spotting system. The resulting *recall-precision* plot can is given in Fig. 9. The *recall-precision* curve of the DTW system can be seen in the lower left corner, indicating that this system might not be well suited for this special task. The system based on HMMs performs much better and can be seen in the middle of the plot. The proposed system, however, works well with diverse handwriting styles and words not encountered in the training set. Transcription and ASCII search keyword spotting result in distinct points even above that line. Clearly, the external language information given in the large LOB corpus has a positive effect on the precision, at he cost of a lower recall value, when compared to using an internal language model only.

In case of the GW database, the proposed BLSTM NN keyword spotting technique outperforms both reference methods which can be seen in Fig. 10(a). The performance of the adapted version are plotted in Fig. 10(b). The system with the lowest performance is the neural network based keyword spotting system trained only on the IAM DB, performing even words than DTW. When the neural networks are adapted to the GW data set, the performance substantially increases. The best system, however, is the one that is trained entirely on the GW DB. In Table I, the average precision of the keyword spotting after the adaptation steps are given. The column *mean* shows the mean of all 50 systems. The column *selected* shows the average precision of the system that performed best on the IAM validation set before adaptation. The last column,

	GWDB		
system	mean	selected	best on valset
IAM NN	0.41	0.43	0.43
IAM NN (2 pages)	0.68	0.71	0.47
IAM NN (5 pages)	0.73	0.76	0.80

TABLE I
THE AVERAGE PRECISION OF THE ADAPTED SYSTEMS.

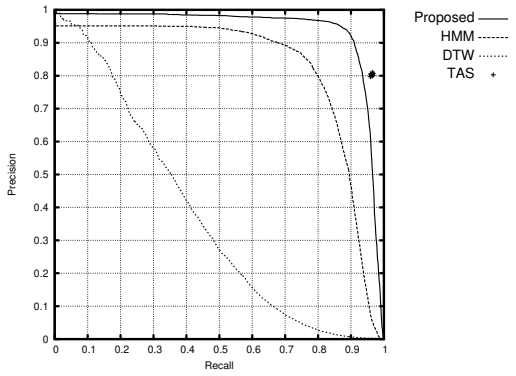


Fig. 11. *Recall-precision* plots of all systems on the PARZIVAL database.

best on valset, shows the average precision of the system that performed best on the corresponding validation sets, i.e. the IAM validation set for the unadapted system, the 1 page GW validation set for the 2 page adaptation, and the 3 page GW validation set for the 5 page adaptation.

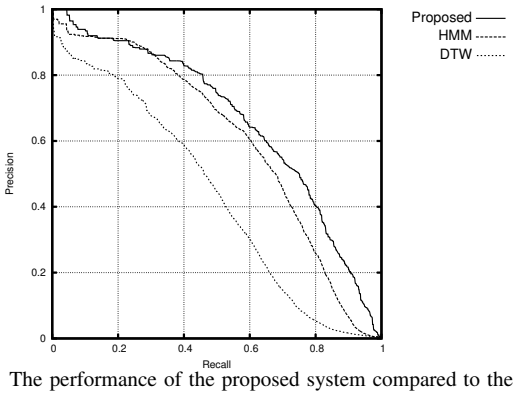
For the results of the TAS systems we refer to Table II. One can see that the adaptation process leads to an increase in precision and recall. The recall percentage, however, of the system that is trained entirely on the GW DB is not met. It also shows that the information derived from the larger LOB corpus may still be useful, although the kind of texts and time of origin are somewhat different.

