Robust Recognition of Degraded Documents Using Character N-Grams

Shrey Dutta, Naveen Sankaran, Pramod Sankar K.*, C.V. Jawahar
Center for Visual Information Technology, IIIT-Hyderabad, INDIA
* *Xerox Research, Webster, USA*
Email: jawahar@iiit.ac.in

***Abstract*—In this paper we present a novel recognition approach that results in a 15% decrease in word error rate on heavily degraded Indian language document images. OCRs have considerably good performance on good quality documents, but fail easily in presence of degradations. Also, classical OCR approaches perform poorly over complex scripts such as those for Indian languages. We address these issues by proposing to *recognize character n-gram images*, which are basically groupings of consecutive character/component segments. Our approach is unique, since we use the character n-grams as a primitive for recognition rather than for post-processing. By exploiting the additional context present in the character n-gram images, we enable better disambiguation between confusing characters in the recognition phase. The labels obtained from recognizing the constituent n-grams are then fused to obtain a label for the word that emitted them. Our method is inherently robust to degradations such as cuts and merges which are common in digital libraries of scanned documents. We also present a reliable and scalable scheme for recognizing character n-gram images. Tests on English and Malayalam document images show considerable improvement in recognition in the case of heavily degraded documents.**

***Keywords*-OCR, Character N-Grams, Degraded Documents**

I. Introduction

Optical Character Recognition (OCR) technology has seen significant progress resulting in many OCR systems such as ABBYY, Tesseract, etc. However, OCRs are known to be sensitive to the quality of the document images, with significant errors observed for even moderately degraded documents. Degradations such as cuts occur in documents due to reasons such as aging of the paper, erosion of ink, poor typesetting, blotting of ink, low resolution/bandwidth in the case of faxed documents, etc. Merges are typically found in documents of non-Latin scripts, which have a complex layout. This is due to the poor spacing between characters in typical word-processors which were initially designed for the simpler script layout of English.

Degradations have long challenged OCR systems. Character recognition becomes hard in cases where degradations can modify the appearance of a character to another similar looking character. For example in Figure 1 the "E" in "English" appears like an "F" due to a cut. The effects of degradations are more pronounced in complex scripts of Indian [1] and Arabic origin [2]. This is due to the presence of a number of similar looking characters, where a small dot or a stroke of a few pixels could alter the character, and thus the meaning of the word. The most popular approach to handle poor recognition on degraded documents, was to use strong post-processing modules such as character error models [3], dictionaries [4], statistical language models [5], or a combination [6]. However, post-processing modules are not easy to construct for Indian languages due to large vocabulary size [7]. Our approach does not require a post-processing step involving a statistical language model.

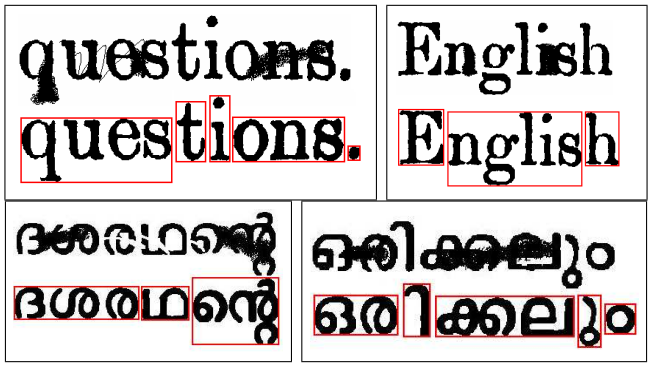

Figure 1. Examples of words where our algorithm correctly recognizes despite degradations. Popular OCRs have failed to recognize these images. We propose a novel n-gram based recognition scheme that addresses challenges in character recognition. In each example, the top word is the test image and the bottom word is the concatenation of the matched n-grams, outlined in red.

Many of the challenges in character recognition could be addressed using a word-recognition approach. Holistic word recognition is popular in the handwriting community [8], where character segmentation is hard due to the cursive nature, but word segmentation is quite reliable. Words present more information than characters, enabling easier disambiguation and thus better recognition performance. However, most word-recognition schemes such as Collection OCR [9], word-shape matching [10], etc. learn classifiers for the vocabulary found in the training set. Since it is impossible to provide all possible words in the training phase, such schemes cannot handle out-of-vocabulary (OOV) words. Recognition based on HMMs [5] can work for OOV words [11], but are not popular due to many practical issues in training. Often discriminative approaches are favored.

