A Framework for Document Specific Error Detection and Corrections in Indic OCR

Rohit Saluja
IITB-Monash Research Academy
Mumbai, India
rohitsaluja@cse.iitb.ac.in

Devaraj Adiga
IIT Bombay
Mumbai, India
pdadiga@iitb.ac.in

Ganesh Ramakrishnan
IIT Bombay
Mumbai, India
ganesh@cse.iitb.ac.in

Parag Chaudhuri
IIT Bombay
Mumbai, India
paragc@cse.iitb.ac.in

Mark Carman
Monash University
Victoria, Australia
mark.carman@monash.edu

***Abstract*—In this paper, we present a framework for assisting word-level corrections in Indic OCR documents by incorporating the ability to identify, segment and combine partially correct word forms. The partially correct word forms themselves may be obtained from corrected parts of the document itself and auxiliary sources such as dictionaries and common OCR character confusions. Our framework updates a domain dictionary and learns OCR specific n-gram confusions from the human feedback on the fly. The framework can also leverage consensus between outputs of multiple OCR systems on the same text as an auxiliary source for dynamic dictionary building. Experimental evaluations confirm that for highly inflectional Indian languages, matching partially correct word forms an result in significant reduction in the amount of manual input required for correction. Furthermore, significant gains are observed when the consolidated output of multiple OCR systems is employed as an auxiliary source of information. We have corrected over 1100 pages (13 books) in Sanskrit, 190 pages (1 book) in Marathi, 50 pages (part of a book) in Hindi and 1000 pages (12 books) in English using our framework. We present a book-wise analysis of improvement in required human interaction for these Languages.**

I. Introduction

Error detection in Optical Character Recognition (OCR) extracted text for of highly inflectional languages from India faces challenges such as large unique-words list, lack of linguistic resources, lack of reliable language models, etc. [1]. The technique of morphological parsing has been applied for character-level error detection and correction in Bangla text [2], and Recurrent Neural Networks have been recently used along with Gaussian Mixture Models to detect the erroneous text in Hindi, Gujarati, Malayalam, and Telugu [3]. A Multi-engine Environment has proven effective in solving the problem of OCR text correction in English [4].

The conventional spell-checkers make use of proximity-based matches, especially Levenshtein-Damerau edit distance to words from a known vocabulary (possibly gathered from the Web), followed by a language model for auto-corrections [5], [6], [7]. The various difficulties involved in developing a high-performing Spellchecker for Hindi, Bengali and English are discussed in Choudhury et al. [8]. Here, an example is given that “fun” being misspelled as “gun” is a real-word error (RWE) and “fun” being misspelled as “vun” is a non-word error (NWE). An observation is made by Choudhury et al. [8], that “hardness of NWE correction is highest for Hindi, followed by Bengali and English”. Intuitively, the larger the number of basic word forms that exist in a language, the more candidates there are for replacing each erroneous word and the harder it is to build a functioning spell-checker. We try to solve the problem of OCR text verification and erroneous word replacement by partially matching the correct word forms obtained from the document or other auxiliary sources. [1]

OCR Word	Correct Word
ज्योतिःशास्त्रीवायकग्रन्थेषु	ज्योतिःशास्त्रीयविषयकग्रन्थेषु
ठिपका-कबडसा	ठिपका-कवडसा
वगक्तिर	वर्गाकार
Niruki«	Nirukta

Fig. 1. Examples of OCR words in Sanskrit, Marathi, Hindi and English (top to bottom) with no correct suggestion from popular engines/spell-checkers. Errors are marked in red and corrections in green. These errors are corrected by our framework.

