6

Comprehensive and Scalable Appraisals of Contemporary Documents

William McFadden[1], Rob Kooper[1], Sang-Chul Lee[2] and Peter Bajcsy[1]
[1]*National Center for Supercomputing Applications, University of Illinois at Urbana-Champaign, Urbana, Illinois, USA*
[2]*Department of Computer and Information Engineering, Inha University, Incheon, Korea*

Abstract
This book chapter describes problems related to contemporary document analyses. Contemporary documents contain multiple digital objects of different type. These digital objects have to be extracted from document containers, represented as data structures, and described by features suitable for comparing digital objects. In many archival and machine learning applications, documents are compared by using multiple metrics, checked for integrity and authenticity, and grouped based on similarity. The objective of our book chapter is to describe methodologies for contemporary document processing, visual exploration, grouping and integrity verification, as well as to include computational scalability challenges and solutions.

1. Introduction

The objective of our work is to design a methodology, algorithms and a framework for document appraisal by (a) enabling exploratory document analyses and integrity/authenticity verification, (b) supporting automation of some analyses and (c) evaluating computational and storage requirements for archival purposes. In order to address the aforementioned criteria, our approach has been to decompose the series of appraisal criteria into a set of focused analyses, such as (a) find groups of records with similar content, (b) rank records according to their creation/last modification time and digital volume, (c) detect inconsistency between ranking and content within a group of records, and (d) compare sampling strategies for preservation of records.
In this work, we had chosen a specific class of electronic documents that (a) correspond to information content found in scientific publications about medical topics, (b) have an incremental nature of their content in time, and (c) contain the types of information representation that are prevalent in contemporary medical environments. Specifically, we narrowed our focus to those electronic documents that contain primarily text, raster and

vector graphics as found in typical medical records in office document file formats. Among the file formats, MS Word can be considered as the most widely used file format for creating documents, while Adobe Portable Document Format (PDF) and Ghostscript could be described as the most widely used for exchanging documents. We selected to work with PDF documents since PDF is an open file format, and the open nature of the file format is critical for automated electronic document appraisal and long term preservation.

In order to address the appraisal criteria [1], we adopted some of the text comparison metrics used in [2], image comparison metrics used in [3] and lessons learnt stated in [4]. Then, we designed a new methodology for grouping electronic documents based on their content similarity (text, image and vector graphics), and prototyped a solution supporting grouping, ranking and integrity verification of any PDF files and HTML files [5]. First, text based, vector based and multi-image based comparisons are performed separately. Multiple images in each document are grouped first and then groups of images across documents are compared to arrive to an image-based similarity score. The current prototype is based on color histogram comparison, line count in vector graphics and word frequency comparison. The image colors and word/ integers/ floating numbers can be analyzed visually to support exploratory analyses. Subsets of the undesirable text and image primitives could be filtered out from document comparisons (e.g., omitting conjunctions, or background colors). The results of text, image and vector based comparisons are fused to create a pair-wise document similarity score. The matrix of pair-wise document similarity scores are used for grouping. The other appraisal criteria are approached by ranking documents within a group of documents based either on time stamps or on file name indicating the version number. The inconsistency between ranking and content within a group of records is based on frequency tracking, where the frequency of text, image and vector primitives is monitored over the time/version dimension of the grouped documents.

Currently, we hypothesized that the correct temporal ranking correlates with the content (images, vector and text) in such a way that the content is being modified without sharp discontinuities. Sharp content discontinuities are perceived as significant changes of document descriptors that would correspond, for instance, to large text/image deletions followed by large text/image additions or large text/image additions followed by large text/image deletions. We have experimented with real PDF documents of journal papers to validate the above hypothesis.

The novelty of our work is in designing a methodology for computer-assisted appraisal, in developing and prototyping a mathematical framework for automation of appraisals based on image, vector graphics and text types of information representation, and in designing a scalable solution using the Map and Reduce parallel programming paradigm for using computer clusters.

2. Previous work

Related work to the proposed framework: Our work is related to the past work of authors in the area of digital libraries [2], content-based image retrieval [3] and appraisal studies [4]. For example, the authors of [6] analyze PDF document by examining the appearance and geometric position of text and image blocks distributed over an entire document. However, they did not use the actual image and vector graphics information in their analyses. Similarly, the focus in [7] is only on the logical structure of PDF documents but not the

content. The work in [8] and [9] is based on analyses of vector graphics objects only since it is focused on diagrams represented by a set of statistics, e.g., the number of horizontal lines and vertical lines. Other authors also focused only on chart images using a model-based approach [10]. There is currently no method that would provide a comprehensive content-based PDF comparison and appraisal strategy according to our knowledge. In order to prototype a solution for comprehensive document comparisons and clustering, the difficulties lie in dealing with vector graphics objects, fusion of similarities of multiple digital objects, and in providing a scalable solution with the increasing number of documents.

Related work on comparing vector graphics objects: Vector graphics objects are described by the most primitive feature of the graphics object, lines, which are practically useless in meaningful comparisons. Due to this, it is necessary to compare objects at a slightly more sophisticated level by comparing groups of lines and their relationship to each other. However, the manner in which this can be done varies, and many techniques for comparison of simple geometric shapes exist, making it not trivial to choose which graphic comparison to use. Veltkamp [11] showed ten distinct methods for the comparison of polygons and open paths, and it is assumed that more may exist. However, the principle difficulty in implementing almost all of these methods is that they rely on a direct, side-by-side computationally expensive comparison between two graphical objects resulting in a real number comparison value in the range between 0 and 1. In addition, the problem of formulating a metric measuring approximate similarity between visual objects has also been known as an open problem [12, 13]. Finally, there is the problem of the sequential arrangement of the line segments since they could be encoded in the document arbitrarily. This introduces a plethora of problems because there is no guarantee that a visually similar object will be encoded in a document such as an Adobe PDF file in anything like the same way.

Related work on scalability: In the past, the scalability of computations has been approached by using parallel programming paradigms based on message-passing interface[1] (MPI) or open multi-processing[2] (OpenMP). MPI is designed for the coordination of a program running as multiple processes in a distributed memory environment by using passing control messages. MPI could also run in a shared memory system. There are several developed libraries supporting MPI[3]. OpenMP is intended for shared memory machines. It uses a multithreading approach where the master threads forks any number of slave threads. Several known software solutions have also used the hybrid parallel programming paradigm combining the strengths of MPI and OpenMP, for example, WRF[4] or NECTAR[5]. The hybrid approach uses MPI across nodes and OpenMP within nodes, which leads to good utilization of shared memory system resource (memory, latency, and bandwidth).
We have investigated the use of Google's MapReduce[6] and Yahoo!'s Pig[7] framework and its associated PigLatin language. MapReduce is a programming model that allows

[1] http://www-unix.mcs.anl.gov/mpi/usingmpi/examples/simplempi/main.htm
[2] http://software.intel.com/en-us/articles/more-work-sharing-with-openmp
[3] http://www-unix.mcs.anl.gov/mpi/usingmpi/examples/simplempi/main.htm
[4] http://www.wrf-model.org/index.php
[5] http://www.cfm.brown.edu/crunch/ATREE/software.html
[6] http://labs.google.com/papers/mapreduce.html

programmers to focus on the tasks instead of the parallel implementation of the tasks. This lets programmers write simple Map function and Reduce function, which are then automatically parallelized without requiring the programmers to code the details of parallel processes and communications. In the past, the Hadoop users have raised several questions about optimal set up and execution of Hadoop [14], such as:
What are the optimum machine configurations for running a Hadoop cluster?
Should I use a smaller number of high end/performance machines or are a larger number of "commodity" machines?
How does the Hadoop/Parallel Distributed Processing community define "commodity"?

The answers to these questions are of a very high value to the end users and several researchers have searched for solutions. For example, in [15] the authors report a new task scheduler to improve Hadoop's performance for clusters that consist of nodes that are not homogeneous. Similarly, we search for better understanding how to setup and execute Hadoop when multiple types of digital objects have to be analyzed in parallel (e.g., text, images, vector graphics). In addition, in most of the real life applications, the Map and Reduce operations are preceded by input/output (load and transfer) operations that force us to balance the computational gains from Hadoop with the cost of opening files and extracting information.

