A Cost Efficient Approach to Correct OCR Errors in Large Document Collections

Deepayan Das, Jerin Philip, Minesh Mathew and C. V. Jawahar
Center for Visual Information Technology, IIIT Hyderabad, India.
{*deepayan.das, jerin.philip, minesh.mathew*}*@research.iiit.ac.in, jawahar@iiit.ac.in*

***Abstract*—Word error rate of an OCR is often higher than its character error rate. This is especially true when OCRs are designed by recognizing characters. High word accuracies are critical for many practical applications like content creation and text-to-speech systems. In order to detect and correct the misrecognised words, it is common for an OCR to employ a post-processor module to improve the word accuracy. However, conventional approaches to post-processing like looking up a dictionary or using a statistical language model (SLM), are still limited. In many such scenarios, it is often required to remove the outstanding errors manually.**

We observe that the traditional post-processing schemes look at error words sequentially since OCRs process documents one at a time. We propose a cost efficient model to address the error words in batches rather than correcting them individually. We exploit the fact that a collection of documents (eg. a book), unlike a single document, has a structure leading to repetition of words. Such words, if efficiently grouped together and corrected together, can lead to significant reduction in the effort. Error correction can be fully automatic or with a human in the loop. We compare the performance of our method with various baseline approaches including the case where all the errors are removed by a human. We demonstrate the efficacy of our solution empirically by reporting more than 70% reduction in the human effort with near perfect error correction. We validate our method on books in both English and Hindi.

***Keywords*-OCR, Batch Correction, Clustering, Post-Processing**

I. Introduction

The past decade witnessed a growing interest towards the creation of huge digital libraries by digitizing books [1, 2, 3]. One of the crucial steps towards digitization involves the recognition and reconstruction of document image collection(s) using an OCR. The recognition module in the context of digitizing collections such as books, could be considerably different from that of recognizing a single document image [4, 5]. In this work, we extend this idea to error correction in document image collections.

Often the recognition module of an OCR has an automatic error correction module embedded. This may be using a dictionary or a statistical language model (SLM). However, many applications need further improvement in accuracy. This demands a human intervention for removing these errors. In this paper, we propose enhancements to the naive human correction approach which reduces the cost for human expert review by more than 70%. Our work is guided by the following two insights. First, the OCR module makes errors consistently. Word images with similar noise drawn from the same type of document leads to same kind of errors. We demonstrate this in Figure 3 where instances of the same word images, drawn from a document collection are misclassified consistently by an OCR. Secondly, books/document collection have a finite set of unique words and many of them are repetitions. These may include named entities and domain specific words which are mostly unknown to the error detection module. This is further demonstrated in Figure 2 where we show that a subset of words in a collection occurs very frequently. A small set of words can cover more than 50% of the total words in the collection. Under this setting, grouping based on image features or similarity in the predictions of the OCR can provide cues for automatic correction or aide the human expert for manual correction.

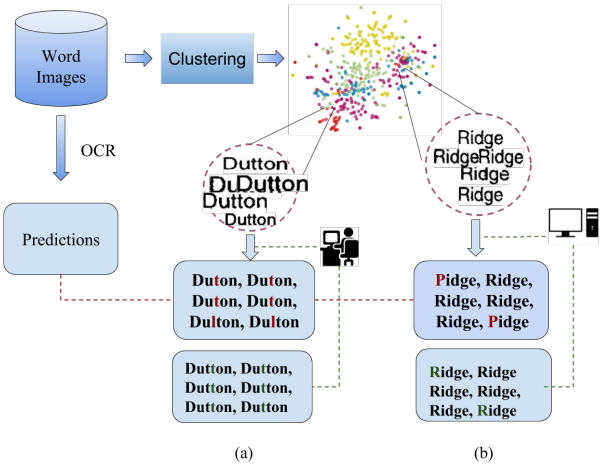


Figure 1: The proposed pipeline for batch correction process where the error instances are clustered and corrected in one go. For a group of error instances, the correct label is chosen and applied. The correct label can be either chosen by a human annotator (a) or generated automatically (b).

We model the problem of word error correction as batch correction where the human expert reviews and corrects errors in batches, than in isolation. Figure 1 presents an overview of our proposed batch correction scheme. Word image-prediction pairs extracted from a collection of documents form groups based on their image and text similarity. In case such a group is recognized incorrectly by the OCR, only one instance from the group needs to be corrected which is then propagated to the rest of the group elements.

Thus, correction needs to be made only once which reduces the cost of correction drastically. The correction can either be automatic or with the help of a human expert. We discuss both these correction processes in detail later in this paper. The major contributions of this work are:

- We demonstrate how clustering can induce an automatic correction and reduce the manual effort in reducing the OCR errors.
- We empirically validate our approach on multiple languages, multiple OCRs, and on a collection of 100 books.