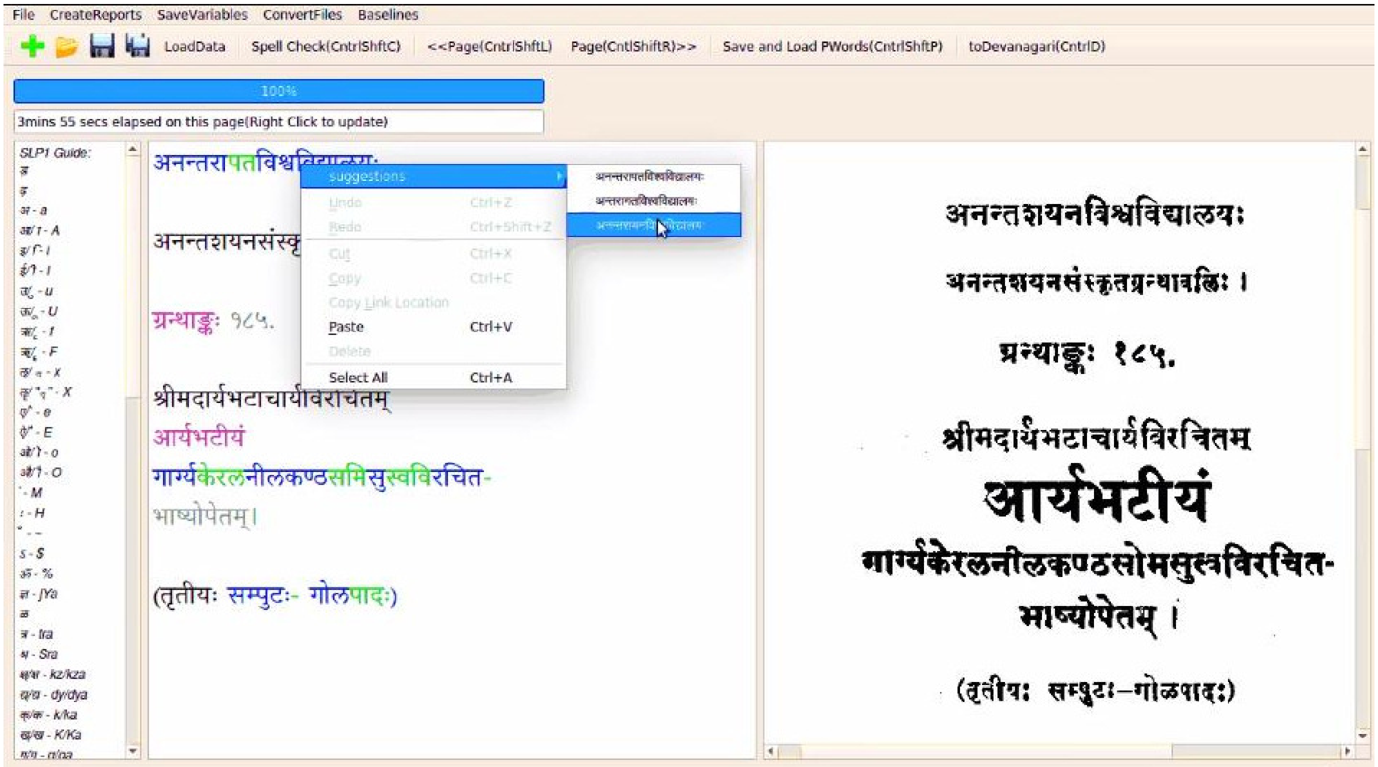
Although the major challenges of error detection in various Indian languages have been discussed in Sankaran et al. [1], we discuss the problem of error detection and correction for Indic OCR and motivate our work in Section II. A systematic discussion on the possible auxiliary sources for getting the document specific suggestions is given in the Section III. The experiments and results are summarized in Section IV, followed by the conclusion in Section V.

II. Problem Scope and Motivation

While investigating OCR error correction, we came across various examples of Indic OCR text which was too hard for various online spell-checkers to verify and/or correct. The suggestions generated were very far from the truth for most of the words. The major reasons for this are:

1) the limited vocabulary is majorly incomplete.
2) limited implementation of language-specific rules for verifying correctly spelled words.
3) the design for such spell-checkers is based on typing errors, and not on frequent OCR n-gram confusions.

[1] The source code of our framework, OpenOCRCorrect, is available at http://tinyurl.com/y9lms89u.



