Fused Text Recogniser and Deep Embeddings Improve Word Recognition and Retrieval

Siddhant Bansal, Praveen Krishnan, and C.V. Jawahar

Center for Visual Information Technology, IIIT Hyderabad, India
siddhant2697@gmail.com, praveen.krishnan@research.iiit.ac.in,
jawahar@iiit.ac.in

Abstract. Recognition and retrieval of textual content from the large document collections have been a powerful use case for the document image analysis community. Often the word is the basic unit for recognition as well as retrieval. Systems that rely only on the text recogniser's (OCR) output are not robust enough in many situations, especially when the word recognition rates are poor, as in the case of historic documents or digital libraries. An alternative has been word spotting based methods that retrieve/match words based on a holistic representation of the word. In this paper, we fuse the noisy output of text recogniser with a deep embeddings representation derived out of the entire word. We use average and max fusion for improving the ranked results in the case of retrieval. We validate our methods on a collection of Hindi documents. We improve word recognition rate by 1.4% and retrieval by 11.13% in the mAP.

Keywords: Word Recognition · Word Retrieval · Deep Embeddings · Text Recogniser.

1 Introduction

Presence of large document collections like the Gutenberg Project and the Digital Library of India (DLI) [4] has provided access to many books in English and Indian languages. These and many other large document collections cover a broad range of disciplines like history, languages, art and science, thereby, providing free access to a vast amount of information. Presence of such libraries proves to be helpful for students and self-learners allowing them to get free access to the content they desire. For the creation of such libraries, the books are converted to machine-readable text by using Optical Character Recognition (OCR) solutions.

We address the issue of enabling content-level access to the document images in these libraries, which are primarily document image collections such as books. In this work, we aim at improving the word recognition and retrieval performance for the Hindi language. This is challenge since the data in these libraries consists of degraded images and printing styles that are no longer used. Current methods providing content-level access to a large corpus can be divided into two classes: (a) text recognition (OCR) and (b) word spotting. Recognition-based approaches

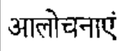
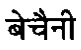
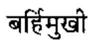
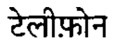
	Input Image	Baseline Prediction	Prediction using Confidence Score	Lexicon based Prediction
(a)		आलोचनाए	आलोचनाएं	आलोचनाएं
(b)		बेचनी	बेचैनी	बेचैनी
(c)		बरहिमुखी	बरहिमुखी	बर्हिमुखी
(d)		टेलीफोन	टेलीफोन	टेलीफ़ोन

Fig. 1. In this figure we show the word retrieval results. In (a) and (b), the correct word is selected using both the confidence score and lexicon based methods proposed in this work. In (c) and (d), the correct output is predicted only by the lexicon based prediction. Both the methods are successful in capturing very low level details that are missed by the traditional text recognition methods.

have showed to perform well in many situations [24,20,8]. Using this technique, scanned documents are converted to machine-readable text and then the search is carried out for the queries on the generated text. On the other hand, word spotting is a recognition-free approach for text recognition. Here, embeddings for the words in the documents are extracted and nearest neighbour search is performed to get a ranked list. Various attempts [6,7,18,23] have been made for creating systems using recognition-free approaches also. However, all of these methods fall short of utilising the full advantages of both, i.e, effectively fusing recognition-based and recognition-free methods.

We aim at exploring the complementary behaviour of recognition-based and recognition-free approaches. For that, we generate the predictions by a CRNN [24] style text recognition network, and get the deep embeddings using the End2End network [16]. Though our experimental validation is limited to printed Hindi books, we believe, the methods is generic, and could find utility in other situations also. They can be used with various types of text recogniser and word spotting techniques. Word retrieval systems using word spotting are known to have a higher recall, whereas, text recognition-based systems provide higher precision [17]. Exploring the aforementioned fact, in this work, we propose various ways of fusing text recognition and word spotting based systems for improving word recognition and retrieval. Our major contributions are:

1. We propose techniques for improving word recognition like - correct hypotheses selection using multiple hypotheses' embeddings (generated using End2End Network [16]), using multiple hypotheses' confidence scores, converting the embeddings to synthetic images and then generating the embeddings, and using lexicon for narrowing down the multiple hypotheses. Using word recognition techniques introduced in this paper, we report an average improvement in the word accuracy of 1.4%.
2. We propose techniques for improving word retrieval like - Naive Merge, Query Expansion, Average Fusion, and Max Fusion. Using word retrieval

techniques introduced in this paper, we report an average improvement in the mAP score by 11.12%.

