Monash University
Faculty of Information Technology

Using fingerprints based on the sequence of a set of selected words in a document for *1*-to-*n* similarity analysis

G. K Gupta
Faculty of Information Technology
Monash University
Clayton, Victoria 3800
Australia
(gopal@infotech.monash.edu.au)

and

A. Singla[1]
Indian Institute of Technology
New Delhi, India
(cs1040157@cse.iitd.ernet.in)

Abstract

Similarity detection is an interesting and challenging problem. It is challenging since the aim often is to compare a very large number of documents efficiently and without excessive storage overheads. Very large numbers of documents needs to be compared since it is believed that as many as about one-third of Web documents are either identical or similar to other pages on the Web. There are other applications of similarity detection. For example, an obvious application is plagiarism detection. Many similarity detection algorithms are based on using fingerprinting in which a fingerprint of each document is built from a set of substrings of the document. These fingerprints can then be compared to detect similarity between documents. We present a new fingerprinting algorithm, called Sequence of Selected Words Fingerprint (or SSWF), that not only uses a set of selected words from the document but also the sequence in which they appear. This approach is shown to work well for the *1*-to-*n* similarity detection problem in which a document is given and we wish to find similar documents in a given collection.

---

[1] This research was carried out during an internship visit by A. Singla to Monash University. Financial support for the visit by Monash is gratefully acknowledged.

1. Introduction

There is a proliferation of similar or identical documents on the Web. It has been found that almost 30% of all Web pages are very similar to other pages and about 22% (much of the 30%) are virtually identical. There are many reasons for identical or similar Web pages. For example, FAQs on important topics and online documentation of popular software may be duplicated for local use to reduce traffic. A paper written by several authors may be available on the Web site of each of them. There might even be several versions of the same paper on some of those sites as the paper undergoes revisions. News articles are often duplicated in newspapers when they are obtained from the same source, for example Reuters. And finally, student assignments including programs, although not on the Web, often have similarity because of plagiarism. In some cases, documents are not identical because different formatting might be used at different sites or a student may have made changes to a plagiarized document to hide plagiarism. Furthermore, a large Web document may be split into smaller documents or a composite document may be joined together to build a single document.

The focus of our study is not limited to Web documents. We are interested in comparing all text documents and finding those that are identical or similar or have significant overlap. Our study is focused only on content-based similarity which is based on comparing the textual content of the documents syntactically.

In document similarity studies, some research aims to analyze a large collection of documents to find those that are similar. This is an *n*-to-*n* problem. Others have studied the problem in which one document is compared with documents from a given collection. This is a *1*-to-*n* problem. Our focus is on the latter problem although we briefly discuss how to extend the proposed technique to the *n*-to-*n* problem.

Broder (1997) defines concepts of resemblance and containment. Resemblance of two documents is defined to be a number between 0 and 1 with 1 indicating that the two documents are identical. Containment of one document in another is also defined as a number between 0 and 1 with 1 indicating that the first document is completely contained in the second.

There are a number of issues that must be considered in document matching. Firstly, if we are looking to compare millions of documents then the additional storage requirement for each document using the similarity analysis method should not be large. Secondly, the method should be robust, that is, it should not be possible to circumvent the matching process with modest changes to a document. Schleimer, Wilkerson and Aiken (2003) suggest that a similarity-detection algorithm should have three properties, namely, whitespace insensitivity, noise suppression and position independence. The last one requires that any scrambling of the order of paragraphs should not impact the resemblance of a document pair. This requirement is somewhat questionable since a document and another that has scrambling of its paragraphs certainly are not the same document and therefore a resemblance of 1 for two such documents is not justified.

The *1*-to-*n* similarity algorithm presented in this paper is based on building a fingerprint using a set of selected words from the document which is to be

compared with the documents in a collection. The fingerprint thus obtained not only captures how many times the selected words appear in the document but also their sequence. The fingerprint thus obtained is relatively simple, short and generally quite unique. We believe the two contributions of this paper are the method of selecting representative words from a document and representing the sequence of their occurrence in the document and in other documents by a string which is each document's fingerprint.