3. Methodology

This section presents the methodology and theoretical framework for addressing grouping, ranking and integrity verification problems

3.1 Overview

The designed methodology consists of the following main steps: (1) Extract components and properties stored in PDF files/containers. (2) Define text, image and vector graphics primitives, and extract their characteristic features. (3) Group images within each document into clusters based on a pair-wise similarity of image primitives and a clustering similarity threshold. (4) Compute a pair-wise similarity of image clusters across two documents based on their corresponding features. (5) Compute a pair-wise similarity of text & vector graphics primitives across two documents. (6) Calculate fusion coefficients per document to weight the contribution of text-based, image-based and vector-based similarities to the final pair-wise document similarity score. (7) Repeat steps (4-6) for all pairs of documents. (8) Group documents into clusters based on the pair-wise document similarity score and a selected similarity threshold. (9) Assign ranks to all documents based on their time stamps and storage file size. (10) Calculate the second difference of the document characteristic features over time and file size dimensions. Report those documents for which the second difference exceeds a given threshold defining allowed discontinuities in content.

[7] http://incubator.apache.org/pig

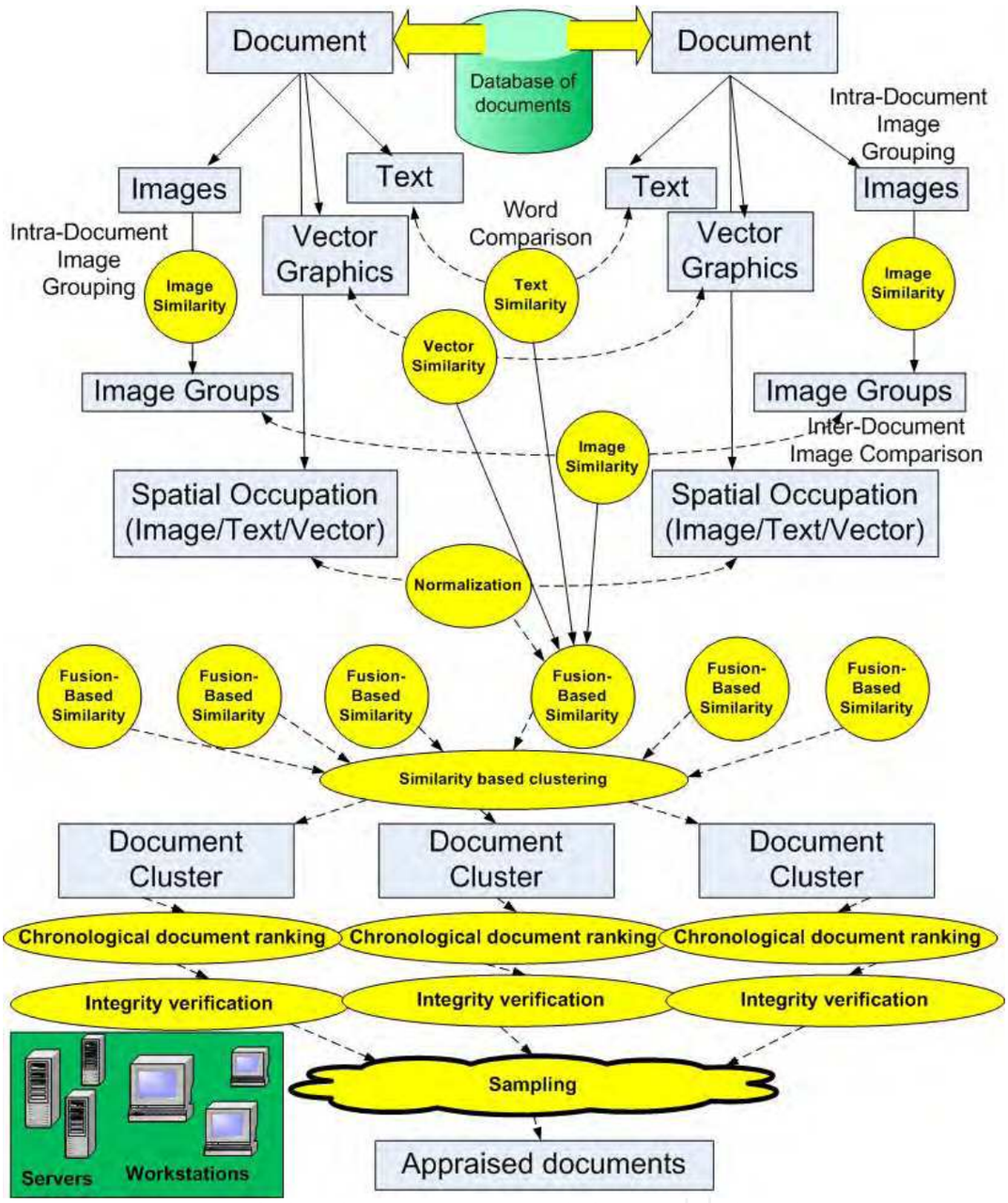


Fig. 1. An overview of the document appraisal framework based on comprehensive comparisons of document's internal digital objects.

3.2 Theoretical Framework

Document grouping problem. Given a set of documents, $\{D_i\}; i = 1, 2, \cdots, N$ compute pair-wise similarity of documents $sim(D_i, D_j)$ and aggregate them into clusters based on the similarity values for further ranking within each cluster.

The similarity of documents is understood as the combined similarity of document components. In our case, it would be the similarity of text, vector and raster (image) graphics components. The three components are decomposed into multiple images I_{ik} and their image primitives e_m^{IMAGE}, vector graphics and their image primitives e_m^{VECTOR}, and text primitives e_m^{TEXT} in textual portions $T_{ik} = T_i$ of a document D_i. The similarity for each component type is derived either directly using the features of its primitives (the case of text) or average features of multiple components of the same type and their primitives (the case of images and vector graphics).

Calculations of Statistical Features. The text feature for the word primitives is the frequency of occurrence of each unique word. The image feature for the color primitive is the frequency of occurrence of each unique color (also denoted a one-dimensional color histogram). The vector graphics feature is the frequency of occurrence of lines forming each vector graphics. The frequency of occurrence provides a statistical estimate of the probability distribution of primitives.

Calculation of Document Similarity. Given two PDF documents D_i, D_j, the similarity is defined as a linear combination of the similarities of the document components. In our case, the formula contains only the text and raster graphics components.

$$\begin{aligned} sim(D_i, D_j) &= w_{TEXT} \cdot sim(T_i, T_j) + \\ & w_{RASTER} \cdot sim(\{I_{ik}\}_{k=1}^{K}, \{I_{jl}\}_{l=1}^{L}) + \\ & w_{VECTOR} \cdot sim(V_i, V_j) \end{aligned} \tag{1}$$

where the $w_{TEXT}, w_{RASTER}, w_{VECTOR}$ are the weighting coefficients.

We have derived the weighting coefficients from the spatial coverage ratio of images, vector graphics and text in two compared documents. The weight assignment could be viewed as the relevance of each PDF component according to the amount of space it occupies in a document. The motivation is based on a typical construction of documents where the space devoted to a textual description or an illustration reflects its importance and hence should be considered in the similarity calculation. Thus, the weighting coefficients are calculated as

$$\begin{aligned} w_{IMAGE}(D_i, D_j) &= \frac{R_{IMAGE}(D_i) + R_{IMAGE}(D_j)}{2} \\ w_{IMAGE}(D_i, D_j) + w_{VECTOR}(D_i, D_j) + w_{TEXT}(D_i, D_j) &= 1 \end{aligned}, \tag{2}$$

where

$$R_{IMAGE}(D) = \frac{Area_{IMAGE}(D)}{Area_{IMAGE}(D) + Area_{VECTOR}(D) + Area_{TEXT}(D)}, \quad R_{IMAGE}(D) + R_{VECTOR}(D) + R_{TEXT}(D) = 1$$

Calculation of Text Similarity. The similarity of text components from two documents $sim(T_i, T_j)$ is computed using the text features and similarity metric defined according to [2]. The equation is provided below.

$$sim(T_i, T_j) = \sum_{k1,k2} \omega_{i,k1} \omega_{j,k2} \tag{3}$$

where $k1, k2$ are those indices of text primitives that occur in both documents (in other words, there exist $e_{i,k1} = e_{j,k2}\ e_{i,k1} \in T_i; e_{j,k2} \in T_j$). The ω terms are the weights of text primitives computed according to the equation below.

$$\omega_{ik} = \frac{f_{ik} \log(N / n_k)}{\sqrt{\sum_{l=1}^{L} (f_{il})^2 (\log(N / n_l))^2}} \tag{4}$$

where f_{ik} is the frequency of occurrence of a word e_k in D_i, N is the number of documents being evaluated, *L* is the number of all unique text primitives (words) in both documents, and n_k is the number of documents in the set of all documents being evaluated that contain the word e_k ($n_k = 1\ or\ 2$).

Calculation of Raster Graphics (Image) Similarity. In contrary to text that is viewed as one whole component, there are multiple instances of raster graphics components (images) in one document. Thus, the similarity of image components in two documents is really a similarity of two sets of images.