In this paper, we shall address the major issue of robust recognition in the presence of heavy degradations by propos-

ing to recognize character n-gram images. Sequences of n character/component segments or character n-grams within a given word image are recognized separately. The label of the given word-image is inferred from the recognition of its constituent character n-gram images. Unlike previous approaches that used character n-grams either for post-processing [12], [13] or in retrieval [14], [15], we use the character n-gram as a primitive for recognition. This approach can potentially recognize an unlimited vocabulary, while a number of advantages of post-processing are realized in the recognition phase itself.

We shall show that in the presence of cuts and merges, the n-gram is a more reliable entity for recognition than either characters or words. Our approach results in a 19% improvement in character error rate (CER) on degraded Malayalam documents, as compared to a state-of-the-art OCR [16]. The techniques presented are independent of the language of the document image and directly applicable to a new script. To demonstrate this, we also show considerable performance on English language documents. The major contributions of our work are:

- A novel re-posing of the OCR problem to one of recognizing character n-grams.
- An efficient and accurate n-gram recognition scheme.
- An optimal fusion technique to obtain word labels.
- An improvement in OCR word-accuracy of more than 15% on a challenging Malayalam dataset in comparison with [16].

II. Recognition Through Character N-Grams

Character n-grams combine the advantages of both characters and words. Through the presence of multiple characters, n-grams have more context than characters. The joint appearance of multiple characters in an n-gram is more distinctive than each character in isolation, which allows for better character disambiguation. Yet, unlike words which are almost unlimited, the number of unique n-grams is finite for a given alphabet and n. This makes it feasible to model all the n-grams, which is not possible at the word-level.

From a clean word-image of k characters, one would find k unigrams, k-1 bigrams, k-2 trigrams, ..., and one k-gram. In the presence of degradations, connected-components (CC) are used instead of characters to build the n-grams. Owing to the inclusion of unigrams and k-grams, this approach unifies both character and word recognition approaches into a single framework. Further, character and word recognition outputs are augmented with recognition of n-grams, which would help in improving recognition performance. All the n-grams are used in a unified recognition framework, which does not bias an n-gram based on its size. Thus, accurate segmentation of a word into the n-grams is not a necessity, allowing the framework to be inherently robust to degradations such as cuts and merges. We shall experimentally demonstrate this hypothesis in Section IV.

Each word-image of k characters emits $k \cdot (k+1)/2$ n-grams. In the training phase, this makes it easier to obtain considerable amount of labeled n-gram exemplars from a small collection of labeled word-images. In the classification phase, n-grams generated from test word-images are recognized using n-gram models learned during training. The different n-grams extracted from an example word image, are shown in Figure 2. The individual n-gram recognitions are merged together to obtain the most suitable label for the word. One of the major advantages of our approach is that all n-grams need not be correctly recognized; even if half the n-grams are erroneously recognized, it is still possible to obtain an accurate word label. Moreover, the scheme implicitly performs a validity check of an n-gram, by always finding the closest valid n-gram seen during training. For example, in the word "Illinois", the first quad gram is always recognized as "Illi" instead of a visually similar "liil", which is most likely unseen in training data.

However, n-gram recognition has these challenges:

1) Building a recognition scheme for n-grams involves classifying against 100K n-gram classes. Classification at such large class sizes is a non-trivial problem.
2) Recognizing n-grams in the test-phase is expensive, as the number of features to classify is multiplied by a factor of $(k+1)/2$ (for a k length word).
3) The label of the word-image needs to be inferred by aggregating the recognition of individual n-grams.

We shall present techniques to address these challenges in the following Sections.

A. N-Gram Recognition Process

In the indexing phase, the design of the n-gram recognition consists of identifying the features and the classifier for the task. In the presence of degradations and multiple fonts, it was observed in [9] that profile-features [17] outperform SIFT and HoG based features. The features extracted for each n-gram image consist of the upper, lower, transition and projection profiles, similar to those in [17]. The issue with profile features is that they are extracted for each column of the n-gram image, making the feature vector dependent on the width of the image. In order to ensure that all the features are of a constant size, all n-gram images are scaled to a canonical size before profile-features are extracted.

