BLSTM Neural Network based Word Retrieval for Hindi Documents

Raman Jain *, Volkmar Frinken †, C.V. Jawahar * and R. Manmatha ‡
* Center for Visual Information Technology
International Institute of Information Technology Hyderabad, Hyderabad, India
ramanjain@research.iiit.ac.in,jawahar@iiit.ac.in
† Institute of Computer Science and Applied Mathematics, University of Bern
Neubruckstrasse 10, CH-3012 Bern, Switzerland
frinken@iam.unibe.ch
‡ Department of Computer Science, University of Massachusetts
140 Governors Drive, Amherst, MA 01003-9264, USA
manmatha@cs.umass.edu

***Abstract*—Retrieval from Hindi document image collections is a challenging task. This is partly due to the complexity of the script, which has more than 800 unique ligatures. In addition, segmentation and recognition of individual characters often becomes difficult due to the writing style as well as degradations in the print. For these reasons, robust OCRs are non existent for Hindi. Therefore, Hindi document repositories are not amenable to indexing and retrieval. In this paper, we propose a scheme for retrieving relevant Hindi documents in response to a query word. This approach uses BLSTM neural networks. Designed to take contextual information into account, these networks can handle word images that can not be robustly segmented into individual characters. By zoning the Hindi words, we simplify the problem and obtain high retrieval rates. Our simplification suits the retrieval problem, while it does not apply to recognition. Our scalable retrieval scheme avoids explicit recognition of characters. An experimental evaluation on a dataset of word images gathered from two complete books demonstrates good accuracy even in the presence of printing variations and degradations. The performance is compared with baseline methods.**

***Keywords*-Word Retrieval, BLSTM neural network, Edit distance;**

I. Introduction

Word spotting and document image retrieval have received significant attention in recent years. The success of word spotting can be attributed to the holistic matching paradigm employed for comparing a query word with the images in the database. Both the query as well as the database images are converted into a sequence of feature vectors and comparison is often carried out with the help of Dynamic Time Warping (DTW) or other matching scheme. This feature representation for the whole image avoids an explicit recognition step. Such word matching schemes have applications in accessing historic handwritten manuscripts [1], searching for relevant documents in a digital library of printed documents [2], and holistic recognition [3].

Word spotting becomes applicable to printed documents in a variety of situations. For the task of document retrieval, word spotting based methods are favored over direct character recognition. OCR systems do not perform well on degraded documents, which leads to an insufficient performance for a robust retrieval systems. This becomes even more apparent when working on Indian scripts where robust OCR systems are not available to the knowledge of the authors.

The Devanagari script for writing Hindi requires as many as 800 classes for recognition [4] and therefore a large amount of training data is required to build a robust recognition system. Many of these classes are formed by adding vowel modifiers consisting of a stroke at the top or at the bottom of a consonant. Although words can not be correctly recognized without the modifiers, we show that they can be ignored for keyword spotting based document retrieval for Hindi. Therefore, we dramatically reducing the amount of training data needed. To disambiguate words which differ only in the upper and lower vowel modifiers, a post processing step is applied.

We use BLSTM neural networks for word spotting in this paper. This category of neural networks was previously used for speech recognition [5], handwriting recognition [6], as well as keyword spotting for English handwritten text [7]. We extend these methods for wordspotting in printed Hindi documents. BLSTM neural networks are recurrent networks with hidden layers consisting of long short-term memory blocks. BLSTM neural network transform a sequence of feature vectors into a sequence of character class probabilities, and thereby building an intermediate representation which is used for word spotting. Another common technique for sequence recognition are Hidden Markov Models (HMMs). However, on a similar keyword spotting task they are outperformed by BLSTM based keyword spotting systems [7].

