

Evaluation Of Spell Correction On Noisy OCR Data

Aayushee Gupta

Department of Computer Science
Indraprastha Institute of Information Technology
Delhi, India¹
aayusheel230@iiitd.ac.in

Haimonti Dutta

Department of Management Science And Systems
University at Buffalo
Buffalo, USA
haimonti@buffalo.edu

Abstract

Optical Character Recognition (OCR) of historical text often leads to several kinds of spelling errors. Existing spell correction algorithms don't present a rigorous performance evaluation of the spell correction process. In this paper, we present a novel N-gram based algorithm for checking the accuracy of spell correction which can handle noisy and cleaned text of different lengths. First, an edit distance based spell correction algorithm is presented which forms the basis of the evaluation mechanism. The goal of the Spell Correction Evaluation (SCE) algorithm is to measure accuracy of the text corrected by the spell correction algorithm against manually corrected data. N word grams are used to make word-to-word correspondences between the corrected text and the noisy OCR text - each token is then annotated as a True/False Positive/Negative of the correction process. The algorithm relies on appropriately choosing a window of N-words and aligning them in three parallel corpora - noisy OCR, corrected and manually cleaned text (ground truth). Empirical results of spell correction on 50 news articles from the "The Sun" newspaper, Nov- Dec 1894 are presented and the SCE algorithm evaluates its accuracy to be 73.1%. We posit that this novel algorithm for spell correction evaluation has a wide applicability and can play a crucial role in analyzing large volumes of digitized OCR text.

Keywords: OCR, Spell Correction, Spell Correction Evaluation, Historical Newspaper Archives

1 Introduction

OCR of typed, handwritten or printed text is widely used to obtain digitized text which can be edited, searched, stored and displayed efficiently ([13, 22]). OCR'd text is used in various applications such as banking, digital libraries [12] and repositories, number plate recognition, handwriting recognition, etc [18]. But the OCR scanning of printed text especially historical newspaper archives generates a lot of noisy text which renders them inadequate for any such tasks. Refinement of such garbled OCR text though spell correction can also make them useful for text mining tasks such as clustering [6], topic identification [9, 24], etc. Spell correction algorithms have seen lots of improvement over the years but their major contribution has been towards improving the correction model. Most of them either do not give a detailed performance evaluation of the algorithm post spell correction or the evaluation measures used are not able to completely analyze the performance of the spell correction algorithm. A major problem encountered while evaluating a spell corrector surfaces when each and every word of the spell corrected text has to be verified against that of the original text (ground truth) to estimate the performance of the spelling correction algorithm. This one to one

¹The author is currently affiliated to the Department of Computer Science and Information Technology at Jaypee Institute of Information Technology, Noida, India.

verification leads to word alignment problem since the corrected text after OCR and original text can be of different lengths due to multiple spelling errors present in the OCR text. In this paper, we describe the development of an N-word grams SCE algorithm that can automatically evaluate a spell correction algorithm by using an N-word window to align the noisy OCR, corrected and original/ manually cleaned text. We present the performance evaluation results of applying our SCE algorithm on a commonly used spell correction technique- Edit distance algorithm on 50 OCR historical newspaper articles of “The Sun” newspaper published in Nov-Dec 1894. The paper is organized as follows: We describe Related Work in Section 2, Characteristics of OCR data in Section 3, Spelling Correction and Evaluation algorithms in Section 4, Empirical Evaluation and Results in Section 5, results followed by Discussion and Future Work in Section 6.

2 Related Work

Kukich [8] comprehensively discusses various spelling correction techniques based on non word, isolated word and real word spelling errors. N-gram analysis, Dictionary lookup and Probabilistic techniques are used for correcting isolated and nonword errors while Context-Dependent techniques are used mostly for correcting real word errors including the correction of word split and join errors [7].

N-gram techniques work by examining each n-gram in the text string and comparing against a pre-compiled table of n-gram statistics to retrieve the correct word while Dictionary look up techniques directly check whether the text string appears in the dictionary using string matching algorithms. Both techniques require a dictionary or a large text corpus and take frequency of n-grams or word occurrence into account in order to find the correct spelling [19], [16]. Probabilistic techniques use transition and word confusion probabilities to estimate likelihood of the correction in order to rank and retrieve correct word spelling.

On the other hand, Context-dependent techniques require contextual information and use either extensive NLP techniques or Statistical Language Modeling (SLM) for spelling correction. Bassil and Alwani [2] use Google 1-5 gram word dataset to gain context information in order to determine the correct words sequence in the text for correction. Tong and Evans [21] use SLM approach involving information from letter n-grams, character confusion and word bi-gram probabilities to perform context sensitive spelling correction obtaining a 60 percent error reduction rate.