1.1 Related works

Modern text recognition solutions are typically modelled as a Seq2Seq problem using RNNs. In such a setting, convolutional layers are used for extracting features, leading to CRNN based solutions. Transcriptions for the same are generated using different forms of recurrent networks. For example, Garain et al. [10] use BLSTMs along with CTC Loss [11] and Adak et al. [2] use CNNs as features extractors coupled with RNN for sequence classification. Pham et al. [20] also use the convolutional network as feature extractor whereas they use multi-dimensional LSTMs (MDLSTM) as the recurrent units. Chen et al. [8] use a variation of the LSTM unit, which they call SeqMDLSTM. Although these approaches help in improving text recognition but fail to achieve reliable results for old degraded documents.

Walker et al. [29] studied the consequences of less accurate text recogniser on document clustering with the help of Latent Dirichlet Allocation (LDA). A detailed study conducted by Taghva et al. [28] shows the effects of text recogniser's errors on the performance of the retrieval system built on a vector space model. Numerous attempts have been made to improve quality of the output from a word recognizer before attempting to index the text. These efforts involve post-processing the output generated by the word recogniser with the help of a lexicon. Methods such as [9,12,30] use the knowledge of topic-categorisation for reducing the document's lexicon size.

Various attempts of retrieving Indian texts using word spotting methods have been illustrated in [6,7,18,23]. The central idea behind all of these attempts can be traced back to [22] where Rath et al. used features to represent the word images and then compared them using a distance metric. The traditional word spotting technique uses dynamic programming [19], handcrafted features [5,18], and Bag of Visual Words approaches [23] for comparing the query image with all other images in the database.

Many deep learning approaches like [13,14] have been proposed in the domain of scene text recognition for improved text spotting. Poznanski et al. [21] adopted VGGNet [25] by using multiple fully connected layers, for recognising attributes of PHOC. Different CNN architectures [15,26,27,31] were proposed which uses PHOC defined embedding spaces for embedding features. On the other hand, Sudholt et al. [26] suggested an architecture which embeds image features to PHOC attributes by using sigmoid activation in the last layer. It uses the final layer to get a holistic representation of the images for word spotting and is referred to as PHOCNet.

In this work, we aim to use the advantages of text recognition and word spotting for improving word recognition and retrieval performances. Deep embeddings generated for word representation yield high recall, whereas, text recognition based approaches provide high precision. In this paper, we explore various methods for merging the results generated by both the methods and getting better-ranked lists for the retrieval. We also use the deep embeddings to choose the best hypothesis from the multiple hypotheses generated by the text recogniser for improving text recognition.

2 Baseline Methods

We first explain the two baseline methods that we are using in this work for our task.

2.1 Text Recognition

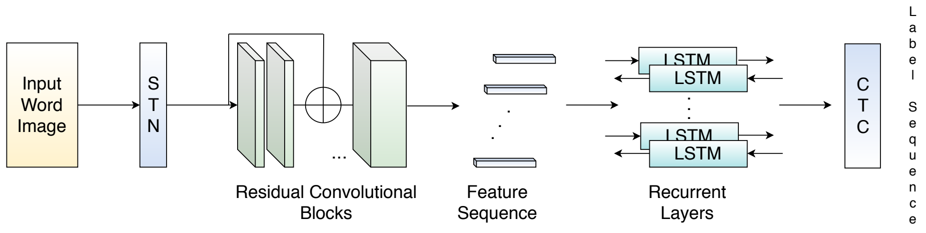


Fig. 2. CRNN Architecture[24] takes in a word image passes it through the Spatial Transform Layer (STN) layer, followed by residual convolutional blocks which learn feature maps. These feature maps are further given as an input to the BLSTM layer.

The problem of text recognition involves converting the content of an image to textual transcriptions. For this purpose, we use CNN-RNN hybrid architecture which was first proposed by [24] shown in Figure 2. Here, from the last convolutional layer we obtain a feature map $F_l \in \mathbb{R}^{\alpha\times\beta\times\gamma}$ which is passed as an input to the BLSTM layers as a sequence of γ feature vectors, each represented as $F_{l+1} \in \mathbb{R}^{\alpha\times\beta}$. CTC loss [11] is used to train the network. It takes in the input of the recurrent layers and decodes the target sequence from the probabilities calculated across all the frames.

2.2 Word Spotting

For learning the textual and image feature space we use the End2End deep network proposed in [16]. Figure 3 shows the architecture of the End2End deep network. Feature extraction and embedding are the major components of the network. The network consists of two input streams - Real Stream and Label Stream as shown in Figure 3. The real stream takes in real-word images as input and feeds it into a deep residual network which computes the features. The label stream gets divided into two different streams - the PHOC [3] feature extractor and a convolutional network. A synthetic image of the current label is given as an input to the convolutional network which in turn calculates its feature representation. This feature representation is in turn concatenated with the vectorial representation which is calculated using PHOC. Features generated from both the streams are fed to the label embedding layer which is responsible for projecting

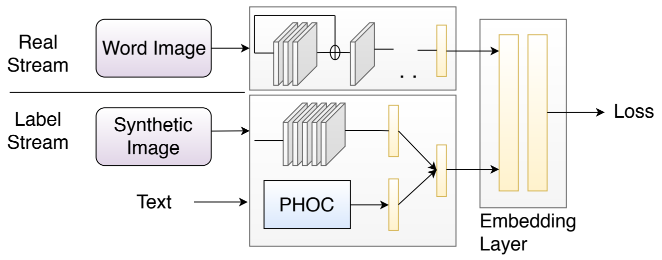


Fig. 3. End2End[16] network for learning both textual embedding using a multi-task loss function.

the embeddings in a common feature space where both the embeddings are in close proximity.