In the presence of thousands of classes to recognize, the classifier of choice needs to be very robust. This means that classifiers should be able to learn from a small set of exemplars per class and also be easy to train. A Nearest Neighbor (NN) classifier would require no training and is highly scalable with the class sizes. A NN classifier was shown to work better than SVMs [9] for a task of classifying 33K words of 1000 classes. Further, the context present in the n-gram is sufficiently distinguishing for many pairs of characters, thereby obviating the need for strong classifiers

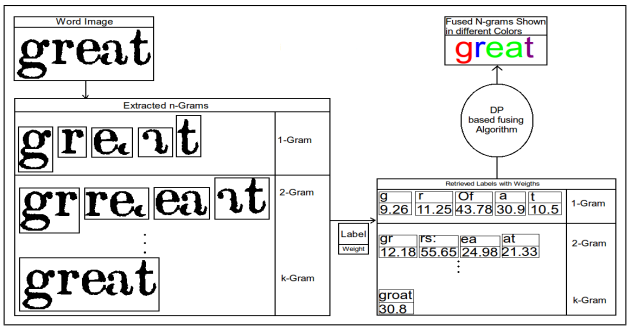

Figure 2. A depiction of word recognition using n-grams. The given word is segmented to its constituent n-grams, each of which is recognized independently. The labels obtained for each n-gram are then fused using dynamic programming to obtain the optimal sequence of n-gram labels.

(and in some cases strong features). We use scaled profile-features with a NN classifier for the n-gram recognition.

The challenge during the test phase is that the test dataset size is increased by an order of magnitude, since each test case generates multiple n-grams. The computational cost of NN classifiers can be significantly reduced by using an Approximate Nearest Neighbor (ANN) search. In ANN, the labeled exemplars are indexed using Hierarchical K-Means and KD-Trees [18]. Due to the indexing, the test point need not be compared against all the exemplars in the labeled data. By looking up a given test n-gram in the built index, one could identify the ANN in about 10 milli-seconds, while a regular NN would take about 5 seconds (about 500$\times$ speedup). The ANN process results in significant speedup that makes recognition of n-grams feasible even on a digital library scale. The obtained ANN is used to identify the label of the given n-gram.

III. Fusing N-Gram Recognition for Word Recognition

Each n-gram provides certain evidence for what the word's label should be. In the situation where every n-gram is correctly recognized, the n-gram labels reinforce the evidence from one another. For example, given the word "most", if the first trigram is correctly recognized as "mos" and the second trigram as "ost", the overlapping "os" implies that the word is very likely to be "most". However, if one of the trigrams is erroneously recognized, the inference is not immediate, thereby requiring to include evidence from the bigrams and unigrams, etc. In this paper, we use an OR scheme where it suffices to correctly recognize only a small subset of all the n-grams.

Given the recognition of each i^{th} n-gram $w_{n,i}$, let the corresponding confidence of recognition be $c_{n,i}$. The objective is to identify the sequence of n-grams that would result in the most confident prediction (with the measure $C_{n,i}$) for the entire word. The objective function is defined as:

$$\begin{aligned} C_{n,i} &= c_{n,i} && \text{, if } n = 1 \\ &= min\{c_{n,i}, W_{n,i}\} && \text{, otherwise} \end{aligned}$$

Where,

$$W_{n,i} = min_{m \in [1,n-1]} \{ \frac{(n-m) \cdot C_{n-m,i} + m \cdot C_{m,n+i-m}}{n} \}$$

When the algorithm is initiated, it has the choice of choosing either the label for the whole word ($c_{n,i}$), or the best combination for the n-grams within the word ($W_{n,i}$). The $W_{n,i}$ is defined to identify the optimal n-gram division of the given word. Each of the n-gram images is recursively recognized using the same definition. If the n-gram in consideration is a unigram, it cannot be further divided, hence the confidence of the unigram label is used as specified in the first condition. The second condition finds the minimum cost between the label for the n^{th} gram and combinations of smaller grams that makes that n-gram.