Finally, the results from the PARZIVAL DB also show the advantage of the proposed system over both reference systems. The neat, regular writing style is advantageous to

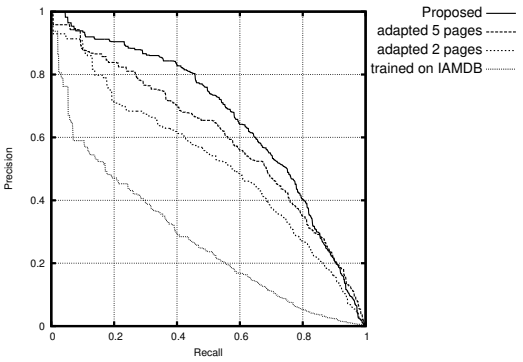
	language	average		best system	
Database	model	prec.	recall	prec.	recall
IAM DB	external	0.77	0.79	0.78	0.79
	internal	0.59	0.84	0.59	0.84
GW DB	external	0.78	0.75	0.82	0.83
	internal	0.66	0.79	0.69	0.85

Database	Adaptation method	language model	prec.	recall
GW DB	unadapted	extern (LOB)	0.67	0.47
	adapted (2 pages)		0.83	0.69
	adapted (5 pages)		0.86	0.74

TABLE II
THE AVERAGE PRECISION AND RECALL VALUES OF THE TRANSCRIPTION AND ASCII SEARCH APPROACH.



(a) The performance of the proposed system compared to the HMM and DTW reference system.



(b) The performance during the adaptation process.

Fig. 10. Recall-Precision plots of all systems on the GW database.

system	IAM DB	GW DB	PARZIVAL DB
Average of NNs	0.76	0.71	0.92
NN best on val set	0.78	0.84	0.94
NN best on test set	0.79	0.84	0.94
HMM	0.36	0.60	0.83
DTW	0.02	0.48	0.37

TABLE III
AVERAGE PRECISION OF THE SPOTTING TASK ON THE IAM DB AND PARZIVAL DB

all systems, and the ability to learn leads to a nearly perfect *recall-precision*-curve (Fig. 11).

A consolidated view clearly indicates the superiority of the proposed word-spotting system not only over the DTW reference system but also over the HMM reference system. Table III shows the average precision of the compared methods. Note that restricting the keywords in the IAM DB experiment to words occurring in the test set does not lead to such a substantial improvement of the DTW performance as it does on the GW DB, which is possibly due to the more diverse writing style encountered in the IAM DB. Without a training set at hand, QBE systems in general or DTW in particular are the only systems applicable to perform word spotting. However, if training data exists it can be exploited to further improve the performance.

D. Comparison with related literature

The experiments presented in this paper, especially the ones on the GW data set, are similar to already published results. Although the task, the data set and the evaluation method are not fully comparable, we will discuss some published results, to put this paper in relation to existing work. The most prominent works on this database have been published by Rath et al [59], [4].

Most of the literature dealing with George Washington's data use automatically segmented words for testing. Additionally, not always the same pages of the manuscript are used. In [13], the authors focus on a subset of 10 selected pages that have a good quality. They report an average precision of 0.65 using a DTW system and 0.62 using corner feature correspondences.

In [22] the authors propose a form of histogram of Oriented Gradients (HOG) features and Continuous Dynamic Programming as a line based approach. They report an average precision of 0.79 for a 15 keywords. Exactly the same words have been used as keywords with an R-precision of 0.6 in [17]. Both of these works use the same 20 pages as we do, although the groud truth might be slightly different, due to rules on how to handle hypthenated words. The authors in [17] use the same database and ground truth as [59], while the authors of [22], similarly to us, created a new ground thruth.

In [41] a decision tree is investigated and in [59] a statistical model using holistic word features. While the authors of the papers use the same 20 page dataset as we do, both references use a different cross-validation set up. They perform a 10-fold cross validation where 90% of the lines constitute the training set while 10% of the lines where used for testing. When using all words with at least one occurrence in the training and test set, the authors report an average precision of 0.54 [59] and 0.79 [41]. This setup is the fairly close to the setup used in our experiments on the GW data base where we have reached an average precision of 0.84.