In this paper, we demonstrate the successful use of a BLSTM neural network for printed Hindi documents. Our focus is on retrieval from a collection of documents written in Devanagari, a script for which robust and reliable OCR system do not exist. Our method takes an intermediate path between the complete recognition, which HMMs and

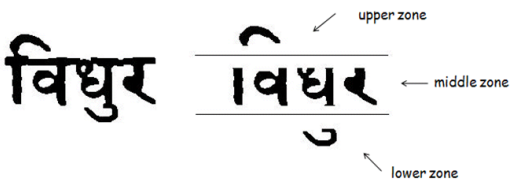

Figure 1. An example word in Devanagari and its upper, middle, and lower zone

BLSTM neural networks are capable of, and matching of feature vectors (popular in traditional word spotting literature) which does not scale to large data sets. We demonstrate our method on a couple of printed books in Hindi.

The paper is organized as follows. We first present the basic properties of the Devanagari script, (used to write the Hindi language) in Section II. We also discuss previous attempts at recognition of this language. We then describe our retrieval scheme based on BLSTM neural network in Section III. The performance of the proposed scheme is reported in Section IV.

II. Hindi and Devanagari

Devanagari is the most popular script in India. Hindi, the national language of India is written in Devanagari. It is the third most popular language in the world [8]. Hindi is written from left to right. It does not have distinct letter cases. Words are often connected with the help of a head line — *sirorekha*. The script contains basically 11 vowels and 33 consonants. Vowels can be written independently, or combined with a consonant to form conjuncts (above, below, before or after). Vowels written in this way are called modifiers or *matra*. Sometimes multiple consonants and vowels are combined to form a new shape which are termed as compound characters. The number of such compound characters can, in principle, be a few thousand. However, in practice, there are around 800 unique ligatures in this language [4]. The exact number depends on the font and the rendering scheme used.

There may not be a sufficient number of examples of each class present in the training set. To avoid incorrect retrieval we adopt the following strategy. Since, the words are characterized by a connecting *sirorekha*, one of the popular initial steps in recognizing Hindi is to remove this line. This separates the vowel modifiers which span the top-zone from the middle (main) zone. This is pictorially explained in Figure 1. Similarly the lower zone is also removed. This reduces the number of character classes to be recognized. Most of the previous work in the recognition of Hindi documents follow this approach or a variant of this for simplifying the class list [9], [10].

One of the earliest work in recognizing Hindi document is from Chaudhuri and Pal [10]. There have also been attempts in recognizing Hindi documents in recent years [9], [11]. In general, all these methods were based on an explicit segmentation of the words into characters. A major alternative to this, is the segmentation free method followed by Natarajan *et al.* [4]. They use an HMM based classification scheme for recognition of words and lines directly.

In general the number of character classes can be anywhere from 300 to 800 [4], [10]. The level of segmentation controls the number of classes to be handled. In this work, our objective is not a complete recognition of Hindi documents. We are interested in spotting words, or retrieving relevant Hindi documents corresponding to a query. We take an intermediate path between complete recognition and feature based word spotting in this work.

III. Neural Network based Word Spotting

In this paper, we perform word spotting based on a recently developed recurrent neural network, termed bidirectional long short-term memory (BLSTM) neural network [6]. We avoid a complete recognition of the Hindi documents here, due to the computational and script related challenges associated with the problem. BLSTM is a bidirectional recurrent neural network (BRNN) that has hidden layers, which are made up of the so-called long short-term memory (LSTM) cells. These networks are capable of processing a finite sequence, and predict a label sequence based on both the past and the future context of the element. BLSTM has many advantages for document recognition and retrieval [7]. It avoids explicit segmentation of words into characters, which are difficult in many situations due to the degradations present in the data set.

A major obstacle in using recurrent networks has been that error gradients vanish exponentially with the size of the time lag between important events. This is called the vanishing gradient problem. Using LSTM hidden layers, BRNN neural networks address this problem by controlling the information flow into and out of each memory blocks by nodes [6].

We represent the word image as a sequence of feature vectors of dimension d. In our implementation, we use $d = 5$ features. BLSTM neural networks contain one node for each of the features in the input layer. Thus the network has d input layer nodes. Each node in the input layer is connected with two separate hidden layers, one of which processes the input sequence of features forward, while the other processes it backward. Both hidden layers are connected to the same output layer. The output layer contains one node for each character class plus a special node ε, to indicate "no character". Thus, There are $K+1$ nodes in the output layer, where K is the number of classes, in our case $K = 81$.