Central to our idea is building the fingerprint that is represented as a string of characters encoding the sequence of occurrences of a set of selected words. The representation is shown to be effective in finding identical as well as similar documents.

The paper is organized as follows. The next section briefly outlines some related previous work in the field of similarity detection. Section 3 presents the proposed approach, called the Sequence of Selected Words Fingerprinting (SSWF). Section 4 presents the results of a number of experiments to evaluate the performance of SSWF. In Section 4, we discuss how SSWF may be extended for *n*-to-*n* similarity detection of large numbers of documents and Section 6 concludes the paper.

2. Related Previous Work

Some techniques for syntactic similarity analysis are based on information retrieval techniques (also called ranking techniques) which were designed for an environment in which a user poses a query and the system finds a collection of documents relevant to the query. This is also what search engines try to do. In this approach, a similarity measure between the query and each document is found usually based on an inverted index of all documents in the collection. The technique is described in detail by Witten, Moffat and Bell (1999).

Another set of similarity analysis techniques are based on fingerprinting. For example, Manber (1994), Brin, Davis and García-Molina (1995), Shivakumar and García-Molina (1995, 1996), Heintze (1996) and Broder (1997) describe fingerprinting techniques. Essentially these techniques develop a fingerprint of each document in the collection and the fingerprints are then compared rather than the documents to find matching pairs of documents. The fingerprints are often based on words or on substrings appearing in each document.

Brin, Davis and García-Molina (1995) developed a system called COPS in the context of copyright protection based on dividing each document into chunks. A chunk is defined as a sequence of consecutive units (sections, paragraphs, sentences, or words). For fingerprinting documents they studied a number of different types of chunks including words and sentences as well as overlapping and non–overlapping chunks. The results were encouraging with about 59% ± 25% success and only about 0.6% failure. Shivakumar and García-Molina (1995, 1996) developed the technique further. They described a technique in which the number of chunks of text that documents share is counted and they then designed a system, SCAM, using words as chunks, which worked a little better than COPS but required significant additional storage, perhaps as much as 30-65% of the size of each document. SCAM still maintains a collection of articles and is available free to anyone who wishes to compare a document with those in the SCAM collection. Cho, Shivakumar and García-Molina (1997) overcome

SCAM's storage problem by hashing each chunk into a 32-bit fingerprint. The fingerprints are then compared to find similarities.

Heintze (1996) uses fingerprinting based on a set of sub-sequences of characters from the document and using their hash values. A document of length N is divided into all possible substrings of length L. These N – L + 1 substrings are called *shingles*. Assuming S(X) and S(Y) to be the sets of shingles for documents X and Y respectively, resemblance R(X, Y) and containment C(X, Y) between the two documents may be defined as follows (Broder, 1997):

$$R(X, Y) = \{S(X) \cap S(Y)\} / \{S(X) \cup S(Y)\},$$

and

$$C(X, Y) = \{S(X) \cap S(Y)\} / \{S(X)\}$$

where {.} represents the cardinality of the set. An algorithm like the following may now be used to find similar documents from a collection:

1. Choose a suitable shingle width and compute the shingles for each document
2. Build a fingerprint for each document based on hashed addresses of (a subset of) its shingles
3. Compare the fingerprints for each pair of documents
4. Identify those documents that are similar based on the number of fingerprints that match

In all fingerprinting techniques, the fingerprint must be small enough not to require too much additional storage and be efficient. To reduce the size of the fingerprints, Heintze suggested that only hash values of some of the shingles be used. One possibility was to only use shingles with the smallest hash values. It is claimed that only about 500 bytes is needed to represent each fingerprint. Broder (1997) outlines an approach that is similar to this and Broder *et al* (1997) use fingerprinting to build clusters from 30 million documents with 50% resemblance. They found 3.6 million clusters with 12.3 million documents, most clusters containing identical documents. Using a shingle size of 10 words, they found a typical document to have about 1000 distinct shingles. Eliminating the most common shingles that were shared by more than 1000 documents, fingerprints were built based on using only 1 in every 25 shingles. Each fingerprint was then represented by about 300 to 400 bytes.