Due to the fact that many documents contain images that are sub-areas or slightly enhanced versions of other images in the same document, we have observed biases in image-based document similarity if all possible image pairs from two documents are evaluated individually and then the average similarity would be computed. The bias is introduced due to large similarity values of one master image in one document with multiple derived images in another document, although many other images would not have any match.

In order to avoid such biases, we approached the similarity calculation by first computing a pair-wise similarity of all images within each document and clustering them. Next, the pair-wise similarity of clusters of images from each document is computed using the average features of clusters.

A. Intra-document image similarity: The similarity of two raster graphics (image) components from one document $sim(I_{ik} \in D_i, I_{il} \in D_i)$ is computed using the one-dimensional color histogram feature and the same similarity metric as defined before for text according to [2]. The equation is provided below.

$$sim(I_{ik} \in D_i, I_{il} \in D_j) = \sum_{k1,k2} \omega_{i,k1} \omega_{i,k2} \tag{5}$$

where $k1, k2$ are those colors that occur in both images (in other words, there exist $e_{i,k1} = e_{i,k2}; e_{i,k1} \in I_{ik}, e_{i,k2} \in I_{il}$). The ω terms are the weights computed the same way as before.

B. Inter-document image similarity: The similarity of two sets of raster graphics (image) components, one from each document, $sim(I_{ik} \in D_i, I_{jl} \in D_j)$ is computed using the average one dimensional color histogram feature of all images in a set and the same similarity metric as defined before for text according to [2]. The equation is provided below.

$$sim(\{I_{ik}\} \in D_i, \{I_{jl}\} \in D_j) = \sum_{k1,k2} \omega_{i,k1} \omega_{j,k2} \tag{6}$$

Calculation of Vector Graphics Similarity. The methodology used text and raster image comparison is based on a statistical comparison technique for comparing large numbers of small information units of precisely defined structure, such as words or colors, which are reused multiple times to construct the document. The difficulty in using this comparison technique with vector graphics is that the most primitive feature of the graphics object, lines, are practically useless in meaningful comparisons. Due to this, it is necessary to compare objects at a slightly more sophisticated level by comparing groups of lines and their relationship to each other. However, the manner in which this can be done varies, and many techniques for comparison of simple geometric shapes exist, making it not trivial to choose which graphic comparison to use.

Several reviews of vector graphics comparison techniques [11, 13, 16] showed distinct methods for the comparison of polygons and open paths. However, the principle difficulty in implementing almost all of these methods is that they rely on a direct, side-by-side computationally expensive comparison between two graphical objects resulting in a real number comparison value in the range between 0 and 1. These types of comparisons would be difficult to integrate into the previously used statistical comparison technique because the current methodology relies on counting exactly reproducible informational units. For example, in a comparison of the text of this document, it would be easy to count the precise number of times the word "text" appears, but if one were to draw many cat graphics whose dimensions were not exactly the same, it would be difficult to count the number of cats because a common definition would not be possible. Instead, it is necessary to develop an approximate characterization of the shape of the graphic, which can then be easily compared with other shape approximations.

In this interest, the algorithm was developed, which binned the angle of each line in any series of connected lines in the document and recorded the binned angles. The angles could be grouped into unordered histograms or stored in the order they are rendered. The angular bin size is also variable. Tests presented in Section 0 were used to find an optimal method. The chosen methodology was unordered grouping with 9 angular bins. Using a textual encoding of the angular description, the algorithm used for text comparison can be used on the vector graphic data.

Finally, rotational and scale invariance of a similarity metric for comparing vector graphics objects has to be considered. For example, defining the angle of a line is simple if the basis axes are taken to be defined by the PDF page, but if this is the case then the comparison technique will not be rotationally invariant. To ensure rotational invariance, there must be a

common definition of the basis line against which all the angles can be computed. Since the main feature of the shape is the location of the vertices, it makes sense to find a basis parameter that correlates the individual vertices of each line to each other. To this end, the average of all the vertices in the shape is used to find a center point about which all reference lines can be drawn. The reference line for each angle computation is then drawn through the center point and through the center of the line whose angle is being computed. These two points will always be in the same relation to the line whose angle is being measured despite rotational variance as is clear from Fig. 2. In Fig. 2, the solid black lines are vector graphic lines measured (which constitute a trapezoid in this example). The red dot is the mean of all the vertices of the shape, and the blue dots are the center points of the individual lines. The dotted line is the reference line, which passes through the red dot and the blue dot of the graphic line being measured. The angle (between 0° and 90°) is then computed from the intersection of these two lines which guarantees the rotational invariance.

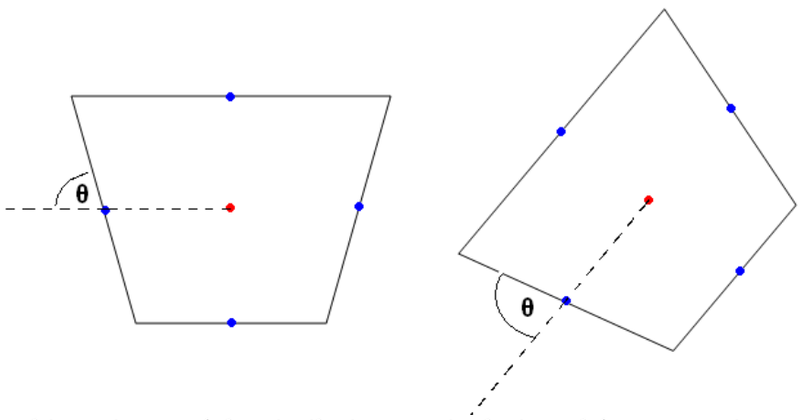


Fig. 2. Rotational invariance of the similarity metric designed for comparing vector graphics objects. The blue points correspond to the middle of each line. The red point is the center of gravity. The angle θ is the angle between each line forming a vector graphics object and the line connecting the center with the midpoint of the vector graphics line. The angle does not change with the rotation of the vector graphics object.

4. Feature Extraction from Adobe PDF Documents

The methodology described earlier is applicable to any set of documents. However, we have chosen to focus on the processing of PDF documents. PDF documents have become almost the defacto standard for information sharing and can be viewed on many different platforms. To be able to compute the similarity of two documents we first need to extract as many features as possible from the PDF documents. In the past we have conducted research to find software toolkits for loading and parsing PDF documents, as well as to identify the cost and functionality tradeoffs of each available software toolkit. The three main toolkits for extracting information from PDF documents are the PDF SDK developed by Adobe, Jpedal developed by IDR and PDFBox an open source SDK (PDFBox has since become a incubation project under the Apache Software Foundation). With these three software toolkits there was a direct relationship between the cost and the completenes of the implementation

according to the PDF specification [17]. Jpedal provides a very complete implementation of the PDF specification for a low price. PDFBox has the main advantage of it being open source allowing us to extend the library if needed.

4.1 Feature Extractions

The choice of text primitives could include characters, words, sentences, paragraphs, sections, etc. Given the choice of these primitives, the word feature is mainly used due to relatively high information granularity comparing with characters and robustness for large number of samples comparing with sentences, paragraphs or sections. In particular, we could further divide the words into multiple types such as alphabetical texts, integers, and float (See Fig. 3). The main idea behind this clustering approach is in the fact that a scientific document could include different types of information based on the types of words.