All these spelling correction techniques have developed over time and have been used in combination to achieve improved accuracy [3]. Agarwal et al. [1] use a combination of Google suggestions, LCS and character confusion probabilities for choosing the correct spelling on a small set of historical newspaper data and achieve recall and precision of 51% and 100% respectively.

Edit distance approach, suggested initially by Wagner and Fischer [23], is a dictionary lookup approach commonly used for OCR data correction because of the large number of substitution errors in OCR data [8] [5] which can be corrected using this technique. String edit distance approaches with faster correction are discussed in [11], [17] with variants like Levenshtein automata and normalized edit distance.

There has not been enough research work regarding automatic evaluation of word-by-word post spelling correction on OCR dataset consisting of Word Split and Join errors. All the above mentioned works evaluate their algorithm performance in terms of percentage of spelling errors corrected by the algorithm or reduction in the word error rate and do not consider the word alignment problem arising due to word split and join errors in the OCR text.

Semi-automatic spelling correction system is discussed by Taghva and Stofsky [20] where they correct these errors but their approach requires user interaction in order to perform complete correction and system evaluation. Rice [15] discusses OCR errors similar to the ones in our dataset. Their algorithm evaluates edit distance spelling correction by estimating word accuracy defined as the percentage of correctly recognized words; the length of LCS between correct and incorrect strings on a page by page level is used as the relevant metric. The evaluation strategy works correctly but the definition of accuracy does not give a complete coverage of the spell correction as it does not provide any information on the errors missed by the spelling corrector due to lack of word by word comparison/alignment during the evaluation procedure.

3 Characteristics of OCR data

An individual OCR text article has at least one or more of the following types of spelling errors:

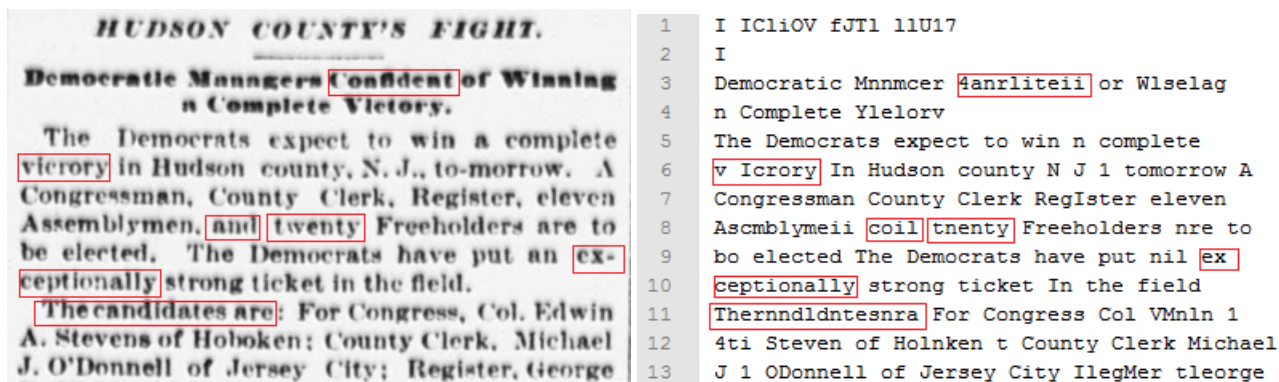


Figure 1: Scanned Image of a Newspaper article (left) and its OCR raw text (right)

- Real word errors** include words that are spelled correctly in the OCR text but still incorrect when compared to the original newspaper article. For example: In Figure 1, the word “coil” has been correctly spelled in the OCR text but should have been “and” according to the original newspaper article.
- Non-real word errors** include words that have been misspelled due to some insertion, deletion, substitution or transposition of characters from a word. For eg. In Figure 1, the word “tnenty” in the OCR text has a substitution error (‘n’ should have been ‘w’) which is actually “twenty” according to the original newspaper article.
- Non-word errors** include words that have been spelled incorrectly and are a combination of alphabets and numerical characters. For example: In Figure 1, the word “4anrliteii” which is a combination of alphabets and number and should have been “confident” as per the original newspaper article.
- New Line errors** include words that are separated by hyphens where part of a word is written on one text line and remaining part in the next line. For example: In Figure 1, the word “ex-ceptionally” where “ex” occurs on one line while “ceptionally” in the next and due to no punctuation in the text, they are treated as separate words in OCR text.
- Word Split and Join errors** include words that either get split into one of more parts or some words in a sentence get joined to a make a single word. For example: In Figure 1, the word “Thernndldntesnra” in the OCR text is actually a combination of three words “The candidates are” while the words “v Icrory” are actually equivalent to a single word “victory” when compared with the original news article.