Chowdhury *et al* (2002) describe a duplicate detection algorithm that can scale to the size of the Web and handle short documents effectively. One of their aims is to help improve search engine performance. The algorithm proposed involves collecting statistics about the given documents (inverse document frequency or *idf* for each term is computed) and then using the *idf* to filter the document, after removing the most common and most infrequent terms. They then derive a single 160-bit hash value for each document and use a number of document collections to evaluate their algorithms.

Schleimer, Wilkerson and Aiken (2003) suggest a number of modifications to the basic fingerprinting method including selecting shingles of sufficient length so that short uninteresting shingles are not included, selecting only shingles with hashes that are $0 \bmod p$ for some suitable p to reduce storage overheads and ensuring that no gap between successive shingles is too large. A concept of a

document window of some suitable size is defined so that at least some part of each window is included in its fingerprint.

Hoad and Zobel's (2003) study compared a single document to a collection of documents to find its coderivatives (documents that originate from the same source). They describe three techniques: ranking, identity measure and fingerprinting several versions and compare their performance on two sets of test documents.

The ranking technique, briefly discussed earlier, is used to find approximate matches to queries on text databases and is not designed to detect coderivatives. A similarity measure is used with three different (inner product, normalized inner product and cosine measure) described and tested. The second technique, called "identity measures", an extension of ranking, is based on the intuition that similar documents should contain a similar number of occurrences of words and be of similar length. A number of different versions of this technique assigning various relative weights to difference in term frequencies and difference in document length are presented. Finally, several versions of fingerprinting techniques based on using a hashing like technique for mapping selected substrings from the document to integers are compared.

It should be noted that no syntactic similarity method that finds two documents to be similar can guarantee that the documents are indeed similar since most techniques, to save on storage and CPU time, use only a small fraction of the documents to analyze similarity. In most cases, one could carefully doctor one of the documents that was found similar to another by a given method without the method detecting that doctoring.

3. Proposed SSWF Approach

SSWF Algorithm

The techniques presented above extract chunks or substrings from each document and then compare (some of) them with those from others. We believe a good fingerprint should not only be based on chunks or substrings that are in the document but also the sequence in which they appear. The proposed approach, which we call Sequence of Selected Words Fingerprint (or SSWF), does that. We now describe the SSWF algorithm for the *1*-to-*n* problem.

1. First find all the words in the given document (call it the *base document* or BD) using an algorithm given in Witten, Moffat and Bell (1999). From these words, select a set of words S, perhaps 50 to 100, as described later in this section.

2. Assign a code to each word in S. For illustration, using only a small set S, we choose the codes to be the letters of the alphabet.

3. Build a string fingerprint for the base document by scanning the document and representing each selected word when encountered in the text by its code. For example if the selected words were "words", "document", and "fingerprint" and their codes were A, B and C respectively then the fingerprint of the text in this algorithm so far is

ABBAACBBAABCC. We have assumed no stemming and therefore "words" does not match with "word".

4. Build fingerprints using the same method for each document in the document collection.

5. Compare the BD's fingerprint with the fingerprint of each document by computing distance between them using a string comparison algorithm. Declare the level of similarity based on the distance compared to the length of the fingerprints.

