STARS

Electronic Theses and Dissertations, 2004-2019

2004

Ocr: A Statistical Model Of Multi-engine Ocr Systems

Mercedes Terre McDonald
University of Central Florida

Part of the Electrical and Computer Engineering Commons

Find similar works at: https://stars.library.ucf.edu/etd

University of Central Florida Libraries http://library.ucf.edu

Showcase of Text, Archives, Research & Scholarship

A STATISTICAL MODEL OF MULTI-ENGINE OCR SYSTEMS

by

MERCEDES TERRE ROGERS
B.S. University of Central Florida, 2000

A thesis submitted in partial fulfillment of the requirements
for the degree of Master of Science
in the Department of Electrical and Computer Engineering
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Summer Term
2004

ABSTRACT

This thesis is a benchmark performed on three commercial Optical Character Recognition (*OCR*) engines. The purpose of this benchmark is to characterize the performance of the *OCR* engines with emphasis on the correlation of errors between each engine. The benchmarks are performed for the evaluation of the effect of a multi-*OCR* system employing a voting scheme to increase overall recognition accuracy. This is desirable since currently *OCR* systems are still unable to recognize characters with 100% accuracy. The existing error rates of *OCR* engines pose a major problem for applications where a single error can possibly effect significant outcomes, such as in legal applications.

The results obtained from this benchmark are the primary determining factor in the decision of implementing a voting scheme. The experiment performed displayed a very high accuracy rate for each of these commercial *OCR* engines. The average accuracy rate found for each engine was near 99.5% based on a less than 6,000 word document. While these error rates are very low, the goal is 100% accuracy in legal applications. Based on the work in this thesis, it has been determined that a simple voting scheme will help to improve the accuracy rate.

TABLE OF CONTENTS

LIST OF FIGURES

v

LIST OF TABLES

LIST OF ABBREVIATIONS

OCR   Optical Character Recognition

ASCII   The American Standard Code for Information Interchange

TIFF   Tag Image File Format

RTF   Rich Text Format

PDF   Portable Document Format

HTML   HyperText Markup Language

SGML   Standard Generalized Markup Language

dpi   Dots Per Inch

WYSIWYG   What You See Is What You Get

DOC   Word Document

XML   Extensible Markup Language

MS Word   MicroSoft Word

CSV   Comma Separated Variable

TXT   Text

DBF   DataBase Format

CHAPTER ONE: INTRODUCTION

<u>OCR</u>

It seems as though the times have changed and for the over the last decade everything has been done electronically. While this may be the truth, it does not hide the fact that over 90% of the information available today is still only available in hard copy form. This truth can pose a problem for industries that require documents to be analyzed.

Optical Character Recognition, *OCR*, is a process that transforms a bitmapped image of printed text into text code, and thereby making it machine-readable.

Digitization refers to the process of converting a paper, or hard copy document, into electronic form. This technology is relatively new and has been developed to expedite the electronic re-keying of documents that are only available in hard copy form. *OCR* is an optional part of this process. Converting the document into text code and making it machine-readable is the idea behind Knowledge Management.

The cost of entering data is the dominant life-cycle cost for Electronic Document Management Systems. *OCR* reduces the cost of converting data from hardcopy form into electronic form with the other option being manual re-keying. Technology advances continue to decrease this life cycle cost while making the use of *OCR* increasingly cost effective.

Today's demand for document-management systems has increased and become popular in offices because these systems allow for rapid retrieval and reuse of documents. Although new technology advances continue to make way to industry at a very rapid pace, the simple truth still holds that *OCR* is unable to recognize characters with 100% accuracy.

OCR Objective

The objective of *OCR*, or pattern recognition, is to interpret input as a sequence of characters taken from a given set of characters. This set of characters can be from any one of the many languages found around the world, hand written or even machine produced. Characters function as media transmitting information based on the agreement of the community. The question is how are characters perceived by the human mind? And how do we transmit this same understanding into a machine-based software program? Human beings can easily recognize characters hand produced and written with an abstract reproduction of every character familiar to their personal knowledge base. This abstraction must be so basic that any variation can be recognized. This simple fact makes it obvious that the character sets memorized by the human brain are not just simple physical images (see figure 1).

Characters are not just simple physical images, or perceived by the community as rigid or mechanical in their representations of the shapes. The abstract reproduction of characters as perceived by the human mind is difficult to explain and machine replicate. An abstracted shape is like a symbol. As the shape is matched or interpreted, there are absorbed changes in size and position, rotation and so on. Shapes are abstract but the human mind is very flexible and able to

make the abstraction. As shown below in Figure 1, the human being does not read character by character but he/she deciphers a thought word by word. Today, *OCR* software can recognize a wide variety of fonts within a wide variety of languages, but handwriting and script fonts the mimic handwriting are still problematic. If hand written, the figure below would be absolutely impossible for any *OCR* software to recognize as anything which makes since. First, the *OCR* engine would have a hard time recognizing the handwritten characters. Second, the engine would not recognize 36 words out of this phrase consisting of 69 words, 47.83 %.

Aoccdrnig to a rscheearch at Cmabrigde Uinervtisy, it deosn't mttaer in waht oredr the ltteers in a wrod are, the olny iprmoetnt tihng is taht the frist and lsat ltteer be at the rghit pclae. The rset can be a total mses and you can sitll raed it wouthit porbelm. Tihs is bcuseae the huamn mnid deos not raed ervey lteter by istlef, but the wrod as a wlohe.