Computation of image and text features We generate the word image embeddings by passing the word image through the real stream shown in Figure 3, they are denoted by E_{wi}. The embedding for a single input word image for word recognition task is denoted by E_{Iin}.The embeddings for the input query text, text recognition's noisy text and text recognition's multiple hypotheses are generated by passing the text through the Label Stream shown in Figure 3 denoted by E_{Tin}, E_n, and E_m respectively.

3 Fusing Word Recognition and Word Spotting

We propose a system that leverages the best attributes of both the recognition-based and recognition-free approaches. For achieving this, we use books from two different internal datasets for Hindi. From the document images in these collections, word images are cropped out using word bounding box information available as part of the annotation. The word images are fed into a pre-trained text recognition network described in Section 2.1 to get the textual transcriptions. These transcriptions are used for creating a system based on recognition-based approach. The word images are also fed into the End2End network [16] described in Section 2.2 for getting word images' deep embeddings. These embeddings help in creating a system based on recognition-free approach.

3.1 Word Recognition using Multiple Hypotheses

To improve word recognition, we use the beam search decoding algorithm for generating K hypotheses from the text recognition system. Figure 4 shows a graph of word accuracy vs. K, where K is the number of hypotheses generated by the text recogniser. Word accuracies generated without any constraints show an increment as K is increased. We can further improve the word accuracy by imposing a constraint with the help of a lexicon and removing words not present in the lexicon. The lexicon, in our case, was taken from [1]. Now the challenge is

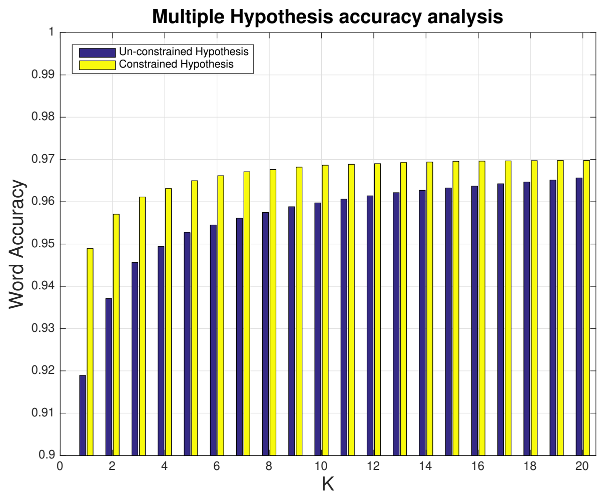


Fig. 4. Word accuracies of K hypotheses generated using beam search decoding algorithm on CRNN's output. Where K is the number of hypotheses generated by the text recogniser.

to come up with a method using which we can select the best hypothesis from these K hypotheses. For this purpose, we use deep embeddings generated by the End2End network. We pass all the hypotheses and input word image through the End2End network [16] and get deep embeddings for them denoted by E_m.

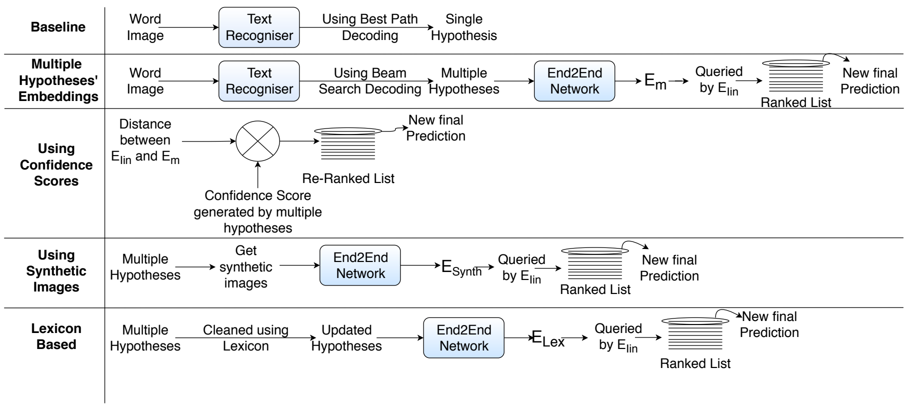


Fig. 5. Multiple Hypotheses' Embeddings are generated using beam search algorithm which are then converted to embedding. It is then queried for getting a ranked list which generates a new prediction. This method is further improved using the confidence scores. Another method uses synthetic images for getting embeddings to query. The final method uses lexicon for reducing noise in the K hypotheses and than generating the embedding for querying.