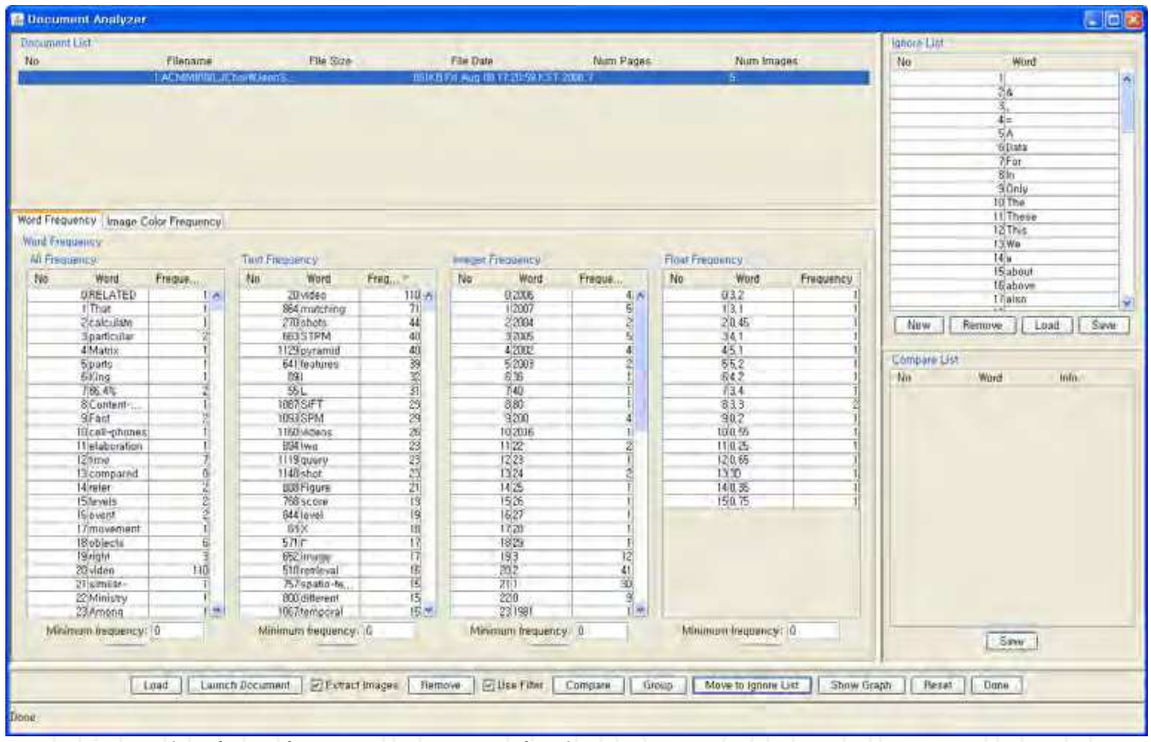


Fig. 3. Example of text features in a scientific document.

Alphabetical words could be further clustered into proper nouns, common nouns, and other frequently used words, e.g., verbs, adjectives, connectives, etc. Analysis on highly frequent common nouns could categorize the document into some scientific topics, and those on proper nouns could label and measure the uniqueness of the document. These classifications could be performed semi- or fully automatically by human or simple dictionary search. In Fig. 3, *text frequency* column shows very high frequency on "video/videos", "matching", and "shots", which implies that the document falls into some apparent topics, such as video matching.

In a scientific document, integer strings could typically represent (a) document numbers, i.e., chapter or page numbers, (b) time information, i.e., years, and/or (c) some data values.

Analyzing the range, i.e., 1900~2009 for years, or non-continuous numerical values, i.e., 2171, 1131400, etc., could give a clue about when the document is written or presence of some data values derived from a study. In Fig. 3, the *integer frequency* column shows some numbers guessed as years and continuous low numerical values guessed as less important information, e.g. page numbers. In addition, due to the fact that the maximum value of the year is 2007, we could also guess that the document was written in or after 2007. The floating point numbers typically appear in a scientific document reporting the measured or derived values from an experiment.The occurrence of these values could provide additional clues about the types of the document.

Similarly, image primitives could become colors at each pixel, shapes of blobs of adjacent pixels with similar color (image segments), color statistics of regular blocks of pixels, etc. In this work, we focused on the statistics of color frequencies in a figure to determine the type of the figures. Fig. 4 shows an example of color frequency analysis on the same scientific document in Fig. 3. Based on the extracted color frequencies after quantification, we could cluster the raster image figures in a typical PDF document into several categories such as charts, natural scenes, or drawings. For example, a chart in Figure 4, shows very high color frequency for the RGB values of [255,255,255] (white), [191,191,255] (cyan), and [255,191,191] (pink). Natural scene would have high color variances in the images depending on the lighting condition, objects in the figure, and zoom level of the scene.

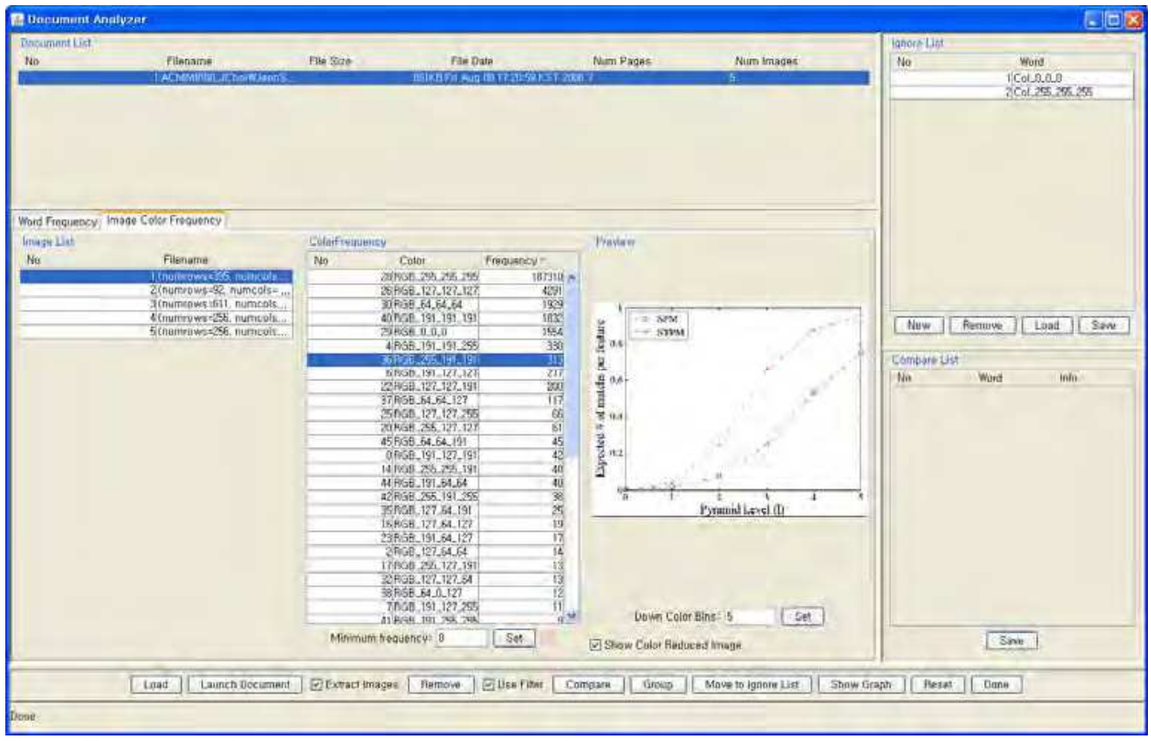


Fig. 4. Example of image color features in a scientific document.

4.2 Scalability of Feature Extractions

To be able to do a better similarity computation of the documents we want a large number of features extracted. The problem becomes that the more features we extract from the PDF

documents, the longer it will take to process a PDF document. The initial approach we took was to store the extracted data for each document so the next time the document is openend we can simply retrieve the cached information. This works well if documents are opened multiple times and not to many complex documents are opened simultanously.
To overcome these problems we have looked at preprocessing the documents. Since we potentially have to preprocess many documents, parallelizing this task is important. The parallelization is achieved by using Apache Hadoop. Each document is preprocessed by our system and the cached information is written to disk. When comparing documents we can simply load the cached information instead of having to process the document.
Hadoop allows us to count occurences in a document. The document is split in smaller pieces (a set of pages) and each page is parsed into objects (text paragraphs, images, vector graphics and in the future other multimedia elements). These objects are sent to a mapping function which will split the object in smaller parts that will be counted. For example text paragraphs are split in words (allowing us to do frequency of occurence of words, dates and numbers) and imaeges are split in pixels (for frequency of occurence of colors). After the map operation the reduce stage will begin which will count the frequency of occurence of the smaller parts generated by the mapper function.
The map and reduce phases of Hadoop can run in parallel, allowing intermediate data to be streamed from the mapper to reducer. This will decrease the time it takes to process a document. In the simple case where we only have one mapper and one reducer we have seen the time be cut in half for parsing the whole document. Since Hadoop codes run in parallel, all the data for a job need to be sent to the processor that will perform the map and reduce operations. For documents that are smaller this overhead can be more than the time it takes to process the document on a local node. On the other hand if we have many documents that need processing we can not run all jobs on the local node and thus no matter what solution is chosen to do the processing, the overhead for distributing the data and results will be the same.

5. Experimental Results

We report experimental results that illustrate the choice of vector graphics comparison metrics, the accuracies of comprehensive document comparisons based on all contained digital objects, the prototype approach to document integrity verification, and the computational scalability achieved using computer clusters and Map & Reduce parallel programming paradigm.

5.1 Evaluations of Vector Graphics Comparisons

In the comparison of images it is standard to ensure that the comparison metric is invariant under translation, rotation, and rescaling. Because the method is based on the angle between lines it stands to reason that there should be no variation under translational variation. Also, as long as rescaling does not distort the image's aspect ratio it should also not make a difference in comparison. To test this, the three documents appearing in Fig. 5 were compared.
Note that the image of the computer in the document on the left has been both translated and rescaled in the center document. The document on the right contains an unrelated image of a CD-ROM drive. The result of a comparison of these three images is 100%

similarity between documents 1 and 2 and 0% similarity between documents 1 and 3 and documents 2 and 3. Therefore the comparison is both translation and scale invariant.
To be sure that this method works a simple test was performed on the three PDF documents shown in Fig. 6.