4 Spelling Correction

Several kinds of spelling errors can exist in the OCR text as described in the previous section.

We use edit distance algorithm for spelling correction because of its speed and ability to correct OCR errors compared to the n-gram approach [4]. Context-dependent spelling correction is not used because of unavailability of n-gram words corpus or ground truth dataset containing OCR and true word pairs.

4.1 Spelling Correction Algorithm

The Edit Distance algorithm based on Levenshtein distance [10] has been used for spelling correction. It is an isolated word correction technique that uses dictionary based-look up method and distance between strings for matching the text and correcting it. An “edit distance” corresponds to the minimum number of insertions, deletions, and substitutions

required to transform one string into another. Our edit distance algorithm corrects non-real word spelling errors up to an edit distance of 2, i.e., it corrects words which have spelling errors that can be corrected by making at most 2 operations of insertion, deletion and substitution of letters in the word. The choice of 2 is governed by the trade off between algorithm runtime and quality of spelling correction. A bigger value improves the spelling correction accuracy but increases the runtime also while a smaller value decreases accuracy and the algorithm runtime. The spelling corrector has been designed as suggested by Peter Norvig². The algorithm requires a dictionary which is used to check if each word of the text exists in it or not. If the word already exists in the dictionary then no change is made to the word and if not, then a candidate list of words is created from the word to be corrected by inserting, substituting or deleting up to 2 letters from it. This list of words is again checked for in the dictionary and returned as suggestions for the word to be corrected. The correction is made with the word formed from lowest edit distance and occurring with more frequency in the dictionary. This makes the edit distance algorithm dependent on the type of dictionary chosen for correction which means the dictionary must be well chosen for spelling correction of a specific document collection. The algorithm is made to run fast by reading the dictionary only once and keeping a data structure in memory for its word counts which can be referred to whenever a word comes up for correction.

4.2 Spelling Correction Algorithm Evaluation

For evaluating the performance of spell correction, the raw OCR text and OCR text after application of spelling correction algorithm (corrected text) needs to be compared with the original newspaper text. The OCR text is extremely garbled with Word Split and Join errors due to which word-to-word alignment with the original newspaper text is impossible, i.e., the raw OCR and original newspaper text are of different lengths. For this purpose, a novel algorithm called SCE (Spelling Correction Evaluation) based on N-gram approach is proposed for automatic evaluation of the corrected text word-by-word against the manually corrected subset of the news articles dataset. The SCE algorithm can be used for evaluation of any type of spelling correction algorithm- dictionary look up, context sensitive or probabilistic spell correction algorithms.

Following sections describe the evaluation parameters for estimating the performance of Spelling Corrector on the OCR dataset used along with the SCE algorithm:

4.2.1 Evaluation Parameters

- **Accuracy** The evaluation metric used by us for measuring the performance of Spelling correction is Accuracy which requires calculation of number of OCR errors that get corrected when compared to the original newspaper text. The measure has been chosen so as to include the complete text coverage and not just check for words that get corrected after spell correction as in the latter case, the number of False Positives and True Negatives get missed which does not give the correct measure of how well the spell corrector works. The formula used for calculating Accuracy is defined by Manning and Schutze, 1999 (p.268-269) as follows:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

where,

TP = Number of True Positives,

TN = Number of True Negatives,

FP = Number of False Positives,

FN = Number of False Negatives.

The aim of the SCE algorithm is to make a word-to-word correspondence between the OCR corrected text and the original newspaper text and to mark each token in the OCR text as a TP , FP , TN or FN . Reynaert and Martin [14] suggest a way to define these terms by distinguishing between correct words and incorrect words in the text through the set of non-target, target and selected words and use Precision and Recall evaluation measures for measuring performance of spelling correction.

²<http://norvig.com/spell-correct.html>

According to our SCE algorithm, a “true positive” is said to occur when a word from the OCR text gets corrected and the corrected spelling matches the one in original article text while a “false positive” occurs if the corrected spelling does not match the corresponding word in the original article text. A “true negative” occurs when a word does not get corrected by the algorithm as it is already correct and matches the correct word in the original text also. On the other hand, a “false negative” occurs when the algorithm is unable to correct the word (there is no change in spelling of the word) and it does not match the corresponding word in the original text but should have been corrected.

- Time taken for Spelling Correction** Time is also an essential parameter while measuring the performance of spelling correction. It is important that the algorithm does not take too long to correct a text article and would be required to be parallelized in case it takes more time for correction for a large dataset.