Word recognition has also been investigated using the GW data set. In [27], the authors experimented with an HMM-based system along with a statistical language model somewhat similar to ours. They experimented with several language models, including an internal model derived from the 19 pages of their cross-validation training set and an extended model using text written at around the same time. Note, that holistic features from single word images are used. Hence their HMM consists of one state per word. The paper reports an accuracy of 0.470 with and 0.606 without out-of-vocabulary (OOV) words when the HMM is trained on 10 pages, like in ourexperiments. When using 19 pages for training and one for testing, the accuracy is increased to 0.551 resp. 0.651. These two figures of performance are used in [18] as a benchmark and topped using a matching technique on the words' contours. This leads to an accuracy of 0.694 with and 0.826 without OOV words. Further reported accuracies are 0.611 with and 0.723 without OOV words in [60] and 0.84 with and 0.71

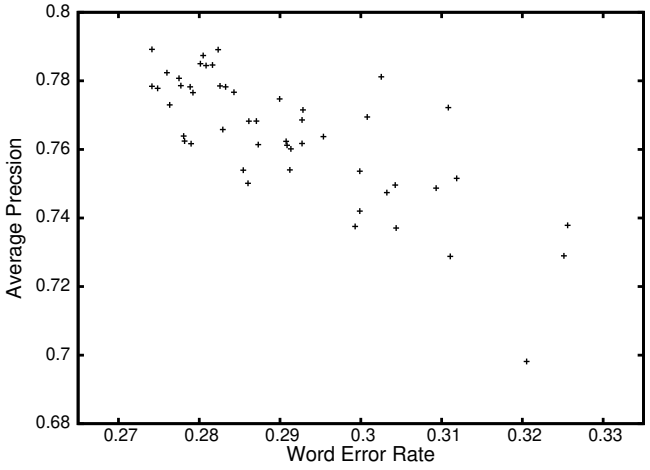


Fig. 12. The average precision of a word spotting system (on the test set) compared to the word error rate when used as a handwriting recognition system (on the validation set) for all 50 networks on the IAM database.

without OOV words in [61]. Both works use also a 20-fold cross validation with a training set of 19 pages and manually segmented words.

We can report on our data set an average precision of 0.84, which is not only significantly higher than the DTW based system, but also the highest reported number among the literature that covers learning based approaches. Although these numbers can not be directly compared, they show that the method proposed in this paper is very well suited for historical data.

E. Correlation of Keyword Spotting and Recognition

As seen above, a neural network used for word spotting can be used, without modification, for handwriting recognition as well. The only change needed concerns the post-processing algorithm (CTC). This makes it possible to compute the correlation between the recognition accuracy of a neural network and its performance when used for keyword spotting.

The scatter plot in Fig. 12 shows, on the IAM database, the correlation between the word error rate when using the neural network as a handwriting recognition system and the average precision when using the neural network for the task of word spotting. The complete list of correlation coefficients can be found in Table IV. It can be seen that, although a high correlation exists between the spotting and recognition performance, this correlation is not perfect.

These results show that the underlying problems to be solved, though similar, are not identical. A handwriting recognition system decides on the best words among a set of possible candidates. Our results show that this works very well resulting in a high precision and a high recall value. Yet, it has the disadvantage that, for a certain position in the text, it can only return the word recognized. A keyword spotting system on the other hand returns the keyword's likelihood. Therefore, keywords can even be found if the transcription of an HWR system misrecognized a word. Another point that distinguishes the two approaches is the capability to decide whether recall or precision is more important in a given retrieval task. However, the most important difference is that keywords can obviously be spotted successfully without a complex handwriting recognition system. E.g., DTW only needs very few sample words and the proposed BLSTM neural networks can be trained on a different database and achieves a similar performance, although a transcription using these neural networks would result in a text having a label error rate of $> 50\%$.

Furthermore, the proposed keyword spotting system needs only a few milliseconds to process one text line, while a transcription of a text line using a language model needs up to several minutes. As long as time requirements do not play any role, it might be beneficial to perform an off-line preprocessing of the entire archive, for example in the form of a *n-best* transcription of each text line on both word and character level to preserve the possibility to find words that do not occur in a dictionary. When this is not possible, our proposed system seems to be the best choice. Examples of this are fast search on newly scanned documents or large databases. Especially for historical documents, databases can be rather large. The entire George Washington collection contains 140'000 pages and the historical document collection of the cabinet of the Dutch Queen [62] contains approximately 300'000 pages. A keyword spotting system based on text recognition does not seem feasible for these databases. Assuming text is written on three quarter of the collection, each page contains 20 lines and a recognition system needs 5 minutes to recognize one text line. Then it takes 28 respectively 40 years to do the preprocessing of the databases. On the other hand, a keyword can be spotted on one text line in one millisecond, leading to a search time of 33, respectively 100 minutes.

