The Effects of OCR Error on the Extraction of Private Information

Kazem Taghva, Russell Beckley, and Jeffrey Coombs

Information Science Research Institute,
University of Nevada, Las Vegas
taghva@isri.unlv.edu

Abstract. OCR error has been shown not to affect the average accuracy of text retrieval or text categorization. Recent studies however have indicated that information extraction is significantly degraded by OCR error. We experimented with information extraction software on two collections, one with OCR-ed documents and another with manually-corrected versions of the former. We discovered a significant reduction in accuracy on the OCR text versus the corrected text. The majority of errors were attributable to zoning problems rather than OCR classification errors.

1 Introduction

Studies have shown that OCR error does not significantly degrade the average effectiveness of information retrieval or text categorization. Recent work suggests that the opposite is the case for information extraction.

In this paper, we offer more evidence for significant degradation in information extraction effectiveness on OCR text. In particular, we examine the performance of two information extractors on corrected and OCR text. We found significant degradation in performance for both extractors on noisy text. Unlike earlier studies, we found that the main source of the degradation was zoning errors in texts containing tabular information rather than classification errors.

In section 2 we describe the problems we attempted to solve using information extraction. Section 3 presents some background concerning experiments on text retrieval and text categorization on OCR text as well as previous experiments applying information extraction to OCR text. We describe our extractors in section 4 and our experiments in section 5. Section 6 reports the effectiveness of using a name-finding module in our extractors. Section 7 presents the results of our experiments. In section 8 we classify and quantify the types of OCR errors affecting the extractors.

2 Problem Description

In order to comply with the Freedom of Information Act (FOIA), documents in a U.S. government online repository must not contain private information such as a person's date of birth or an employee's identification number [1].

There are two ways to approach the identification and removal of such information. First, one can view this as a *text categorization* task. That is, one attempts to separate documents (or pages) into two categories, those likely to contain private information and those that do not. When the Information Science Research Institute (ISRI) originally began studying this problem, we took the text categorization approach.

One obvious difficulty with this type of solution is what one does with the document or page categorized as containing private information. The document or page as a whole could be removed from the online collection, but then any information therein which is not private is kept from public view. Such a result violates the spirit of the FOIA which strives to make federal agency records publicly accessible insofar as that information is not otherwise protected by law.

Alternately, the private information could be targeted more narrowly and only those items within a document or page that are private would be located and removed. For example, one could automatically *redact* information considered private by replacing likely private strings with the familiar black rectangles. The task of finding such strings of interest is usually called *information extraction.*

Information extraction (IE) is the automatic discovery of information classes in unstructured text. For our tasks, we wish to identify strings in the classes of birth dates and employee identification numbers. This definition of IE is essentially equivalent to the idea that it is the discovery of structured data in unstructured text since the placement of found items into classes such as "birth date" provide the structure [3, 8]. The information discovered likely will be suitable for storage in a relational database, but this is not a necessary feature of IE.

3 Background

Categorization and extraction work at different levels. The categorization task requires in this case a binary decision of labeling a given document as containing privacy or not. On the other hand, the information extraction task consists of identifying all and only the *instances* of privacy information in a given document.

The differences between the two approaches raise an important question for the processing of OCR texts. ISRI has extensively studied the effectiveness of *text retrieval* on OCR versus clean text [15]. Our testing revealed that average precision and recall are not affected by OCR errors. A similar result was found for *text categorization.* Analyzing the application of the BOW [5] text categorizer on OCR text, ISRI determined that the accuracy of text categorization was not significantly affected by the presence of OCR errors, especially if typical dimensionality reduction techniques are applied [17, 16].

In recent studies it has become apparent that the *information extraction* task differs from text retrieval and categorization in that performance is affected for the worse by OCR errors. Miller et al. [7] noted degradation in the accuracy of their Hidden Markov Model (HMM) information extraction system IdentiFinder. They printed copies of Wall Street Journal articles and then ran experiments on progressively degraded images of these with progressively higher word error rates. The system suffered a 0.6 point loss in F1-measure for each percentage point increase in word error. We had similar difficulties applying an address extracting HMM to noisy texts [14].

Jing et al. [4] address the related problem of summarizing documents containing OCR error. They report degradation in performance at every step in the summarizing process. Errors resulted from sentence tokenizers failing to correctly identify sentence boundaries. Syntactic parsers failed to create correct parse trees for the resulting sentence tokens. Sentence reduction operations failed to remove redundant information and correctly combine sentences sharing co-references.