We can find the distance between two strings by finding the number of symbols that need to be changed or deleted in either string to make them identical. Given that the document's fingerprint is based on a set of words, the probability that some words appearing in the BD will appear as noise in the second document is high. To remove such noise from fingerprint matching, we identify any isolated matching symbols or isolated two matching symbols and consider them noise which is not counted as a match. Furthermore, any match of three successive symbols that is separated from other matches by at least two symbols shows that although three matching words were found, at least two words on each side of the match were not. As an example consider the two strings and positions of their symbols in Table 1. For convenience we have assumed that the strings are of equal length. If the matches between the two strings are as follows:

S1 Position 1 = S2 Position 3
S1 Position 6 = S2 Position 6
S1 Position 7 = S2 Position 7
S1 Position 8 = S2 Position 8
S1 Position 9 = S2 Position 11
S1 Position 10 = S2 Position 12

All these matches will be considered noise and not considered matches. The similarity between the two fingerprints will be zero.

Position	1	2	3	4	5	6	7	8	9	10	11	12	13
String S1	A	B	B	A	A	C	B	B	A	A	B	C	C
String S2	C	C	A	C	C	C	B	B	B	B	A	A	C

Table 1. Identifying Noise in Fingerprint Matching

We believe the number of symbol matches after removing the noise matches is a good indication of similarity between the two fingerprints.

SSWF builds a document's fingerprint by encoding some selected words in the sequence they appear in the document. We believe SSWF is a short and generally unique signature of the document which is unlikely to match the fingerprint of a dissimilar document.

Selecting the Set of Words

A number of considerations need to be taken into account when selecting the words S. These are:

1. The set S must be reproducible for the same document. S therefore cannot be selected randomly.
2. The words in S should not be very common in the document (otherwise the fingerprint will be long), nor should they be appearing only once, for example misspelt words (otherwise the fingerprint alphabet will be large). We believe to capture the essence of a document, of say 5000 words, S should include at least 50 words but not more than 100.
3. The words in S should, if possible, be fairly evenly "peppered" through the document so that the fingerprint obtained captures the whole document without leaving large gaps. We believe each such word should at least appear three times in the document and should not appear more than say 25 times although the frequency may vary with the length of the document.
4. The fingerprint should have appropriate granularity. The granularity in our representation is based on the number of words selected and their frequency in the document. If the fingerprint is too large, it will require significant storage and be costly to compare with other fingerprints. To find a good balance between a good document signature and matching efficiency, we believe the fingerprint should be neither very small nor very large.

Fingerprints in the range of 100 to 200 symbols in length are perhaps the best although a very large document might require a longer fingerprint. A fingerprint of length 200 implies that for a document of 10,000 words, one symbol on the average is likely to occur in every 50 words or about ten symbols per page of the document. For larger documents, say 100,000 words (about 200 pages), only one symbol in a fingerprint of 200 symbols will on average occur in each page. Our experiments, described in the next section, show that even small fingerprints relative to the size of a document can be effective in similarity detection. There is of course the option of breaking a large document into smaller documents with separate fingerprints for each part.

If we wish to build fingerprints of about 200 symbols and want at least 50 distinct words in S, it follows that each word in S should on average appear in a document perhaps three to five times.

Similarity Measure

Earlier we presented definitions for resemblance and containment. We will use the slightly modified definitions given below.

Let the lengths of the fingerprints of the two documents X and Y be P and Q respectively and let P be longer than Q. Let the number of matching symbols between the two fingerprints be M. We define identical documents (resemblance = 1) to be those that have fingerprints that match exactly, that is, P = M. Documents that are not identical but very similar should also get a score close to 1 based on the number of symbols matched compared to P. In defining containment we compare the number of symbols matched to Q, the length of the smaller fingerprint. Resemblance and containment may therefore be defined as follows:

$$R(X, Y) = M/P$$

and

$$C(X, Y) = M/Q$$

4. Performance Evaluation

To evaluate the effectiveness of the SSWF technique we have carried out a number of experiments that are described below. The documents used in these experiments consist of news reports, technical reports, Web pages and documentation for the UNIX operating system.

In the experiments we will use the following *1*-to-*n* matching algorithm which is based on the SSWF description given above. We assume that the BD is large enough so that S with 70 words can be selected from it. Smaller documents, for example Web pages or news articles, need to be handled slightly differently by selecting a smaller number of words. We will discuss this later.