For large corpora with this approach, the recognition task is unrealistic while the word spotting task is potentially useful. They are complementary and one could imagine doing word spotting to locate the item of interest and then recognizing only the page of interest.

As fas as time is concerned, all learning based systems have the disadvantage of needing the time it takes to train the system. However, once the system is trained, the keyword spotting can be performed faster than example based systems. While the number of words being checked in each text line is the same, the words do not have to be compared to all suitable training data but only to the model parameters.

V. Conclusion

In this paper we presented a novel keyword spotting approach using bidirectional long-short term neural networks in combination with a modified version of the Connectionist Token Passing algorithm. This system has several advantages compared to existing techniques. First, it is a line based approach and does not need any word segmentation. Secondly, although the system needs to be trained, it does not require bounding boxes around characters or words as often needed in the keyword spotting literature. The only requirement is a transcription of the text lines in the training set. Finally, being derived from a general neural network based handwritten text

Database	Language Model	Accuracy valset	Accuracy testset
IAM DB	external	70.90	65.43
	internal	68.10	65.43
GW DB	external	67.11	55.98
	internal	76.3	52.92

Database	Language Model	Correlation valset	Correlation testset
IAM DB	external	-0.768	-0.850
	internal	-0.812	-0.806
GW DB	external	-0.914	-0.896
	internal	-0.961	-0.835

TABLE IV
THE RECOGNITION ACCURACY AND THE CORRELATION BETWEEN THE AVERAGE WORD SPOTTING PRECISION AND THE HANDWRITING RECOGNITION.

recognition system, any arbitrary string can be searched for, not just the words appearing in the training set.

We compared this approach to the common dynamic time warping approach on three data sets as well as a modern HMM-based keyword spotting system, which include both modern and historical handwritten data. Furthermore we compared these methods with the results returned by recognizing the text using a complete handwriting recognition system followed by a plain search in the ASCII character output string.

We demonstrated that dynamic time warping, while applicable with some success to historical data, has difficulties on the modern handwriting data set to cope with the differences in handwriting styles encountered between the training and the test set. The same argument seems valid for all spotting systems tried so far that treat word spotting as an image retrieval problem where similar sub-images of a given prototype are to be found. The HMM-based approach performs better and is able to cope with diverse writing styles. Yet, is is also constantly outperformed by the proposed system. To do keyword spotting for words that do not occur in the training set, it seems that more sophisticated methods or even handwriting recognition are necessary, especially for arbitrary and diverse writings.

The system proposed in this paper – a handwriting recognition system adjusted to the task of word-spotting – is flexible enough to deal with a variety of diverse handwritten texts. Due to the initialization of the neural networks using random weights, a natural variance of several different neural networks can be observed. However, this is not a problem because it is possible to select high performing candidates on the validation set.

We analyzed the performance of the neural networks when using them in a recognition task followed by word spotting on the output and investigated the influence of external information in form of a language model. The increased precision achieved gives rise to new research directions. A keyword spotting system that, after finding a candidate, decodes the word preceding it to include the bi-gram probability, merits further investigation.

The combination of several systems, as it has been done successfully with handwriting recognition systems, seems an interesting option to be investigated in future research.

ACKNOWLEDGMENTS

This work has been supported by the Swiss National Science Foundation (Project CRSI22_125220). We thank Alex Graves for kindly providing us with the BLSTM Neural Network source code.