3mins 55 secs elapsed on this page(Right Click to update)

अनन्तरापतविश्वविद्यालयः

अनन्तशयनसंस्कृ

ग्रन्थाङ्कः १८५.

श्रीमदार्यभटाचार्यविराचतम्
आर्यभटीयं
गार्ग्यकेरलनीलकण्ठसमिसुस्वविरचित-
भाष्योपेतम्।

(तृतीयः सम्पुटः- गोलपादः)

अनन्तरापतविश्वविद्यालयः
अन्तरापतविश्वविद्यालयः
अनन्तशयनविश्वविद्यालयः

अनन्तशयनविश्वविद्यालयः

अनन्तशयनसंस्कृतग्रन्थावलिः ।

ग्रन्थाङ्कः १८५.

श्रीमदार्यभटाचार्यविरचितम्
आर्यभटीयं
गार्ग्यकेरलनीलकण्ठसोमसुत्वविरचित-
भाष्योपेतम् ।

(तृतीयः सम्पुटः–गोळपादः)

Fig. 2. A screen-shot of our framework.

4) ambiguities in suggestions from dictionary due to large no. of basic word forms.

Some examples of erroneous OCR outputs for which the online spelling correction systems failed are given in Figure 1. We are able to provide correct suggestions for these words through our system.

Generating the spelling suggestions is even more of a problem for Indian languages, since the average typing speed of professional Indian typists is much slower than the average typing speed of a professional typist in English, which is 75 words per minute (WPM) at a word error rate of roughly 0.5% [9]. The reason for this is that the keyboard is designed for a total of 26 characters whereas there are over 50 characters in Sanskrit, out of which vowels exist in different forms at different locations. Moreover, the average length of a word in Indian languages is much longer than English due to conjoining words which make typing a more difficult task for curating errors. [2]

Our research is motivated by various observations; a) as auxiliary sources and the methods to correct the word increases, the possibility of getting the correct word suggestion increases, b) a suggestion that is not helpful in complete word correction may be helpful in partial word correction and c) a correct conjoined word is formed by combination of a minimum number of words from the word dictionary. The problem in reading the conjoined words, due to their large length, is also an important factor while curating the OCR errors. To overcome this, we provide a user-friendly color coding scheme in our framework for the partial dictionary string matches, for each combined word. Figure 2 shows a screen-shot of our system. The words verified as correct are marked as black by our framework. The gray words are the words that have been marked as correct by the user (previously at a different location in the document) and the purple words are ones that have been auto-corrected by the system. The user is required to right click to generate suggestions. Each multi-colored (green and blue) word is a conjoined word consisting of substrings which are valid words in either the word dictionary or the domain vocabulary (which is updated on the fly with corrections). The colors (green and blue) differentiate the adjacent valid substrings of the conjoined word. An error is more likely to be present in the places where the green/blue substring is short (of length 2/3 chars).

III. Auxiliary sources

Various auxiliary sources are found to be helpful in generating the correct suggestions for an erroneous word. The same is discussed in the decreasing order of relevance.

[2] In our system we provide the facility to type in SLP1 (an ASCII transliteration scheme) format, since typing in English is much easier; once the user gets well conversant with typing in the SLP1 format. If "Ctrl+D" is clicked after typing in SLP1 format, the word under the cursor is automatically converted to Devanagari.

A. *Domain specific vocabulary*

One of the powerful auxiliary sources for OCR error corrections could be a domain specific vocabulary. Our work aims to digitize out-of-print books in Indian languages. Initially no domain-specific vocabulary is available, but as the user corrects a word in the document, we update a domain-specific vocabulary to further help in correcting the remaining words. We further use this vocabulary for correcting other books written in the same domain.

B. *OCR documents from different systems*

In our experiments, it was observed that the OCR document itself is one of the most powerful auxiliary sources in correcting the erroneous text since it contains the domain information. Such an auxiliary source is helpful when the OCR document has decent word level accuracy and hence frequently occurring words can be used to generate the correct suggestion for the erroneous OCR words. Specifically, the words which are incorrect due to location-specific imaging errors can be corrected with this source. Another powerful auxiliary source in the dual-engine environment is the OCR document from the secondary OCR.