1. Given *n* documents, carry out case folding and remove all stop words.
2. Select one of these documents randomly to be the BD.
3. Find all the words and their frequencies in the BD. Sort the words in increasing order of frequency. From these words, select a set of 70 words starting from those that have a frequency of 3. Continue with words of frequency 4 and 5 if necessary.
4. Assign a code to each word selected using the ASCII codes.
5. Build a fingerprint as a string of symbols for the document by scanning it and representing each word in S when encountered in the document by its code.
6. Using the same method, build a fingerprint for each document in the collection.
7. Compare the BD's fingerprint with the fingerprint of each document using a string comparison algorithm. The algorithm used is based on the work of Wagner and Fischer (1974), modified to eliminate noise matches of length one, two and three as discussed earlier. Compute the number of symbols that match between the two documents.
8. Compute the resemblance and containment between the BD and every other document in the collection.

The above algorithm was used for the following experiments.

Experiment Set 1 – Comparing a technical report with other technical reports in a collection

A collection of Hewlett-Packard technical reports is available from the HP Web site. These PDF documents were converted to text and one of the technical reports was randomly chosen as the BD to be used in a *1*-to-*n* similarity analysis. Since the reports deal with a variety of topics, it was expected that the resemblance between them will be low.

There were a total of 40 technical reports. Results of comparing a BD with ten of them are presented in Table 2. The remaining results were similar. For each document that the BD was compared with, the table presents its length, its fingerprint's length, the number of matching symbols and the distance between

the two fingerprints. Overall average resemblance found by SSWF between the BD and each of the 39 reports was 0.37% and containment 0.75%. For one technical report (Document 9), resemblance was almost 14% and containment over 28%. For all other documents resemblance and containment were 0%. We checked document 9 and the BD and found that the authors for the reports were the same and the topic of Document 9 and the BD was the same. In fact, Document 9 included some of the BD in the first half of the report.

Length of the Base Document = 23590 Characters						
Fingerprint Length of the Base Document = 211 Characters						
Document Number	Length of Document in Characters	Fingerprint Length in Characters	Number of Matching Symbols	Distance Between the Two Fingerprints	Resemblance	Containment
1	36127	107	0	185	0.00%	0.00%
2	40467	140	0	190	0.00%	0.00%
3	29790	125	0	191	0.00%	0.00%
4	50585	116	0	189	0.00%	0.00%
5	32221	65	0	187	0.00%	0.00%
6	63746	132	0	190	0.00%	0.00%
7	43073	93	0	189	0.00%	0.00%
8	48980	380	0	335	0.00%	0.00%
9	60170	429	60	333	13.99%	28.44%
10	41226	127	0	191	0.00%	0.00%

Table 2. Results of Experiment with a Technical Report Collection

These results show that SSWF is effective in identifying similarity in documents. The results are based on using S with 70 words of length 3 and 4 and 5 when required. Since 3, 4 and 5 were chosen based on our intuition we decided that it was appropriate to try other word frequencies. We used 6, 7 and 8 in some experiments and found that to make fingerprints of length about 200 we needed only 25-30 words in S. The performance using higher frequency words was similar to Table 2 although the number of non-zero resemblance values was higher and the average resemblance was 1.1% compared to 0.57% for words with frequencies 3, 4 and 5. It appears that the likelihood of noise matches is somewhat higher if higher frequency words are used.

Experiment Set 2 – Comparing individual pages of the same document

In this experiment, we compared individual pages of the same technical report expecting to find higher resemblance between them since each of them deals with the same subject matter. The BD used was the front page of the report which included the report's abstract. It was compared with each remaining page of that technical report.

Five technical reports of 12, 20, 37, 12 and 7 pages respectively were compared page by page with the BD from the corresponding technical report. Typical results are presented in Table 3 for two technical reports, the first five results are from one report and the second five from another. Clearly the first page after the front page had similarity with the front page but there was little similarity between the front page and the remaining pages.

