

# An expert system for automatically correcting OCR output

Kazem Taghva, Julie Borsack, and Allen Condit

Information Science Research Institute  
University of Nevada, Las Vegas

## ABSTRACT

This paper describes a new expert system for automatically correcting errors made by optical character recognition (OCR) devices. The system, which we call the post-processing system, is designed to improve the quality of text produced by an OCR device in preparation for subsequent retrieval from an information system.

The system is composed of numerous parts: an information retrieval system, an English dictionary, a domain-specific dictionary, and a collection of algorithms and heuristics designed to correct as many OCR errors as possible. For the remaining errors that cannot be corrected, the system passes them on to a user-level editing program.

This post-processing system can be viewed as part of a larger system that would streamline the steps of taking a document from its hard copy form to its usable electronic form, or it can be considered a stand alone system for OCR error correction.

An earlier version of this system has been used to process approximately 10,000 pages of OCR generated text. Among the OCR errors discovered by this version, about 87% were corrected. We will be implementing numerous new parts of the system, testing this new version, and presenting the results in this paper.

## 1. INTRODUCTION

A common complaint users of OCR technology have is that even with today's best devices, they feel manual correction must still be performed. The cost though of manual correction is high. For anything but a small quantity of text, correction involves a substantial amount of personnel, hardware, organization and management. The costs associated with this kind of task are monetarily high as well as time intensive. In a number of applications, the decision has been to type documents directly rather than OCR the documents and perform subsequent manual correction.<sup>4</sup>

Since text is generally stored electronically for search and retrieval, the critical accuracy rate becomes the rate at which manual correction can be avoided without affecting retrieval results. Studies have shown that 100% accurate text may not be necessary for full text retrieval applications.<sup>3,16,15</sup> Our objective then should be to bring the text to a level with which a retrieval system can cope.

Most commercially available retrieval systems presently rely on the presence or absence of the words to distinguish between relevant and non-relevant documents. Some of the advanced systems, such as SMART<sup>11</sup> and INQUERY,<sup>2</sup> use statistical information regarding word frequencies in the individual documents and the collection to enhance retrieval. The post-processing system we designed was built to improve the quality of text for information retrieval (IR) systems. Its intent is to correct the words used by these systems, i.e., the significant index terms. For example, a typical spell-checker will treat all words of the document equally, with the goal of detecting all unrecognizable terms.<sup>5,9,16</sup> In contrast, this post-processing system only concentrates on correcting terms that will ultimately affect retrieval. The underlying philosophy on which this system is built depends on the assumption that for each significant misrecognized word in the collection, there are correctly recognized occurrences of the same word.

The current version of the system is equipped with a user level interface, special recognizers that handle acronyms and proper nouns. In particular, these terms are dealt with specially. They bypass the first phase of processing and are routed directly to the user interface phase for manual verification. The basis for exempting these terms from the initial phases is: 1) they cannot be corrected by consulting a dictionary and 2) they usually have a higher retrieval value than other index words.<sup>27</sup> This new system also handles inflected word forms to more accurately select the correct substitution. This system has been modified to work in concert with statistical IR systems.

The purpose of the completed system is to automatically correct OCR generated text to an acceptable level of accuracy for use in an IR application. With this system, this level of accuracy can be attained at virtually no cost when compared to manual correction. The details of the post-processing system are described in this paper.

## 2. DESCRIPTION OF THE SYSTEM

What discriminates this post-processing system from corrections that can be made by the OCR device at the time of page recognition, is the use of contextual information found in the complete document as well as the full document collection. Using more global information reflects the methods that might be used by a person when discerning unrecognizable words. The system is composed of several parts:

**an information retrieval system** into which all the text is loaded to facilitate index construction of all significant words. Index information, such as document-term location, allows for straightforward substitution of correct words for erroneous ones.

**special recognizers** for locating acronyms and proper names.

**an English dictionary** for spell checking purposes.

**a domain-specific dictionary**, in our case, a list of words derived from the Licensing Support System (LSS) Thesaurus.<sup>13</sup>

**a discriminator** implementing a heuristic for deciding whether a given string is a word (correctly spelled or misrecognized) or if the string is simply garbage generated by the OCR device.