REFERENCES

[1] A. Vinciarelli, "A Survey On Off-Line Cursive Word Recognition," *Pattern Recognition*, vol. 35, no. 7, pp. 1433–1446, 2002.
[2] R. Plamondon and S. N. Srihari, "On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 63–84, 2000.
[3] C. Choisy, "Dynamic Handwritten Keyword Spotting Based on the NSHP-HMM," in *9th Int'l Conf. on Document Analysis and Recognition*, 2007, pp. 242–246.
[4] T. M. Rath and R. Manmatha, "Word Spotting for Historical Documents," *Int'l Journal of Document Analysis and Recognition*, vol. 9, pp. 139–152, 2007.
[5] Y. Leydier, A. Ouji, LeBourgeois, and H. Emptoz, "Towards an Omnilingual Word Retrieval System for Ancient Manuscripts," *Pattern Recognition*, vol. 42, no. 9, pp. 2089–2105, 2009.
[6] K. Khurshid, C. Faure, and N. Vincent, "Fusion of Word Spotting and Spartial Information for Figure Caption Retrival in Historical Document Images," in *10th Int'l Conf. on Document Analysis and Recognition*, vol. 1, 2009, pp. 266–270.
[7] S. Levy, "Google's Two Revolutions," *Newsweek Dec. 27 / Jan. 3*, 2004, Available: http://www.msnbc.msn.com/id/6733225/site/newsweek/
[8] S.-S. Kuo and O. E. Agazzi, "Keyword Spotting in Poorly Printed Documents using Pseudo 2-d Hidden Markov Models," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 16, no. 8, pp. 842–848, 1994.
[9] R. Manmatha, C. Han, and E. Riseman, "Word Spotting: a New Approach to Indexing Handwriting," in *Int'l Conf. on Computer Vision and Pattern Recognition*, 1996, pp. 631–637.
[10] R. Manmatha and W. B. Croft, *Word Spotting: Indexing Handwritten Archives*. MIT Press, 1997, ch. 3, pp. 43–64.
[11] R. Manmatha and T. M. Rath, "Indexing of Handwritten Historical Documents - Recent Progress," in *Symposium on Document Image Understanding Technology*, 2003, pp. 77–85.
[12] Y. Lu and C. L. Tan, "Word Spotting in Chinese Document Images without Layout Analysis," in *16th Int'l Conference on Pattern Recognition*, 2002, pp. 57–60.
[13] J. Rothfeder, S. Feng, and T. M. Rath, "Using Corner Feature Correspondences to Rank Word Images by Similarity," in *Workshop on Document Image Analysis and Retrieval*, 2003, p. 30.
[14] A. Bhardwaj, D. Jose, and V. Govindaraju, "Script Independent Word Spotting in Multilingual Documents," in *2nd Int'l Workshop on Cross Lingual Information Access*, 2008, pp. 48–54.
[15] B. Zhang, S. N. Srihari, and C. Huang, "Word Image Retrieval Using Binary Features," in *Proceedings of the SPIE*, vol. 5296, 2004, pp. 45–53.
[16] S. Srihari, H. Srinivasan, P. Babu, and C. Bhole, "Spotting Words in Handwritten Arabic Documents," in *Proc. of SPIE*, vol. 6067, Document Recognition and Retrieval XIII, 2006.
[17] Y. Leydier, F. Lebourgeois, and H. Emptoz, "Text Search for Medieval Manuscript Images," *Pattern Recognition*, vol. 40, pp. 3552–3567, 2007.
[18] T. Adamek, N. E. Connor, and A. F. Smeaton, "Word matching using Single Closed Contours for Indexing Historical Documents," *Journal on Document Analysis and Recognition*, vol. 9, no. 2, pp. 153–165, 2007.
[19] H. Cao and V. Govindaraju, "Template-free Word Spotting in Low-Quality Manuscripts," in *6th Int'l Conf. on Advances in Pattern Recognition*, 2007.
[20] T. M. Rath, R. Manmatha, and V. Lavrenko, "A Search Engine for Historical Manuscript Images," in *27th Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval*, 2004, pp. 369–376.
[21] T. M. Rath and R. Manmatha, "Word Image Matching Using Dynamic Time Warping," in *Computer Vision and Pattern Recognition*, vol. 2, 2003, pp. 521–527.