For each element position in the input feature sequence, the hidden layers provide access to the past as well as future context to the output layer. The output layer sums up the values which comes from both forward and backward hidden layers. These output activations of the nodes in the output layer are normalized (such that they sum up to one) and

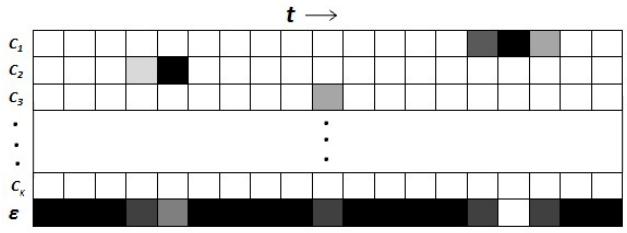

Figure 2. The sequence of output activations of the neural network.

then treated as a probability vector of the characters present at the position. All the probability vectors together form a matrix of character probabilities, as shown in Figure 2. The dark cells show high probabilities and the probability is one when it is completely black. Similarly, light cells means low probabilities and the probability is zero when the cell is completely white. If there are W samples in the input feature sequence, the matrix has a dimension of $(K+1) \times W$. Each path through this matrix which corresponds to a possible character sequence. The overall probability of the sequence is directly given by the product of all elements along its path. Note, however, that such a naive decoding will not result in a semantically meaningful character decoding.

The most likely sequence of characters is given by choosing the character with the highest probability for each step in the sequence. Afterwards, this character sequence needs to be shortened. A single character in the word image can lead to multiple subsequent activations of the same node. Also, activations of the output node corresponding to ε indicate that no decision about a character can be made at that position. To transform a path to a character sequence, all subsequent occurrences of the same character are merged and then the ε symbol is deleted. Hence, the two paths $c_1, \varepsilon, \varepsilon, c_2, c_2, \varepsilon$ and $\varepsilon, c_1, c_1, c_2, c_2, \varepsilon$ indicate the same character sequence $c_1 c_2$. For more details on BLSTM neural networks please refer to [6].

We work with word images instead characters and lines. This is primarily motivated by the fact that we are interested in retrieving documents based on queries which are primarily words. First, document images are segmented into words. Since the *sirorekha* connects the whole word and increases the number of possible shapes to be recognized, we segment the word by removing *sirorekha*. This is done with the help of a horizontal projection. We remove the top and bottom zone and work with the middle zone. This can result in lowering of the precision at the same recall. However, this reduction is not very significant. In addition, with a simple re-ranking, we overcome most of the loss in precision incurred due to this. We manipulate the ground-truth label sequence of the images to obtain the corresponding ground-truth after zoning it in the middle. After removing upper and lower zone, each image is represented as a sequence of feature vectors. This does not require any further segmentation of the word images. This makes the process robust to the typical cuts and merges seen in the document images.

Word images are represented as a sequence of feature vectors. For all our experiments, we use features extracted from vertical stripes of uniform length (sliding window). We extract five different features, (a) the lower profile, (b) the upper profile, (c) the ink-background transitions, (d) the number of black pixels, and (e) the span of the foreground pixels. The upper and lower profiles measure the distance of the top and bottom foreground pixel from the respective baselines. Ink-background transitions measures the number of transitions from Ink to background and reverse. the number of black pixels provides the information about the density of ink in the vertical stripe.

These input sequences of feature vectors are fed to the network for training together with their corresponding label sequences. Once the system is trained, it uses the neural network output probabilities to return the most likely label sequence for given test sequence. Then, all database images are stored as the corresponding output label sequence returned by the neural network For word retrieval, these output labels are compared with the query image's label sequence using edit distance and ranked based on this score.

IV. Results

In this section, we present experimental results to validate the utility of the proposed method. We also quantitatively compare the performance against baseline methods. To evaluate the performance, we have conducted two experiments. For the first experiment, we consider a complete book (we call this book as *book1*) for training and testing. For the second experiment, we train on *book1* and test on a different book (*book2*). Both these books are annotated at the word level [12]. Details of these two books are given in Table I.