In this paper we offer more support for the conclusion that OCR error degrade the accuracy of IE systems. Specifically we look at two extraction tasks and test the performance of two extraction programs on both OCR and corrected text versions of the data. Both programs show significant degradation when run on OCR text as opposed to corrected text. However, word error caused by classification errors was not the main source of degradation for our systems. Unlike the collection of newspaper articles from a single source tested by Miller et al, our collection of government documents is less homogeneous. Our collection contains documents and forms which display private information in tables. Such data, after passing through zoning algorithms in the OCR system, were sometimes transformed in ways which created difficulties for IE systems like ours which use proximity information to identify items.

4 System Description

ISRI constructed two information extractors for identifying two types of private information. We were looking for (1) birthdates and (2) employee identification numbers appearing in documents collected by the Department of Energy (DOE).

The extractors used in these tasks look for relations among features in the documents. To detect a particular type of private information, it is useful to identify other types of entities which often appear near that particular type. In some cases, to be considered private information, there must be a relationship between data of different types. For example, a date in isolation is not private information. Nor does it become private if it is identified as a birthdate with an identifier such as *date of birth*. A date becomes private information only when it is correctly associated with a person. The statement *John Dough's date of birth is 5/17/55* is private, but the phrase *someone's birthdate is 5/17/55* is not, unless a referent for *someone* is implied contextually.

In some cases, isolated items such as social security numbers are private. However, relational information still proves useful for their identification because the format of a string alone does not guarantee that it is a social security number. Human beings, like machines, cannot identify a social security number in text unless the context provides some evidence. This is fortunate for machines because it ideally requires human authors to provide clues for the identification of items in the context which can in turn be used by a machine.

A set of textual features, such as patterns identifiable be regular expressions, is defined for each extractor. Often a subset of these features serves as an *anchor set*. Occurrences of anchor features are found first, and a context or "window" is defined surrounding them. This context is then searched for other features. All such features provide evidence for the identification (or lack thereof) of the target information type.

The nature of the anchor set differs for the two extraction tasks. The birthdate extractor anchors are keywords and phrases such as *birthdate* and *date of birth*. In the employee identification number extractor, the anchors are patterns for identification numbers.

We describe each extractor in more detail in sections 4.1 and 4.2 respectively.

4.1 Birthdate Extractor

The date of birth extractor functions as follows. First, it tries to find an anchor pattern, which in this case is a key word pattern such as *born*, *birth date*, *date of birth*, etc. When an anchor pattern is identified, the program searches a window of pre-defined size for strings that match a set of date patterns. If found, the date is output.

Although a birthdate without a name is not considered to be private information, searching for a name is deemed impractical. For one thing, although we require the person's name to be in the context, we do not actually need to identify that name. Furthermore, experiments (reported elsewhere [11]) have suggested that name-finding algorithms do not improve accuracy for this task. Also, the co-occurrence of birthdate phrases with dates is nearly always associated with a name in the text, i.e. the presence of other features allow us to assume with acceptable accuracy the presence of a name.

4.2 Employee Identification Number Extractor

The employee identification number extractor uses Bayesian probability to combine the evidence provided by multiple textual features. In practice this amounts to a summation of points associated with features found in a context, with each occurrence weighted according to its position.

For each feature f we train two values using positive and negative samples respectively: a) the frequency of f's occurrences in contexts of employee numbers ($frequency(f|eid)$), and b) the frequency of f's occurrences outside any such context ($frequency(f|\overline{eid})$). The value of $frequency$ is defined as

$$frequency = \frac{|occurrences|}{|bytes\ of\ text\ sampled|} \tag{1}$$

The *anchor set* of features consists of patterns matching employee identification numbers, e.g. "11987". Non-anchor features include the words *employee*, *identification*, and *number* and their many abbreviations. Also included are the names of various employers. Furthermore, when certain features are found consecutively, e.g. *employee identification*, the juxtaposition is regarded as an additional feature, contributing further evidence and more points.

The set of features is next partitioned into subsets, or "groups", such that if x,y are in the group and x contributes evidence, y does not contribute evidence. For example, if *employee* and *emp* occur in a candidate context, only the occurrence with the highest score will count.