**an algorithm** for performing the similarity measure between correctly spelled words (centroids) and misrecognized terms.

**a confusion matrix** for resolving ambiguity among centroids equally similar to a single misrecognized term. This confusion matrix contains specific information about the kinds of errors an OCR device is likely to make, and therefore, eliminates centroids where the character transformations are not likely to be made by the device.

**other heuristics** for deciding which correctly spelled word should be substituted for the erroneous one when there are multiple possibilities. These methods use local information (i.e. document level information as opposed to collection information) about the words and suffix information to determine which form of the word should be substituted.

**an optional user-level interface** for entry of the correctly spelled word, when the above heuristics cannot decide with reasonable confidence which of several choices is correct.

Viewing the system on a more technical level, it is composed of some twenty programs, nearly all of which are written in the Perl programming language.<sup>18</sup> Makefiles are used to coordinate individual program execution and control the flow of data through the system. There are five distinct phases of the system which are run sequentially. First, special recognizers are used, followed by pre-clustering, clustering, and post-clustering phases. An optional user-level interface phase completes the post-processing system.

### 2.1. Special recognizers

Objects such as acronyms and proper nouns tend to have a high retrieval value.<sup>7</sup> While correcting occurrences of these items when possible, the post-processing system should also ensure that correct occurrences of these objects are not corrupted. Corruption may occur when an acronym or proper noun does not appear in the dictionary and as a result it is listed as a misspelling. After clustering the misspelled words around the centroids, sometimes a correctly spelled acronym or proper noun is clustered with a centroid and subsequently all occurrences of it are equated to the centroid—resulting in a loss of information. The clustering process is described in detail in Section 2.3.

To identify and exclude these items from consideration, we search the full text of the collection prior to running the rest of the post-processing system. As occurrences of acronyms and proper nouns are identified, they are recorded and excluded from the clustering process if they are not in the dictionary. The result is that while some misspelled occurrences of these items would have otherwise been corrected, correct occurrences are preserved and no longer run the risk of being corrupted.

We define an acronym as a string of three to six uppercase letters, bounded by non-uppercase letters, for which we are able to locate a definition. The full text of the collection is searched for the pattern. Lines containing a

Acronym	Definition
ALARA	As Low As Reasonably Achievable
DOE	Department of Energy
GSC	Geological Survey of Canada
HMTA	Hazardous Materials Transportation Act
INEL	Idaho Nuclear Engineering Laboratory
LANL	Los Alamos National Laboratory
LVDT	linear variable differential transducer
NBMG	Nevada Bureau of Mines and Geology
NEIC	National Earthquake Information Center
OCRWM	Office of Civilian Radioactive Management

**Table 1.** Acronym sample

series of all uppercase words occur frequently in the text, such as titles and headings. Many non acronyms are listed as a result, so an additional heuristic is used. Lines of text having at least two consecutive all lowercase words are considered, while all other lines are excluded. Then for each occurrence of the acronym pattern, we try and locate a definition for the acronym in the vicinity. If we are able to find one, the acronym is recorded; otherwise, it is discarded. See Table 1 for examples of acronyms identified. The procedure for identifying proper nouns uses similar pattern matching and other heuristics.

## 2.2. Pre-clustering

The first step of the pre-clustering phase determines the significant words from the text using an IR system. These *indexed* terms may vary slightly for different IR systems. But in general, the terms indexed are similar. The most typical method for storing these words by an IR system is with an *inverted file*. An inverted file is simply a list of the document's *non-stop* words (stop words are inconsequential words, such as **the**, **this**, etc.) with pointers to the documents in which they are found. The inverted file usually contains other word information such as term frequency and the position of the word within the document. It is this information that is exploited in our post-processing system.

Once the significant words have been determined for the collection and the other word information is in usable form, the next three pieces of the post-processing system (an English dictionary, the domain specific dictionary, and the discriminator) are employed to classify each word into one of three categories: correct words, misspelled words, or useless strings of random characters (what we call "garbage").

Of course the definitiveness of these resulting sets is limited by the completeness of the dictionaries and the heuristics designed to discriminate between real words and "garbage". We use *ispell*<sup>5</sup> for our English dictionary and a manually constructed thesaurus for our domain specific dictionary. This thesaurus was assembled at the time the Licensing Support System Prototype was constructed so the words are directly related to the document collection.