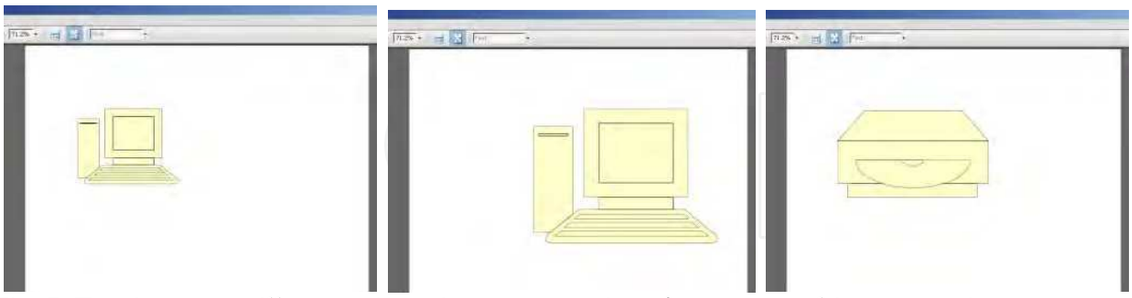


Fig. 5. Test images to illustrate invariance properties of vector graphics comparisons

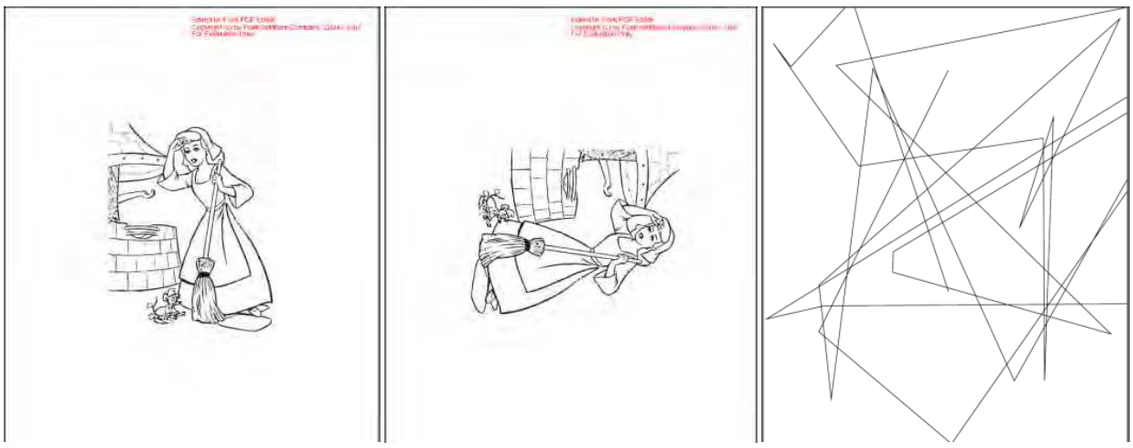


Fig. 6. Three test PDF documents with vector graphics objects. The leftmost document and the center document are just 90° rotations of each other and the rightmost document contains an unrelated set of random lines.

Using the unordered implementation with 9 bins without the rotationally invariant modification, the similarity between the rotated documents is only 0.18, while the similarity between unrelated documents is 0. Using the modification to allow rotational invariance brings the similarity between the rotated documents to 1, while the similarity between the unrelated documents remains at 0. This process was repeated for rotations of 25°, 50°, 75°, and 180°. The 180° rotation also showed a similarity of 1, but when the graphic was rotated by the intermediate angles, the similarity dropped to 0.72. However, the rotations at intermediate angles all showed similarity 1 between other rotations at intermediate angles. This indicates that during partial rotations some line segments may become distorted in the PDF rendering, but this type of problem cannot be rectified because the actual shape is being distorted and there is no way to recover it.

5.2 Evaluations of Comprehensive Document Comparisons and Clustering Results

Using the presented framework, we appraised sets of PDF documents generated during scientific medical journal preparations. The document sample set consisted of 10 documents 4 of which were modified versions of one article and 6 were of a different though related article. The pair-wise comparisons of the features are presented in the following graphs.