C. *Sub-strings from OCR words conforming to word conjoining rules*

A powerful auxiliary source which is helpful for generating the suggestions for many erroneous words is the sub-strings from the OCR words that are corrected, verified as correct, or conform to the rules for conjoining word forms. The sub-word forms for conjoined words are searched in the word dictionary III-E and the updated domain specific vocabulary III-B.

D. *Document and OCR specific n-gram confusions*

We observed that the error confusions in a word from the primary OCR engine are generally different from the error confusions in the corresponding word from the secondary OCR engine. This is because two different OCR systems use different preprocessing techniques and different classifier models [4]. Thus, the OCR specific confusions of the primary OCR, or the only OCR in single engine environment, can be helpful in deciding whether the part of the erroneous word should be changed or not. For example: if the nearest word to the erroneous word “iiet” are “net” and “pet” from the dictionary, the tie can be broken by the common OCR character confusion ii→n, and hence the word “net” can be given preference over “pet”. Another interesting example would be NWE “iiiternet” matching to “interpret”, where the common OCR confusion ii→n can be helpful in correcting the erroneous word to “internet”. The change to “interpret” is be avoided if m→pr is not a valid OCR character confusion for the primary OCR system. We also take care of n-gram confusions involving two or more characters on either or both sides, e.g.: in English iii→m or iii→in are common OCR confusions. With each correction made by the user, we update the OCR and document specific confusions to be further used to correct the remaining words in the same document, and further correct the remaining OCR documents.[3]

E. *Off-the-shelf dictionary*

Though the vocabulary is generally incomplete in Indian Languages due to the rich inflections, still the frequent words can be corrected via a fixed word dictionary.

IV. Experiments and Results

Lang.	TP	FP	TN	FN	Prec	Recall	F-Scr.
Sanskrit							
LB	87.45	39.02	60.98	12.55	30.36	87.45	45.08
UB	91.62	0	100	8.38	100	91.62	95.63
Dual eng.	82.35	17.64	82.29	17.70	48.04	82.29	60.66
Loglinear	85.13	17.84	82.16	14.87	48.62	85.13	61.89
Marathi							
LB	33.80	25.02	74.98	66.20	36.10	33.80	34.91
UB	15.20	0.03	99.97	84.80	99.49	15.20	26.37
Dual eng.	29.15	12.87	87.13	70.85	48.64	29.15	36.46
Loglinear	76.93	3.83	96.17	23.07	78.77	76.93	77.84
Hindi							
LB	53.15	19.21	80.79	46.85	49.83	53.15	51.43
UB	44.72	1.55	98.45	55.28	91.18	44.72	60.01
Dual eng.	61.76	18.74	81.26	38.23	54.19	61.76	57.73
Loglinear	64.34	15.25	84.75	35.66	55.97	64.33	59.86

TABLE I

Error Detection results. The results of both Dual engine agreement (used in framework) and Log Linear Classifier are better than LB (Lower Baseline) and are in-between (or better than) UB (Idealized Upper Baseline). Also, Log Linear Classifier beats dual engine results.

A. *Error detection*

We analyzed various methods for detecting errors in the OCR text. We observed that commonly used dictionary lookup approach gave a high percentage of True Positives (errors detected as errors) but a lower percentage of True Negatives (correct words detected as correct). Marking all words that can be formed by applying conjoining rules to words from the dictionary as correct increased the True Negatives but reduced the True Positives and hence was not used. We observed through data analysis that task of achieving high F-Score depends upon the complexity of the data and the dictionary being used for error detection. The difficulty can be analyzed by the two baselines as follows:-

- Lower Baseline (LB): Dictionary Lookup based detection with off-the-shelf Dictionary.
- Upper Baseline (UB): Dictionary Lookup based detection with Dictionary set to contain all the words in the Ground Truth. This is an idealized baseline as OOV (Out of Vocabulary) Ground Truth words are never known in advance.

For our framework, we mark words common to the output of two OCR systems as correct as it is highly unlikely for two OCR systems to come up with same erroneous word [4]. Words, where the OCR systems disagree, are marked

[3] Word alignment and confusions extraction is done using dynamic programming.

as incorrect. The results for different Indian languages are summarized in Table I. It can be observed that F-Score for the dual OCR system is better than LB (Lower Baseline) and are in-between (or better than) UB (Idealized Upper Baseline). For comparison purposes, we also trained a logistic linear regression based plug-in classifier [10] on 60% of data with 80:20 as the train:val split, and tested the results of remaining 40% of document words. The plug-in classifier is trained to optimize the F-score of validation data and can also be tuned incrementally on-the-fly. This is done by tuning the probability threshold of the logistic linear classifier to maximize the F-Score on validation data. Using the simple features like frequencies of n-grams (up-to 8) of the OCR word in a fixed dictionary, it gave better F-Score on the test data than Dual OCR agreement. For Sanskrit, the Indian language with highest inflections, we additionally used features such as edit distance between OCR words from two OCR systems, no. of dictionary word components both obtained by applying simple and complex word conjoining rules, and all possible products of these three features as the input features. We also divided each of the first three features with primary OCR word length, and used all of their possible products as another set of features in addition to using two binary features; i) marked as 1 for word common to both OCR systems, else 0, and ii) 1 if the word is from dictionary else 0.

Sugg. Rank	%age of "correct","uniquely correct" suggestions in Sanskrit	Marathi	Hindi
1	29.07, 29.07	15.73, 15.73	14.24, 14.24
2	10.76, 4.45	13.23, 5.11	13.05, 0.01
3	23.42, 4.20	14.09, 3.93	3.47, 0.36
4	15.99, 2.86	3.34, 0.60	3.83, 0.72
5	6.84, 2.06	15.20, 11.99	10.973, 8.11
Total (Uniq.) Suggs.	42.64	37.63	23.44

TABLE II
PERCENTAGE OF ERRONEOUS WORDS FOR WHICH THE CORRECT SUGGESTIONS WERE GENERATED BY VARIOUS AUXILIARY SOURCES AND METHODS IN DECREASING ORDER OF RELEVANCE.

B. Suggestion generation

We avoid using frequency based Language Models for OCR corrections as they can do more harm than good in the OCR setting [11]. Moreover, Language Models are already used in post-processing stage of an OCR system, and hence OCR output is likely to exhibit a lower percentage of contextual errors. We generated various suggestions based on the auxiliary sources mentioned in the previous sections. The suggestion generation results are shared in Table II. The top two suggestions are the nearest suggestions from the secondary and primary OCR documents respectively. The third suggestion is generated with the nearest sub-string search from the secondary OCR document as explained in section III-C. The fourth suggestion is generated by partially correcting the primary OCR word in accordance with the secondary OCR word for the same image as explained in Section III-D. Here the OCR confusions of primary OCR, that are updated on the fly, are used. The fifth suggestion is generated by applying OCR confusions to the OCR word to reach to a word that follows the conjoining rules. Some frequently used word conjoining rules are simultaneously used to split the OCR word until we reach a word that follows the conjoining rules. We look for the word forms for such conjoined words in off-the-shelf as well as domain vocabulary that is updated on-the-fly.