Baseline word recognition system Figure 5 shows the baseline method that generates a single output. The baseline results for constrained and unconstrained approaches can be seen at $K = 1$ in Figure 4. It can be concluded from Figure 4 that, correct word is more likely to be present when $K > 1$, and not in the first ($K = 1$) position. The task of selecting the correct prediction out of the K predictions is a challenge because of inherent ambiguity in the problem. We attempt to overcome this challenge using various methods proposed in this section.

Using multiple hypotheses' embeddings As it can be seen in Figure 5, E_{Iin} is queried on E_m to get a ranked list. The result at the top is considered to be the text recogniser's output.

Using confidence scores A confidence score is associated with E_m generated by the beam search decoding algorithm. This method uses the confidence score to select the better prediction. As seen in Figure 5 the distance between E_{Iin} and E_m is summed with the confidence score. This new summed score is used to re-rank the ranked list obtained by querying E_{Iin} on E_m. The top-1 result of the re-ranked list in considered as the final output from the text recognizer.

Using synthetic images This is a method in which we take E_m closer to E_{Iin} by converting multiple hypotheses from text to synthetic images. Here, we exploit the fact that in the same subspace, the image embeddings will lie closer to the E_{Iin} as compared to E_m. As shown in Figure 5, we generate synthetic images for the multiple hypotheses generated by the text recognition system. These synthetic images are then passed through the End2End network [16] to get the embeddings denoted by E_{synth}. E_{synth} is queried using E_{Iin} to get a ranked list which contains the final text recognition hypothesis at the top. This method performs better as compared to the one using E_m because image embeddings will be closer to the input word-image embedding even if they are in the same representation space. However, one drawback of this method is that it does not perform well if E_m is highly noisy. In this case, the synthetic images generated are highly incorrect and far from the input word image.

Lexicon based recognition Figure 4 shows that using lexicon-based constrained hypotheses, we can get much better word accuracy as compared to the hypotheses generated in an unconstrained fashion. In this method, we exploit this fact and limit the multiple hypotheses generated from the text recognition system. This approach is different from the lexicon-based decoding technique. As shown in Figure 5, here we limit the multiple hypotheses after they are generated. This helps in getting desired results with comparatively less computation requirements. We then use the limited hypotheses E_{Lex} for selecting our final hypothesis by querying the E_{Iin} and getting a ranked list of hypotheses. This method outperforms the previous methods as it constrains the number of words to choose from and hence decreases noise in the hypotheses.

3.2 Word Retrieval using Fusion and Re-Ranking

For creating a retrieval system, in this work, we get the deep embeddings for the input query text (E_{inp}) and also for the word images (E_{wi}) from the documents on which we wish to query the input text. To use the attributes of the text recognition system, we also convert the text generated by the text recognition system to deep embeddings (E_n). In this section, we propose various techniques for improving word retrieval.