The discriminator was designed based on the qualities of an English word in contrast to the characteristics of certain meaningless strings generated by an OCR device. When certain *non-text* document sections (such as maps, figures, graphs, etc.) are translated by the device to ASCII, the strings are usually senseless and do not resemble real words or even misspellings. We use this contrast to determine which strings should be eliminated from further processing. Four tests are performed. First, if a string is unusually long (in number of characters), it is discarded. Next, if a character appears four consecutive times in a string, that string is discarded. Third, the ratio of alphabetic to non-alphabetic characters is measured. If too many non-alphabetic characters appear, the string is discarded. Finally, of the words that contain nothing but alphabetic characters, if the vowel to consonant ratio is too one-sided, the string is discarded. These simple heuristics work quite well; a sample list of discarded "garbage strings" appears in Table 2.

After garbage strings have been removed, the remaining terms are spell-checked and divided into correctly spelled and misspelled words using the dictionaries described. Correctly spelled words occurring more than once in the collection are labelled *centroids*. All subsequent correction stems from these selected terms. Note that we do not include correctly spelled words occurring only once in the collection. Since they occur only once, we assume their importance to the collection or any particular document is negligible; further, they increase the number of centroids

Garbage string	Reason
ddmetymemmedimdlmldmmmmylor'mm	length
maditswntermdutkmeemmotkzmmmfymmm	length
f3	length
deffiiiioil	repetition
emeeeeeee	repetition
f4141111	repetition
cq243ib80	alphabetic
ef5300	alphabetic
hn3ai25i3012	alphabetic
crlhch	v/c ratio
ddrcrcnt	v/c ratio
hhnntnn	v/c ratio

**Table 2.** Example garbage strings

causing more multiply clustered misspellings. We found from our analysis that removing these words from the centroid list was more beneficial to successful clustering. Therefore, these words were discarded from further processing. In the next phase, the misspelled words are *clustered* around the centroids.

### 2.3. Clustering

The clustering phase approximates the closeness between the centroids and the list of misspellings. Our current similarity measure rests on the principles of *edit distance*. The notion of edit distance can be described as the number of operations performed to make one string identical to another. More specifically, following Wagner and Fisher,<sup>17</sup> let  $\Sigma$  be a finite alphabet. An *edit operation* is a pair of strings  $(a, b) \neq (\epsilon, \epsilon)$  over  $\Sigma$  of length less than or equal to 1. We say string  $x$  results from  $w$ , in notation,  $w \implies x$ , if there are strings  $\alpha$  and  $\beta$  such that  $x = \alpha\beta$  and  $w = \alpha a \beta$ . We say  $(a, b)$  is a *substitution* if  $a \neq \epsilon$  and  $b \neq \epsilon$ , a *delete* operation if  $b = \epsilon$  and  $a \neq \epsilon$ , and an *insertion* operation if  $a = \epsilon$  and  $b \neq \epsilon$ . In the rest of the paper, we use the term edit distance  $n$  to refer to some combination of  $n$  applications of substitution, insertion, or deletion.

Currently, we use *agrep*<sup>19</sup> to determine edit distance. *Agrep* allows the user to specify a number indicating the maximum edit distance between the pattern and the data being searched. When we find a match we say that the misspelled word has been clustered with the centroid. A cluster can be defined as a set of words with a single centroid representing the set. Our goal is to cluster all misspelled words with one and only one centroid. However, in practice, it is quite common for a misspelled word to be clustered with multiple centroids. In other words, the edit distance of a misspelling could be sufficiently close to more than one centroid. Obviously, we cannot substitute multiple centroids for a single occurrence of a misspelling.

### 2.4. Post-clustering

The post-clustering phase of the system is designed to discriminate among centroids and select the correct substitute. There are actually two distinct issues to be dealt with when selecting the appropriate centroid for a misspelling. First, with multiply clustered misspellings, there is more than one centroid that could be correct for that particular term. Also, for separate occurrences of the same misspelling, distinct centroids could be the correct substitution. We have three separate algorithms to deal with these indeterminate sets: local information, confusion matrix, and suffix handling.