This objective function lends itself to be optimally solved using dynamic programming (DP). Each entry in the DP array stores the cumulative confidence of the n-grams that contribute to the word label. The backtracked path of the DP array is the most confident sequence of n-grams. The word label is obtained by simply concatenating these n-grams. This process is shown in Figure 2. The n-grams formed by the CCs are individually recognized and a label is obtained from the closest match. The word label is obtained as a concatenation of the most confident n-gram recognition sequence. For the given example, the word is recognized as the sequence of n-grams forming *g-r-ea-t*.

A. Analysis of the Algorithm

A summary of the n-gram based recognition process is provided in Algorithm 1.

Algorithm 1 N-Gram based Word Recognition Framework

Training Phase
for all Word-images in *Training* data **do**
  Segment words to connected-components.
  Obtain character n-gram images and extract features.
end for
Build index over all features extracted over *Training* data.
Test Phase
for all Word-images in *Test* data **do**
  Segment words to connected-components, extract features for character n-grams.
  Recognize n-grams against index built on *Training* data. Obtain confidence of each recognition.
  Apply Dynamic Programming on the confidence scores and the n-gram size to fuse n-gram recognitions. Obtain the most likely word label.
end for

Recognizing an Unseen Word: Consider the case where the word "modulation" is OOV. A holistic word-recognition would fail to obtain a label for such a word-image. In our

	Character Error Rate(CER)				Word Error Rate (WER)			
Malayalam	**MOCR**	**Char-Rec (Uni).**	**Word-rec(k)**	**N-Gram Rec.**	**MOCR**	**Char-Rec(Uni).**	**Word-Rec(k)**	**N-Gram Rec.**
Good	4.43	4.98	50.07	4.27	20.86	24.35	64.48	20.26
Bad	19.62	12.87	59.02	7.07	55.23	41.8	73.21	29.27
Ugly	37.64	36.03	67.5	18.12	62.94	64.76	84.65	47.73
English	**Tesseract**	**Char-Rec (Uni).**	**Word-rec(k)**	**N-Gram Rec.**	**Tesseract**	**Char-Rec(Uni).**	**Word-rec(k)**	**N-Gram Rec.**
Good	0.64	2.36	17.48	2.0	3.24	8.15	19.98	7.08
Bad	3.7	25.27	20.72	7.9	10.66	49.47	24.74	21.21

Table I
CHARACTER AND WORD ERROR RATES FOR MALAYALAM AND ENGLISH DATASETS. N-GRAM BASED RECOGNITION CONSISTENTLY OUTPERFORMS THE CHARACTER AND WORD RECOGNITION BASELINES. BY USING N-GRAMS, WE ARE ABLE TO IMPROVE CHARACTER RECOGNITION BY 19%.

	# Words	# N-Grams	# Cuts	# Merges
Good	81K	3M	5388 (6%)	783 (0.01%)
Bad	141K	6.7M	55,149 (39%)	16781 (12%)
Ugly	23K	0.5M	1575 (6%)	16,923 (74%)

Table II
DETAILS OF THE TEST DATASETS FOR THE MALAYALAM COLLECTION.

approach, suppose that exemplars are present for the words "module" and "integration", the constituent n-grams from these exemplars would be indexed. During the test phase, the n-grams "modul" and "ation" in the test word are correctly recognized against the corresponding n-gram exemplars. When the recognition result is fused, our algorithm would generate the correct label for the *unseen* word.

Effect of Cuts & Merges: In the case of cuts, a character would be split to two components. A standard OCR would treat them as separate characters and classify them separately, deferring error-correction to the post-processing step. In our algorithm, the split components form a valid bigram and is thus correctly classified in spite of a cut. Similarly, in the case of merges multiple characters in the test word would be combined to appear like one CC. In such cases, a unigram from test data would match its corresponding n-gram from training data, resulting in a correct recognition. We have effectively negated the effect of cuts and merges by using a single indexing scheme over all n-grams, such that it ignores the number of components and only focuses on how they appear together. Hence, the approach is quite robust to degradations such as cuts and merges.