In the first experiment, the document images were scanned and segmented into word images. We used 60% of the images as the training set and 20% as the validation set. The remaining 20% was used as a test set. We trained the network using the training set and used the validation set to decide the stopping criteria during the iterative training process. Each word image in test set is given to the trained network as an input. The network outputs a most probable sequence of labels for each image with the help of output probabilities. This label sequence is further used to compute the edit distance from the query words for word retrieval. To show the word retrieval performance, we selected two sets of frequently occurring keywords. The first set contains words which are already present in the training set (in-vocabulary) and the other set contains words which are not present in the training set (out-of-vocabulary). Both these sets have 100 queries each. These key words are selected such that they (i) have multiple occurrences in the data set (ii) have mostly functional words and (iii) have very few stop words. The

Table I
BOOKS USED FOR THE EXPERIMENTS.

Book	#Pages	#Lines	#Words
book1	98	2463	27764
book2	108	2590	28265

Table II
RESULTS OF BLSTM BASED METHOD ON IN-VOCABULARY AND OUT-OF-VOCABULARY QUERIES (100 EACH).

Queries	P	MAP
in-vocabulary	92.90	84.18
out-of-vocabulary	92.17	82.91

Table III
A COMPARISON WITH THE REFERENCE SYSTEMS.

Method	P	MAP
DTW	84.64	77.39
Euclidean	78.23	71.82
BLSTM based	91.73	84.77
BLSTM with reranking	93.26	89.02

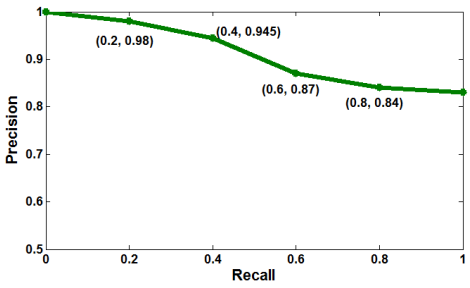

Figure 3. Precision-recall curve evaluated for 100 queries.

results of in-vocabulary and out-of-vocabulary queries are shown separately in Table II. Here P represents the precision at a recall rate of 50%. Mean Average Precision (MAP) is the mean of the area under the Precision-Recall curve for all the queries.

In the second experiment, we trained the neural network on *book1* and tested on *book2*. We selected 100 most frequently occurring function keywords from the test set as queries. For retrieval, we compute their edit distances from the rest of the test images using their label sequence representations. We then ranked the retrieved results based on their distances from the query keyword and computed the MAP. We also compute the precision at a recall rate of 50%. We obtained MAP of 84.77% in this experiment. We observed that most confusions come from dissimilar words with similar middle zones. To solve this issue, we do a re-ranking with the help of the upper zone. We re-rank the retrieved set based on the number of connected component present in the upper zone to avoid dissimilar images with the same middle zone. This scheme does not depend on the

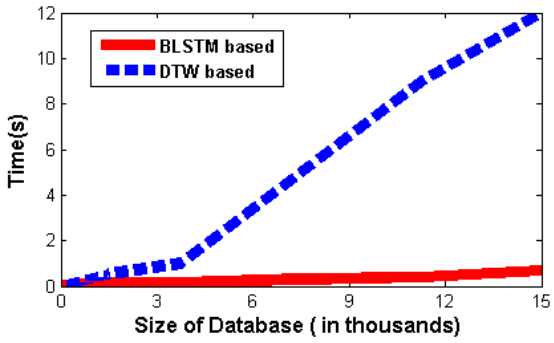

Figure 4. This graphs compares the average time takes by two different methods, BLSTM and DTW distance, per query on increasing database size.

size or font of characters in the text word. The use of simple features, which are easy to compute, make our system fast and robust. Results of our method are shown in the last two rows of Table III. The precision-recall curve corresponding to the BLSTM with re-ranking is shown in Figure 3.

Traditional word spotting/retrieval methods are also applicable for Hindi documents. We compare our results with word spotting methods with DTW and Euclidean distance. In both cases, the the feature vectors are the same. For Euclidean, all words are normalized to the same width (145 pixels in our case). We show the comparison of the neural network based method with DTW distance and Euclidean distance based approach in the Table III. It may be seen that the proposed methods are superior in retrieving relevant words.