The algorithm first seeks an occurrence of an anchor feature, f_a, then establishes a context around f_a. The variable *test_sum* is set as follows:

$$test\_sum = \log\left(\frac{frequency(f_a|eid)}{frequency(f_a|\overline{eid})}\right) \tag{2}$$

Then we search the context for all other defined features. For each occurrence f in the context, we find the textual distance d to f_a. We use a Poisson distribution to estimate the probability of f occurring once within d bytes of f_a. This probability is determined for the in-context frequency ($frequency(f|eid)$) and for the out-of-context frequency ($frequency(f|\overline{eid})$) The score for the occurrence is the logarithm of the quotient of the two probabilities. In a given context, an occurrence counts if and only if it is the highest scoring occurrence among the features of its group in that context:

$$test\_sum = test\_sum + \log\left(\frac{poisson\_prob(d, frequency(f|eid))}{poisson\_prob(d, frequency(f|\overline{eid}))}\right)$$

For a given candidate context, if *test_sum* exceeds an established threshold, the algorithm returns the text matching the anchor pattern.

The Poisson probability p is defined as:

$$p(y) = \frac{\lambda^y e^{-\lambda}}{y!} \quad (y = 1, 2, 3, \ldots) \tag{3}$$

where λ is the mean number of occurrences within a window [6]. In our case, $y = 1$ always.

As with the birthdate extractor, we tested a version of the employee id extractor with a name-finding tool. If a personal name was identified, this fact increased the probability that the anchor match was an employee identification number. Results of these tests are discussed in section 6.

5 Experiments

In order to test the effectiveness of our extraction programs we constructed four document collections. The documents used for these collections consist of Department of Energy documents being considered for inclusion in an online repository. Reports, e-mails, forms and technical documents are included.

For our date of birth finder, we constructed two collections, one with 745 documents and the other with 1076. In the first, 40 contained at least one occurrence of a date of birth and 705 document had no birthdates. In the 1076 document collection, 76 had at least one birthdate and 1000 documents did not. These are denoted as *dob-745* and *dob-1076.*

For testing the employee identification number extractor, we again created two document collections. The first contained 579 documents, 97 of which contained at least one employee id, and 482 without. The second collection contained 617 documents, 95 with occurrences of an employee id and 522 without. These two collections are called *empid-579* and *empid-617*.

For each of the four collections, corrected versions of the documents containing privacy information were manually prepared. Experts then tagged the privacy items using ISRI's MetaMarker tool [13]. This tagging made it possible to evaluate our extractors on an item by item basis more appropriate to the information extraction task.

We evaluate the results using the standard measures of recall, precision, and F1 but in two ways. First, we computed these measures on a per-document basis. At this level, the effectiveness of our programs as document classifiers is determined. That is, we can test how effective the programs are at answering the question, which document contains at least one example of private information of the given type?

Second, we also evaluate our programs as to how well they identify specific instances of a given privacy type. Thus, we can determine how well our programs identify all and only those instances of a given type.

The standard definitions of precision, recall, and F1 appear below. In each case, the formula can be interpreted as referring to *documents* on one hand, and to *specific occurrences of privacy strings* on the other.

$$recall = \frac{\text{number of privacy items located by program}}{\text{total number of privacy items}} \tag{4}$$

$$precision = \frac{\text{number of privacy items located by program}}{\text{total number of items located by program}} \tag{5}$$

$$\text{F1} = \frac{2(Precision \times Recall)}{(Precision + Recall)}. \tag{6}$$

6 Name Finding

The ISRI name-finding algorithm uses a Hidden Markov Model (HMM) whose states consist of various name formats, such as "`first-name last-name`", "`last-name, first-name`" (the latter "`,`" is part of the pattern), and so forth [10]. Each input term is identified as having various features. Several knowledge bases are included to determine these features, such as a list common last names, male and female first names, a list of common "non-name" words, and common titles including `Mr.`, `Mrs.`, etc. The name lists were derived from the 1990 census list [2]. Another feature identified is the capitalization pattern of the word. Transition probabilities depend on where connecting marks such as commas appear between words.

We found that including a module for personal name extraction did not help the performance of the birthdate extractor [11]. Our tests on the employee identification number extractor produced somewhat mixed results. Figure 1 compares

		Clean			OCR			
	Collection	Recall	Precision	F1	Recall	Precision	F1	Δ_{F1}
Per Document								
with name	empid-579	0.970	0.839	0.900	0.938	0.835	0.884	0.016
without name	empid-579	0.949	0.968	0.958	0.928	0.968	0.947	0.011
with name	empid-617	0.979	0.816	0.890	0.979	0.816	0.890	0.000
without name	empid-617	0.979	0.903	0.940	0.979	0.903	0.939	0.001
Per Item								
with name	empid-579	0.836	0.656	0.735	0.712	0.309	0.431	0.304
without name	empid-579	0.815	0.783	0.799	0.712	0.406	0.517	0.282
with name	empid-617	0.768	0.604	0.677	0.678	0.445	0.541	0.136
without name	empid-617	0.695	0.683	0.689	0.644	0.556	0.597	0.092

Fig. 1. Effectiveness of Name-finding

the effectiveness of the employee number finder both using a name finding module and not using the module. The results for the two collections, the *empid-579* and the *empid-617*, are shown with per-document and per-item counts.

Although the F1 result is lower when the name-finding module is used, recall for the per-item task can be increased with some loss in precision if name-finding is used. For some tasks, the increased recall may be worth the loss in precision, especially if missing examples of privacy items is considered more costly than identifying false positives.

7 OCR Effectiveness

Figure 2 presents the results of experiments using both the birthdate and employee id extraction programs. The column Δ_{F1} showing the change in F1 reveals that F1 diminished in each case, both on a per-document and per-item basis.

It is interesting to compare the per-document and the per-item mean change in F1. Using a one-sided paired-t test to determine how significant these changes

		Clean			OCR			
Task	Collection	Recall	Precision	F1	Recall	Precision	F1	Δ_{F1}
Per-Document								
employee id	empid-617	0.979	0.903	0.940	0.979	0.903	0.940	0.000
employee id	empid-579	0.949	0.968	0.958	0.928	0.968	0.947	0.041
birthdate	dob-745	1.000	0.930	0.964	0.925	0.925	0.925	0.036
birthdate	dob-076	0.974	0.961	0.967	0.869	0.957	0.910	0.057
Per-Item								
employee id	empid-617	0.695	0.683	0.689	0.644	0.556	0.597	0.092
employee id	empid-579	0.815	0.783	0.799	0.712	0.406	0.517	0.282
birthdate	dob-745	0.990	0.850	0.915	0.938	0.835	0.884	0.034
birthdate	dob-1076	0.678	0.779	0.725	0.464	0.534	0.497	0.228

Fig. 2. Performance on OCR versus clean text

were, it turns out that the per-item degradation in F1 from the clean text test to the OCR test was more significant than the per-document (p-value of 0.036 versus 0.067).

The reason for the difference is that on the per-document level, only one example of private information needs to be identified correctly for the document to be correctly categorized. If there are multiple items in a document, and all but one contain OCR errors, the document will still be correctly marked as containing private information. Individual items not correctly identified on the per-item level are each counted as mistakes. Also, the per-document approach is more like a traditional document categorization task as we noted in section 3. That the per-document approach is less affected by OCR error is consistent with earlier studies showing that text categorization performance is not on average negatively affected by OCR error.

8 Analysis of Results

We discovered two types of OCR error affecting the extractors. The first consists of error traditionally called "classification" errors [18]. A classification error occurs if one or more characters in a string are transformed by the OCR software, or characters are added or deleted. For example, one date was changed from *07/20/1954* to *0?/20/1954*. In several cases, dates and employee ids became completely unrecognizable. In addition, some failures of the extractors could be attributable to OCR-errors in keywords. In one instance, *DATE OF BIRTH* became *DATR OF III RTN*. The counts of these errors are presented in the "Classification" column in figure 3.

A second type of error was caused by zoning. In several of the documents there were tables which listed private information. These were sometimes indicated by column headings. So for example, a list of birthdates would be presented under a single column heading "birthdate". However, the OCR engine occasionally would determine that the heading should be zoned separately from the dates. The heading would then be placed farther from the birthdates than it appears to the eye in the document image. Our experts when creating the corrected text versions of documents sometimes made the text more "readable." This resulted in headings being placed closer to items. These types of errors are enumerated in the "Zoning" column in figure 3. We calculate that 37.5% of the errors were classification errors and 63.5% due to zoning.

Collection	Zoning	Classification	Correctable
dob-745	3	2	1
dob-1076	41	28	15
empid-579	6	5	1
empid-617	10	1	1
Total	**60**	**36**	**18**

Fig. 3. OCR Error Types

9 Conclusion and Future Work

Because OCR text can weaken the usefulness of information extraction programs, steps should be taken to improve the OCR output of such texts. However, it is currently unknown what techniques, if any, can be applied to improve the precision and recall of information extraction programs.

The problems caused by zoning errors will obviously have to be approached differently from classification errors. The classification errors may be correctable to a certain extent using aggressive error correction techniques. For example, spelling correction algorithms enhanced with general and collection-specific lexicons have been shown to improve OCR-accuracy [9, 15]. Such techniques have been incorporated into OCRspell [18] and MANICURE [12].

It should be stressed that the "Correctable" column in figure 3 lists the number of potentially correctable errors. These include keywords, birthdates, or employee id's for which some characters survived the OCR-process. It may be the case that no algorithm will actually correct all the errors. However, only truly impossible cases were excluded, generally those in which no trace of the phrase in question survived. It is a question for further research if indeed ocr-correcting algorithms will be able to correct these errors and if so, which types. In future work we hope to study various ameliorative approaches for dealing with both types of error.

In the case of zoning problems, a minimal strategy would be to expand the size of the context searched around anchor features. Also, more aggressive use of page geometry information from the OCR process to identify columnar information may prove helpful.

Another idea would be to abandon the reliance on "anchor" features. In an extractor dependent on anchor features, extraction fails if the anchor features are note recognized. However, if all features which contribute to the location of an item are given weight in the search, then a desired item may be discoverable even if "anchor" features are corrupted.

References

[1] U.S. Government. The freedom of information act 5 U.S.C. sec. 552 as amended in 2002. Url: `http://www.usdoj.gov/oip/foia_updates/Vol_XVII_4/page2.htm`. Viewed June 30, 2004.

[2] U.S. Government. Frequently occurring first names and surnames from the 1990 census. `http://www.census.gov/genealogy/www/freqnames.html`. Viewed August, 2005.

[3] Ralph Grishman. Information extraction: Techniques and challenges. In *SCIE1997*, pages 10–27, 1997.

[4] Hongyan Jing, Daniel Lopresti, and Chilin Shih. Summarizing noisy documents. In *Proceedings of SDIUT'03*, pages 111–119, Greenbelt, MD, April 2003.

[5] Andrew McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. `http://www.cs.cmu.edu/~mccallum/bow`, 1996.

[6] William Mendenhall and Terry Sincich. *Statistics for Engineering and the Sciences.* Prentice Hall, 4th edition, 1995.

[7] David Miller, Sean Boisen, Richard Schwartz, Rebecca Stone, and Ralph Weischedel. Named entity extraction from noisy input: Speech and OCR. In *Proceedings of the Sixth Conference on Applied Natural Languae Processing*, pages 316–324, 2000.

[8] Raymond Mooney and Razvan Bunescu. Mining knowledge from text using information extraction. In *SIGKDD Explorations*, volume 7, pages 3–10, June 2005.

[9] Thomas Nartker, Kazem Taghva, Ron Young, Julie Borsack, and Allen Condit. OCR correction based on document level knowledge. In *Proc. IS&T/SPIE 2003 Intl. Symp. on Electronic Imaging Science and Technology*, volume 5010, pages 103–110, Santa Clara, CA, January 2003.

[10] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals of Speech Recognition.* Prentice Hall, 1993.

[11] Kazem Taghva, Russell Beckley, Jeffrey Coombs, Julie Borsack, Ray Pereda, and Thomas Nartker. Automatic redaction of private information using relational information extraction. In *Proc. IS&T/SPIE 2006 Intl. Symp. on Electronic Imaging Science and Technology".* Submitted.

[12] Kazem Taghva, Julie Borsack, and Tom Nartker. A process flow for realizing high accuracy for ocr text. In *SDIUT 2006.* Forthcoming.

[13] Kazem Taghva and Marc Cartright. An efficient tool for XML data preparation. In *Proc. ISNG 2005 Information Systems: New Generations*, Las Vegas, NV, April 2005.

[14] Kazem Taghva, Jeffrey Coombs, and Ray Pereda. Address extraction using hidden markov models. In *Proc. IS&T/SPIE 2005 Intl. Symp. on Electronic Imaging Science and Technology*, San Jose, CA, January 2005.

[15] Kazem Taghva, Thomas Nartker, and Julie Borsack. Information access in the presence of OCR errors. In *Proc. of ACM Hardcopy Document Processing Workshop*, pages 1–8, Washington, DC, November 2004.

[16] Kazem Taghva, Thomas A. Nartker, and Julie Borsack. Recognize, categorize, and retrieve. In *Proc. of the Symposium on Document Image Understanding Technology*, pages 227–232, Columbia, MD, April 2001. Laboratory for Language and Media Processing, University of Maryland.

[17] Kazem Taghva, Tom Nartker, Julie Borsack, Steve Lumos, Allen Condit, and Ron Young. Evaluating text categorization in the presence of OCR errors. In *Proc. IS&T/SPIE 2001 Intl. Symp. on Electronic Imaging Science and Technology*, pages 68–74, San Jose, CA, January 2001.

[18] Kazem Taghva and Eric Stofsky. Ocrspell: An interactive spelling correction system for OCR errors in text. *Intl. Journal on Document Analysis and Recognition*, 3(3):125–137, March 2001.