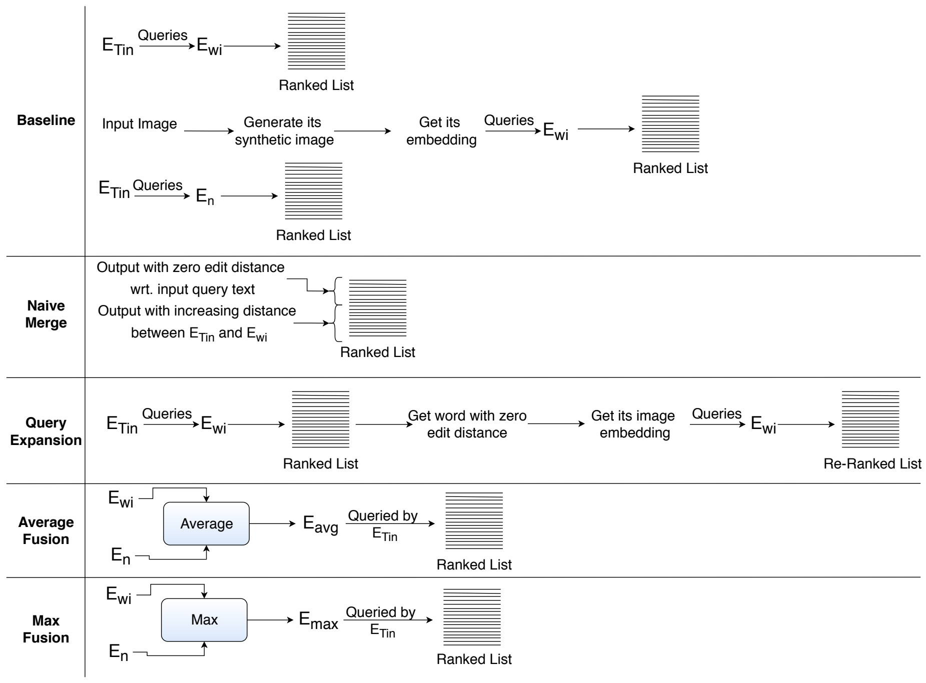


Fig. 6. Baseline for word recognition queries E_{Tin} on E_{wi} and E_n. Naive Merge attempts to exploit the best attributes of text recognition and word spotting. In Query Expansion we merge the "query by string" and "query by example" setting. Average and Max Fusion methods fuse E_{wi} and E_n by performing average and max operation respectively.

Baseline Word Retrieval System There are three major ways to perform baseline word retrieval experiment. The first method focuses on demonstrating the word retrieval capabilities of embeddings in a "query by string" setting. Shown in Figure 6, E_{Tin} is queried on E_{wi} to get a ranked list. The second method focuses on demonstrating the word retrieval capabilities of embeddings in a "query by example" setting. Here, E_{Iin} of E_{Tin} is used to query the E_{wi}, and get a ranked list. The third method focuses on demonstrating the word retrieval when done on E_n in a "query by string" setting. Here, E_{Tin} is queried on E_n

to get a ranked list. This method is a measure of how well the text recogniser is performing without the help of E_{wi}. The performance of this method is directly proportional to the performance of the text recogniser.

Another way in which we can measure the performance of E_n is by using the edit distance for ranking the word images. Edit distance measures the number of operations needed to transform one string into another. We calculate the edit distance between E_{Tin} and E_n. The ranked list is created in an increasing order of edit distance. This experiment gauges the contribution of the text recogniser in the word retrieval process.

Naive Merge Here as shown in Figure 6, we get the initial ranked list by calculating the edit distance between input text and noisy text recogniser's output. The remaining queries are arranged in the order of increasing Euclidean distance between E_{Tin} and E_{wi}. This method exploits the best attributes of both the methods - text recognition and word spotting for creating a ranked list.

Query Expansion In this method, we merge the "query by string" and "query by example" setting. The "query by example" setting works much better than the "query by string" setting, as image embedding is used to query on E_{wi} and image embeddings lie closer in the subspace. As shown in Figure 6, from the initial ranked list, we get a word with zero edit distance. We then get that word's image embedding. It is then queried on E_{wi} to get a re-ranked list. The only case in which this method can fail is, when the text recogniser generates a wrong word with respect to the input image and the generated word matches with the input query; in this particular case, we end up selecting a incorrect image embedding for re-ranking and get a wrong ranked list.

Average Fusion In this method, we merge E_{wi} and E_n. As shown in 6, we perform average of corresponding E_{wi} and E_n, which is called the averaged embedding (E_{avg}). The E_{Tin} is then queried on E_{avg} to get a ranked list.

Max Fusion In this method, we merge E_{wi} and E_n. As shown in 6, we the output having a maximum value between E_{wi} and E_n, which is called the max Embedding (E_{max}). The E_{Tin} is then queried on E_{max} to get a ranked list.

4 Experiments

In this section, we discuss the dataset details and results on the experiments described in Section 3

4.1 Dataset and evaluation metrics

We use two type of data collections for implementing and evaluating various strategies for word retrieval and recognition. In the first collection (Dataset1), the

Table 1. Dataset Details & Text recogniser Accuracy

Dataset name	Ann.	#Books	#Pages	#Words	Char%	Word%
*Dataset*1	Yes	13	1389	396087	-	-
*Dataset*2	Yes	32	4290	1265504	93.93%	86.61%

books were scanned and annotated internally. As Dataset1 was created internally, it has good quality books. Whereas, for the second collection (Dateset2) the books were randomly sampled from the DLI [4] collection. Books in Dataset2 range from different time periods, consists of variety of font size and are highly degraded. Table 1 summarises the two datasets used in this work. To get models which generalize well and are unbiased towards any particular dataset, we train our End2End network and word recogniser on Dataset1 and test the models on Dataset2. Both of these datasets contain annotated books.

We evaluate our word recognition system in terms of word accuracy which is $1 - WER$ (Word Error Rate), where WER is defined as $\frac{S+D+I}{S+D+C}$. Here S is the count of substitutions, D is the count of deletions, I is the count of insertions and C is the count of correct words. All our word retrieval methods are evaluated using the mAP score, which is defined as $mAP = \frac{\sum_{q=1}^{Q} AvgP(q)}{Q}$. Here Q is the number of queries, $AvgP(q)$ is the average precision for each query.