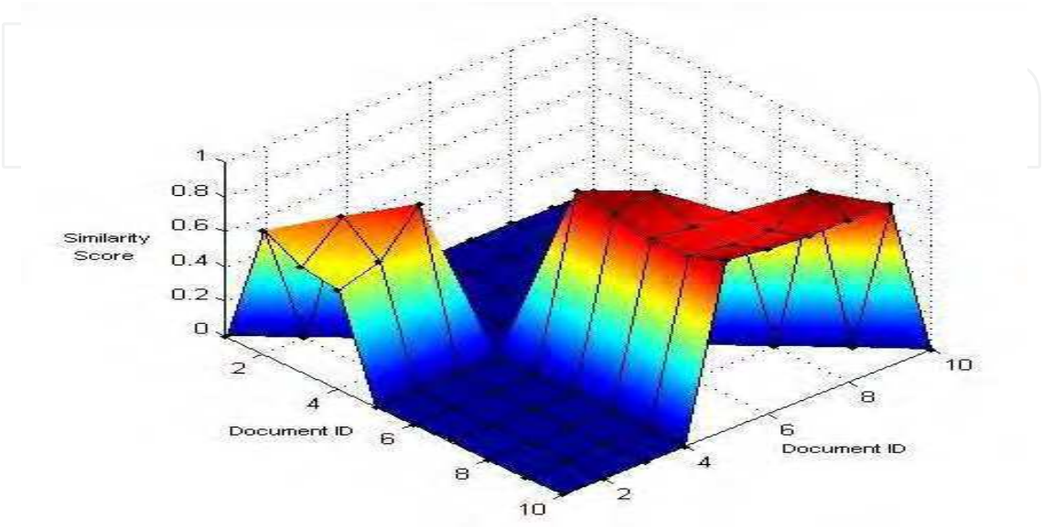


Fig. 7. Word similarity comparison

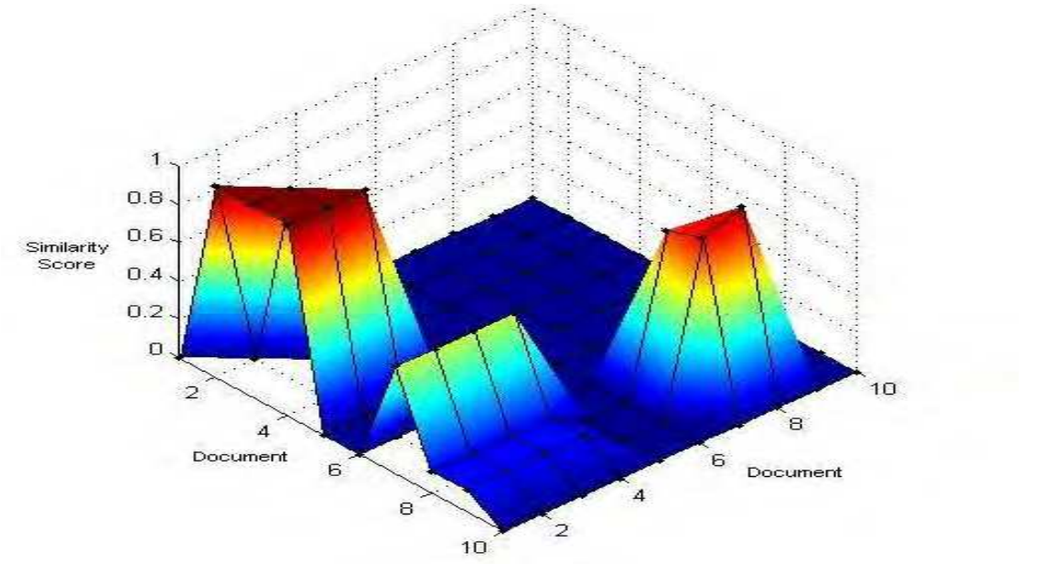


Fig. 8. Vector graphics similarity comparison

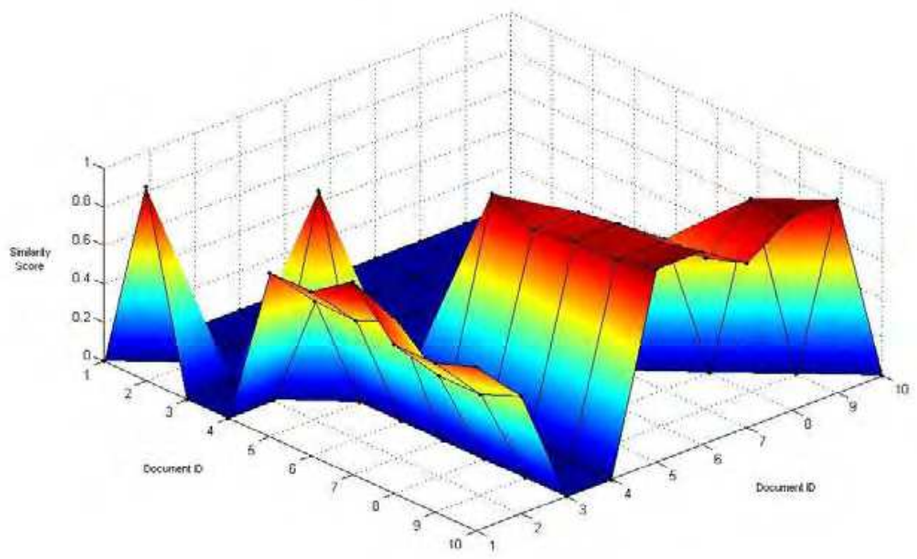


Fig. 9. Raster image similarity comparison

The z values of the graphs in Figures 1-3 represent the similarities between the documents identified with the numbers by the x and y axes. The word similarity comparison clearly shows that the documents 5 through 10 score highly in comparison with each other while they score poorly in comparison to the documents 1 through 4. Likewise, documents 1 through 4 show higher scores for comparison within the group. The vector graphics of documents 1 through 4 are nearly identical while in the second subgroup only documents 7 through 9 are conclusively linked. The raster images within the two subgroups of the documents shows high similarities for the documents 5 through 10 score but the rest are not obvious how they are related. The combination of vector graphics comparison along with word comparison results in a clear consensus about which documents belong together as shown in Figure 4.

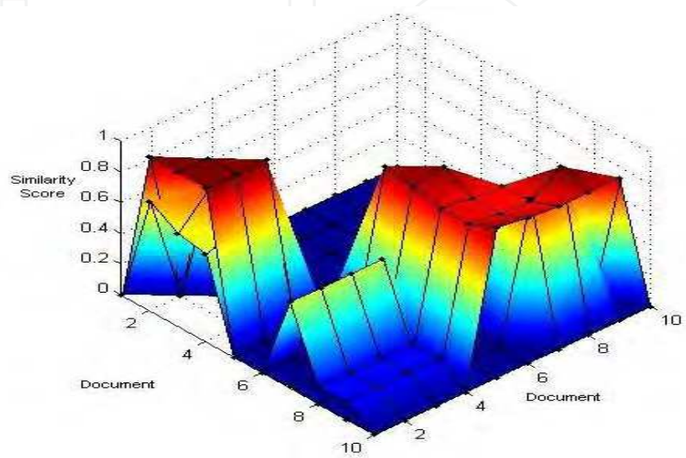


Fig. 10. Vector graphics similarity and word similarity combined

Throughout the documents the relative apportionment of visible space of the three document features varies. Figure 5 shows the fraction of each document covered by words, images and vector graphics.

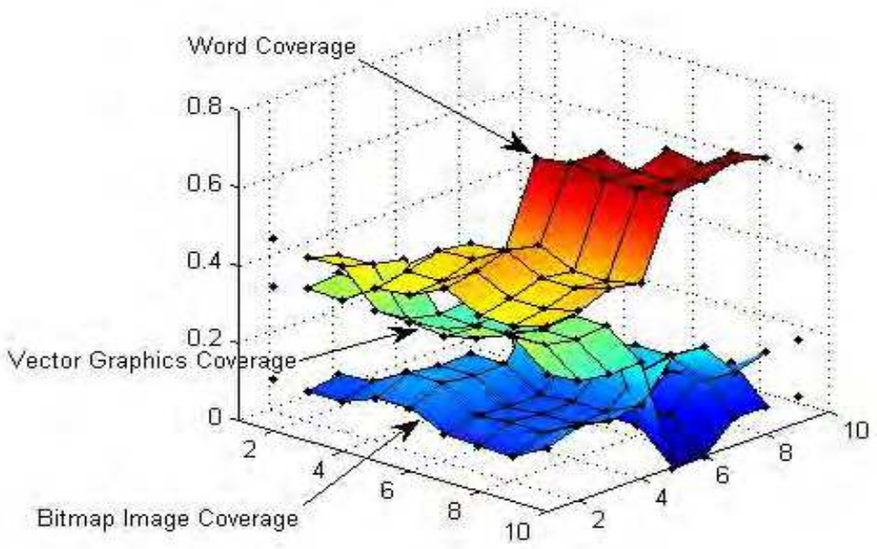


Fig. 11. Portion of document surface allotted to each document feature

Combining the three comparison techniques with weights allotted by the proportion of coverage of the feature represented by that comparison allows for a final similarity score to be established. Figure 6 displays the final comparison matrix, which clearly distinguishes the two subgroups of the original document set.

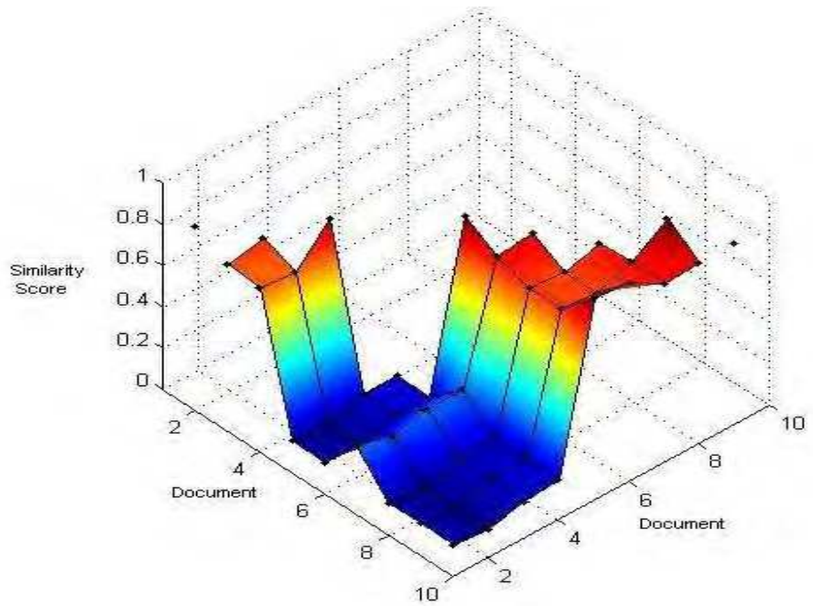


Fig. 12. Comparison using combination of document features in proportion to coverage

5.3 Preliminary Results of Document Integrity Verification

With the documents adequately grouped and ordered by PDF timestamp, the verification process looks for conspicuous editing habits from one document to the next. A successful document order verification relies on multiple failures of the tests displayed in Figures 7 and 8. The current tests conducted search for (a) appearance and disappearance of (1) identical document dates images or (2) identical, and (b) increase or decrease in (1) file size, (2) image count, (3) sentence count, and (4) average date value from one document to the next.