Finally, we compare the computational complexity of the proposed retrieval scheme with the DTW based baseline. In both the cases, a dynamic programming based retrieval is carried out. However, the feature representation is different. The plot in Figure 4 compares the retrieval time for both neural network method and DTW based retrieval. It is clearly seen that the neural network based method is faster than the DTW based matching and search procedure.

We have also shown the qualitative results on five different queries in Figure 5. Variation in the retrieved results shows that the network based method is robust towards degradations and print variations. Training on *book1* and testing on *book2* shows some level of generalization across print variations. However, more work is required for demonstrating font and style independence across a larger spectrum.

V. CONCLUSIONS

In this paper, a word retrieval scheme for Hindi documents is presented. It bases on two principles. First, instead of an OCR system, a sequential approach to word recognition in followed. Words are stored as possible character sequences computed by BLSTM neural networks and the edit distance of this sequence to the query word is used. Secondly, in order

Figure 5. First column shows the query words. Their retrieved images are shown in decreasing order, from left-to-right.

to circumvent the problems that arise due to the large number of possible character classes encountered in Devanagari script, vowel modifications are removed. This reduces the number of classes drastically. Disambiguations that may arise are resolved in a post-processing step. This way, the proposed method takes an intermediate path between the complete recognition and feature based word spotting task.

On a set of experiments using printed Devanagari script, the superiority of the proposed method over a range of reference systems can be shown. The success originates from a well-balanced system. On the one hand, we adopt a sophisticated recognition technique for the retrieval task. On the other hand, by avoiding a complete recognition, we bypass the challenges posed by the script. As a future work, we would like to support text and sub-word queries. We also plan to scale the work to larger dataset.

VI. Acknowledgments

This work was supported by the Ministry of Communication and Information Technology, Government of India and by the EU project FP7-PEOPLE-2008-IAPP: 230653. R. Manmatha was supported in part by the Center for Intelligent Information Retrieval and in part by NSF grant #IIS-0910884. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor. We also thank Alex Graves for providing us the source code of the BLSTM neural network.

References

[1] T. M. Rath and R. Manmatha, "Word spotting for historical documents," *IJDAR*, vol. 9, no. 2-4, pp. 139–152, 2007.

[2] A. Balasubramanian, M. Meshesha, and C. V. Jawahar, "Retrieval from document image collections," in *Document Analysis Systems*, 2006, pp. 1–12.

[3] V. Lavrenko, T. M. Rath, and R. Manmatha, "Holistic word recognition for handwritten historical documents," in *DIAL*, 2004, pp. 278–287.

[4] P. Natarajan, E. MacRostie, and M. Decerbo, *Guide to OCR for Indic Scripts*. BBN Technologies, Cambridge, MA 02138, USA: Springer London, 2010, ch. PartI,9.

[5] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Processing*, vol. 45, pp. 2673–2681, 1997.

[6] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 5, pp. 855–868, 2009.

[7] V. Frinken, A. Fischer, H. Bunke, and R. Manmatha, "Adapting blstm neural network based keyword spotting trained on modern data to historical documents," in *ICFHR*, 2010, pp. 352–357.

[8] U. Pal and B. B. Chaudhuri, "Indian script character recognition: a survey," *Pattern Recognition*, vol. 37, no. 9, pp. 1887–1899, 2004.

[9] S. Kompalli, S. Setlur, and V. Govindaraju, "Devanagari ocr using a recognition driven segmentation framework and stochastic language models," *IJDAR*, vol. 12, no. 2, pp. 123–138, 2009.

[10] B. B. Chaudhuri and U. Pal, "An ocr system to read two indian language scripts: Bangla and devnagari (hindi)," in *ICDAR*, 1997, pp. 1011–1015.

[11] A. Bhardwaj, S. Kompalli, S. Setlur, and V. Govindaraju, "An ocr based approach for word spotting in devanagari documents," in *DRR*, 2008, p. 68150.

[12] C. V. Jawahar and A. Kumar, "Content-level annotation of large collection of printed document images," in *ICDAR*, 2007, pp. 799–803.