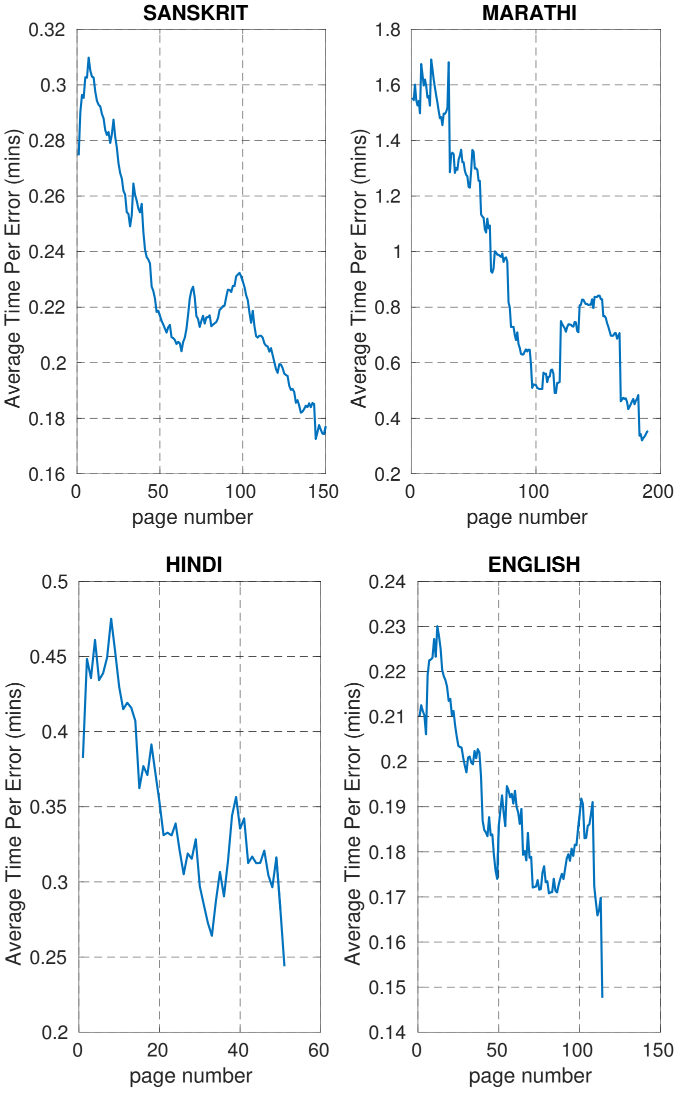


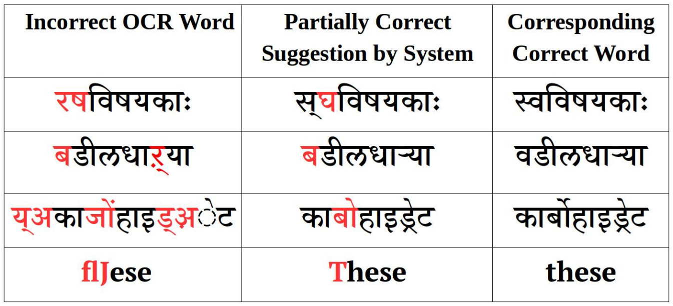
Fig. 3. System analysis of documents in different Languages. For Indian Languages, there is overall decrease in time per error as the user progresses in page number. The system also works well for the English document.

C. System analysis

As mentioned earlier, we have corrected over 1000 pages in English and Sanskrit, 190 pages in Marathi and 50 pages in Hindi using our application.

Here, in Figure 3, we present the real time system analysis for a book in each language. Each book is corrected by a single user so that the efficiency of the framework can be truly analyzed. After correction of each page by the user, the time taken to correct the page is saved. The number of erroneous

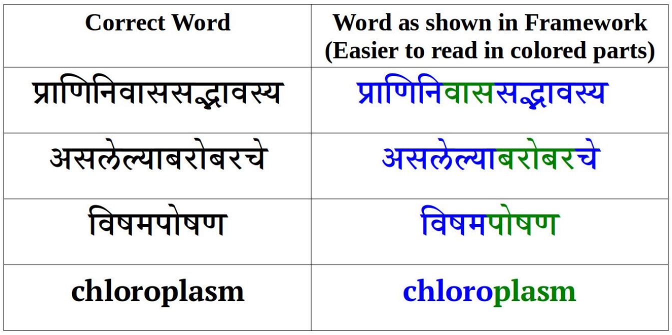
words on each page, calculated using dynamic programming (by comparing original OCR page and corrected page), is also saved and used for calculating time per error on each page. For Indian languages, the average (with the averaging window of the quarter size of the document) time per error decreases with page number for the majority of the document. Ideally, the time per error should be dropping throughout the documents, but fatigue and other factors cause the user to slow down at times. Thus, the system is effective in reducing human efforts for corrections in Indic OCR.
Although the system is designed specifically for Indian languages, we nevertheless used it for English documents. For the particular document in English, refer Figure 3 (bottom right), the average time per error drops similar to the documents in Indian languages.



Incorrect OCR Word	Partially Correct Suggestion by System	Corresponding Correct Word
रषविषयकाः	स्घविषयकाः	स्वविषयकाः
बडीलधार्या	बडीलधाऱ्या	वडीलधाऱ्या
य्अकाजोंहाइड्अ◌ेट	काबोहाइड्रेट	कार्बोहाइड्रेट
flJese	These	these

Fig. 4. Examples of erroneous OCR words for which partially correct suggestions were obtained by our framework.

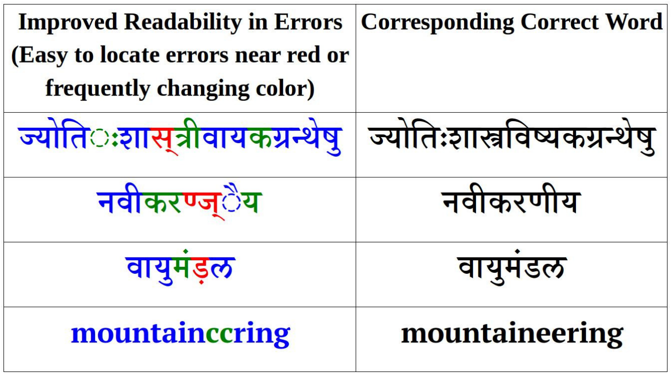
Some errors in OCR output for which the correct suggestions are obtained by our framework are shown in Figure 1. Some examples of erroneous OCR words, for which partially correct words were suggested by our framework, are shown in Figure 4.



Correct Word	Word as shown in Framework (Easier to read in colored parts)
प्राणिनिवाससद्भावस्य	प्राणिनिवाससद्भावस्य
असलेल्याबरोबरचे	असलेल्याबरोबरचे
विषमपोषण	विषमपोषण
chloroplasm	chloroplasm

Fig. 5. Examples of correct Out of Vocabulary (OOV) words (marked as errors) for which readability is improved by our framework.

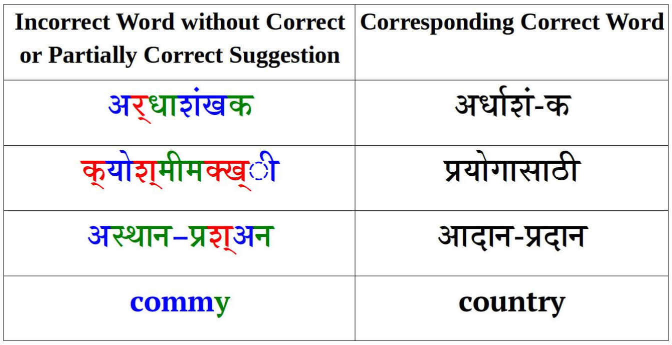
In Figure 5, we show examples of the correct Out of Vocabulary (OOV) words in the OCR output, which are marked as errors (colored) by our framework. However, the user can easily understand that such words as correct due to improved readability. This happens due to adequate color coding of dictionary strings in a combined word. Such coding is also helpful in identifying the errors such as “Therehegoes” where the OCR system fails to recognize the whitespace characters.



Improved Readability in Errors (Easy to locate errors near red or frequently changing color)	Corresponding Correct Word
ज्योति◌ःशास्त्रीवायकग्रन्थेषु	ज्योतिःशास्त्रविष्यकग्रन्थेषु
नवीकरण्ज्◌ैय	नवीकरणीय
वायुमंड़ल	वायुमंडल
mountainccring	mountaineering

Fig. 6. Examples of incorrect OCR words with improved readability.