4.2 Word Recognition

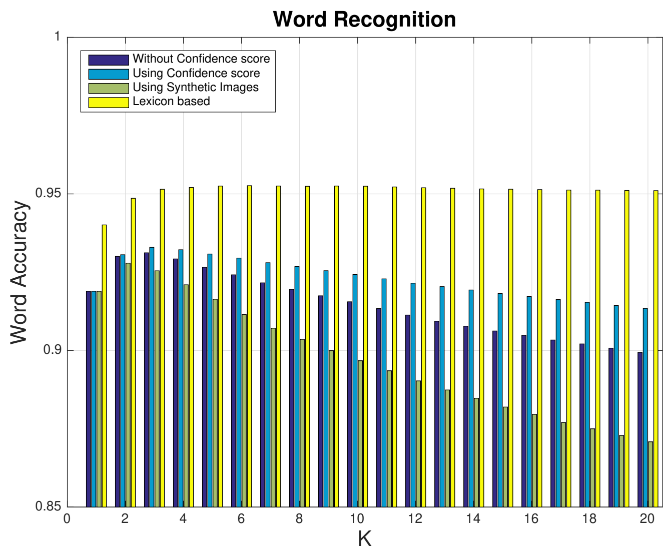


Fig. 7. Word Recognition results. Here K denotes the number of hypotheses generated by the text recogniser.

This section shows the results obtained by applying the techniques discussed in Section 3.1 for Word Recognition on Dataset2 for $1,05,475$ words. There is no pre-processing done on these words and contain a high amount of noise, like presence of punctuation marks, skewing, marks and cuts. Figure 7 shows the results of word recognition experiments. The bars in dark blue colour show the variation in the word accuracies as K is increased for the experiment in which we use the multiple hypotheses' embeddings as described in Section 3.1. It is evident as the K is increased, at first, the word accuracy increases and then, starts to decrease; the reason is as K increases the noise starts to increase and the algorithm tends to choose more wrong predictions. In this method for $K = 3$, we report the highest word accuracy that is 93.12%, which is 1.2% more than accuracy at $K = 1$.

The method where we incorporate the confidence information shows a higher gain in the accuracy score as K is increased. At $K = 3$, we report the highest word accuracy, which is 1.4% more than the accuracy at $K = 1$. The method, in which we use the synthetic images for getting the embeddings; we are able to get similar results as compared to the experiment when we use the confidence score and more when compared to the base experiment only when the input word image is clean i.e. it does not have any punctuation marks, skewing, marks and cuts. Results in Figure 7 are on words with no pre-processing due to which at $K = 3$, we report the highest word accuracy that is 0.65% more than accuracy at $K = 1$. The lexicon-based method shows the best result of all with the highest accuracy reaching to 95.26%, which is 1.25% more as compared to $K = 1$. And it can be observed that the results are more consistent in this case, and don't drop as they do for the other methods; the reason being there is a very little amount of noise in the top predictions.

4.3 Word Retrieval

This section shows the results obtained by applying the techniques discussed in Section 3.1 for word retrieval on Dataset2. All the experiments in this section are performed using $17,337$ queries on a subset of data from $Dataset2$ consisting of a total of $1,20,000$ words from 500 pages. In table 2, row 1-3 shows the baseline results for word retrieval. The mAP of the ranked list generated by calculating edit distance on text recogniser's noisy output is 82.15%, whereas mAP for query by string method on text recogniser's noisy word embeddings is 90.18%. This proves that converting E_n to deep embeddings helps in isolating more useful information. The mAP of the ranked list generated by querying E_{Tin} on E_{wi} is 92.80%; this shows that text recogniser's output is noisier as compared to E_{wi}. These results lead us to the conclusion that we can use the complementary information from the text recogniser's noisy text and word images for improving word retrieval.

Row 3-4 in table 2 shows the results obtained by re-ranking the ranked lists. The mAP of the ranked list obtained by naive re-ranking is 92.10%. Naive re-ranking shows an improvement over baselines by using the best output of the text recogniser and remaining by querying the word images' embeddings. The mAP of

Table 2. This table summarises the results for all the word retrieval experiments. It can be observed that methods using "query by example" setting work best in the baseline as well as the re-ranking case. In the case of fusion, average fusion proves to be the one showing the most improvement.

Experiment Type	Experiment	mAP
Baseline	Edit Distance on Text Recogniser's Outputs	82.15
	QbS on Text Recogniser's Embeddings	90.18
	QbS Word Image Embeddings	92.80
	QbI Word Image Embeddings	**96.52**
Re-ranking	Naive re-ranking	92.10
	Query Expansion	**93.18**
Fusion	**Average Fusion**	**93.07**
	Max. Fusion	92.79

the ranked list obtained by Query Expansion is 93.18%. Query Expansion shows improvement over all the previous methods as here we get a second-ranked list by using the text recogniser's best output's word image, it is intuitive that, if we use a word-image embedding for querying E_{wi} the mAP will improve.