[22] K. Terasawa and Y. Tanaka, "Slit Style HOG Features for Document Image Word Spotting," in *10th Int'l Conf. on Document Analysis and Recognition*, vol. 1, 2009, pp. 116–120.

[23] S. N. Srihari, H. Srinivasan, C. Huang, and S. Shetty, "Spotting Words in Latin, Devanagari and Arabic Scripts," *Indian Jounal of Artificial Intelligence*, vol. 16, no. 3, pp. 2–9, 2006.

[24] J. Kesheta, D. Grangierb, and S. Bengioc, "Discriminative Keyword Spotting," *Speech Communication*, vol. 51, no. 4, pp. 317 – 329, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/B6V1C-4TPHRJ2-1/2/170cdbfefa41f6916f1d1d7aa1e70c55

[25] A. Kołcz, J. Alspector, M. F. Augusteijn, R. Carlson, and G. V. Popescu, "A Line-Oriented Approach to Word Spotting in Handwritten Documents," *Pattern Analysis and Applications*, vol. 3, pp. 153–168, 2000.

[26] H. Cao, A. Bhardwaj, and V. Govindaraju, "A Probabilistic Method for Keyword Retrieval in Handwritten Document Images," *Pattern Recognition*, vol. 42, no. 12, pp. 3374–3382, December 2009. [Online]. Available: http://dx.doi.org/10.1016/j.patcog.2009.02.003

[27] V. Lavrenko, T. M. Rath, and R. Manmatha, "Holistic Word Recognition for Handwritten Historical Documents," in *Int'l Workshop on Document Image Analysis for Libraries*, 2004, pp. 278–287.

[28] J. Chan, C. Ziftci, and D. Forsyth, "Searching Off-Line Arabic Documents," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2006, pp. 1455–1462.

[29] J. A. Rodríguez and F. Perronnin, "Local Gradient Histrogram Features For Word Spotting in Unconstrained Handwritten Documents," in *11th Int'l Conf. Frontiers in Handwriting Recognition*, 2008, pp. 7–12.

[30] A. Fischer, A. Keller, V. Frinken, and H. Bunke, "HMM-Based Word Spotting in Handwritten Documents Using Subword Models," in *20th Int'l Conf. on Pattern Recognition*, 2010, pp. 3416–3419.

[31] J. Edwards, Y. Whye, T. David, F. Roger, B. M. Maire, and G. Vesom, "Making Latin Manuscripts Searchable using gHMM's," in *Advances in Neural Information Processing Systems (NIPS) 17*. MIT Press, 2004, pp. 385–392.

[32] J. A. Rodríguez, F. Perronnin, G. Sánchez, and J. Lladós, "Unsupervised Writer Style Adaptation for Handwritten Word Spotting," in *19th Int'l Conf. on Pattern Recognition*, 2008, pp. 1–4.

[33] F. Perronnin and J. Rodriguez-Serrano, "Fisher Kernels for Handwritten Word-spotting," in *10th Int'l Conf. on Document Analysis and Recognition*, vol. 1, 2009, pp. 106–110.

[34] S. Fernández, A. Graves, and J. Schmidhuber, "An Application of Recurrent Neural Networks to Discriminative Keyword Spotting," in *17th Int'l Conf. on Artificial Neural Networks*, ser. Lecture Notes in Computer Science, vol. 4669, 2007, pp. 220–229.

[35] M. Wollmer, F. Eyben, J. Keshet, A. Graves, B. Schuller, and G. Rigoll, "Robust Discriminative Keyword Spotting for Emotionally Colored Spontaneous Speech Using Bidirectional LSTM Networks," in *IEEE Int'l Conf. on Acustics, Speech and Signal Processing*, 2009, pp. 3949–3952.

[36] E. Saykol, A. K. Sinop, U. Gdkbay, O. Ulusoy, and A. E. Cetin, "Content-based Retrieval of Historical Ottoman Documents stored as Textual Images," *IEEE Trans. on Image Processing*, vol. 13, no. 3, pp. 314–325, 2004.

[37] R. F. Moghaddam and M. Cheriet, "Application on Multi-level Classifier and Clustering for automatic Word spotting in Historical Document Images," in *10th Int'l Conf. on Document Analysis and Recognition*, vol. 2, 2009, pp. 511–515.