Some examples of incorrect OCR words for which error locations are easily identifiable due to adequate color coding are shown in Figure 6.



Incorrect Word without Correct or Partially Correct Suggestion	Corresponding Correct Word
अर्धाशंखक	अर्धाशं-क
क्योश्मीमक्ख्◌ी	प्रयोगासाठी
अस्थान–प्रश्अन	आदान-प्रदान
commy	country

Fig. 7. Examples of complex OCR errors not corrected by our framework.

In certain complex cases, our framework is not able to suggest the correct word to the user. Such examples are shown in Figure 7.

V. Conclusions

We designed an interactive approach for word level corrections applicable to Indian languages with varying degree of inflections. The system can easily be adapted to other Indian languages by changing the ASCII transliteration scheme which it uses to store and process the data. Our framework leverages generic word dictionaries and a domain-specific vocabulary grown incrementally based on user corrections from the current on the OCR document. It also learns OCR specific confusions on-the-fly. We have incorporated word conjoining rules to parse OCR words and discover their potentially correct sub-strings. Furthermore, we have presented a dual engine environment to cross-verify potential errors and corrections. We empirically verify that the dual engine environment in conjunction with the previously mentioned resources, yields error detection performance close to the idealized baseline, while additionally providing for accurate suggestion generation. We also presented a plug-in classification approach to further improve the error detection by tuning the probability threshold for classification. Given the role of user interaction in our framework, we have carefully designed the UI to reduce the overall cognitive load by use of transliteration schemes, suitable color coding, and learning on-the-fly from interactions.

References

[1] N. Sankaran and C. Jawahar, “Error Detection in Highly Inflectional Languages,” in *Document Analysis and Recognition (ICDAR), 12th International Conference on*, 2013, pp. 1135–1139.

[2] U. Pal, P. K. Kundu, and B. B. Chaudhuri, “OCR error correction of an inflectional indian language using morphological parsing,” *Journal of Information Science and Engineering*, vol. 16, no. 6, pp. 903–922, 2000.

[3] V. Vinitha and C. Jawahar, “Error Detection in Indic OCRs,” in *12th IAPR Workshop on Document Analysis Systems (DAS)*, 2016, pp. 180–185.

[4] A. Abdulkader and M. R. Casey, “Low cost correction of ocr errors using learning in a multi-engine environment,” in *Proceedings of the 10th international conference on document analysis and recognition*, 2009.

[5] C. Whitelaw, B. Hutchinson, G. Y. Chung, and G. Ellis, “Using the web for language independent spellchecking and autocorrection,” in *Proceedings of the Conference on Empirical Methods in Natural Language Processing: Volume 2*. Association for Computational Linguistics, 2009, pp. 890–899.

[6] S. Hanov, “Fast and easy Levenshtein distance using a Trie,” *http://stevehanov.ca/blog/index.php?id=114*, last accessed on 11/15/2016.

[7] P. Norvig, “How to write a spelling corrector?” *http://norvig.com/spell-correct.html*, 2011, last accessed on 11/15/2016.

[8] M. Choudhury, M. Thomas, A. Mukherjee, A. Basu, and N. Ganguly, “How Difficult is it to Develop a Perfect Spell-checker? A Cross-linguistic Analysis through Complex Network Approach,” *arXiv preprint physics/0703198*, 2007.

[9] F. Vorbeck, A. Ba-Ssalamah, J. Kettenbach, and P. Huebsch, “Report generation using digital speech recognition in radiology,” *European Radiology*, vol. 10, no. 12, pp. 1976–1982, 2000.

[10] H. Narasimhan, R. Vaish, and S. Agarwal, “On the Statistical Consistency of Plug-in Classifiers for Non-decomposable Performance Measures,” in *Proceedings of NIPS*, 2014.

[11] R. Smith, “Limits on the Application of Frequency-based Language Models to OCR,” in *International Conference on Document Analysis and Recognition (ICDAR)*, 2011, pp. 538–542.