Rows 5-6 in table 2 shows the results obtained by fusing W_n and W_{wi}. The ranked list obtained by W_{avg} gives us an mAP score of 93.07%, and the one obtained by W_{max} is 92.79%. The fusion methods show an improvement over the baselines as they contain information of both W_{wi} and W_n.

4.4 Failure cases

Figure 8 shows instances where both word recognition and retrieval fail to achieve the desired results.

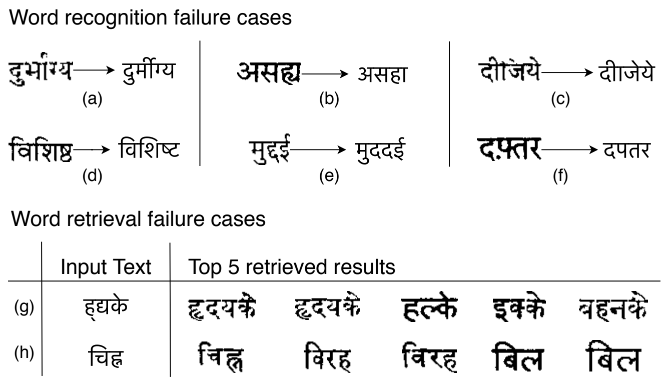


Fig. 8. Failure cases

In case (a) and (c) the image is degraded due to which we are not able to capture the image information well and end up generating the wrong prediction. In case (b), (d), (e), and (f) the characters resemble closely to other characters which leads to a wrong output, to add to this in the case of (b), (d), and (e) the characters are rare which increases the confusion. Cases (g) and (h) show the example where word retrieval was not able to perform well. For case (g) and (h), the input query has a rare character which resembles other characters, due to which the words are incorrectly retrieved. Though, for (h), using query expansion technique we were successful in retrieving one out of two instances of the query.

5 Conclusion

To summarise, we improve the word retrieval process by using the deep embeddings generated by the End2End network [16]. We have shown that by using the complementary information of text recogniser and word spotting methods. We can create word recognition and retrieval system capable of performing better than both of the individual systems. We plan to explore various other fusion techniques apart from average and max fusion for improving word retrieval.

References

1. Hindi word frequency list. bluehttp://cse.iitkgp.ac.in/resgrp/cnerg/qa/fire13translit/index.html, [Online; Last accessed 26 Dec 2019]
2. Adak, C., Chaudhuri, B.B., Blumenstein, M.: Offline cursive bengali word recognition using cnns with a recurrent model. In: 2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR). pp. 429–434 (Oct 2016). https://doi.org/10.1109/ICFHR.2016.0086
3. Almazán, J., Gordo, A., Fornés, A., Valveny, E.: Word spotting and recognition with embedded attributes. IEEE Transactions on Pattern Analysis and Machine Intelligence **36**(12), 2552–2566 (Dec 2014). https://doi.org/10.1109/TPAMI.2014.2339814
4. Ambati, V., Balakrishnan, N., Reddy, R., Pratha, L., Jawahar, C.V.: The digital library of india project: Process, policies and architecture (2006)
5. Balasubramanian, A., Meshesha, M., Jawahar, C.V.: Retrieval from document image collections. In: Bunke, H., Spitz, A.L. (eds.) Document Analysis Systems VII. pp. 1–12. Springer Berlin Heidelberg, Berlin, Heidelberg (2006)
6. Bhardwaj, A., Kompalli, S., Setlur, S., Govindaraju, V.: An ocr based approach for word spotting in devanagari documents. vol. 6815, p. 68150 (01 2008). https://doi.org/10.1117/12.767289
7. Chaudhury, S., Sethi, G., Vyas, A., Harit, G.: Devising interactive access techniques for indian language document images. Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings. pp. 885–889 (2003)
8. Chen, Z., Wu, Y., Yin, F., Liu, C.: Simultaneous script identification and handwriting recognition via multi-task learning of recurrent neural networks. In: 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR). vol. 01, pp. 525–530 (Nov 2017). https://doi.org/10.1109/ICDAR.2017.92