First we apply local information. Remember, certain term data are stored in the inverted index. The *localinfo* program uses this information to determine which centroid is more plausibly correct for the clustered misspelling. We found that if we look at the frequency of the centroids in the documents in which the misspelling occurs, we can reduce the number of centroids for that misspelling with a high degree of confidence. Further, this heuristic can correctly select different centroids for the same misspelling, resolving the confusion previously mentioned.

Using an example to clarify, suppose the misspelled word *bjective* occurred in document 1797, and was clustered with centroids *objective* and *subjective*. Further suppose *objective* occurs in document 1797 at least once and *subjective* does not occur at all. Then we decide to eliminate the centroid *subjective* and equate *bjective* to

Misspelled word	Centroids
aapproximate	approximate, approximated, approximates
communicatic	communicate, communicated, communicates, communicating, communication
idemification	identification, identifications

**Table 3.** Example suffix problems

**objective.** At the localinfo level, we eliminate all centroids that do not occur in the document for that misspelling. If we have reduced the number of centroids to one, the misspelling is equated to that centroid. If more than one centroid appears in that document, no resolution is made. The misspelling and the remaining centroids are passed to the next post-clustering process.

The next attempt at eliminating multiple centroids applies information about the kinds of errors an OCR device is likely to make. This information is collected based on the observation that approximate matching of longer words has a high percentage of accuracy. We use misspellings from the device itself to correct other misspellings of shorter character length. Therefore, the confusion matrix built reflects errors made by the device used. Consistent device errors will inherently be a part of the confusion matrix. For more detailed information about the automatic construction of the confusion matrix, see.<sup>14</sup> An example of its ability to discriminate among several centroids follows.

Suppose we have the misspelled word **aration** clustered with centroids **aeration**, **oration**, and **ration**. Considering the errors the OCR device has made, according to the confusion matrix, the device will more likely produce **aration** when the correct word was **aeration**, rather than **oration** or **ration**.

In this way, the system can further reduce the number of clusterings, and in many cases, can select exactly one. Both the local information and confusion matrix methods are quite similar to the versions described in,<sup>14</sup> except that we now generate the confusion matrix automatically within the system, rather than requiring it from the OCR device externally.

The final technique used to eliminate misspellings clustered with multiple centroids involves the consideration of suffixes. In our original post-processing system, we used a text retrieval system which indexed full words, but also used stemming. So ordinarily only one or two forms of a word would become centroids. In the development of our new version of the system, we used INQUERY<sup>2</sup> which gave us the option of using stemming, but if used, it does not index full words. Instead, it stems the words and then indexes them. This was unacceptable for the post-processing system, so stemming was not used. The result was clusterings such as those shown in Table 3. From a text retrieval standpoint, all the centroids are equivalent in clusterings such as these. For these, the post-processing system has identified the correct root word, but not the correct suffix. We believe it is better to try and select one of the centroids, rather than disregard the clustering altogether.

First, the program only considers centroids which are of a sufficient length (6 characters), and checks to make sure they all have the same prefix so that we do not confuse words which do not have the same root. Next, we eliminate all centroids which do not have the same last letter as the misspelled word. If there are still multiple centroids, we determine if there exists exactly one centroid which is closer (in terms of edit distance) to the misspelled word, and pick it if one does exist. As a last resort, if there are still multiple centroids, we simply pick the first one listed which had the same last letter as the misspelled word. This suffix handling procedure is quite effective in dealing with multiple centroids when they are all the same root word.

## 2.5. User-level interface

After all attempts at elimination have been made, there may still be some misspelled words clustered with multiple centroids. At this point, the post-processing system cannot decide which centroid is the correct one and the user has the choice of simply deleting all such clusterings, or sending them to the user-level interface program.

This part of the post-processing system steps through each clustering with multiple centroids and displays a block of text, highlighting the misspelled word. The user can then choose which of the centroids is the correct one, or can optionally type in the correct centroid if none of the listed centroids are correct. The user also has the option of either globally replacing all occurrences of the misspelled word in the collection, globally replacing occurrences in each document, or stepping through all occurrences of the misspelled word and making a choice for each one. Normally it is sufficient to make the change globally throughout the collection, but there are occasions when the misspelled word should be equated to one centroid in one document, and another centroid in another document.