[38] Y. Leydier, F. L. Bourgois, and H. Emptoz, "Omnilingual Segmentation-free Word Spotting for Ancient Manuscripts Indexation," in *8th Int'l Conf. on Document Analysis and Recognition*, 2005, pp. 533–537.

[39] B. Gatos and I. Pratikakis, "Segmentation-free Word Spotting in Historical Printed Documents," in *10th Int'l Conf. on Document Analysis and Recognition*, vol. 1, 2009, pp. 271–275.

[40] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, "A Novel Connectionist System for Unconstrained Handwriting Recognition," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 855–868, 2009.

[41] N. R. Howe, T. M. Rath, and R. Manmatha, "Boosted Decision Trees for Word Recognition in Handwritten Document Retrieval," in *28th Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval*, 2005, pp. 377–383.

[42] D. Metzler and W. B. Croft, "A Markov Random Field Model for Term Dependencies," in *28th Annuar ACM SIGIR Conf. on Research and Development in Information Retrieval*, 2005, pp. 472–479.

[43] V. Frinken, A. Fischer, and H. Bunke, "A Novel Word Spotting Algorithm Using Bidirectional Long Short-Term Memory Neural Networks," in *4th Workshop on Artificial Neural Networks in Pattern Recognition*, 2010, pp. 185–196.

[44] V. Frinken, A. Fischer, R. Manmatha, and H. Bunke, "Adapting BLSTM Neural Network based Keyword Spotting trained on Modern Data to Historical Documents," in *10th Int'l Conf. on Frontiers in Handwriting Recognition*, 2010, pp. 352–257.

[45] A. Fischer, M. Wüthrich, M. Liwicki, V. Frinken, H. Bunke, G. Viehhauser, and M. Stolz, "Automatic Transcription of Handwritten Medieval Documents," in *15th International Conference on Virtual Systems and Multimedia*, 2009, pp. 137–142.

[46] U.-V. Marti and H. Bunke, "Using a Statistical Language Model to Improve the Performance of an HMM-Based Cursive Handwriting Recognition System," *Int'l Journal of Pattern Recognition and Artificial Intelligence*, vol. 15, pp. 65–90, 2001.

[47] J. A. Rodríguez-Serrano and F. Perronnin, "Handwritten Word-Spotting Using Hidden Markov Models and Universal Vocabularies," *Pattern Recognition*, vol. 42, no. 9, pp. 2106–2116, 2009.

[48] A. E. R. Cory S. Myers, Lawrence R. Rabiner, "An Investigation of the Use of Dynamic Time Warping for Word Spotting and Connected Speech Recognition," in *IEEE Int'l Conf. on Acoustics, Speech, and Signal Processing*, 1980, pp. 173–177.

[49] T. M. Rath and R. Manmatha, "Features for word spotting in historical manuscripts," in *7th Int'l Conf. Document Analysis and Recognition*, 2003, pp. 218–222.

[50] H. Sakoe and S. Chiba, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition," *Trans. on Acoustics, Speech, & Signal Processing*, vol. 26, pp. 43–49, 1978.

[51] T. Ploetz and G. A. Fink, "Markov Models for Offline Handwriting Recognition: a Survey," *Int. Journal on Document Analysis and Recognition*, vol. 12, no. 12, pp. 269–298, 2009.

[52] M. A. El-Yacoubi, M. Gilloux, and J.-M. Bertille, "A Statistical Approach for Phrase Location and Recognition within a Text Line: an Application to Street Name Recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, pp. 172–188, 2002.

[53] S. Thomas, C. Chatelain, L. Heutte, and T. Paquet, "An Information Extraction Model for Unconstrained Handwritten Documents," in *20th Int'l Conf. on Pattern Recognition*, 2010, pp. 3412–3415.

[54] H. Kucera and W. N. Francis, *Manual of Information to accompany A Standard Corpus of Present-Day Edited American English, for use with Digital Computers*, Brown University, Department of Linguistics, Providence, Rhode Island, 1964, revised 1971. Revised and amplified 1979.