4.2.2 N-Word Grams SCE (Spelling Correction Evaluation) Algorithm

The SCE algorithm is based on an N-word grams approach. To make the correspondence between corrected and original OCR text, a window of N-word grams in the original newspaper text is considered (Original.txt) which can be seen in a diagrammatic representation in Figure 2.

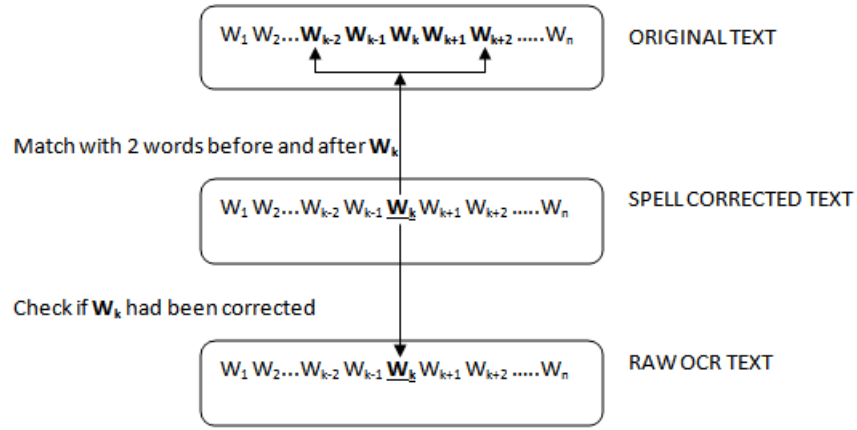


Figure 2: Schematic diagram for alignment of spell corrected article text with original article text for a word W_k

For each token in the spell corrected text (Corrected.txt), the corresponding token in the original text article along with 2 tokens before and 2 tokens after it are considered for alignment³ If the token being considered matches with any word in the original text article words window and its spelling has been corrected when compared to the corresponding token in raw OCR text (OCR.txt), then it is marked as a “True Positive” which is actually rewarding the Spell corrector for making the correct spelling change. A “False Positive” is marked if it does not match any of the words despite its spelling being corrected. If the token being considered matches any of the words in the words window but no spelling correction has been made for it, then it is marked as a “True Negative” and if it does not match any word in the window and the spelling corrector also did not correct it, then it is marked as a “False Negative” as the word got missed by the corrector.

Several cases could occur during alignment like difference in the lengths of linetext between OCR and Original text or while considering the first, second or the last tokens from the Corrected linetext for which the corresponding word window in Original text needs to be smaller. All such cases have been covered in SCE Algorithm 1 which calls function ‘MatchWordGrams’ (Algorithm2) for these different cases.

³The choice of N=2 is based on the Word Split and Join errors in the dataset. Choosing N=2 allows a window of words to be considered to make up for the alignment lost because of Word Split and Join errors. This value can be set up appropriately by considering the maximum difference of lengths in each line of Ocr and Original text in the dataset.

A limitation of the SCE algorithm is that it requires all 3 versions of a newspaper article (Original, Corrected and OCR) to have the same number of lines as alignment of linetexts is performed. In case of difference in the number of lines of text due to some Word Split and Join errors, the words window needs to be extended so as to cover previous and next line texts also for alignment.

```

Input: Ocr.txt,Corrected.txt,Original.txt
Output: Spell Corrector Accuracy
OcrLine:=a line of text from Ocr.txt;
CorrectedLine:=a line of text from Corrected.txt;
OriginalLine:=a line of text from Original.txt;
 $tp \leftarrow 0$   $fp \leftarrow 0$   $tn \leftarrow 0$   $fn \leftarrow 0$ ;
for (int  $i=0$ ;  $i < \text{CorrectedLine.length}$  ;  $i++$ ) do
    if ( $\text{CorrectedLine.length} < 4 \parallel \text{OriginalLine.length} < 4$ ) then
        MatchWordGrams ((OcrLine,CorrectedLine,OriginalLine,0,OriginalLine.length, $i$ )) ;
    end
    else
        if ( $i==0$ ) then
            MatchWordGrams ((OcrLine,CorrectedLine,OriginalLine,0,3,0)) ;
        end
        else if ( $i==1$ ) then
            MatchWordGrams ((OcrLine,CorrectedLine,OriginalLine,0,4,1)) ;
        end
        else if ( $i==( \text{CorrectedLine.length}-2) \parallel ( \text{CorrectedLine.length}-1) \parallel$ 
            ( $\text{CorrectedLine.length}$ )  $\parallel ( \text{CorrectedLine.length}+1)$ ) then
            MatchWordGrams ((OcrLine,CorrectedLine,OriginalLine, $i-2$ ,OriginalLine.length, $i$ )) ;
        end
        else if ( $i \geq \text{CorrectedLine.length}+2$ ) then
            MatchWordGrams ((OcrLine,CorrectedLine,OriginalLine,OriginalLine.length-
            3,OriginalLine.length, $i$ )) ;
        end
        else
            MatchWordGrams ((OcrLine,CorrectedLine,OriginalLine, $i-2$ , $i+2$ , $i$ )) ;
        end
    end
end
 $\text{Accuracy} = (tp + tn) / (tp + tn + fp + fn)$ ;

```

Algorithm 1: SCE Algorithm for Spell Correction

```

function MATCHWORDGRAMS(OcrLine, CorrectedLine, OriginalLine, jstart, jend, i)
  for (int j=jstart; j<jend; j++) do
    if ((CorrectedLine[i].equals(OriginalLine[j]))&&!(OcrLine[i].equals(CorrectedLine[i]))) then
      tp = tp + 1;
      flag0=false;
      return tp;
    end
    else if ((CorrectedLine[i].equals(OriginalLine[j]))&&(OcrLine[i].equals(CorrectedLine[i]))) then
      tn = tn + 1;
      flag1=false;
      return tn;
    end
  end
  if (!(OcrLine[i].equals(CorrectedLine[i]))&&flag0==true) then
    fp = fp + 1;
    return fp;
  end
  else if ((OcrLine[i].equals(CorrectedLine[i])) && flag1==true) then
    fn = fn + 1;
    return fn;
  end
end function

```

Algorithm 2: MatchWordGrams Function called by Algorithm 1

An Example

Working of the SCE algorithm can be demonstrated with the help of the following example: Consider 3 versions of a scanned image of a newspaper article, the original text of the scanned image in Figure 3 and the raw OCR text and the corrected text (after spell correction) in Figure 4. As highlighted in the figures, for line 6 the line texts are:

OcrLine= *by tltn rejmr of th cepert aocountauts who*

CorrectedLine= *by than report of the expert accountants who*

OriginalLine= *by the report of the expert accountants who*

Here, for each token of CorrectedLine, we find its index and call the MatchWordGrams function accordingly. For the first token 'by' at index i=0 in CorrectedLine, we consider the word window to be "by the report" (index j=0 to 2) in OriginalLine by matching iteratively with each token to see if there is a match and also if there has been a spelling correction by comparing with the corresponding token in OcrLine. Here, no change was made to the spelling of 'by' and it matches with a word in words window, so it is marked as a FN. For the second token 'than' at index i=1, we consider the word window to be "by the report of" (index j=0 to 3) for which there is no match in the window but there has been a spelling correction from 'tltn' to 'than', which implies the correction was wrong and the token is marked as a FP. For the third token 'report' at index i=2, we consider the window as "by the report of the" (index j=0 to 4) in Original Line and find that there is a match in the word window and there has been a spelling correction too from 'rejmr' to 'report' which makes this token a TP. Similarly, rest of the tokens get marked for each line in the Corrected.txt.

Another example can be considered from Line 10 in Figure 3 and Figure 4 where the number of tokens is different in CorrectedLine and OriginalLine. In such a case, direct alignment between tokens is not possible because of which the words window becomes useful. Here, when the last token 'Richmond' of CorrectedLine is considered at index i=3, the corresponding words window becomes "Jury now sitting at Richmond" (index j=1 to 5) for which there is a match in the words windows and corresponding spelling has also been changed from 'tilchmond' to 'Richmond' which makes it a TP. Had the word window not been considered, the corresponding token at index j=3 in OriginalLine would have been chosen as 'sitting' which would have resulted in a FP.

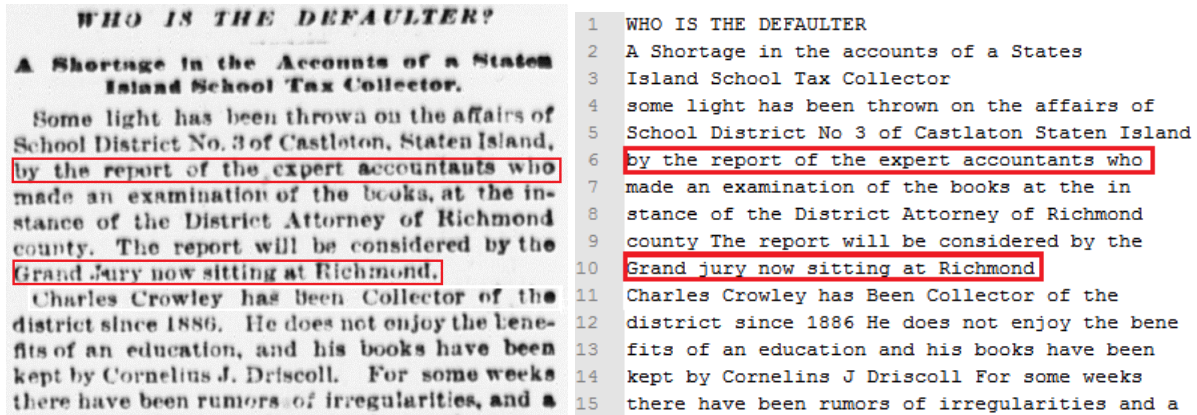


Figure 3: Scanned image of a newspaper article (left) along with its original text (right)

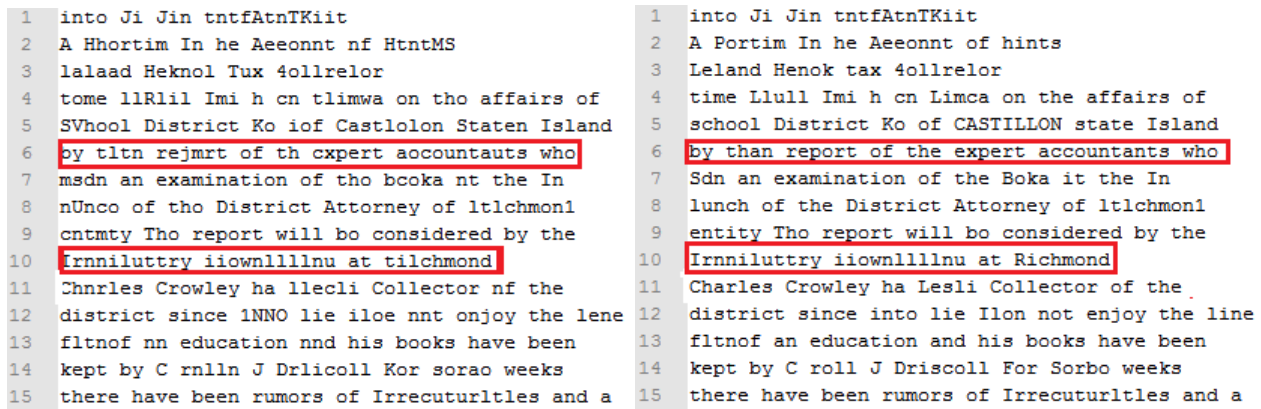


Figure 4: OCR raw text (left) and Spell corrected text (right) of the article

5 Empirical Evaluation and Results

5.1 Data Source

The dataset has been taken from Chronicling America. *Chronicling America*⁴ is an initiative of the National Endowment for Humanities (NEH) and the Library of Congress (LC) whose goal is to develop an online, searchable database of historically significant newspapers between 1836 and 1922. The New York Public Library (NYPL) is part of this initiative and has scanned 200,000 newspaper pages published between 1890 and 1920 from microfilm.

In order to make a newspaper available for searching on the Internet, the following processes used in [6] must take place: (1) the microfilm copy or paper original is scanned; (2) master and Web image files are generated; (3) metadata is assigned for each page to improve the search capability of the newspaper; (4) OCR software is run over high resolution images to create searchable full text and (5) OCR text, images, and metadata are imported into a digital library software program. The scanned newspaper holdings of the NYPL offers a wealth of data and opinion for researchers and historians.

The newspaper titles and digitized pages available through the Chronicling America website can be searched using the OpenSearch protocol⁵. Unfortunately, the current search facilities are rudimentary and irrelevant documents are

⁴<http://chroniclingamerica.loc.gov/>

⁵<http://www.opensearch.org/Home>

often more highly ranked than relevant ones. The newspapers are scanned on a page-by-page basis and article level segmentation is poor or non-existent; the OCR scanning process is far from perfect and the documents generated from it contains a large amount of garbled text. In a bid to serve its patrons better, the New York Public Library employed human annotators to clean headlines of articles and text, but the process of manually reading all the old newspapers article-by-article and cleaning them soon became very expensive.

5.2 Data Statistics

The OCR text available from Chronicling America website is on a page by page level and no article level segmentation is provided. OCR text dataset is therefore, taken from a PostgreSQL database where article level segmentation of page-level OCR text from Chronicling America is available for two months of articles of “The Sun” newspaper from November-December 1894 consisting of 14020 news articles with a total of 8,403,844 tokens. The newspaper database ER diagram ⁶ is used to extract the required articles text from the database by dumping complete dataset and extracting individual articles linetext based on their unique ID. The individual text articles generated from the database do not have any punctuation and contain a large amount of garbled text containing OCR errors mentioned in Section 3.

5.3 Experimental Procedure and Results

Aim: The aim of our experiments is to answer the following question: How good is the spell corrector? The metrics for evaluation are accuracy and time taken to correct the text.

Materials: The spelling correction algorithm is used to correct all the 14020 OCR raw text articles in the dataset. The dictionary used for look-up is a concatenation of several public domain books from Project Gutenberg and lists of most frequent words from Wiktionary and the British National Corpus⁷. This is augmented with a large people names list which is obtained by running Stanford NER-CRF parser on subsets of the ClueWeb12 dataset made available in the TREC 2013 Crowdsourcing Track⁸. This enhanced dictionary has been used to give special consideration to correction of person names in the dataset.

Methods: In order to answer our research question for checking the performance of spell corrector, we do the following:

3 versions of each newspaper article are required: OCR raw text, spelling corrector corrected text and the original scanned newspaper article text. Since the dataset is quite large (14020) and it is not possible to get original text of each of these newspaper images, a smaller number of articles are chosen to study the results of spelling correction. 50 scanned newspaper images are taken and an online OCR ⁹ is run on them followed by some manual correction to get the original articles text. Accuracy can then be calculated for all 3 versions of 50 newspaper articles using the SCE algorithm by marking each word in the OCR text article as a *TP*, *FP*, *TN* or *FN*.

Results: The spell corrector takes 9 seconds on an average to correct the newspaper OCR articles. It takes a total of 36 hours to run on 14020 articles. The spell corrector also shows an Accuracy of 73.1% when corrected text is compared to OCR text and original article text using our SCE algorithm. We believe that the results are less accurate due to the presence of a large number of non-word, new line, word split and join errors in the OCR data which can not be corrected by the edit distance spelling corrector used for this research.

5.4 Discussion

The edit distance based spell corrector used in this work corrects non-real word errors by focusing on isolated words in the dataset. We believe a better accuracy of spell correction can be obtained by correcting the new line errors in the articles. This can be done by checking for if the word at last index of a linetext or the word at first index of the

⁶[https://power.ldeo.columbia.edu/twiki/pub/Incubator/BodhiDBDesign/Final ERD.pdf](https://power.ldeo.columbia.edu/twiki/pub/Incubator/BodhiDBDesign/Final%20ERD.pdf)

⁷<http://norvig.com/big.txt>

⁸<http://boston.lti.cs.cmu.edu/clueweb12/TRECcrowdsourcing2013/>

⁹www.onlineocr.net

next linetext is a word not present in the dictionary and combining the two and checking again in the dictionary for a valid word. The new word, if present in the dictionary can be replaced by the two words from which it is formed thereby removing the New Line error. Similar approach can be applied for word split and join errors but would require each word of an article not present in the dictionary to be analyzed along with some window of words before and after it to make a correction. Since edit distance algorithm is governed by the dictionary choice, using a dictionary with historical terms, places and people names can also help perform spelling correction better and improve its accuracy.

We also compared our N-gram based SCE algorithm with the LCS (Longest Common Subsequence) algorithm.¹⁰ The LCS of corrected and original text gives a list of matching corrected words found in the original text. Following the similar evaluation procedure of calculating accuracy as in the N-word gram approach, if a corrected word finds a match in the LCS and its spelling is found to be corrected, then it is marked as a TP otherwise a TN and if no match is found in LCS and spelling has been corrected, then it is marked a FP or else FN. We didn't find a significant difference in the accuracy obtained through both the algorithms and conclude that LCS is a special case of the N-word gram algorithm when the window size N is set to the complete linetext.

6 Conclusion and Future Work

In this paper, we presented a novel approach for automatic performance evaluation of spell correction on noisy OCR text through N-word grams alignment of the OCR, corrected and manually cleaned text. Preliminary results of application of our algorithm on an Edit distance based spell corrector evaluate its accuracy to be 73.1%. We believe the N-word gram SCE algorithm can be used to evaluate accuracy of any kind of spell correction algorithm and provide a detailed performance by considering complete text coverage compared to simpler measures like percentage of corrected words or word error rate used previously. We plan to use other spelling correction algorithms like context dependent spelling correction to correct the OCR text and measure the accuracy using our SCE algorithm to compare among multiple algorithms and decide which one suits the dataset better and gives best accuracy. We shall also work towards developing a measure so that the N-word gram window size used for alignment can be calculated automatically for any dataset.

7 Acknowledgement

References

- [1] Shivam Agarwal. Utilizing big data in identification and correction of ocr errors. 2013.
- [2] Youssef Bassil and Mohammad Alwani. Ocr context-sensitive error correction based on google web 1t 5-gram data set. *arXiv preprint arXiv:1204.0188*, 2012.
- [3] Eric Brill and Robert C Moore. An improved error model for noisy channel spelling correction. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 286–293. Association for Computational Linguistics, 2000.
- [4] Ishan Chattopadhyaya, Kannappan Sirchabesan, and Krishanu Seal. A fast generative spell corrector based on edit distance. In *Advances in Information Retrieval*, pages 404–410. Springer, 2013.
- [5] Peter Christen. A comparison of personal name matching: Techniques and practical issues. In *Data Mining Workshops, 2006. ICDM Workshops 2006. Sixth IEEE International Conference on*, pages 290–294. IEEE, 2006.
- [6] Haimonti Dutta, Rebecca J Passonneau, Austin Lee, Axinia Radeva, Boyi Xie, David L Waltz, and Barbara Taranto. Learning parameters of the k-means algorithm from subjective human annotation. In *FLAIRS Conference*, 2011.
- [7] Mohammad Ali Elmi and Martha Evens. Spelling correction using context. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 1*, pages 360–364. Association for Computational Linguistics, 1998.

¹⁰https://en.wikipedia.org/wiki/Longest_common_subsequence_problem

- [8] Karen Kukich. Techniques for automatically correcting words in text. *ACM Computing Surveys (CSUR)*, 24(4):377–439, 1992.
- [9] Austin Lee, Haimonti Dutta, Rebecca Passonneau, David Waltz, and Barbara Taranto. Topic identification from historic newspaper articles of the new york public library: A case study. In *5th Annual Machine Learning Symposium at the New York Academy of Sciences*, 2010.
- [10] Vladimir I Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. In *Soviet Physics Doklady*, volume 10, page 707, 1966.
- [11] Andres Marzal and Enrique Vidal. Computation of normalized edit distance and applications. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(9):926–932, 1993.
- [12] Thomas McMurdo and Birdie MacLennan. The vermont digital newspaper project and the national digital newspaper program. *Library Resources & Technical Services*, 57(3):148–163, 2013.
- [13] Thomas Palfray, David Hebert, Stéphane Nicolas, Pierrick Tranouez, and Thierry Paquet. Logical segmentation for article extraction in digitized old newspapers. In *Proceedings of the 2012 ACM symposium on Document engineering*, pages 129–132. ACM, 2012.
- [14] Martin Reynaert. All, and only, the errors: more complete and consistent spelling and ocr-error correction evaluation. In *LREC*, 2008.
- [15] Stephen V Rice. *Measuring the accuracy of page-reading systems*. PhD thesis, University of Nevada, 1996.
- [16] Christoph Ringlstetter, Max Hadersbeck, Klaus U Schulz, and Stoyan Mihov. Text correction using domain dependent bigram models from web crawls. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-2007) Workshop on Analytics for Noisy Unstructured Text Data*, 2007.
- [17] Klaus U Schulz and Stoyan Mihov. Fast string correction with levenshtein automata. *International Journal on Document Analysis and Recognition*, 5(1):67–85, 2002.
- [18] Amarjot Singh, Ketan Bacchuwar, and Akshay Bhasin. A survey of ocr applications. *International Journal of Machine Learning and Computing*, 2(3):314–318, 2012.
- [19] Christian M Strohmaier, Christoph Ringlstetter, Klaus U Schulz, and Stoyan Mihov. Lexical postcorrection of ocr-results: The web as a dynamic secondary dictionary? In *ICDAR*, pages 1133–1137. Citeseer, 2003.
- [20] Kazem Taghva and Eric Stofsky. Ocrspell: an interactive spelling correction system for ocr errors in text. *International Journal on Document Analysis and Recognition*, 3(3):125–137, 2001.
- [21] Xiang Tong and David A Evans. A statistical approach to automatic ocr error correction in context. In *Proceedings of the fourth workshop on very large corpora*, pages 88–100, 1996.
- [22] Andrew J Torget, Rada Mihalcea, Jon Christensen, and Geoff McGhee. Mapping texts: Combining text-mining and geo-visualization to unlock the research potential of historical newspapers. 2011.
- [23] Robert A Wagner and Michael J Fischer. The string-to-string correction problem. *Journal of the ACM (JACM)*, 21(1):168–173, 1974.
- [24] Tze-I Yang, Andrew J Torget, and Rada Mihalcea. Topic modeling on historical newspapers. In *Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 96–104. Association for Computational Linguistics, 2011.