There are even rare occasions when different occurrences of the misspelled word should be equated to two different centroids within the same document, so the default behavior of the user interface is to step through each occurrence of the misspelled word in the collection. Almost always, though, there are only a few occurrences of the misspelled word in the collection so this procedure of manually choosing the correct centroid is not very time-consuming.

Currently, to help the user decide on the correct centroid, only a block of ASCII text is displayed, with the misspelled word highlighted. We are currently improving the user interface and when complete it will also display a portion of the original page image containing the misspelled word. An early beta version is working, having been implemented using the Tool Command Language (Tcl) and Tk, an X Window System toolkit based on Tcl.<sup>8</sup>

### 3. EVALUATION

After all the clustering data has been processed, the result is a list of misspelled words, an optional document number, and a single correct word (we filter out any remaining misspelled words with multiple centroids). The text is then reloaded into a text retrieval system, making the substitutions indicated by the clustering data. If a document number is given for a clustering, then that substitution is only made for the indicated document, otherwise the substitution is made for each occurrence of the misspelled word in the collection. An evaluation of the performance of the system is given in this section.

The magnitude of this project may not be apparent from the count of the documents (674) with which we evaluate our results. But the documents we use are *full text* documents with an average of 9300 non-stop words per document; this yields over 6 million total terms (this figure includes “garbage strings”). The post-processing system clusters 48,500 unique misspellings with 167,000 occurrences. Since different occurrences of the same misspelling might be clustered with different centroids, each occurrence of each misspelling should be checked to determine correctness. In many cases the context of a term should be verified to validate the correctness of the substitution. To examine each misspelling to this detail for 167,000 occurrences would be an extremely tedious task. But some examination of the clusterings and an evaluation of the results are necessary. So we settled on the following two methods to evaluate the soundness of our post processing system.

1. We run a set of queries pertaining to our document collection before and after post-processing.
2. We evaluate manually all *query* term clusterings.

Since the ultimate goal of the post-processing system is to improve the quality of the text for use with an IR system, we felt a fair evaluation would be to judge its worth using the standard IR techniques of *recall* and *precision*. Recall can be defined as the ratio of the number of relevant documents returned to the total number of relevant documents for the query, in the collection. Precision is the ratio of the number of relevant documents returned to the total number of documents returned for a given query.<sup>12</sup> The document-to-query relevance had been manually determined by geology students proficient in the topics covered in this collection. Table 4 shows the improvement in precision at ten standard recall points for the post-processed text over the original *raw* OCR-generated text. This increase in precision is evidence that automatic correction can render notable improvement in retrieval.

Table 4 is a good measure of improvement at the IR level. But closer inspection of the changes made by the post-processing system is also necessary. Since we know that these queries include words that are applicable to our collection, we felt that the words in the queries would be a good representative sample. Although not complete, we feel this term selection is unbiased. There were 300 distinct query terms excluding stop-words. For these 300 terms, 11,730 misspellings were singly clustered with a query term centroid and substituted in the text. Of these replacements, 10,700 were replaced correctly. The post-processing system selected the correct substitutions for 91% for the misspellings it found. These two means of verification show the usefulness of the post-processing system on OCR-generated text.

### 4. CONCLUSION AND FUTURE WORK

The post-processing system described in this paper is an extension of our original post-processing system.<sup>14</sup> Although we did not evaluate this version to the extent to which we evaluated the original version, we believe this post-processing system is superior for the following reasons. As we mentioned, the post-processing system is designed to prepare OCR text for subsequent retrieval. We found that more sophisticated IR systems are more sensitive to

Recall	Precision (% change) – 68 queries		
	raw OCR text before pps	OCR text after pps	
10	53.1	54.6	(+2.8)
20	44.8	48.0	(+7.2)
30	38.9	39.8	(+2.5)
40	33.8	34.7	(+2.6)
50	30.5	30.8	(+1.1)
60	24.6	24.4	(-0.9)
70	17.3	18.0	(+4.2)
80	13.5	15.3	(+13.4)
90	11.1	12.3	(+11.0)
100	10.5	11.4	(+8.1)
average	27.8	28.9	(+4.1)

**Table 4.** Precision at standard recall points

OCR error. This is due to the fact that more knowledge and statistical information is extracted from the text. This phenomenon is apparent in our evaluation of the probabilistic retrieval systems.<sup>15</sup> We also know proper names, acronyms, and other special recognizers (to be added in the future) have proven to be very useful in IR, and thus our system tries to give special attention to these special terms.