Figure 1: Example of Human Intelligence

In the case of printed characters, shapes vary in stroke, ornament and so on. In the case of human produced or hand printed characters, there are many more nonlinear transformations.

Applications

As mentioned earlier, knowledge management continues to reap the benefits of this

advancing technology. Knowledge management includes electronic document management, publishing, and control of forms. The idea behind the conversion from hardcopy form into electronic form is to make the information readily available. These electronic images can be indexed and stored in a document storage system or database which gives users the ability to search through millions of documents in a matter of moments. Electronic data can then be distributed via a company's Intranet or even via the Internet.

Optical Character Recognition is a technology long used by libraries and government agencies to make documents readily available electronically. Advances in this technology have stimulated use by other organizations and have also increased the demand for *OCR*. Not only is *OCR* the most cost-effective and speedy method available for document input-tasks, but *OCR* also frees acres of storage space once given to file cabinets and boxes full of paper documents [1].

The OCR Process

Before *OCR* can be used, the source material must be scanned using a scanner to read in the page(s) as a bitmapped image or a pattern of dots. Once the document is stored in the computer as an image, the *OCR* software then processes the scan to differentiate between images and text and this software also determines what letters are represented in the light and dark areas. Older *OCR* engines match these scanned images against stored bitmaps based on specific fonts. The finishing results of these older *OCR* engines helped to establish *OCR*'s reputation for inaccuracy because of this hit-or-miss approach taken to recognize the bitmap.

Today, multiple algorithms of neural network technology are used to help analyze the stroke edge, the line of discontinuity between the text characters and the background. *OCR* engines and the algorithms supporting the technology now allow for irregularities of printed ink on paper and average the light and dark along the side of a character stroke matching it to known characters. Once this is complete, the *OCR* software makes a 'best guess' as to which character it is and polls the results from all the algorithms available to obtain a single reading.

There are five discrete processes subsequent to image capture involved in Optical Character Recognition [3]:

1. Identification of text and image blocks: Most software engines use white space to try to recognize the text in appropriate order. Complex formatting can cause problems within the *OCR* process. This can include formatting such as cross-column headings or tables that should be manually delineated by "zoning" prior to *OCR*. Zoning is the identifying and numbering of text blocks. Images interspersed throughout the text will usually be ignored by the *OCR* software and dropped from simple output formats such as ASCII.

2. Character recognition: The most common method of character recognition identifies a character by analyzing its shape and comparing its features against a set of rules that distinguish each character/font. This method is called "feature extraction".

3. Word identification/recognition: After step two, character recognition, is complete, character strings are then compared with dictionaries appropriate to the language of the original document *OCR*'ed.

4. Correction: Once the TIFF image is recognized, the *OCR* output is stored in a proprietary file format specific to the *OCR* software used. The software highlights non-recognized

characters or suspicious strings. It is then the responsibility of the operator to input the corrections.

5. Formatting output: Once the character recognition process is complete, the post-*OCR* process converts the file into one or more of the output formats offered by the software. These output file types can include ASCII, Word, RTF, other file formats, and in the case of Adobe Capture, PDF.

Weighing the Need for Machine Readability

When a document is scanned and displayed onto the computer monitor screen with clearly legible text, it appears to the human being as a readable document, however to the computer, the image is not "readable". The image cannot be "understood" as anything other than dots. *OCR* will make the text machine-readable at which time the computer can "read" the document following any constraints or operations intended from the operator. The objective of *OCR* is to make people more productive. This goal is achieved by reducing the time it takes to translate printed text or image files into editable text. Some common time consuming activities include: manually defining page layout; assigning text, graphic and/or numeric zones; proofing recognition errors and reformatting documents after export [5]. *OCR* is designed to help make people more productive and to maximize throughput. *OCR* should be considered if:

- The text is to be reused, edited, or reformatted;
- The text should be available for full-text information retrieval, i.e. for internet search engines;

- The text is to be coded in HTML or SGML;

- The text should be available to adaptive equipment for the visually impaired;

- File size is of concern (in terms of storage or bandwidth to transmit);

- Resources are available to perform *OCR* and correct the output.

In short, the decision to run *OCR* on a document should be based on the projected use of the document. [3]

When a document is first scanned into the computer the document is filed as a TIFF image. Below is a table that compares image and *OCR* characteristics:

Table 1: Image & OCR Characteristics

Characteristic	Bitmapped Image	OCR text
Size of file	Large	Fraction (e.g., <5%) of size of bitmapped image.
Original formatting	Retained	Lost if ASCII; retained in part or entirely with some other output formats.
Editing and reformatting	Not possible	Possible
Information retrieval (indexing)	Not possible	Possible
Reproduction accuracy	Provides a facsimile of the original. It can be manipulated (e.g. sharpened) and provides a true representation of the appearance of the original.	Not a true representation of original. Processing can affect both text and format accuracy.

After a document has been identified as requiring the capability to be machine-read, there are more considerations that need to be weighed. Is the material within the document in question conducive to the *OCR* process? As mentioned earlier, the regeneration of the document by manually re-keying the text into the computer can be quite costly. This alternative overhead cost

should be compared to both the *OCR* accuracy rate and the throughput rate for the entire process.

An accuracy rate exceeding 98% is often cited as necessary for document conversion (*OCR*) to be more efficient than re-keying [3]. The number of edits required expressed as a percentage of the number of characters in the image determines the accuracy rate. Edits required can include insertions, deletions and substitutions. High accuracy rates have continually proven difficult to achieve for certain types of media. This includes media types such as catalogue cards, multi-language texts, historical items with faded type or unusual fonts, handwritten documents and/or letters. Accuracy can be affected by a number of factors:

- Hardware and software variables: including scanner quality, recognition method and algorithm, number and sophistication of font and word glossaries.

- Scan resolution: The number of dots per inch, or the resolution, can affect the clarity of the image and accuracy of *OCR*. It has been shown that reducing from 300 dpi to 200 dpi increased the *OCR* error rate for a complex document by 75%. However it has also been shown that increasing the dpi from 300 to 400 has negligible impact on *OCR* accuracy.

- Generation of original: The generation of the document can affect the Accuracy rate. Second generation scans will reflect quality factors that affected the first generation copy. Second generation scans can include photocopies or microforms. These factors can include resolution, condition, accuracy, completeness and legibility.

- Binding of original: Inadequate margins will distort text on a typical flatbed scanner. Book cradle-scanners ensure better image capture while preserving bindings.

- Paper quality/typeface clarity: Filled or touching characters stemming from excess ink or paper degradation can prevent the characters from being recognized. Also broken

characters resulting from pale type may cause non-recognition. Stains or marks on the document may be captured on the scanned bitmapped image and *OCR* may try to interpret these paper irregularities as characters. An example of this might be specks on the original document mistaken as accents on the *OCR*'ed duplicate.

- Typographical and formatting complexities: Variations in typeface or font size, for instance bold or italics, may be lost or introduced and/or result in "misunderstood" characters. The software may not recognize mathematical symbols and other unusual fonts or characters. Standard *OCR* software does not recognize handwriting. Formatting can also cause recognition problems. Cross-column headings, tables, indented text, footnotes, headers, text wrapped around images, and margin notes can all present problems to the resulting text unless the scanned image has been zoned.

- Linguistic complexities: Problems can arise when more than one language dictionary has been loaded. Character sets can be mixed and character sets of certain languages might not be supported.

When choosing *OCR* software, there are different selection criteria that should be considered. The criteria can include the *OCR* recognition method, recognition speed, font glossaries, output formats supported, languages supported, sophistication of dictionaries, advanced features such as spell checkers and WYSIWYG editors, and price. Chapter 2 will briefly discuss the three software engines used for this benchmarking effort and the advantages and/or disadvantages of each.

Problem Statement

The purpose of this experiment is to understand the available uplift of creating a voting scheme using three *OCR* engines verses one to recognize a document image. The three engines will be analyzed and benchmarked to see the relative position of the software on the market. Some expect to see results in the low 80th percentile while others contend to see results in the low 90th percentile. [Doculex]. The results obtained from the benchmark will ultimately be the determining factor in the decision of implementing a voting scheme or not. The lower the base accuracy the greater the opportunity for improvement by a realized scheme. Several software houses have used the voting scheme. [Doculex]. But for the most part, this approach has been abandoned.

The results from the first part of this experiment will be the benchmark. If the uplift opportunity is great (i.e. >6 percentage points) then a reasonable explanation of the statistical approach to be taken will be provided. If the recommendation provides the go-ahead for a rationalization scheme, further experimental testing will be realized in the future.

Who's Done What

Advances are being made to recognize characters based on the context of the word in which they appear, as with the Predictive Optical Word Recognition algorithm from Peabody, Mass.-based ScanSoft Inc [1]. The next step for *OCR* software recognition is to use knowledge of the parts of speech and grammar to recognize individual characters. This is called document

recognition. Developers are taking different approaches to improve script and handwriting recognition. For example, *OCR* software from ExperVision Inc. in Fremont, Calif. first identifies the font and then runs its character-recognition algorithms [1].

In support of the goal of automatically selecting methods of enhancing an image to improve the accuracy of *OCR* on that image, [1] considered the problem of determining whether to apply each of a set of methods as a supervised classification problem for machine learning. Each image was characterized according to a combination of two sets of measures. One of the measures reflect the degree of particular types of noise present in documents in a single font of Roman or similar script. The other measure used to characterize an image was a more general set based on connected component statistics. Several potential methods of image improvement where considered each constituting its own 2-class classification problem. The problem was classified according to whether transforming the image with the selected method improved *OCR* accuracy. The experiment results varied for the different image transformation methods, but the system made the correct choice in 77% of the cases in which the decision affected the *OCR* score by at least .01. The correct choice was made 64% of the time overall.

[3] Offers a perspective on the performance of current *OCR* systems by illustrating actual *OCR* errors made by three commercial devices. This paper presents illustrated examples of recognition errors. The main causes of errors found in this experiment are errors caused from Imaging Defects, Similar Symbols, Punctuation, and Typography. The analysis done for this research was performed on examples drawn from large-scale tests conducted by the authors at the Information Science Research Institute of the University of Nevada, Las Vegas.

Procedures Followed

Three commercial *OCR* engines were benchmarked for the purpose of this experiment. The engines obtained for the benchmark were ABBYY FineReader *OCR* 7.0, Doculex OCR-It, and OmniPage Pro 12. In testing *OCR* accuracy, there are certain guidelines and key steps that should be followed.

- It is important to compare similar *OCR* products.

- Many pages should be tested because accuracy can vary greatly based on the type of page and settings selected.

- A wide variety of pages should be tested, i.e. newspaper articles, magazine pages, laser prints, photocopies, spreadsheets, and books. (For the purposes of this experiment, only one word document applied with several different settings was analyzed.)

- The scanner settings should be selected and remain the same throughout testing. Settings to consider are resolution and depth to name a couple.

- For a fair comparison, it is recommended to use a third-party scanning software to scan in paper documents and to save each document as an uncompressed TIFF file. It will be then possible to import the same image into each of the different *OCR* engines rather than scanning in the document at separate occasions producing unequal effects on accuracy.

- The next step is to perform the *OCR* process. This should be done automatically for each of the engines using the default or original settings provided. Manual zoning can improve *OCR* accuracy, but automatic processing should be selected. This will ensure a valid comparison of different *OCR* applications is possible.

- The documents should be saved for each of the engines and then compared for word and formatting errors. *OCR* errors can be categorized in two different ways:

  1. Character errors – every incorrect character or punctuation mark is classified as an error
  2. Word errors – regardless of how many mistakes occurred within the incorrect word, each incorrect word counts as just one error

- Formatting errors are difficult to count objectively and for the purpose of this experiment, were not considered.

There are four steps to input a document.

  1. Scanning
  2. Reading
  3. Spell-checking
  4. Saving

Once the scanning is complete, the document is saved as an image. The image is then imported into the desired *OCR* application and read – or recognized by the engine. The option is then available to spell-check the errors the *OCR* engine has identified. After spell-check is complete, the document can be saved in one of several different format types depending on what is available through the *OCR* engine utilized.

CHAPTER TWO: OCR ENGINES BENCHMARKED

ABBYY FineReader OCR 7.0

"ABBYY Fine Reader is an Optical Character Recognition (*OCR*) system that helps convert printed and PDF documents into editable formats while retaining the original layout of the document. The program allows users to create a digital copy of any document in minutes without manually retyping it. Although incredibly easy to use, ABBYY FineReader also provides more sophisticated settings and options to meet the needs of professional users who want to fine-tune the application to suit their needs." [ABBYY user guide, pg7].

ABBYY FineReader's recognition process is based on ABBYY's IPA Technology. The IPA perception is based on three principles used to determine the behavior of the system. The first principle, Integrity, is the identification of recognition objects based on a set of basic elements and their interrelations. The second principle Purposefulness is the generation and purposeful verification of a recognition hypothesis. Adaptability is the third principle from which ABBYY's IPA recognition technology is based on and this is the system's ability to learn and be trained.

After a document has been scanned and an image is acquired, FineReader analyzes the image file and recognizes each character. The layout analysis involves selecting the recognition

areas, tables, pictures, lines, and individual characters. FineReader layout analysis is more

accurate when the nature of the test is known. The image reading process is closely related to

the page layout analysis.

ABBYY FineReader recognition system generates a hypothesis about a recognition

object and then accepts or rejects the hypothesis according to whether the structural elements are

present. A recognition object is a character, part of a character or several glued characters. The

structural elements can be arcs, circles, dots, etc., or they actually represent computer equivalents

of character parts crucial for human perception arcs. The FineReader application then adapts

itself to the text according to the degree of accuracy attained. Once recognition is complete, the

user can edit and save the recognized text in any convenient format. The supported document

saving formats for ABBYY FineReader are:

- Microsoft Word Document (*.DOC)

- Rich Text Format (*.RTF)

- Microsoft Word XML Document (*.XML) (MS Word 2003 only)

- Adobe Acrobat Format (*.PDF)

- Hypertext Markup Language HTML

- Microsoft PowerPoint Format (*.PPT)

- Comma Separated Values (*.CSV)

- Plain Text (*.TXT). FineReader supports various code pages (Windows, DOC, Mac, ISO) and Unicode encoding

- Microsoft Excel Spreadsheet (*.XLS)

- Database Format (*.DBF)

Doculex OCR-It

Doculex provides another Optical Character Recognition Engine that they claim accommodates the demand for high-speed, reliable production conversion of captured document images into machine-readable text. The product offered by Doculex, OCR-It can be run as a stand-alone product or can also be used as a distributed component. If used as a distributed component, OCR-It will work with batches generated with Doculex Capture. Doculex offers many features available with the OCR-It package, including [Doculex web-site]:

- Searchable text from scanned images

- Enhanced *OCR* engine, exporting to RTF, TXT, delimited text and others

- Allows the user to strip away noise characters and excessive white space

- Allows conversion of selected images based on bar code target, mark sense or function key field prompting

- Utilize converted text according to document groupings

- Stores *OCR* data with other image information collected during scanning

- Detects page orientation

- Allows the user to recognize many prominent world languages including:

    1. English

    2. American (Grabar)

    3. American (Western)

    4. Bulgarian

    5. Catalan

6. Croatian

7. Czech

8. Danish

9. Dutch

- Enables text validation

- Option to eliminate white space

- Choose from several single and multi-page TIFF file import methods

  1. PageName = DocName = FileName

  2. DocName = SubFolderName

  3. DocName = FileName Prefix

  4. Incremental Naming

  5. User-Created Import File

  6. Convert MultiPage Images

- Export to ASCII, RTF, TXT, delimited text, and to information management systems including Summation, Concordance, Ringtail, and Hummingbird

- Incorporate user defined dictionaries for case specific language

Once the media of interest is scanned and stored as a single page Group IV TIFF file, OCR-It can extract the text from the scanned image. The Expervision Character Recognition engine, orientation detection, and spelling check/correction are the tools required by the OCR-It to create the text data. Doculex Batches, used to store a list of image pointers for a single batch, may be created by scanning pages with a Doculex Capture Program or by importing

existing Group IV TIFF files into the program itself. Doculex Batches are a set of database

tables that contain a unique identifier for each page and also categorize individual pages into

documents. These tables or batches, contain information about scanned images including the

following [Doculex]:

- Document and page numbers

- Page number

- Directory and filename

- Batch label

- Annotations

- Index Fields

- Bookmark Records (Coding Pages) – place-mark items created during scanning

- *OCR* Text-Optical Character Recognition data gathered while creating PDF files, the data
    is reusable

OmniPage Pro 12

OmniPage Pro is the third commercial *OCR* engine used for the benchmark. After a

document is scanned and an image is obtained, OmniPage Pro analyzes the character shapes in

an image during the *OCR* process and defines solutions to produce editable text. After the *OCR*

process is complete, OmniPage Pro also offers the capability to save the resulting text to a

variety of word-processing, desktop publishing or spreadsheet application.

Throughout the *OCR* process, OmniPage Pro offers additional capabilities including of

course, text recognition. The following elements are retained throughout the *OCR* process performed by OmniPage Pro:

- Graphics – photos, logos, and drawings

- Text Formatting – font types, sizes and styles, i.e. **bold,** *italic,* <u>underlines,</u> indents, margins, and line spacing

- Page Formatting – column structure, table formats, and placement of graphics and headings

OmniPage Pro handles documents in two main ways:

1. Automatic processing

2. Manual processing

The basic steps for both of the above mentioned processing methods are relatively the same. Once a document image is acquired and has been imported into the OmniPage Pro application, *OCR* is performed to generate editable text. During the *OCR* process, this engine creates zones around elements on each page. The engine will interpret the text characters or graphics within each zone. It is possible to manual and/or template zone if so desired. Once *OCR* is complete, the user can check for errors and correct the document using the *OCR* Proofreader within the Text Editor.

OmniPage Pro offers document export capabilities as well. The user can save the document to a specified file name and type, place the document on the clipboard, or send the document as a mail attachment. The file types supported for saving recognition results include but are not limited to the following:

- Microsoft Word Document (*.DOC)

- Rich Text Format (*.RTF)

- Microsoft Word XML Document (*.XML)

- Adobe Acrobat Format (*.PDF)

- Hypertext Markup Language HTML

- Microsoft PowerPoint Format '97 (*.RTF)

- Comma Separated Values (*.CSV)

- Plain Text (*.TXT)

- Microsoft Excel Spreadsheet (*.XLS)

CHAPTE THREE: RESEARCH DESIGN AND METHODOLOGY

Experiment to be Done

The process to perform this benchmark was fairly straightforward. As mentioned in the previous section, it is important to compare each *OCR* engine using the default settings so that the results are fair.   Manually changing the settings would create an unfair advantage not allowing the remaining engines to be as equally competitive during the character recognition stage.

The benchmark performed created a method for grading *OCR* performance.   The idea is to create an image set of practical and impractical documents. For the purposes of this research and experiment, a simple 10-page document was manipulated several different ways. Font size was varied, as was the font type. The manipulations were done in such a way to spot the points at which each engine would fail. Below is the experiment set up as followed with the different character manipulations (size and shape) performed.

Experiment Set Up

- 1st set of experiments – Completely electronic - ~10 pg CLEAN document – Word document converted to TIFF image file to be imported into *OCR* applications.

- The experiment will be done in 4 sets of font styles:

1. Times New Roman

2. Arial

3. Courier

4. Garamond

- For each of the above font types, the following manipulations will be performed:

  o 12 pt font size

    1st set – original, no induced errors

    2nd set – Bold font type

    3rd set – Italic type font

  o 8 pt font size

    1st set – original, no induced errors

    2nd set – Bold font type

    3rd set – Italic type font

  o 16 pt font size

    1st set – original, no induced errors

    2nd set – Bold font type

    3rd set – Italic type font

- 108 output .txt files will be created from the *OCR* engines

<u>Why Benchmark Needed</u>

The purpose of this benchmark is to determine whether or not a scheme utilizing three

different commercial *OCR* engines would be beneficial for character recognition, most

specifically for the Law Community. Legal representatives are often provided hundreds of hard-

copy documents and given only a limited amount of time to search for needed content and

evidence. *OCR* technology has provided a capability much needed in the legal industries for data

entry but still the imaging system life cycle costs are high. Despite the *OCR* progress the costs

of correcting errors from recognized images still remains the biggest portion of the imaging

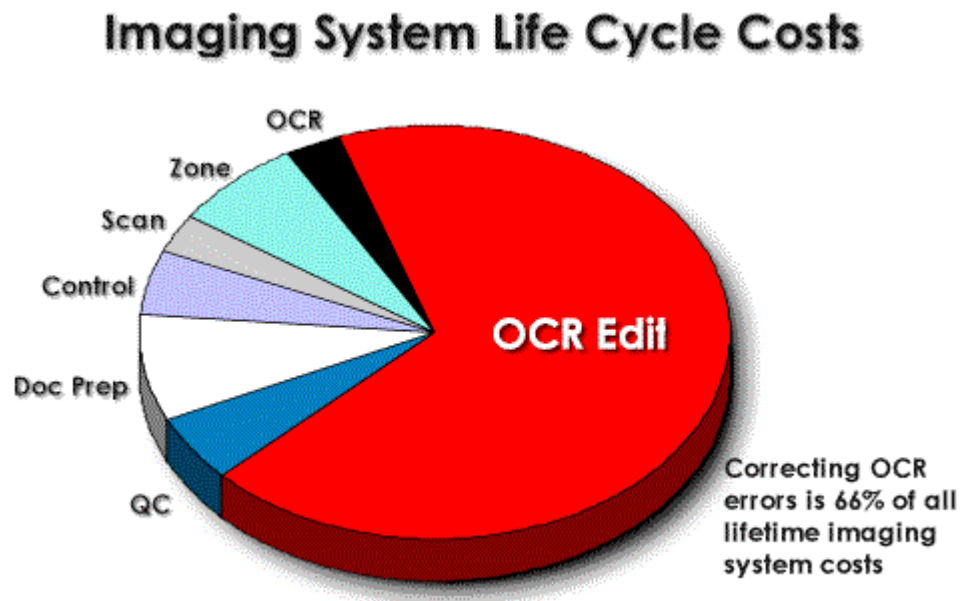system cost. Shown below is a typical *OCR* project cost structure. [6].

Figure 2: Imaging System Life Cycle Costs

A voting scheme utilizing three commercial *OCR* engines to recognize a document image

has the potential to increase the accuracy of error detection and therefore will lower the number

of errors generated. Increasing the *OCR* accuracy rate will lower the time needed for *OCR* Edit

and will therefore directly lower the *OCR* error correction cost. This increase in accuracy will

also reduce the amount of person-hours required to spend on data correction and for the legal

industry, this means more time available to identify evidence within the *OCR*'ed text.

As stated by [6], "A high accuracy *OCR* engine, particularly one which is based on

voting engines, has a lot more data to use to decide whether to mark a character as suspicious,

and hence is much better at marking its errors as suspicious." Because the time allotted for data

correction in the legal community is minimal, the amount of errors retained within the *OCR*'ed

document is relatively high and therefore data search will not be optimal. A high accuracy *OCR*

engine similar to that mentioned above will generate much fewer errors to start with and

therefore the *OCR*'ed document will require less time for data correction.

Manual *OCR* error correction is the process of reviewing the *OCR* results and correcting

mistakes. This process is time consuming and will not guarantee that all mistakes are corrected.

An example of a difficult error to catch is the *OCR* engine replacing the 'm' with the two letters

'r' and 'n' next to each other, or 'rn'. As shown in figure 3 , this type of error could easily be

missed by human interpretation and unless the editor is reviewing the document text very slowly

and meticulously, while incurring high overhead labor costs, the error will most likely be missed.

Commercial → Cornmercial

OR

Automobile → Autornobile

Figure 3: m Versus rn

Automated and manual *OCR* error correction can only find and fix a fraction of the errors that are created in recognizing images through *OCR* [6]. This statement would conclude that the cleaner the *OCR*'ed output data is from the start, the better. In the legal community where time does not permit for automated or manual *OCR* error correction, there is no substitute for accurate *OCR*. Implementing a voting scheme utilizing three commercial *OCR* engines with an increase in accuracy will provide improved capabilities, two-fold, to the legal industries.

CHAPTER FOUR: RESULTS

The manipulations performed displayed a high character recognition rate the majority of the time. Shown below are sample tables of calculations found. The accuracy is expressed as a percentage of errors found based on the 10 page document with 5843 words, 32690 characters with no spaces, and 38566 characters with spaces.   The first table displays the average accuracy rate found for each of the three *OCR* engines. The accuracy rate is very high for each engine but there is still room for improvement, and as mentioned previously – within legal applications, any amount of improvement is priceless.

Table 2: Average Overall Accuracy (%) for Each Engine

	Average Accuracy Found
Abbyy	99.67054595
Doculex	99.73900394
Omni	99.57071139

The table above is the overall accuracy rate for each engine expressed as a percentage. The rate for each engine was calculated using the accuracy percentages from each of the different manipulations performed. During the error recognition for each of the manipulations, a percentage was determined using each character error weighed against the amount of words in

the document examined. Some examples of the percentage of errors found weighed against the number of words for different manipulations, i.e. Garamond font size 16 normal print, from this 'perfect' 10-page document are shown below.

Table 3: Garamond Font Type, 16 Point

Manipulation		Garamond 16 Norm	
# of Errors	25	13	7
Accuracy %	99.5721376	99.77751155	99.88019853
OCR Engine	Omni	Doculex	Abbyy

As shown in the above table, the 'perfect' 10-page document was reprinted normal, versus italic or bold, in the font style Garamond using a large print of 16 point. Once the document was manipulated in this particular format, the appropriate steps were taken to produce a TIFF image that each of the engines could perform character recognition on. The number of errors found versus the number of words was relatively small producing a high accuracy rate for each engine. This accuracy rate is less than perfect and will continue to decrease as the document examined becomes larger and more irregular.

Table 4: Garamond Font Type, 12 Point

Manipulation	Garamond 12 Norm		
# of Errors	57	1	11
Accuracy %	99.02447373	99.9828855	99.81174054
OCR Engine	Omni	Doculex	Abbyy

Table 4 is the accuracy rate for each of the engines determined using the perfect 10-page document manipulated just as the previous table but with a smaller font size. As shown in the variations of accuracies, each of the engines will have a different failure point. It is this variation that supports the idea and decision to create a voting scheme utilizing three different engines.

Table 5: Courier Font Type, 12 Point Bold

Manipulation	Courier 12 Bold		
# of Errors	30	22	17
Accuracy %	99.48656512	99.62348109	99.70905357
OCR Engine	Omni	Doculex	Abbyy

Table 5 displays the accuracy for each of the engines calculated using the 10-page document with bold Courier font style, size 12 point.

Because of time constraints, the 10-page document was broken down into a 'perfect' 5-page document and the same style manipulations were performed to determine *OCR* engine

accuracies. Below are the results of the testing done with the 5-page document.

Table 6: Examples of Errors Found

Types of Errors:	Type 1	Type 2	Type 3	Type 4: Only 1 engine displayed error
Original - Correct	chief elected	war-laden	April 3, 1959	
Omni	chiefelected	warladen	April 3,1959	
Doculex	chief elected	war- laden	April 3, 1959	
Abbyy	chiefelected	war-laden	April 3,1959	
Outcome w/ voting	**CORRECT**	**CORRECT**	**INCORRECT**	**CORRECT**

Table 6 is an example of the different types of errors found. The document scanned was a 5-page 'perfect' document manipulated in different font types. This ideal, unrealistic document created common and specific error types. The 4 types of error shown above were the most common found throughout the experiment. Omni often left a space out of two words, inserting an unrecognized character, creating one word or string that would be determined as a spelling error during the spell check phase. The human eye could miss this type of error. In this example a voting scheme could resolve the problem by inserting a space in the correct character place. The voting scheme would use the Doculex engine at that failure point; insert the space and move forward creating the correct character string.

The second type of common error found resulted in 2 out of the 3 engines failing at the

hyphen character. One of the engines would completely drop the hyphen while another engine would see the hyphen but then insert a space. The third engine didn't have a problem with this character type and again, a voting scheme would solve this error.

The third type of error shown in Table 6 resulted from number recognition. The engines had a difficult time placing the numbers in the correct character space with regards to a comma. The engines would recognize the numbers correctly but would delete a space between the comma and the next number. 2 of the engines would typically fail at this word type but the 3rd engine would recognize the string correctly. A voting scheme would not resolve this problem because 2 of the 3 engines incorrectly recognize the word, resulting in the same character string.

The fourth type of error typical within this experiment was the result of only one engine failing at a character string. One example of this error type was found at character position 8939 in the Garamond 12 point normal font manipulation. At this position, ABBYY recognized the word Title as Titie. A voting scheme would correctly identify this sting using the other 2 engines. Regardless of the error, i.e. an incorrect letter or a missing space or even a wrong number, a voting scheme would resolve the error correctly identifying the character string.

Once the error types and numbers were identified and accounted for, accuracy rates were calculated. To increase the accuracy rates, it was determined if the error would vanish while utilizing a simple voting scheme. If the answer to this question was yes, the error was not counted and the accuracy was refigured. Below are the results from the 5-page document manipulations and engine recognition.

Table 7: Correction Rate for Simple Voting Scheme, Garamond

Document Manipulation	Total # Errors (all 3 engines)	Omni	Doculex	ABBYY	Total errors corrected w/ voting solution	% Rate of Correction
Garamond 8 Normal font	52	Total Errors - 12 Type 1 - 3 Type 2 - 1 Type 3 - Type 4 - 8	Total Errors - 32 Type 1 - Type 2 - 1 Type 3 - Type 4 - 31	Total Errors - 8 Type 1 - 3 Type 2 - 1 Type 3 - Type 4 - 4	52	100
Garamond 8 Bold font	40	Total Errors - 13 Type 1 - 4 Type 2 - Type 3 - Type 4 - 9	Total Errors - 6 Type 1 - Type 2 - Type 3 - 6 Type 4 -	Total Errors - 21 Type 1 - 4 Type 2 - Type 3 - 6 Type 4 - 11	28	70
Garamond 12 Normal font	69	Total Errors - 57 Type 1 - 9 Type 2 - 1 Type 3 - Type 4 - 47	Total Errors - 1 Type 1 - 1 Type 2 - Type 3 - Type 4 -	Total Errors - 11 Type 1 - 9 Type 2 - Type 3 - Type 4 - 2	69	100
Garamond 12 Bold font	56	Total Errors - 20 Type 1 - 9 Type 2 - 4 Type 3 - Type 4 - 7	Total Errors - 8 Type 1 - Type 2 - 4 Type 3 - 2 Type 4 - 2	Total Errors - 28 Type 1 - 9 Type 2 - Type 3 - 2 Type 4 - 17	52	92.8571429
Garamond 16 Normal font	45	Total Errors - 25 Type 1 - Type 2 - 3 Type 3 - 1 Type 4 - 21	Total Errors - 13 Type 1 - Type 2 - 3 Type 3 - 1 Type 4 - 9	Total Errors - 7 Type 1 - Type 2 - Type 3 - Type 4 - 7	43	95.5555556
Garamond 16 Bold font	92	Total Errors - 37 Type 1 - 13 Type 2 - 3 Type 3 - 8 Type 4 - 13	Total Errors - 15 Type 1 - Type 2 - 3 Type 3 - Type 4 - 12	Total Errors - 40 Type 1 - 16 Type 2 - Type 3 - 8 Type 4 - 16	76	82.6086957

Table 7 shows the number of errors found that would be correctly identified by a simple voting scheme. The first column shows the particular experiment run for the 5-page document. The next column is a tally of the total number of errors found regardless of error type or engine. The 3rd, 4th, and 5th columns further break down the errors found, the number and type of each error is identified for each of the 3 engines. The 6th column is the number of errors that would be resolved using a simple voting scheme. Finally, a percent rate of correction is determined using the number of errors corrected against the number of errors found. As can be seen, a voting scheme would be highly valuable.

Table 8 below is a continuation of data found for the experiment. As shown in the 1st column, the manipulation is slightly different using a different font type.

Table 8: Correction Rate for Simple Voting Scheme, Courier

Document Manipulation	Total # Errors (all 3 engines)	Omni	Doculex	ABBYY	Total errors corrected w/ voting solution	% Rate of Correction
Courier 8 Bold font	26	Total Errors - 10 Type 1 - 4 Type 2 - 2 Type 3 - Type 4 - 4	Total Errors - 9 Type 1 - Type 2 - 2 Type 3 - Type 4 - 7	Total Errors - 7 Type 1 - 4 Type 2 - Type 3 - Type 4 - 3	26	100
Courier 8 Normal font	23	Total Errors - 9 Type 1 - 5 Type 2 - 1 Type 3 - 1 Type 4 - 2	Total Errors - 9 Type 1 - Type 2 - 1 Type 3 - 1 Type 4 - 7	Total Errors - 5 Type 1 - 5 Type 2 - Type 3 - Type 4 -	21	91.3043478
Courier 12 Bold font	69	Total Errors - 30 Type 1 - 14 Type 2 - 2 Type 3 - Type 4 - 14	Total Errors - 22 Type 1 - 2 Type 2 - Type 3 - Type 4 - 20	Total Errors - 17 Type 1 - 14 Type 2 - Type 3 - Type 4 - 3	69	100
Courier 12 Normal font	47	Total Errors - 21 Type 1 - 14 Type 2 - 1 Type 3 - Type 4 - 6	Total Errors - 12 Type 1 - Type 2 - 1 Type 3 - Type 4 - 11	Total Errors - 14 Type 1 - 14 Type 2 - Type 3 - Type 4 -	47	100

Table 8 is identical to the previous table showing the different number of errors found and the type of each error. Again, it can be seen that a voting scheme would be beneficial in the character recognition process utilizing all three engines.

The experiment performed was successful in determining the need and usefulness of a

simple voting scheme using 3 *OCR* engines.

CHAPTER FIVE: CONCLUSION

This thesis provided a benchmark on three Commercial Optical Character Recognition engines available today. The three engines were ABBYY FinerReader, Doculex OCR-IT and OmniPage Pro. The document used to compare the three engines against each other was a clean-cut, 10-page paragraph. This document does not represent the average document that will require *OCR* be performed in the typical industry. The average document that will be scanned and converted into machine-readable text is more of a complex document often containing tables and figures along with formatting not easily read by *OCR* engines. Because the document was of simple form, the accuracy rates measured were very high.

Unfortunately, within every experiment there will be shortcomings that need to be accounted for. The experiment setup for this research was faulted because of the actual document statistics. First, as previously mentioned, the document used was a perfect, 5-10 page paragraph. While this is ideal, this is not practical within any market. Second, time constraints placed on this research prohibited the number of scans run. The amount of raw data used to determine the accuracy rates was limited. Third, the errors found and counted were determined using each character versus each word, and the accuracy was calculated using the number of edits required expressed as a percentage of the number of words in the image/document. If the accuracy was calculated using the number of edits required expressed as a percentage of the

number of characters (not words) within the image/document, the accuracy rate would be higher than shown.

Media diversity was limited for the purposes of this research and experiment. Ideally, the data obtained should have been from scans of all different types of documents. This could include newspaper articles, magazine pages, hand written letters and files, copies of photocopies, book pages and aged documents. The data collected in this research was performed with loss-less scanning which is virtually impossible when actually scanning a document. The *OCR* engines did a comparable job against each other recognizing the text.

The results of this experiment, while limited, did successfully conclude that a simple voting scheme should be realized to better fulfill the *OCR* requirements of certain markets, i.e. the legal community. Future work related to this research will include the software solution for combining three *OCR* engines during the character recognition stage of the *OCR* process.

LIST OF REFERENCES

1. Optical Character Recognition Quickstudy by Sami Lais

   http://www.computerworld.com/softwaretopics/software/apps/story/0,10801,73023,00.html

2. After the Scan Is Over Story by Sami Lais

   http://www.computerworld.com/softwaretopics/software/apps/story/0,10801,72989,00.html

3. Optical Character Recognition (OCR) as a Digitization Technology by Susan Haigh

   Network Notes #37, ISSN 1201-4338 Information Technology Services National Library

   of Canada http://www.nlc-bnc.ca/9/1/p1-236-e.html

4. Webopedia Definition: Optical Character Recognition

   http://www.webopedia.com/TERM/O/optical_character_recognition.html

5. Expervision http://www.expervision.com

6. Doculex http://www.doculex.com/

7. ABBYY Software House http://www.abbyy.com

8. ScanSoft http://www.scansoft.com/