Limitations of Approach: One of the limitations of our approach is that it assumes word segmentation to be provided as input. This assumption might be tough to satisfy for Arabic documents, in which cases HMM solutions can perform well. We could address this limitation easily by recognizing at the line-level instead of word-level. Another limitation of the approach is that the ANN classifier scheme is memory intensive due to the need for a large set of labeled exemplars and the inherent indexing structure. However, this is an acceptable design for large-scale digital libraries that are usually well-equipped with computing nodes or have access to cloud-computing.

IV. EXPERIMENTAL SETUP & RESULTS

The data for our experiments is obtained from multiple sources such as scanned books and newspapers for the Indian language of Malayalam. Groundtruth was obtained by manual typesetting. The *Training* dataset consists of around 100K words. The *Test* dataset is divided into three groups based on their degradations: Good, Bad and Ugly based on increasing percentage of cuts or merges. The details of the number and type of degradations in the test datasets are given in Table II. The Bad dataset consists of more cuts while the Ugly dataset has lot more merges. Bad dataset comes mostly from old books while ugly comes from newspapers. We also build an English dataset from 140K words, which is divided equally for training and testing.

There are two baselines that we compare our approach against: i) a character classification scheme and ii) a word recognition scheme. Character recognition is measured using only single character unigrams and word recognition is measured using only the $k$grams. To ensure that there is no bias in terms of features and classifiers, we use the same features and classifier as used for n-gram recognition (namely scaled profile-features and NN classification). The recognition accuracy is measured at both character and word levels respectively by Character Error Rate (CER) and Word Error Rate (WER). We also compare our approach with a state-of-the-art OCR for both languages.

The quantitative results are provided in Table I. On comparing columns 3 and 5 as well as 7 and 9, we can see that the proposed ngram recognition is consistently superior to character (unigram) recognition, for both Malayalam and English. We can see that the word-recognition (k-gram) performs poorly over all datasets, since words that are unseen in training do not receive valid labels in the test-phase. The higher word recognition in English compared to Malayalam can be attributed to a smaller vocabulary of the language. It is clear that the performance of all the recognition methods degrade in performance with poor quality data. However, the loss of accuracy is much less pronounced in our method. This makes our method highly suited for recognizing degraded words. We obtain an improvement of 17% in WER for the ugly data set.

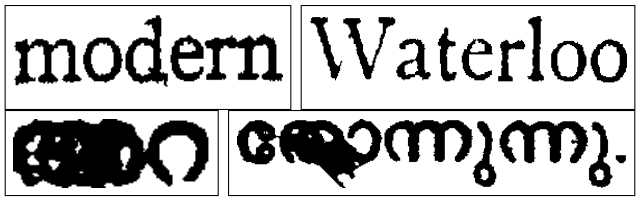

Figure 3. Examples of words where our algorithm fails to correctly recognize. In the English words (top row) the degradations result in labels "modem" and "\Vaterloo" respectively. For the given Malayalam words, the excessive degradations result in unrecognizable characters and thus errors. State of the art OCRs have also failed to recognize these words.

We also compare our results with OCRs for both Malayalam and English. Malayalam OCR (MOCR) does not use any language model and therefore, the recognition rates are comparable to the character (unigram) recognition. We obtain significant improvement of 19% in CER and 15% in WER improvements in the ugly dataset. Comparing with Tesseract, we do not observe any improvement in performance using n-gram recognition. This can be attributed to the better features, more discriminative classifiers and a strong post-processor used in Tesseract. Also, Tesseract has the advantage of better design and engineering due to contributions from the open-source community. Our ngram based recognition is better suited for highly degraded words, and thereby we complement the present day OCRs.

A. Error Analysis

A few erroneous examples are shown in Figure 3. The errors in the examples are due to heavy degradations, resulting in either unrecognizable characters or in visually similar but erroneous characters. For example, due to a merge of "rn" in "modern", the word was recognized as "modem", while the "W" in "Waterloo" was mis-recognized as "\V" due to a cut. Also, we observed that some of the errors in our recognition scheme are due to erroneous, yet confident recognition of n-grams. For example, the letter "c" is sometimes confused with "e" with very similar confidence values. In the absence of more confident n-grams, this error is retained in the final word. Similarly, some errors are found in similar looking n-grams that cannot be disambiguated, such as "lil" (in *lily*) and "ill" (in *pill*). This could possibly be addressed in the future by using stronger features for matching, or by using multiple labels for each n-grams from which the most appropriate is picked in the fusing scheme.