9. Farooq, F., Bhardwaj, A., Govindaraju, V.: Using topic models for ocr correction. IJDAR **12**, 153–164 (09 2009). https://doi.org/10.1007/s10032-009-0095-7
10. Garain, U., Mioulet, L., Chaudhuri, B.B., Chatelain, C., Paquet, T.: Unconstrained bengali handwriting recognition with recurrent models. In: 2015 13th International Conference on Document Analysis and Recognition (ICDAR). pp. 1056–1060 (Aug 2015). https://doi.org/10.1109/ICDAR.2015.7333923
11. Graves, A., Fernández, S., Gomez, F., Schmidhuber, J.: Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In: Proceedings of the 23rd International Conference on Machine Learning. p. 369–376. ICML '06, Association for Computing Machinery, New York, NY, USA (2006). https://doi.org/10.1145/1143844.1143891, bluehttps://doi.org/10.1145/1143844.1143891
12. Hassan, E., Garg, V., Haque, S., Chaudhury, S., Gopal, M.: Searching ocr'ed text: An lda based approach. pp. 1210–1214 (09 2011). https://doi.org/10.1109/ICDAR.2011.244
13. Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A.: Synthetic data and artificial neural networks for natural scene text recognition. CoRR **abs/1406.2227** (2014), bluehttp://arxiv.org/abs/1406.2227
14. Jaderberg, M., Vedaldi, A., Zisserman, A.: Deep features for text spotting. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) Computer Vision – ECCV 2014. pp. 512–528. Springer International Publishing, Cham (2014)
15. Krishnan, P., Dutta, K., Jawahar, C.V.: Deep feature embedding for accurate recognition and retrieval of handwritten text. In: 2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR). pp. 289–294 (Oct 2016). https://doi.org/10.1109/ICFHR.2016.0062
16. Krishnan, P., Dutta, K., Jawahar, C.V.: Word spotting and recognition using deep embedding. In: 2018 13th IAPR International Workshop on Document Analysis Systems (DAS). pp. 1–6 (April 2018). https://doi.org/10.1109/DAS.2018.70
17. Krishnan, P., Shekhar, R., Jawahar, C.: Content level access to digital library of india pages (12 2012). https://doi.org/10.1145/2425333.2425338
18. Meshesha, M., Jawahar, C.V.: Matching word images for content-based retrieval from printed document images. International Journal of Document Analysis and Recognition (IJDAR) **11**(1), 29–38 (Oct 2008). https://doi.org/10.1007/s10032-008-0067-3, bluehttps://doi.org/10.1007/s10032-008-0067-3
19. Myers, C., Rabiner, L., Rosenberg, A.: Performance tradeoffs in dynamic time warping algorithms for isolated word recognition. IEEE Transactions on Acoustics, Speech, and Signal Processing **28**(6), 623–635 (December 1980). https://doi.org/10.1109/TASSP.1980.1163491
20. Pham, V., Bluche, T., Kermorvant, C., Louradour, J.: Dropout improves recurrent neural networks for handwriting recognition. In: 2014 14th International Conference on Frontiers in Handwriting Recognition. pp. 285–290 (Sep 2014). https://doi.org/10.1109/ICFHR.2014.55
21. Poznanski, A., Wolf, L.: Cnn-n-gram for handwritingword recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2305–2314 (June 2016). https://doi.org/10.1109/CVPR.2016.253
22. Rath, T., Manmatha, R.: Word spotting for historical documents. International Journal of Document Analysis and Recognition (IJDAR) **9**, 139–152 (04 2007). https://doi.org/10.1007/s10032-006-0027-8
23. Shekhar, R., Jawahar, C.V.: Word image retrieval using bag of visual words. In: 2012 10th IAPR International Workshop on Document Analysis Systems. pp. 297–301 (March 2012). https://doi.org/10.1109/DAS.2012.96

24. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. CoRR **abs/1507.05717** (2015), bluehttp://arxiv.org/abs/1507.05717
25. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv 1409.1556 (09 2014)
26. Sudholt, S., Fink, G.A.: Phocnet: A deep convolutional neural network for word spotting in handwritten documents. CoRR **abs/1604.00187** (2016), bluehttp://arxiv.org/abs/1604.00187
27. Sudholt, S., Fink, G.A.: Attribute cnns for word spotting in handwritten documents. CoRR **abs/1712.07487** (2017), bluehttp://arxiv.org/abs/1712.07487
28. Taghva, K., Borsack, J., Condit, A.: Effects of ocr errors on ranking and feedback using the vector space model. Information Processing Management **32**(3), 317 – 327 (1996). https://doi.org/https://doi.org/10.1016/0306-4573(95)00058-5, bluehttp://www.sciencedirect.com/science/article/pii/0306457395000585
29. Walker, D., Lund, W.B., Ringger, E.K.: Evaluating models of latent document semantics in the presence of OCR errors. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing. pp. 240–250. Association for Computational Linguistics, Cambridge, MA (Oct 2010), bluehttps://www.aclweb.org/anthology/D10-1024
30. Wick, M., Ross, M., Learned-Miller, E.: Context-sensitive error correction: Using topic models to improve ocr. In: Ninth International Conference on Document Analysis and Recognition (ICDAR 2007). vol. 2, pp. 1168–1172 (Sep 2007). https://doi.org/10.1109/ICDAR.2007.4377099
31. Wilkinson, T., Brun, A.: Semantic and verbatim word spotting using deep neural networks. In: 2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR). pp. 307–312 (Oct 2016). https://doi.org/10.1109/ICFHR.2016.0065