[55] J. T. Goodman, "A Bit of Progress in Language Modeling - Extended Version," Microsoft Research, One Microsoft Way Redmond, WA 98052, Tech. Rep. MSR-TR-2001-72, 8 2001.

[56] A. Stolke, "SRILM - An Extensible Language Modeling Toolkit," in *Int'l Conf. Spoken Language Processing*, 2002, pp. 901–904.

[57] U.-V. Marti and H. Bunke, "The IAM-Database: An English Sentence Database for Offline Handwriting Recognition," *Int'l Journal on Document Analysis and Recognition*, vol. 5, pp. 39–46, 2002.

[58] G. Salton, *The SMART Retrieval System – Experiments in Automatic Document Processing*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1971.

[59] T. M. Rath, V. Lavrenko, and R. Manmatha, "A Statistical Approach to Retrieving Historical Manuscript Images without Recognition," Center for Intelligent Information Retrival, Tech. Rep. MM-42, 2003.

[60] S. Feng, "Statistical Models for Text Query-Based Image Retrieval," Ph.D. dissertation, University of Massachusetts, Amherst, May 2008.

[61] N. R. Howe, S. Feng, and R. Manmatha, "Finding Words in Alphabet Soup: Inference on Freeform Character Recognition for Historical Scripts," *Pattern Recognition*, vol. 42, no. 12, pp. 3338–3347, 12 2009.

[62] M. Bulacu, R. van Koert, L. Schomaker, and T. van der Zant, "Layout Analysis of Historical Documents for Searching the Archives of the Cabinet of the Dutch Queen," in *9'th Int'l Conf. on Document Analysis and Recognition*, 2007, pp. 367–361.

Volkmar Frinken received his M.S. in Computer Science from the University of Dortmund, Germany, in 2007. He recently completed his PhD at the University of Bern, Switzerland, where he is also employed as a research assistant at the artificial intelligence research group. In summer 2011 he will take on a post-doc position at the Autonomous University of Barcelona, Spain. His research areas include handwriting recognition and keyword spotting, semi-supervised learning, and document analysis for historical and modern document.

Andreas Fischer received his MSc degree in computer science from the University of Bern, Switzerland, in 2008. Since then, he is a PhD student and research assistant at the Institute of Computer Science and Applied Mathematics of the University of Bern. The research topic of his PhD thesis is handwriting recognition in historical documents. His research interests include handwriting recognition, interactive systems, hidden Markov models, neural networks, and syntactic pattern recognition.

R. Manmatha is a Research Associate Professor in the Department of Computer Science at the University of Massachusetts in Amherst. He obtained his Phd from the University of Massachusetts. His research is in the areas of retrieving and organizing images/videos, scanned handwritten and printed documents. He has worked on automatic image annotation and retrieval and on word spotting for scanned handwritten documents. One of his current projects involves organizing millions of scanned printed books. His students and he built the first automatic demonstration retrieval system for historical handwritten documents - a portion of George Washington's documents. He was a co-founder of SnapTell, a mobile image search company, bought by Amazon.

Horst Bunke is a professor of Computer Science at the University of Bern, Switzerland. From 1998 to 2000 he served as the 1st Vice-President of the International Association for Pattern Recognition (IAPR) and in 2000 he was Acting President of IAPR. Horst Bunke is a Fellow of the IAPR, former Editor-in-Charge of the International Journal of Pattern Recognition and Artificial Intelligence, former Editor-in-Chief of the journal Electronic Letters of Computer Vision and Image Analysis, Editor-in-Chief of the book series on Machine Perception and Artificial Intelligence by World Scientific Publ. Co., Advisory Editor of Pattern Recognition, Associate Editor of Acta Cybernetica and Frontiers of Computer Science in China, and former Associate Editor of the International Journal of Document Analysis and Recognition, and Pattern Analysis and Applications. Horst Bunke 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 has more than 650 publications, including over 40 authored, co-authored, edited or co-edited books and special editions of journals.