V. Conclusions and Future Work

In this paper, we propose a new n-gram recognition based scheme to convert document images to text. Our approach addresses challenges such as degradations and confusing characters, where classical OCRs fail. Our results show significant improvements in recognition performance over challenging datasets. In future work, we shall improve the classification performance of individual n-grams. We shall explore the possibility of obtaining multiple labels for each n-gram which can be fused to obtain a ranked list of labels for the word. One could also look at applying similar techniques to other scripts and to handwritten documents. Another direction of this work would be to reduce the memory and time requirements by building classifiers and indexing schemes with a smaller footprint.

VI. Acknowledgements

This work was supported in part by the Ministry of Communications and Information Technology, Govt. of India.

References

[1] S. Kompalli, S. Nayak, S. Setlur, and V. Govindaraju, "Challenges in OCR of Devanagari documents," in *Proc. ICDAR*, 2005, pp. 327–331.

[2] M. S. Khorsheed, "Off-Line Arabic Character Recognition - A Review," *Pattern Analysis and Applications*, vol. 5, pp. 31–45, 2002.

[3] S. Kahan, T. Pavlidis, and H. S. Baird, "On the recognition of printed character of any font and size," *IEEE PAMI*, vol. 9, no. 2, pp. 274–288, 1987.

[4] G. S. Lehal, C. Singh, and R. Lehal, "A shape based post processor for gurumukhi OCR," in *Proc. ICDAR*, 2001, pp. 1105–1109.

[5] P. Natarajan, E. MacRostie, and M. Decerbo, "The BBN Byblos Hindi OCR System," *Guide to OCR for Indic Scripts*, pp. 173–180, 2009.

[6] K. Taghva and E. Stofsky, "OCRSpell: an interactive spelling correction system for ocr errors in text," *IJDAR*, vol. 3, p. 2001, 2001.

[7] A. Bharati, P. Rao, R. Sangal, and S. M. Bendre, "Basic statistical analaysis of corpus and cross comparision," in *Proc. ICON*, 2002.

[8] S. Srihari, H. Srinivasan, P. Babu, and C. Bhole, "Handwritten arabic word spotting using the cedarabic document analysis system," in *Proc. Symposium on Document Image Understanding Technology (SDIUT-05)*, 2005, pp. 123–132.

[9] Pramod Sankar K., C. V. Jawahar and R. Manmatha, "Nearest Neighbor based Collection OCR," in *Proc. DAS*, 2010.

[10] T. Adamek, N. E. O'Connor, and A. F. Smeaton, "Word matching using single closed contours for indexing handwritten historical documents," *IJDAR*, vol. 9, no. 2-4, pp. 153–165, 2007.

[11] I. Bazzi, R. M. Schwartz, and J. Makhoul, "An Omnifont Open-Vocabulary OCR System for English and Arabic," *IEEE PAMI*, vol. 21, no. 6, pp. 495–504, 1999.

[12] A. Brakensiek, D. Willett, and G. Rigoll, "Improved degraded document recognition with hybrid modeling techniques and character n-grams," in *Proc. ICPR*, 2000, pp. 438–441.

[13] Y. Fataicha, M. Cheriet, Y. Nie, and Y. Suen, "Retrieving poorly degraded ocr documents," *IJDAR*, vol. 8, 2006.

[14] S. Harding, W. B. Croft, and C. Weir, "Probabilistic retrieval of OCR degraded text using n-grams," in *Proc. ECDL*, 1997, pp. 345–359.

[15] Sudha Praveen M., Pramod Sankar K. and C. V. Jawahar, "Character n-gram spotting in document images," in *Proc. ICDAR*, 2011, pp. 941–945.

[16] D. Arya, T. Patnaik, S. Chaudhury, C. V. Jawahar, B. B. Chaudhury, A. G. Ramakrishnan, G. S. Lehal, and C. Bhagavati, "Experiences of Integration and Performance Testing of Multilingual OCR for Printed Indian Scripts," in *ICDAR MOCR Workshop*, 2011.

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

[18] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *Proc. VISAPP*, 2009, pp. 331–340.