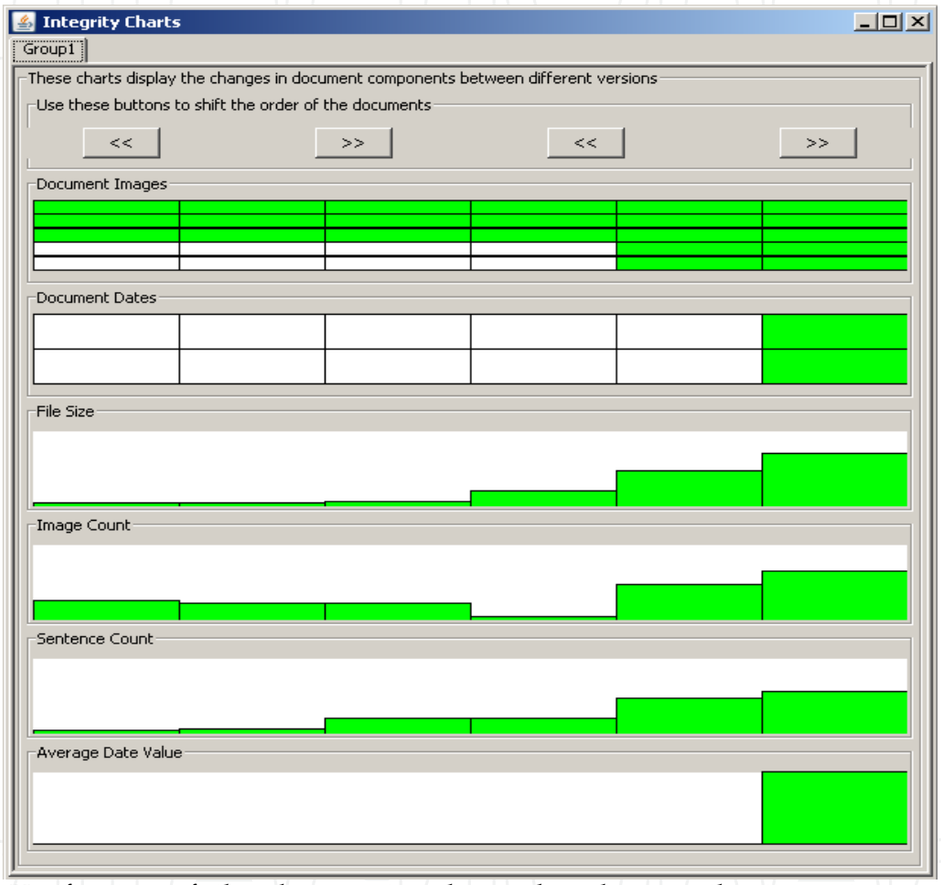


Fig. 13. Verification of the document ordering based upon the time stamps of PDF documents. Documents are aligned from earliest (left) to latest (right). Integrity tests are aligned from top to bottom. These are: (1) appearance or disappearance of document images, (2) appearance and disappearance of dates appearing in documents, (3) file size, (4) image count, (5) number of sentence, and (6) average value of dates found in document.

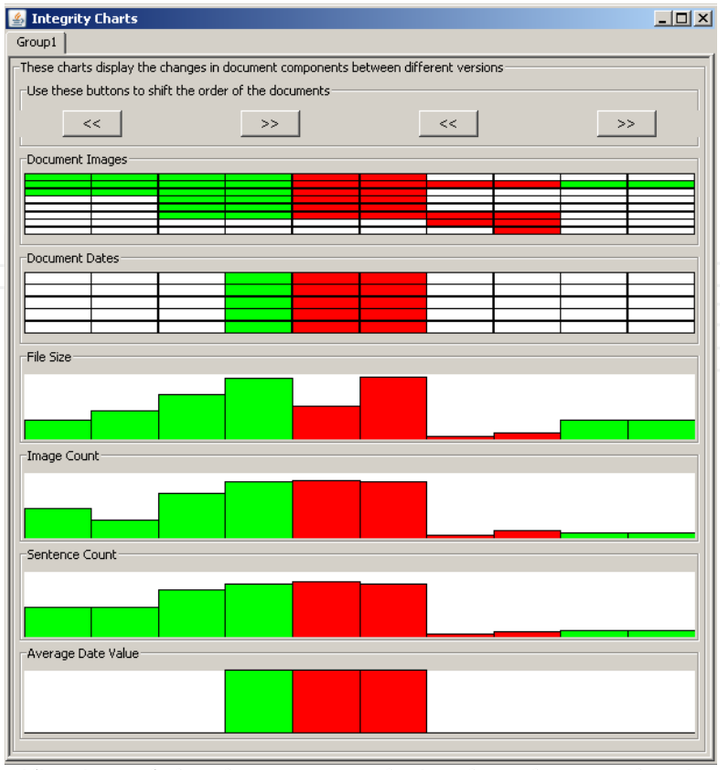


Fig. 14. Failed Verification of the document ordering based upon the time stamps of PDF documents. Green bars indicate reasonable changes to documents while red bars indicate suspicious document editing behavior such as drastic deletions.

5.4 Evaluations of Computational Scalability Using Computer Clusters

Our experimental design focuses on quantifying the speed of the statistical summarization of digital objects in Adobe PDF documents. The choice of statistical summarization is based on the observations that word counting in text, line counting in vector graphics objects and color-based pixel counting in images are all counting operations suitable for distributed computing.

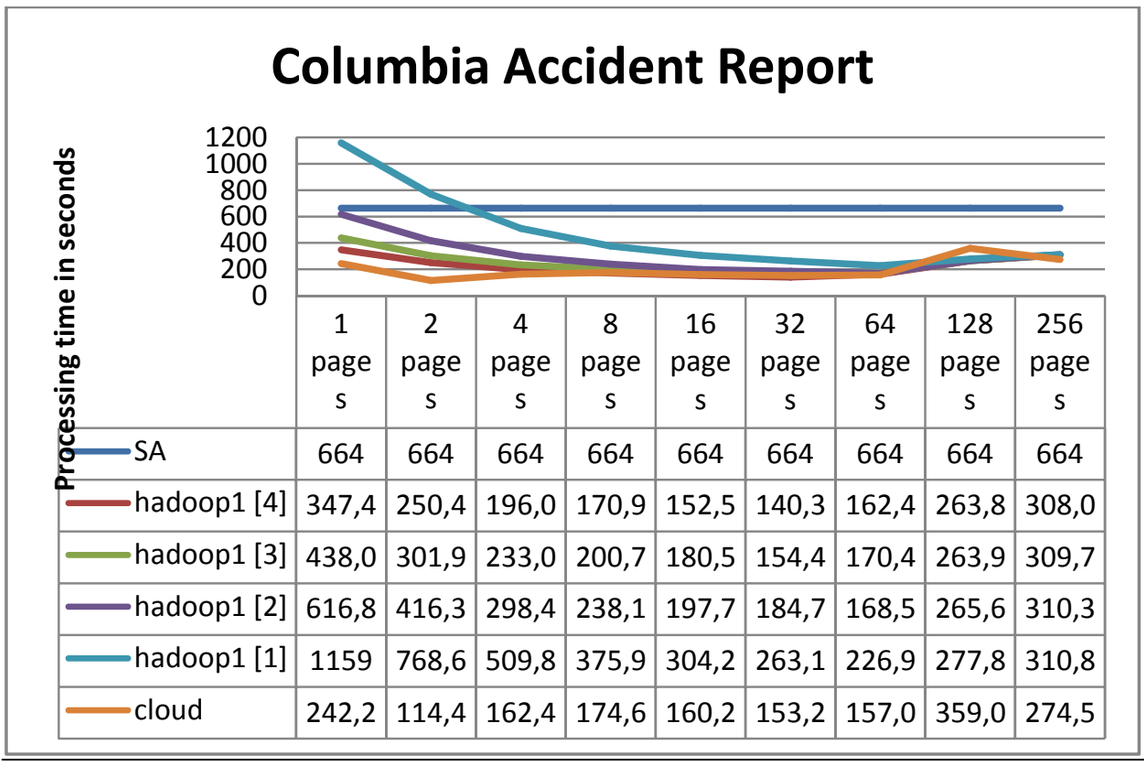
Hardware Configuration: We tested the prototype implementation on several machines that were configured with the Linux operating system (Ubuntu flavor) and Hadoop. There was one cluster at National Center for Supercomputing Applications (NCSA) and one cluster in the Computer Science Department at the University of Illinois at Urbana-Champaign (UIUC) that we used during the last period of performance. The cluster specifications are provided below.

NCSA cluster: The NCSA cluster consisted of four identical desktop machines. Each machine has a Core 2 processor running at 2.4 GHz with 2GB of memory and 100GB of local disk space. The cluster created from these machines was set up to do five Map and one Reduce task per node, resulting in the ability to do 4 * 5 = 20 Map tasks and 4 * 1 = 4 Reduce tasks simultaneously. Each node was also set up to be part of the shared file system with a replication the same as number of nodes in the cluster (resulting in each file being locally on the file system).

Illinois Cloud Computing Testbed cluster: We were given access to the Illinois Cloud Computing Testbed (CCT) in the Computer Science Department, at the University of Illinois at Urbana-Champaign (UIUC). The CCT testbed was funded by Intel, Yahoo! and Hewlett Packard, and provides resources for doing cluster computing research. After arranging the access to the test-bed, we could run jobs on a larger system than the ones at NCSA. However, we did not have control of the CCT system and could not reduce the number of hosts running. Thus, we always run with the maximum number of hosts up equal to 64 nodes. Each node has a dual quad core CPU with 16GB of memory. The cluster created from these machines was set up to do six Map and two Reduce tasks per node, resulting in the ability to do 64*6=384 Map tasks and 64*2=128 Reduce tasks simultaneously. Each node has 2TB of local disk space which is part of the larger distributed file system. Unlike the NCSA clusters this file system has a replication factor of 3 resulting not in each file being locally available.

Software Configuration: For the experiments, we took the NCSA cluster and created two configurations. One configuration had a single machine with both master and slave processes running. The master process is responsible for distributing the jobs and the slaves execute the jobs (the Map and Reduce operations). The second configuration had two machines, where each machine had the slave processes running and one of them had the master process running. The Illinois CCT cluster has a dedicated machine for running the master process and all other machines are running slave processes. The dedicated machine running the master process is not running any slave processes.
We have run experiments with Hadoop on both NCSA cluster and Illinois CCT cluster (also denoted as cloud in the figure below).