We also know that one of the most common errors made by OCR devices is the insertion or deletion of a space.<sup>10</sup> Two routines are presently under implementation to capture these kinds of errors. The former splits a single word into two misspellings while the latter concatenates two words into a single misspelling. Since the distance between these misspellings and their correct forms is usually greater than the edit distance the post-processing system currently detects, these erroneous terms are not caught and corrected.

There are a number of improvements that could be made to the post-processing system. Many though are dependent on the collection used, the required level of accuracy expected, and the way the text will be applied. The point is, post-processing of OCR text can render notable improvement that can make a difference and this improvement can be gotten without manual intervention at little cost.

## REFERENCES

1. Richard C. Angell, George E. Freund, and Peter Willett. Automatic spelling correction using a trigram similarity measure. *Inf. Proc. and Management*, 19(4):255–261, 1983.
2. J. P. Callan, W. B. Croft, and S. M. Harding. The INQUERY retrieval system. In *Proc. 3rd Intl. Conf. on Database and Expert Systems Applications*, pages 78–83, 1992.
3. W. B. Croft, S. Harding, K. Taghva, and J. Borsack. An evaluation of information retrieval accuracy with simulated OCR output. In *Proc. 3rd Symposium on Document Analysis and Information Retrieval*, pages 115–126, Las Vegas, NV, April 1994.
4. Robert P. Futrelle et al. Document analysis, understanding, and knowledge access. In *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR)*, pages 101–111, St. Malo, France, 1991.
5. R. E. Gorin, Pace Willisson, Walt Buehring, Geoff Kuenning, et al. Ispell, a free software package for spell checking files. The UNIX community, 1971. version 2.0.02.
6. Karen Kukich. Techniques for automatically correcting words in text. *ACM Computing Surveys*, 24(4):377–439, December 1992.
7. Michael L. Mauldin. *Information Retrieval by Text Skimming*. Ph.D. dissertation, Carnegie Mellon University, 1989.
8. John K. Ousterhout. *An Introduction to Tcl and Tk*. Addison-Wesley, Reading, MA, 1993.
9. Joseph J. Pollock and Antonio Zamora. Automatic spelling correction in scientific and scholarly text. *Communications of the ACM*, 27(4):358–368, April 1984.
10. Stephen V. Rice, Junichi Kanai, and Thomas A. Nartker. An evaluation of OCR accuracy. Technical Report 93-01, Information Science Research Institute, University of Nevada, Las Vegas, April 1993.

11. G. Salton. *Automatic Text Processing*. Addison-Wesley, Reading, MA, 1989.
12. G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw Hill, New York, 1983.
13. Science Applications Intl. Corp. Capture station simulation: Lessons learned, Final Report, for the Licensing Support System, November 1990.
14. Kazem Taghva, Julie Borsack, Bryan Bullard, and Allen Condit. Post-editing through approximation and global correction. *International Journal of Pattern Recognition and Artificial Intelligence*, 9(6):911–923, 1995.
15. Kazem Taghva, Julie Borsack, and Allen Condit. Results of applying probabilistic IR to OCR text. In *Proc. 17th Intl. ACM/SIGIR Conf. on Research and Development in Information Retrieval*, pages 202–211, Dublin, Ireland, July 1994.
16. Kazem Taghva, Julie Borsack, Allen Condit, and Srinivas Erva. The effects of noisy data on text retrieval. *J. American Soc. for Inf. Sci.*, 45(1):50–58, January 1994.
17. Robert A. Wagner and Michael J. Fischer. The string-to-string correction problem. *Journal of the ACM*, 21(1):168–173, January 1974.
18. Larry Wall and Randal L. Schwartz. *Programming perl*. O'Reilly and Associates, 1990.
19. Sun Wu and Udi Manber. Fast text searching allowing errors. *Communications of the ACM*, 35(10):83–91, October 1992.