A. Related Work

Conventional approaches to error detection and correction reduce to finding the closest match for an invalid word in a known vocabulary [6, 7]. Bassil and Alwani [7] put forth one of the first works which explored in detail OCR post-processing methods. They consider three modes of correction. In the simplest of approaches, corrections could be performed manually by a human expert. Next, a dictionary-based method similar to what modern day word processors are equipped with was proposed. A possible correction is suggested once an error word was detected. This is accomplished by finding a word in the dictionary with minimum edit distance to the error word which becomes the correction proposal. Dictionary-based approaches could not capture errors in the grammar where words were correct according to the dictionary, but not in the surrounding context. Ability to correct such mismatches was attempted by bringing Statistical Language Models (SLM) using larger language context [8, 9]. The SLMs do not work well for many languages which lack large text corpus to build the language models. Also, they run into issues when newer out-of language words (such as a technical word from a foreign language) come in books. Smith [10] argues that, unless carefully applied, a language model can do more harm than good. Hence it becomes necessary to review the results of a conventional OCR system by bringing a human in the loop for the digital reproduction of a book. To involve human in the loop, projects as early as Project Gutenberg [1] introduced distributed proofreading [11] approaches. Two proofreaders, having access to a book, refine its OCR outputs in turns. A demerit, in this case, is that the entire book has to be visited for proofreading. Von Ahn et al. [12] suggested *ReCaptcha* and report the use of crowdsourcing to transcribe word images. While the corrections are made only in the case of suspected errors, the efforts ignore the possibility of grouping similar misrecognized images and propagating the correct label to each instance in one go. Observing OCR errors to be highly correlated, Abdulkader and Casey [13] proposes a low-cost method to improve the required human-hours needed for correction using clustering. They group OCR outputs first, followed by finding subgroups using the word-image similarities. A similar method based on word image clustering had been shown [14] to be effective in creating index over a collection of documents. However the above two approaches, assume the clusters to be completely homogeneous and thus fail to address cases where the clusters might contain more than one label.

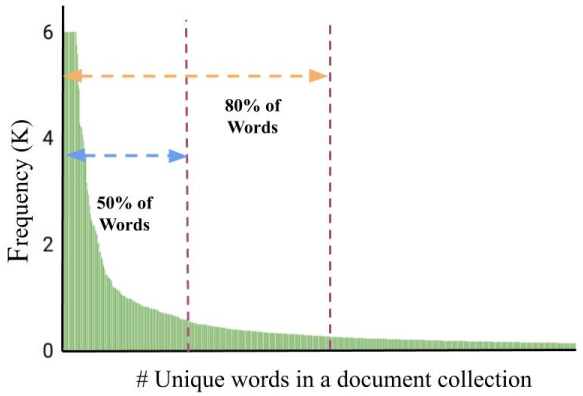


Figure 2: The frequency of unique words in a collection of documents. A subset of words in the collection vocabulary have a very high frequency and accounting for 50% of the words present in the collection. Thus it is safe to assume that if errors occurring in this subset are grouped and corrected in a batch, it can lead to a significant reduction in correction cost.

Baird et al. [15], Taghva et al. [16] note that enabling information retrieval in document image databases is hampered by errors in the OCR outputs [17]. This is very important for the effective use of digital library for books through large scale digitization which include Project Gutenburg [1], Digital Library of India [2] and Google Books [3]. One of the main objectives of such projects is to provide content level access (enable search and retrieval) over the entire digitized collection. Past works turn to humans for correcting the last array of errors left in the pipeline post recognition [12, 13]. All these leave scope for improvement in the space of error correction, especially addressing challenges while scaling up the number of books. Our work is motivated by the works such as [18, 19, 13] that cluster word images to improve the efficiency and accuracy. In this work, we group the erroneous predictions, and present them to a human editor. The human editor then decides the label for the cluster and also the components (elements present in a cluster) to which the label shall be assigned to. The instances where the cluster label do not match the content of the word image are addressed separately by the editor. This mitigates the propagation of errors for clusters that are not homogeneous.

II. Cost Effective Correction

In this section we formulate the problem of error correction and propose two strategies of batch correction.

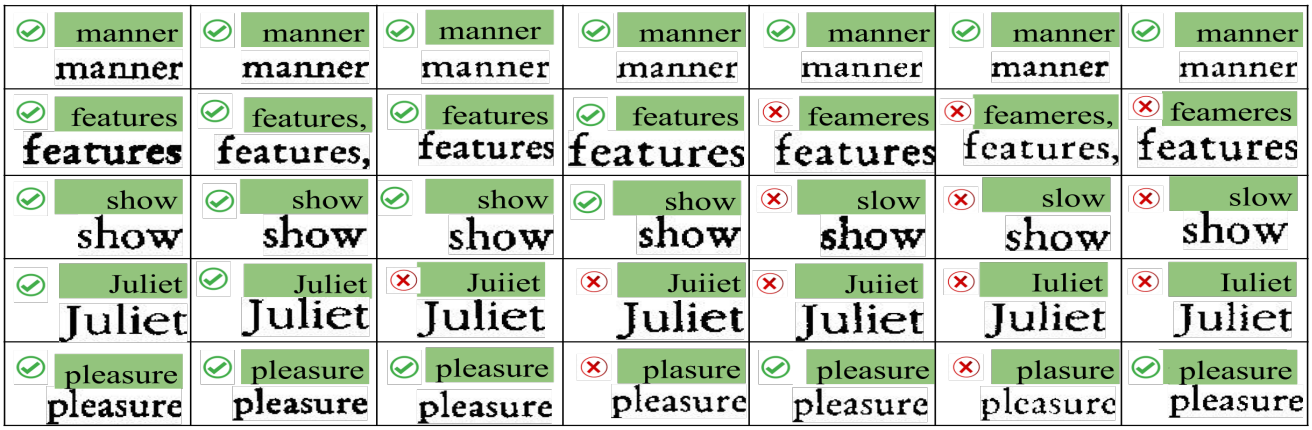


Figure 3: Consistent errors generated by OCR for a given document collection. Each row represents different images for the same word and it's corresponding OCR prediction in the green text box. We can observe that for similar degradations, the OCR outputs similar error patterns.

A. Types of Errors

Recognition modules of OCR systems operate at a character or word level resulting in transcribing word-images into a textual string. Errors in such a setup are inevitable and the cost of manual correction is significantly high. Since it is practically impossible to verify each word manually, we propose to have an independent error detection mechanism operating on the OCR predictions. Assuming that such a system has a low False Negative Rate, only instances where the OCR prediction is not agreed upon by the error detection pipeline, which we denote hereafter as *error instances*, would then need to be corrected. We assume that the errors are detected with a dictionary or an appropriate error detection module. One can categorize the agreement between the recognition module and the error detector into four categories:

1) Error False Positives (EFP): Set of words that are falsely flagged as error by the detection module since they do not exist in the dictionary, such as out of vocabulary (OOV).
2) Error True Positives (ETP): Errors of the OCR which are correctly detected by the error detection module.
3) Recognizer False Negatives (RFN): Words exist in the dictionary but are not the correct transcriptions of the word image.
4) True Negatives (TN): Recognizer correctly predicts word image, and the detection module is in agreement.

Note that the words in TN after error detection are correct words and nothing needs to be done. The words in RFN cannot be detected as an error in isolation. Their correction needs larger language context and is out of scope for this paper. As far as the error correction is concerned, we would like to take human help or automatically correct the words categorized as ETP.

Our objective is to further reduce the word error rate of an OCR. We exploit the fact that OCR system is prone to make systematic errors. Due to the nature of the prediction, multiple instances of the same word could be misclassified to the same wrong label. We propose a grouping of such misclassifications in a collection of documents which enable correcting these multiple errors in one go.

B. Evaluation of Error Correction Effort

We propose a cost based evaluation to demonstrate the efficacy of our method. To this end, we first enumerate all possible edit actions a human in the loop has available and associate a cost with each action.

- We define a verification cost C_v for the case where the reviewer just has to verify an already correct prediction to be a valid word.
- We define average word typing cost C_t for cases where corrections have to be fully typed out.
- For cases where a dictionary provides correction proposals in a drop-down fashion, we define a cost C_d.

For a naive correction process (process where no batching is involved), the editor will have to type out corrections in ETP and verify a word wrongly classified in EFP. The total cost involved turns out to be $C_1 = |\text{ETP}| \cdot C_t + |\text{EFP}| \cdot C_v$. (Here, $|X|$ denotes the cardinality of set X.) We denote this method hereafter as *Typing*. If for some error instances, the editor has an additional option to select from a set of correction proposals, the cost reduces to $C_2 = |\text{ETP}_t| \cdot C_t + |\text{ETP}_d| \cdot C_d + |\text{EFP}| \cdot C_v$ such that ETP_t, ETP_d forms a partition of ETP. Here ETP_t refers to the error true positives which can only be corrected via *Typing* whereas ETP_d refers to the error true positives that can be corrected by choosing the correct suggestion from the set of correction proposals. This method is denoted as *Typing+Selection* hereafter.

C. Overview of the method

We hypothesize that correcting similar instances in EFP and ETP together can make digitization efforts more efficient.

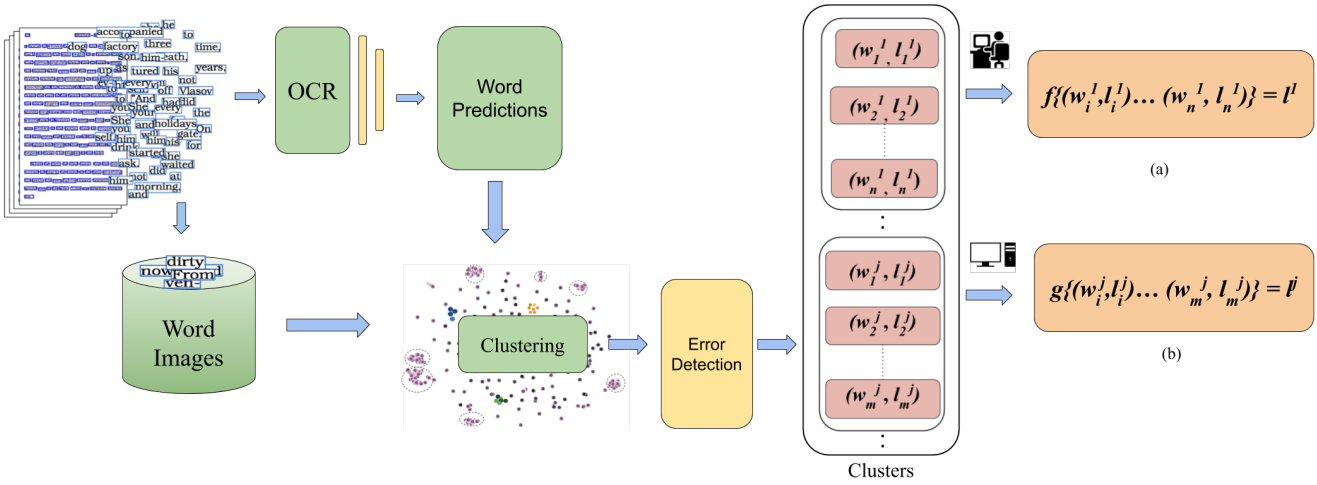


Figure 4: Pipeline of proposed batch correction approach. Given word images ($w_i \dots w_n$) and its corresponding OCR predictions ($l_i \dots l_n$) we form clusters. Next, the clusters containing error instances are sent for correction. We employ two forms of correction approaches which are shown in (a) when the human editor decides the label for a cluster and in (b) when the cluster label is generated automatically.

As mentioned above, we propose an approach that groups error instances together, based on some similarity metric and propagate the correction of one of these to the rest of the group. Correction candidate for a group of error words can either be fully automated or done with human aid. We discuss both the propositions in detail later in this section. In the ideal case, word images with same ground truth will be grouped together, and the ability to correct them in one go would provide an efficient way for humans in the loop to correct large document collections. If we could group the error instances based on their ground truths as $C_1, C_2 \dots C_{|V|}$, each of these groups could be corrected in just one action from editor leading to a cost of $V_t C_t + V_d C_d + V_v C_v$ such that $V_t + V_d + V_v = |V|$. Here V_t, V_d and V_v are numbers of clusters requiring typing, selection from dictionary and verification respectively.

Our proposed model for error correction is presented in Figure 4. The document images, segmented at word level go through the OCR pipeline which assigns them labels/predictions. The word images and their corresponding predictions are subsequently sent through a clustering pipeline, which groups the word images based on their image and text similarity. We discuss the clustering pipeline along with the features on which the clustering is performed in Section III. Next we perform an error detection on the components of each cluster and identify those clusters in which error instances occur. Only those clusters which contain error instances are sent for either of the two correction techniques-automated or human aided, which are dicussed below.

Automated approach: For a given cluster containing word images and their corresponding OCR predictions, the most frequent prediction is chosen to be the representative of the whole cluster and its label is propagated to the remaining cluster elements. Two scenarios arise out of such a setting. For a given cluster, the number of correct predictions is more or less than incorrect predictions. This leads to two situations.

In the first case, words appearing in ETP get corrected automatically without any further manual corrective action other than verification. In the second case, words appearing in EFP (proper nouns, acronyms, technical keyword, etc.) get corrected without much cost, while for clusters containing ETP, even the correct predictions end up being assigned the wrong label. Thus a human editor is required to verify the assigned label with the actual word image for every erroneous prediction and make keyboard entries wherever necessary. This leads to an added correction cost.

Human aided approach: We allow a human editor to pick the representative of the cluster. This reduces the cost by eliminating the chances of error propagation which arise when labels are generated automatically. However, this also mandates that a human editor be present throughout the correction process. In case of ETP, the editor can enter the correction once and the correction is propagated to all matching images. Our method here reduces the cognitive load for the human, thereby improving efficiency.

In the above two approaches we consider the clusters to be completely homogeneous. Clusters containing impurities and the relevant correction approach is discussed later in the paper.

III. Grouping Error Words

In this section, we provide the details of our approach for grouping error words together. As discussed earlier, this significantly reduces the human cost.

A. Features and Clustering

For every error instance, we have two types of features for use in clustering: text predictions of OCR and features from word-image.

Image Features: We use the pre-final layer representations from deep neural networks trained to classify word-images. Such representations capture the discriminatory information between different word-images and have demonstrated success in embedding similar images together [20]. The activation for an image can be considered as a compact representation in a continuous space. For clustering the above features, we employ the *k-means* [21] algorithm. The number of clusters k is set to number of unique words in a collection. The *k-means* algorithm has a time complexity of $O(n^2)$ where n is the number of error instances detected by our pipeline.

Text Features: For text features we propose using the word predictions of the OCR. A natural distance measure for such features is the edit distance, which has been found to be of significant help for error detection in past work. However, approaches like *k-means* are ill-suited to the discrete nature of these features and our distance measure. Therefore, we propose using a Minimum Spanning Tree (MST) based approach [21] using pairwise edit-distance to cluster variants of text predictions. This could also group consistent errors which comprise of error instances where the (1) Prediction is right but error detection is in disagreement. (2) for the same kind of word-image (different word iamges of same textual word) OCR consistently give the same erroneous prediction due to bias in training data.

For clustering, we consider the predictions as vertices of a weighted undirected graph, and the pairwise edit-distance between two vertices form the edge weights. Distances between vertices are scaled to $[0, 1]$. A MST is constructed and edges with weights greater than a threshold are discarded, which results in a forest where each connected component forms a cluster.

Image Features and Text: Word images with high visual similarity but having different text content can be grouped into the same cluster since they might be close to each other in the image feature space. This leads to fragmentation or formation of impure cluster. Assuming one true label per cluster can induce an additional cost of correcting word instances whose ground truth is different from the assigned label. To address the intra-cluster variability we further partition each cluster into sub-clusters by leveraging the textual transcription of each word image such that words that lie within a predefined edit distance, can be grouped into the same sub-cluster.

B. Practical Issues in Clustering Algorithms

In a simpler first approach over a fewer number of books, we use *k-means* and MST based clustering algorithm to group error instances together. While the two algorithms work well for a fewer number of books, they don't scale to much larger book collections. We address this by using a Locality Sensitive Hashing (LSH) based nearest neighbour computation [22] in our clustering pipeline. We discuss in detail the algorithms and their suitability below.

Degradations in print, paper or both over time are prevalent in older documents. Font styles and variations different from the OCR's training distribution used by a common publishing system across these books could be similar in the image space. Similar noise in the images like the cuts and merges lead to consistent errors in OCR. This prior domain knowledge can be incorporated and taken advantage of while clustering. Under these circumstances, we find LSH well suited for scaling up correction in our problem setting. LSH tends to approximate the nearest neighbour search in a way such that items which are similar are hashed into the same 'bucket'. Consistency in noise leads to similar hashes for features from images with similar content. Search space is now limited to the bucket of word-images for which hash matches the query image. This makes the process orders faster.

IV. Dataset and Evaluation Protocols

We experiment on two datasets (both are collection of books). First one fully annotated (FA) and the other one partly annotated (PA). Annotation here imply that we have an error free transcription that allows us to validate whether our corrections are valid or not. This is done by having an error free transcription of the book.

Table I summarizes these dataset.

scale	Language	#books	#pages	#words	# unique
FA	English	19	2417	0.73M	30K
	Hindi	32	4287	1.20M	63K
PA	Hindi				
	- Annotated	50	200	30K	6K
	- Unannotated	100	25K	5M*	80K*

Table I: Details of the books used in our work. Here FA refers to the fully annotated books whereas PA refers to the partially annotated books.

Fully Annotated(FA): This annotated dataset comprises of 19 books in English and 32 books in Hindi. Pages from the books are segmented at a word level and annotated by humans. Five books are set aside from each of the languages to train the OCR while rest of the books are used for testing and further batch correction experiments.

Partially Annotated(PA): In order to demonstrate the scalability of our approach, we run our experiments on a larger collection containing 100 Hindi books. Most of these books were printed decades ago, resulting in degradation in quality of pages. The collection consists of approximately 25K pages with around 5M words. A small subset of 200 pages across 50 books are annotated and set aside as test set

Method	English						Hindi					
	Automated			Human			Automated			Human		
	Type	Type + Select		Type	Type + Select		Type	Type + Select		Type	Type + Select	
	-	Static	Growing	-	Static	Growing	-	Static	Growing	-	Static	Growing
k-means(I)	1.130	0.873	0.692	0.689	0.527	0.372	1.013	0.714	0.648	0.494	0.366	0.234
LSH(I)	0.939	0.732	0.695	0.283	0.232	0.222	0.944	0.664	0.659	0.162	0.135	0.134
MST(T)	1.000	0.740	0.695	0.199	0.187	**0.187**	1.000	0.695	0.681	0.142	0.133	0.132
k-means(I) + MST(T)	1.000	0.853	**0.653**	0.607	0.459	0.327	0.960	0.681	**0.634**	0.281	0.217	0.191
LSH(I) + MST(T)	0.947	0.739	0.689	0.285	0.232	0.222	0.949	0.666	0.651	0.153	0.129	**0.128**

Table II: Evaluation of costs of each approach proposed in this paper. The numbers reported are relative to Full Typing method. We observe a decrease in cost as we go left to right for each clustering approach for books of a given language. 'I' stands for image features, and 'T' stands for prediction text.

A. Evaluation Protocol

Our primary objective of evaluation is to find the effectiveness our proposed batch correction method, as compared to the naive approach where each error is corrected individually. To that effect we use units of seconds that a human expert is required to put in for each method. We also analyze the gains across the various clustering approaches as well as different features used for clustering. Having fully annotated (FA) ground-truth information allows to estimate the costs.

In partially annotated (PA), we evaluate the performance of our approach on a large collection of 25K pages. Though we use the entire 25K pages for clustering, performance is estimated using the 200 annotated pages, randomly sampled from the books.

We hypothesize that the increase in word accuracy of the OCR translates to a reduction in correction cost. Also, since the subset of pages used in evaluation belong to the same pool of books which the larger clustering algorithm is run on, it is reasonable to assume that decrease in cost during evaluation is indicative of a decrease in the whole collection.

V. Experiments, Results and Discussion

In this section we present the implementation aspects of the batch correction model and the results on the two datasets.

A. Experimental Setup

The OCR we use to recognize the cropped word images follows CRNN style hybrid image to text transcription model first proposed by Shi et al. [23] in their work for scene-text recognition. We also have an error detection module for verifying the accuracy of these predictions, implemented using a dictionary. We trained two OCRs – one for each language. For training, we set aside 5 books each from English and Hindi book datasets respectively. The English language OCR was trained on word images from nearly 600 pages ($\sim$160K words) while the Hindi language OCR was trained on word images from approximately 650 pages ($\sim$180K words).

The CNN based feature extractor for clustering word images is adopted from Krishnan and Jawahar [20]. The network was initially trained on synthetic handwritten word images and later fine-tuned on a real-world corpus. Real data used in training is the same 160K word images which are used for training the OCR. The segmented word-images are fed to the network and the pre-final layer activations are used as features for clustering.

Error detection is performed using a dictionary. An instance is determined to be erroneous if its OCR prediction is not present in the dictionary. To suggest corrections for an error instance, the dictionary requires a reasonably good vocabulary. We generate a base dictionary by using Wikipedia dumps for the respective language. For each book while testing, we enrich the corresponding base dictionary further using ground truths of books used for training but not the ones we are testing. We use two variants of this dictionary - one *Static* and the other *Growing*. The *Growing* allows for addition of new words to dictionary, like how modern word processors do. In our grouped correction scenario same words could be scattered across clusters and *Growing* dictionary speeds up correction by not having to type the words that are already corrected.

B. Cluster Impurity

One of the limitations of clustering algorithms like *k-means* or MST is their inability to form perfect, homogeneous clusters. Despite our efforts in fusing image and text features together in order to minimize the impurities, outliers manage to creep into the clusters. This is shown in Figure 5. Cluster impurity poses a serious drawback in our error correction pipeline. Up until now we considered our clusters to be homogeneous and formulated our cost accordingly. However, in practice this can lead to wrong cost estimation. For automated approach, cluster impurity can lead to assignment of labels to instances which do not share the same ground truth. Thus an annotator needs to revisit each cluster and correct all unwarranted cluster assignments.

For human in the loop, we let the human assign labels to the cluster components. A human can correct impure parts of the cluster by visual inspection through *Typing* or *Selection*. Consistent errors can be corrected for a group in this case, unlike in automated approach giving this method an advantage.

C. Results and Discussions

Our empirical studies compare costs of error correction across different methods/situations. The cost is measured

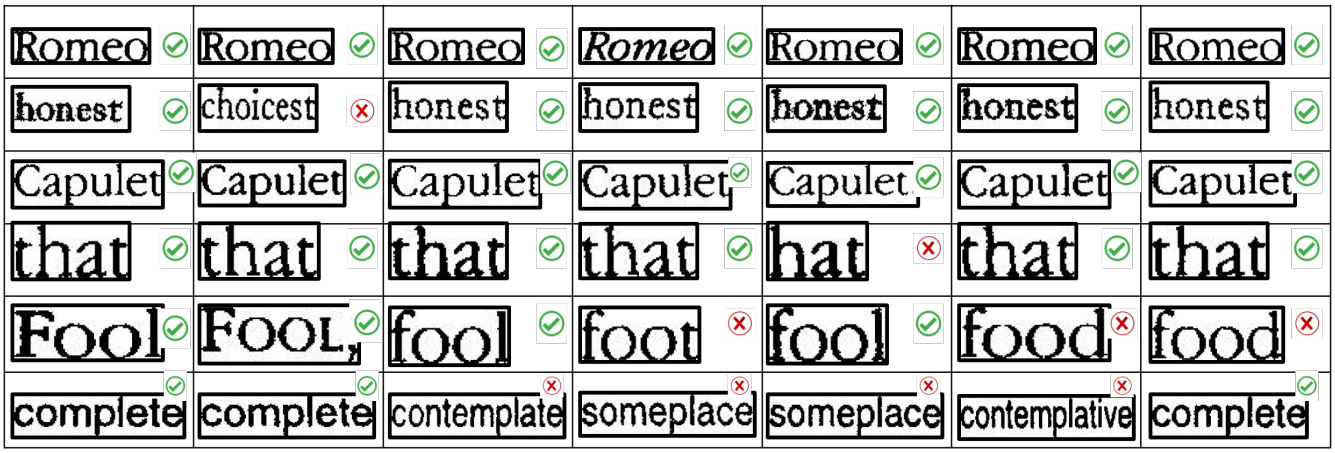


Figure 5: Qualitative results of *k-means* + MST clustering on English dataset. Images, relevant to the cluster are marked correct while the false positives are crossed out.

in units of seconds of human effort put into correction. The following values are used for computing the cost in simulations. We assume C_v as 1, C_d as 5 and C_t as 15 seconds.

All costs in this work are computed relative to *Typing*. Table III delineates the cost for correction without grouping efforts. We experiment with setups involving no dictionary, static as well as growing dictionaries, restricting the edit actions available accordingly. We find *Typing* + *Selection* outperforms *Typing* and *Growing* outperforms *Static*, as expected.

	Type	Type + Select	
	-	Static	Growing
English	1.000	0.740	**0.695**
Hindi	1.000	0.686	**0.681**

Table III: Relative cost of correction with respect to full typing when no batching is involved.

In Table II, we compare the cost of correction when we employ different clustering schemes. Here corrections are performed in batches. Our results are across the two correction approaches - the first which is automated and the second involving a human editor.

The order among relative costs for edit actions and dictionary variants are consistent with the case without batch correction (Table III). Further, we find sequential refinement of clusters using image features and then text-features perform best among different clustering schemes. For the automated approach, *k-means* on image features followed by MST on text features achieve the lowest cost for both the languages. When involving human editor in the process, for Hindi, LSH on image features and refining with MST works best, while for English data MST on text word predictions seems to achieve the lowest cost ($\sim$ 0.13 and $\sim$ 0.2 times the actual typing cost respectively) which is equivalent to more than 70% reduction in human effort.

Correction methods involving human editor consistently outperforms the automated correction approach, even with the former restricted in actions and in dictionary. This can be attributed to the failure of automatic approach in determining which is the correct prediction in a cluster which is largely impure.

D. Results on Large Dataset

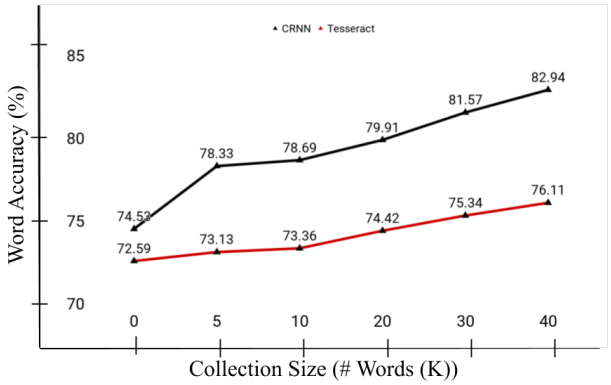


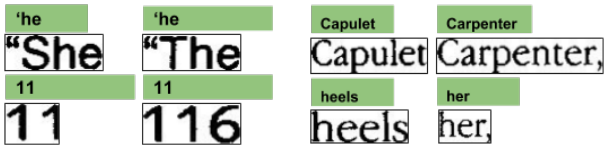
Figure 6: Result on the unannotated data. We observe that as the number of words in the collection increases the automated batch correction method's ability to correct the errored predictions improve which is reflected by the increase in OCR accuracy.

We vary the size of the collection and estimate the accuracy on the 200 fully annotated pages. In addition to using the aforementioned hybrid CRNN for text recognition, we also test the effectiveness of our batch correction technique with the publicly available *Tesseract* OCR for Hindi. We observe that the as the collection size increases from 200 to 25K, our batch correction algorithm becomes better at selecting the right candidate and assigning them to the errored predictions. As a result of which the word accuracy post-correction increases systematically. It can be

seen from Figure 6 that the word accuracy for the dataset improves as the size of collection increases. This implies that for the larger unannotated data, the proposed batch correction method will lead to a better improvement in word accuracy and thus reduction in overall correction cost. We also observe that the gain in word accuracy for our hybrid CRNN-OCR is much more significant than the *Tesseract* OCR. On closer inspection we find that the errors made by the CRNN-OCR mostly fall in the category of EFP i.e. words that were recognized correctly but were flagged down as error by the error detection module. As claimed, these errors can be corrected easily by propagating the most frequent label to all the cluster components. However, this does not seem to be the case with the *Tesseract* OCR where most of the errors fall into the category of ETP. This suggests the the improvement of the OCR word accuracy depends upon the quality and robustness of the existing OCR module. Performance of the traditional methods for error correction does not change with the size of the collection. Our method scales well to large collections and yields superior performance, making it an ideal candidate for large scale efforts like digital libraries.

E. Error Analysis

We discuss failure cases of our proposed correction process with a few qualitative examples. For text predictions clustered using MST algorithm, a few error cases are illustrated in Figure 7a. The recognition module's high confusion in predicting numbers and punctuation extends to clustering using text predictions. But there exists strong cues here in the image feature space which can be used to group samples separately.



(a) Text predictions (b) Image features

Figure 7: Failure cases for clustering on text and image features respectively. Each row in the above figure represents one cluster. The text predictions are depicted in green text box.

Figure 7b shows failures in clustering solely using image features. Instances containing 'Carpenter' and 'Capulet' are grouped into the same cluster although there is a significant difference between their text predictions. Image feature based clustering alone fails to obtain a pure cluster here, but text predictions' similarity can be used to make clusters more pure. We demonstrate such successful refinement in Figure 5. 'Capulet' is one such correction proposal, but the entry corresponding to 'Carpenter' is no longer associated. Failure cases of the combined clustering approach are indicated in Figure 5. Predictions 'fool' and 'food' are inherently different, but still managed to be clustered together. This is likely due to these being very near in image and text space.

VI. Conclusion

In this work we propose a cost efficient batch correction scheme for error reduction in OCRs. We achieve this by discovering clusters of similar errors and processing them together. We empirically study our method in multiple situations and demonstrate the utility. We also demonstrate the scalability of approach. We see two immediate extensions: (i) active learning techniques to find clusters/subclusters that need post-processing. (ii) adapting recognizer to a collection, not just the post-processing module.

References

[1] Project Gutenberg. www.gutenberg.org.

[2] Pramod Sankar K, Vamshi Ambati, Lakshmi Pratha, and CV Jawahar. Digitizing a million books: Challenges for document analysis. In *DAS*, 2006.

[3] Google Books. https://books.google.co.in/.

[4] Pingping Xiu and Henry S Baird. Whole-book recognition using mutual-entropy-driven model adaptation. In *DRR*, 2008.

[5] Neeba NV and CV Jawahar. Recognition of books by verification and retraining. In *ICPR*, 2008.

[6] Karen Kukich. Techniques for automatically correcting words in text. *ACM-CSUR*, 1992.

[7] Youssef Bassil and Mohammad Alwani. OCR Post-Processing Error Correction Algorithm using Google Online Spelling Suggestion. *CoRR*, 2012.

[8] Youssef Bassil and Mohammad Alwani. OCR context-sensitive error correction based on Google Web 1T 5-gram data set. *AJSR*, 2012.

[9] Rohit Saluja, Devaraj Adiga, Parag Chaudhuri, Ganesh Ramakrishnan, and Mark Carman. Error Detection and Corrections in Indic OCR Using LSTMs. In *ICDAR*, 2017.

[10] Ray Smith. Limits on the application of frequency-based language models to ocr. In *ICDAR*, 2011.

[11] Gregory B Newby and Charles Franks. Distributed proofreading. In *JCDL*, 2003.

[12] Luis Von Ahn, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum. recaptcha: Human-based character recognition via web security measures. *Science*, 321, 2008.

[13] Ahmad Abdulkader and Mathew R Casey. Low cost correction of OCR errors using learning in a multi-engine environment. In *ICDAR*, 2009.

[14] K Pramod Sankar and CV Jawahar. Probabilistic reverse annotation for large scale image retrieval. In *CVPR*, 2007.

[15] Henry S Baird, Venugopal Govindaraju, and Daniel P Lopresti. Document analysis systems for digital libraries: Challenges and opportunities. In *DAS*, 2004.

[16] Kazem Taghva, Julie Borsack, and Allen Condit. Evaluation of model-based retrieval effectiveness with OCR text. *ACM-TOIS*, 1996.

[17] Naveen Sankaran and CV Jawahar. Error detection in highly inflectional languages. In *ICDAR*, 2013.

[18] Venkat Rasagna, Anand Kumar, CV Jawahar, and Raghavan Manmatha. Robust recognition of documents by fusing results of word clusters. In *ICDAR*, 2009.

[19] Pramod Sankar K, CV Jawahar, and Raghavan Manmatha. Nearest neighbor based collection OCR. In *DAS*, 2010.

[20] Praveen Krishnan and CV Jawahar. HWNet v2: An Efficient Word Image Representation for Handwritten Documents. *arXiv preprint arXiv:1802.06194*, 2018.

[21] Richard O Duda, Peter E Hart, and David G Stork. *Pattern classification and scene analysis*. Wiley New York, 1973.

[22] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *ACM-STOC*, 1998.

[23] Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. In *PAMI*, 2017.