Benchmarking: As a reference measurement, we timed the execution without using Hadoop on a single machine. To be able to execute the code with the larger size documents we had to increase the memory to 2GB for the Java VM. The code used to obtain the reference measurement has not been optimized. The reference measurement always corresponds to the blue line in all graphs denoted as SA (stand-alone).
All timing measurements are conducted after the system started and the document to be processed has been uploaded to the distributed file system (in the case of Hadoop). For the case of the stand-alone machine, the timing starts when the document is about to be opened and it stops when the counting is finished (all of this is done in java and averaged over 10 runs so that the Java virtual machine has the time to optimize the code). In the case of Hadoop, the timing starts in the main function when the job is submitted and stops when Hadoop has finished its computation.



	1 pages	2 pages	4 pages	8 pages	16 pages	32 pages	64 pages	128 pages	256 pages
SA	664	664	664	664	664	664	664	664	664
hadoop1 [4]	347,4	250,4	196,0	170,9	152,5	140,3	162,4	263,8	308,0
hadoop1 [3]	438,0	301,9	233,0	200,7	180,5	154,4	170,4	263,9	309,7
hadoop1 [2]	616,8	416,3	298,4	238,1	197,7	184,7	168,5	265,6	310,3
hadoop1 [1]	1159	768,6	509,8	375,9	304,2	263,1	226,9	277,8	310,8
cloud	242,2	114,4	162,4	174,6	160,2	153,2	157,0	359,0	274,5

Fig. 15. Time to parse the Columbia Investigation PDF document in seconds as a function of the number of pages in each distributed data chunk. The PDF document contains 248 pages, 179.187 words, 236 images, and 30.924 vector graphics; and its size is 10.330.897 bytes. Dark blue curve corresponds to the reference measurement (single machine, no Hadoop). Orange curve (denoted as cloud) corresponds to the Illinois CCT cluster with 64 machines. Other curves correspond to NCSA cluster with a variable number of nodes utilized for the computation (listed in brackets as hadoop1[number of nodes]).

Experimental Results: We have analyzed several documents of various complexity (number of digital objects contained) and various size. The initial experiments took the document and split it up in clusters of 1, 2, 4, 8, …, 256 pages per map operation (or to the closest power of two larger than the total number of pages in a document). The map operation would take the data from the document and split it up in the smallest possible item. For text it would create a list of words, for images a list of pixels and for vector graphics a list of vector graphic operations (connect points to lines, create a rectangle, etc.). Once these lists are generated, Hadoop would start to reduce these lists by counting how often certain items appeared in the list. Processing time for an example document is shown Fig 15. As it can be seen in Fig 15, the use of Hadoop is definitely advantageous over a stand-alone implementation for complex documents such as the Columbia Investigation report.

6. Summary

We have described a framework for addressing document appraisal criteria. The framework consists of feature extraction from multiple digital objects contained in contemporray

documents, pair-wise similarity metrics for comparing digital objects, a comprehensive content-based grouping of documents, ranking based on temporal or file size attributes and verification of document integrity, as well as the parallel algorithms supporting applications with large volumes of documents and computationally intensive processing. Although we selected to work with documents in PDF format, the framework is applicable to any file format as long as the information can be loaded from any proprietary file format. The framework implementation called Document To Learn (Doc2Learn) is available for downloading from http://isda.ncsa.uiuc.edu/.In future, we will be exploring other hypotheses to increase the likelihood of detecting inconsistencies and understanding the high-performance computing requirements of computer-assisted appraisal of electronic records.

7. Acknowledgment

This research was partially supported by a National Archive and Records Administration (NARA) supplement to NSF PACI cooperative agreement CA #SCI-9619019. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the National Science Foundation, the National Archive and Records Administration, or the U.S. government.

8. References

[1] P. Bajcsy, and S.-C. Lee, "Computer Assisted Appraisal of Contemporary PDF Documents," in ARCHIVES 2008: Archival R/Evolution & Identities, 72nd Annual Meeting Pre-conference Programs:, San Francisco, CA, 2008.

[2] G. Salton, J. Allan, and C. Buckley, "Automatic structuring and retrieval of large text files," *in Communication of the ACM,* vol. 37, no. 2, pp. 97-108, 1994.

[3] D. M. Squire, W. Müller, H. Müller *et al.*, "Content-Based query of image databases: inspirations from text retrieval," *Pattern Recognition Letters,* vol. 21, pp. 1193-1198, 2000.

[4] J. A. Marshall, "Accounting For Disposition: A Comparative Case Study of Appraisal Documentation at the NARA in the US, Library and Archives Canada, and the NAA," Dep. of Library and Information Science, Univ. of Pittsburg, 2006.

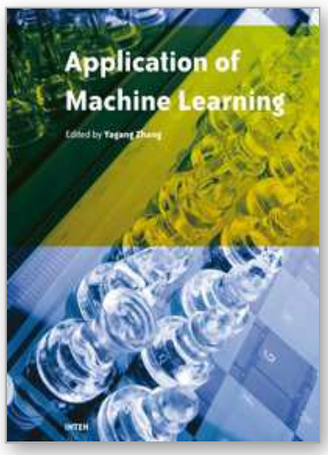
[5] S.-C. Lee, W. McFadden, and P. Bajcsy, "Text, Image and Vector Graphics Based Appraisal of Contemporary Documents," in The 7th International Conference on Machine Learning and Applications, San Diego, CA., 2008.

[6] W. S. Lovegrove, and D. F. Brailsford, " Document analysis of PDF files: methods, results and implications," *ELECTRONIC PUBLISHING,* vol. 8, no. 2 & 3, pp. 207-220, JUNE & SEPTEMBER 1995, 1995.

[7] A. Anjewierden, "AIDAS: incremental logical structure discovery in PDF documents," in International Conference on Document Analysis and Recognition, 2001.

[8] R. P. Futrelle, M. Shao, C. Cieslik *et al.*, "Extraction, layout analysis and classification of diagrams in PDF documents. Intl. Conf.Document Analysis & Recognition" pp. 1007-1014 year 2003.

[9] M. Shao, and R. P. Futrelle, "Recognition and Classification of Figures in PDF Documents," *Lecture Notes in Computer Science*: Springer Berlin / Heidelberg 2006.
[10] W. Huang, C. L. Tan, and W. K. Leow, "Model-based chart image recognition," in Fifth IAPR International Workshop on Graphics Recognition Computer Vision Center, Barcelona, Catalonia, Spain, 2003, pp. 87-99.
[11] R. C. Veltkamp, and L. J. Latecki, "Properties and Performances of Shape Similarity Measures.," *Data Science and Classification,*. pp. 47-56.
[12] M. Tanase and R. C. Veltkamp, "Part-based Shape Retrieval." In Proceedings of the 13th annual ACM international conference on Multimedia, pages 543–546. ACM New York, NY, USA, 2005.
[13] L. Hess and B.Mayoh. "Graphics and theUnderstanding of Perceptual Mechanisms: Analogies and Similarities". Geometric Modeling and Imaging–New Trends, 2006, pages 107–112, 2006.
[14] Enis. "Machine Scaling," 14th of June 2009; http://wiki.apache.org/hadoop/MachineScaling.
[15] M. Zaharia, A. Konwinski, A. D. Joseph *et al.*, *Improving MapReduce Performance in Heterogeneous Environments,* UCB/EECS-2008-99, University of California at Berkeley, 2008.
[16] R. C. Veltkamp, "Shape Matching: Similarity Measures and Algorithms." pp. 188-97.
[17] Adobe Systems, "PDF Reference version 1.7," *ISO 32000-1 standard* http://www.adobe.com/devnet/pdf/pdf_reference.html, [June 12th, 2009, 2009].



Application of Machine Learning

Edited by Yagang Zhang

ISBN 978-953-307-035-3
Hard cover, 280 pages
Publisher InTech
Published online 01, February, 2010
Published in print edition February, 2010

The goal of this book is to present the latest applications of machine learning, which mainly include: speech recognition, traffic and fault classification, surface quality prediction in laser machining, network security and bioinformatics, enterprise credit risk evaluation, and so on. This book will be of interest to industrial engineers and scientists as well as academics who wish to pursue machine learning. The book is intended for both graduate and postgraduate students in fields such as computer science, cybernetics, system sciences, engineering, statistics, and social sciences, and as a reference for software professionals and practitioners. The wide scope of the book provides them with a good introduction to many application researches of machine learning, and it is also the source of useful bibliographical information.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen