# Utilizing Web Data in Identification and Correction of OCR Errors

Kazem Taghva and Shivam Agarwal
Department of Computer Science
University of Nevada, Las Vegas
Las Vegas, NV

## 1 Abstract

In this paper, we report on our experiments for detection and correction of OCR errors with web data. More specifically, we utilize Google search to access the big data resources available to identify possible candidates for correction. We then use a combination of the Longest Common Subsequences (LCS) and Bayesian estimates to automatically pick the proper candidate.

Our experimental results on a small set of historical newspaper data show a recall and precision of 51% and 100%, respectively. The work in this paper further provides a detailed classification and analysis of all errors. In particular, we point out the shortcomings of our approach in its ability to suggest proper candidates to correct the remaining errors.

**Keywords:** Post Processing, Information Extraction, Mining, Error Identification, Error Correction, Big Data

## 2 Introduction

The conversion of a large collection of paper documents to electronic is typically done using Optical Character Recognition followed by some sort of post processing [8]. The latter is performed:

- To remove extraneous characters produced as a result of misrecognition of graphics.
- To eliminate end-of-line hyphenations.
- To possibly identify and correct word errors.

The post processing can be very expensive and labor intensive depending on conversion requirements and textual applications. For retrieval purposes, experiments have shown that average precision is unaffected when comparing OCR text to its nearly perfect text version [9]. It is also shown that some characteristics of OCR text can cause problems for certain functionalities of Information Retrieval (IR) and Information Extraction (IE) systems[8].

In this paper, we report on an experiment to automatically identify and correct OCR errors based on the availability of big data on the web.

In section 3, we provide some background and related work on error identification and correction. Section 4 describes our motivation for use of web data for error correction. Section 5 is the description of our algorithm. Section 6 is a summary of our results, and section 7 is a conclusion and proposes future work.

# 3 Background

Identification and correction of misspellings in text has a long history and many of techniques are summarized by Kukich [6]. Other notable work in this area are the work of Brill [3] and Hauser et al.[4]. The latter deals with identification of various spelling forms of words in historical documents. Most of the methodologies cited in these papers deal with typical spelling errors caused by the arrangement of alphabets on the computer key boards. The OCR errors are due to character segmentation and recognition and hence are less responsive to standard spelling correction techniques.

The character accuracy of an OCR engine is obviously a major factor in the number of misspellings in a converted document. A standard text page has about 2500 characters and an OCR engine with 99% character accuracy can produce approximately 25 errors. This does not translate to 25 misspellings as character errors tend to be 1.5 errors per misrecognized word on the average. As a result, 99% character accuracy produces around 15 misspelling per page.

Many studies have been done to understand the nature of OCR errors and possibly suggest remedies for specific text applications such as retrieval or extraction. One such approach is proposed by Jin and Hauptman [5] that incorporates a general content-based correction model that works on the top of an existing OCR correction tool to boost retrieval performance. An example of an OCR correction tool is OCRspell proposed by Taghva and Stofsky [12]. OCRspell is especially designed as a semi-automatic approach. A learning mechanism is used based on the corrections applied by the user. By comparing error-prone tokens and the manual replacement by the user, dynamic mappings are derived. For instance, from iiiount@in $\rightarrow$ mountain the mappings iii $\rightarrow$ m and @ $\rightarrow$ a are derived.

The Hauser[4] work is interesting in the sense that supervised and unsupervised learning are applied to solve spelling variation of words in historical documents. Taghva [11] used supervised learning capabilities of Hidden Markov Model (HMMs) to identify and correct OCR errors in a post processing application. This work showed that the second order HMM does not improve the accuracy over the first order when we require high precision despite the additional context.

In the next section, we report on our work that uses the word sequence and context to correct OCR errors. We will point out that the context beyond four surrounding words does not contribute to more corrections.

# 4 Web Data and OCR Post Processing

The training of HMM based post processing systems collects statistics on character frequency and sequencing to calculate probabilities for decoding of errors via Viterbi. Based on [11], the HMM can correct about 11% of the errors at precision of 100%. In other words, we get 11% improvement without corrupting the text by replacing a correct word with an incorrect

one. We are of the opinion that 100% precision is a good requirement for an ideal post processor. Most of us have observed that spell checkers cannot detect a class of errors in which one correct word is substituted for another correct word erroneously. These type of errors require more context information for identification and correction. One such approach based on statistical language modeling is proposed by Tong and Evans[13]. Another example of a post processor that uses phrases and document content to correct errors is Manicure [10].

One recent work [2, 1] uses Googles Web 1T 5-gram data set to detect and correct OCR spelling. The algorithm works on an OCR text file as follows:

1. Each word is spell checked against Google Web 1T unigram to identify all mispellings in the file.

2. Each misspelling is matched with Google Web 1T to identify the top 10 correctly spelled words based on the number of common bigrams.

3. Each misspelling along with four preceding words in the text file is matched with Google Web 1T to identify five-word sequences.

The correct spelling is picked from the ten candidates based on the ranking of the five-word sequences. The algorithm shows good potential based on a small experiment by authors.

We see at least two ways that the algorithm can be improved. The first one is the fact that OCR errors are different than other spelling errors. For example, we do not type `rn` for `m`. The second one is the fact that document content beyond five-word sequencing can play a role in error correction. We incorporate both of these observations in our algorithm and show examples that can benefit from these changes.

# 5 Error Correction Algorithm

The proposed OCR error correction starts by first cleaning OCR corpus $T$ to remove all characters other than `a` to `z` as well as all the stopwords. Then the cleaned text $T_c$ is screened through spell checker `Jspell` which produces the set of all probable errors $E$.

We concatenate each OCR error with words immediately preceding or following it to generate queries of variable length. Formally each query $Q$ can be denoted as: $Q = w_{-n}, \ldots w_{-2}, w_{-1}, e, w_1, w_2, \ldots w_n$, where $w_{-i}$ represents the ith word that precedes $e$, and $w_i$ represent ith word following $e$. The number of words $2n+1$ can be theoretically as large as one wishes but in our experiments, it ranges over 1,3,5,7 and 9. The query $Q$ is then fed to Google as a search string.

The algorithm selects the `HTML` summary of the top ten ranked documents for further processing. These summaries are parsed to pick the highlighted words to be used as correction candidates for the error $e$. The Google's result page is also searched for Google's *Did you mean* or *Showing results for* . If any suggestion is found then it is appended to the list of Correction Candidates. Let this list be $C = \{c_1, c_2, \ldots c_n\}$. Both Levenstein Edit distance and LCS are used to identify candidates $c_j$ and $c_k$ for correction of error $e$, respectively. Since the possibility that $e$ may be correctly spelled exists, we allow edit distance 0. In

case there are more than one best candidate $c_j$ or $c_k$, we use the OCR confusion matrix $M$ to pick a candidate. The matrix $M$ is used to compute the Bayes probability $p(c \mid e)$ ,the probability of $c$ being the correct candidate given error $e$.

In the next section, we provide details on our data, confusion matrix, and experimental results.

# 6 Experimental Results

To evaluate the performance of our approach, we need to determine the evaluation measures. There are four possible outcomes that can occur when applying a post processing to correct OCR errors:

1. correct → correct: A correct character is still correct after processing. This is a true negative (TN).

2. correct → wrong: A correct character is changed to a wrong character at by the process. This is a false positive (FP).

3. wrong → correct: A character is corrected by the procedure. This is a true positive (TP).

4. wrong → wrong: A wrong character is still wrong after the process. This is a false negative (FN).

Recall and Precision values are calculated using TP, FP and FN as follows:
Recall = $\frac{TP}{TP+FN}$
Precision = $\frac{TP}{TP+FP}$
The definition of these measures has been proposed and discussed by Reynaert [7]. Recall values show the ability of the system to correct errors and the precision values indicate the accuracy of the system.

In order to test the performance of our approach, we used two small data sets. The first set was used to build a generic confusion matrix. This data has been taken from a book titled Notes on Witchcraft with 60 pages, which has been manually corrected with reference to a non-OCR version image of the book. The second set of data is taken from Library of Congress

(http://www.loc.gov/index.html)

.

We picked seven pages based on various criteria such as readability, date of publication, and convenience to map with its corresponding OCR text. These pages are from newspaper articles written between 1836 to 1922 and have been digitized as grayscale with 400 dpi resolution. There are ninety five misspellings in these pages. We are interested in having a high recall at precision of close to 100%. The table 6 shows the recall and precision values associated with queries of different length.

| Distance | 1 word | 3 words | 5 words | 7 words | 9 words |
|---|---|---|---|---|---|
| Levenstein | R=32<br>P=100 | R=44<br>P=100 | R=48.5<br>P=100 | R=43<br>P=100 | R=40<br>P=100 |
| LCS | R=30<br>P=100 | R=45<br>P=100 | R=51<br>P=100 | R=44<br>P=100 | R=41<br>P=100 |

Table 1: The Performance at Different Query Length

Table 6 shows that recall is the lowest value for the one word query. The value increases as query length expands. It reaches its maximum value 51.5% at a query length five. The LCS method gives the highest values for recall at 100% Precision.

It is observed that our approach corrects more errors than the correction suggested by Google's *Did you mean*. The Google's suggestion only, corrects 42 errors out of 95 misspellings but our approach is able to correct 49 misspellings. This is an improvement of around 16.6%. More analysis of data shows that if the context surrounding the error is also misspelled then Google fails to suggest a candidate. For instance, the query tne+territory+mnke+advances+tho does not get a candidate for `mnke` by Google. Further Google has no suggestion for any word in the query day+Oclock+Mrther+tf+jnd. Our approach corrects both `mnke` and `Mrther`. The confusion matrix also provides correction for some words. For example, in the query work underway roaa surVs Yucca, the error `roaa` gets corrected by higher probability of `rosa` over `road`.

# 7 Conclusion and Future Work

This paper is a preliminary explanation of our project. We have shown that the use of document content can improve the post processing beyond Google's *Did you mean* suggestions. We have also shown that the use of a confusion matrix can improve the automated correction procedure for OCR text. As it is stated, this is a preliminary experiment and we intend to report on extended work in the future.

# References


[1] Youssef Bassil and Mohammad Alwani. Ocr context-sensitive error correction based on google web 1t 5-gram data set. *CoRR*, abs/1204.0188, 2012.

[2] Youssef Bassil and Mohammad Alwani. Ocr post-processing error correction algorithm using google online spelling suggestion. *CoRR*, abs/1204.0191, 2012.

[3] E. Brill and R. C. Moore. An improved error model for noisy channel spelling correction. *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, page 286293.

[4] M. Hauser, E. Leiss Heller, K. U. Schulz, and C. Wanzeck. Information access to historical documents from the early new high german period. In *In Proceedings of IJCAI-07 Workshop on Analytics for Noisy Unstructured Text Data (AND-07)*, page 147 154, 2007.

[5] Rong Jin, ChengXiang Zhai, and Alexander G. Hauptmann. Information retrieval for ocr documents: a content-based probabilistic correction model. In *DRR*, pages 128–135, 2003.

[6] Karen Kukich. Spelling correction for the telecommunications network for the deaf. *Commun. ACM*, 35(5):80–90, May 1992.

[7] Martin Reynaert. Parallel identification of the spelling variants in corpora. In *Proceedings of The Third Workshop on Analytics for Noisy Unstructured Text Data*, AND '09, pages 77–84, New York, NY, USA, 2009. ACM.

[8] Kazem Taghva, Russell Beckley, and Jeffrey Combs. The effects of ocr error on the extraction of private information. In *DAS'06 Proceedings of the 7th international conference on Document Analysis Systems*, pages 348–357.

[9] Kazem Taghva, Julie Borsack, and Allen Condit. Evaluation of model-based retrieval effectiveness with ocr text. *ACM Transactions on Information Systems*, 14:64–93, 1996.

[10] Kazem Taghva, Allen Condit, Julie Borsack, John Kilburg, Changshi Wu, and Jeff Gilbreth. The manicure document processing system. In *In Proc. SPIE 1998 Intl. Symp. on Electronic Imaging Science and Technology*, 1998.

[11] Kazem Taghva, Srijana Poudel, and Spandana Malreddy. Post processing with first- and second-order hidden markov models. In *DRR*, 2013.

[12] Kazem Taghva and Eric Stofsky. Ocrspell: an interactive spelling correction system for ocr errors in text. *International Journal of Document Analysis and Recognition*, 3:2001, 2001.

[13] X. Tong and D. A. Evan. A statistical approach to automatic ocr error correction in context. in proceedings of the fourth workshop on very large corpora. In *Proceedings of the fourth workshop on very large corpora*, 1996.