As discussed earlier, for small documents it is often not possible to find a set S with as many as 70 words after removal of stop words and frequent words. We therefore decided to use all the words that occurred in the BD 3, 4 or 5 times which often resulted in 30-40 words and therefore smaller fingerprints.

Document Number	Length of Document in Characters	Fingerprint Length in Characters	Number of Matching Characters	Distance Between the Two Fingerprints	Resemblance	Containment
Length of Base Page for the First report = 3862 Characters Fingerprint Length of the Base Page for the First report = 172 Characters						
1	1512	79	53	132	30.81%	67.09%
2	3298	53	0	151	0.00%	0.00%
3	2786	47	0	156	0.00%	0.00%
4	4595	69	0	152	0.00%	0.00%
5	4635	66	0	152	0.00%	0.00%
Length of the Base Page for the Second report = 4268 Characters Fingerprint Length of the Base Page for the Second report = 126 Characters						
1	1515	44	18	119	11.54%	40.91%
2	4322	60	0	139	0.00%	0.00%
3	4753	73	0	141	0.00%	0.00%
4	4369	67	0	136	0.00%	0.00%
5	2589	3	0	153	0.00%	0.00%

Table 3. Results of Experiments with Individual Pages of Two Technical Reports

Experiment Set 3 – News reports from the Web

A number of news articles were collected by searching Google using a query with two keywords, Bush and Iraq, and the first 30 documents were selected. A document was randomly chosen as the BD. Since the reports dealt with the same topic, we expected the resemblance between the BD and the others to be high.

As in the last experiment, we used all the words that occurred in the BD with a frequency of 3, 4 or 5 which often resulted in smaller fingerprints. Results obtained for ten news reports are given in Table 4.

Length of Base Page = 7829 Characters						
Fingerprint Length of Base Page = 186 Characters						
Document Number	Length of Document in Characters	Fingerprint Length in Characters	Number of Matching Characters	Distance Between the Two Fingerprints	Resemblance	Containment
1	4102	28	0	172	0.00%	0.00%
2	4757	27	0	172	0.00%	0.00%
3	6952	43	0	168	0.00%	0.00%
4	19444	179	0	177	0.00%	0.00%
5	11500	110	0	165	0.00%	0.00%
6	4181	35	0	168	0.00%	0.00%
7	11467	73	0	167	0.00%	0.00%
8	8719	52	0	168	0.00%	0.00%
9	7852	31	0	171	0.00%	0.00%
10	4858	33	0	170	0.00%	0.00%

Table 4. Results of Experiments with News Articles retrieved by Google Search Engine

The results were somewhat surprising, so we examined all the documents and found that there was little similarity between them. Some discussed the economics of the war; some discussed weapons of mass destruction while others reported President Bush's speeches which were not similar to the other documents.

Experiment Set 4 – Documents with rearranged text

In these experiments, we rearranged a given document by moving parts of it to elsewhere in the document. These rearranged documents therefore had the same words as the original document but the sequence of sections of the document was different. The parts moved varied from 10% of the document to 50%. A variety of documents were used in the tests. The results in Table 5 show results of modifying a page of UNIX documentation. The resemblance was similar to the larger portion of the document that was still together. As in the last two sets of experiments, we used all the words that occurred in the BD with a frequency of 3, 4 or 5. Only these 30-50 words were selected to build fingerprints.

Length of Each Document = 9606 Characters					
Length of Each Fingerprint = 192 Characters					
Document Number	Fraction of Document Moved	Distance Between Fingerprints	Number of Matching Symbols	Resemblance	Resemblance by Splitting the Fingerprint
1	10%	24	181	94.27%	83.10%
2	20%	63	161	83.85%	82.63%
3	30%	94	143	74.48%	83.10%
4	40%	128	129	67.19%	81.22%
5	50%	178	104	54.17%	84.04%

Table 5. Results of Experiments with Portions of a Document Rearranged Within the Document

These results show that sequence of text is important in SSWF similarity detection. Although the words in all the documents in Table 5 were the same, the rearrangement resulted in resemblance values much below 100%.

We can overcome this difficulty by dividing the BD's fingerprint into say 10 equal pieces and then matching each piece with the fingerprint of the target document. Each piece is able to find a match in the rearranged document and the total number of matches should be close to the length of the fingerprint. We tried this approach of dividing the fingerprint into 10 and found that resemblance of over 80% was obtained for all rearranged documents. The results were not closer to 100% since smaller pieces often found partial matches in other parts of the document. We are studying how SSWF may be modified so that more accurate estimates of resemblance are obtained.

Experiment Set 5 – Evaluating shorter fingerprints

Although results of the last two sets of experiments show that shorter fingerprints can be effective for smaller documents, we conducted a set of

experiments to compare performance of shorter fingerprints with longer fingerprints. Shorter fingerprints were obtained by reducing the number of words in S. While the experiments with technical reports discussed above selected 70 words, these experiments selected only 25 words. Table 6 shows the results of using the shorter fingerprints on the same technical reports as in Table 2. The results show that even the shorter fingerprints are effective although the resemblance values are a bit higher (average resemblance = 1.1 compared with 0.37 with longer fingerprints).

Length of the Base Document = 23590 Characters						
Fingerprint Length of the Base Document = 76 Characters						
Document Number	Fingerprint Length using 25 Words	Fingerprint Length using 70 Words	Number of Matching Symbols with 25	Number of Matching Symbols with 70	Resemblance with 25 Words	Resemblance with 70 Words
1	16	107	0	0	0.00%	0.00%
2	25	140	3	0	3.95%	0.00%
3	15	125	0	0	0.00%	0.00%
4	32	116	0	0	0.00%	0.00%
5	18	65	0	0	0.00%	0.00%
6	19	132	0	0	0.00%	0.00%
7	25	93	0	0	0.00%	0.00%
8	97	380	0	11	0.00%	2.89%
9	97	429	20	81	20.62%	18.88%
10	25	127	0	0	0.00%	0.00%

Table 6. Results of Experiments Comparing Fingerprints of Two Different Lengths

Experiment Set 6 – Making changes to the given document

In this set of experiments, we removed a section of a document and replaced it with a section of similar length from another unrelated document. This was done to test the proposed technique for plagiarism detection since it is commonly believed that plagiarism involves cutting and pasting sections of documents from the Web. The results show that when part of a BD is combined with part of another document, the fingerprinting method correctly finds the resemblance. As a typical example, when the section replaced from a document was 50%, 40%, 30%, 20% and 10%, the resemblance discovered using SSWF was 49%, 35%, 29%, 16% and 8% respectively.

Experiment Set 7 – Comparing short Web documents

In this set of experiments, we collected several hundred Web documents by crawling the Monash University Web site. These documents were often quite small, some as small as 100 words while others were over 4000 words. Selecting one base document we compared it with 325 other Web pages. Typical results are given in Table 7.

Document Number	Signature Length	Distance Between Fingerprints	Number of Matching Symbols	Resemblance	Containment
2065	26	29	3	7.89%	11.54%
2426	23	28	4	10.53%	17.39%
2551	23	26	8	21.05%	34.78%
5725	39	29	4	10.26%	10.53%
1327	9	30	0	0.00%	0.00%
7313	52	41	4	7.69%	10.53%
8446	39	28	4	10.26%	10.53%
2308	21	25	9	23.68%	42.86%
3795	19	30	4	10.53%	21.05%
6621	16	28	0	0.00%	0.00%

Table 7. Similarity Detection amongst Short Web Documents

Most of the matches between the base document and the other Web documents are four symbols representing words like Monash University. Only two documents in Table 7 show more than four symbols similarity. We found that all such documents were about introduction to various faculty units in the University and involved very similar templates describing each faculty.

5. Extending the Technique to Large Document Collections

It is possible to extend the proposed SSWF technique to *n*-to-*n* similarity analysis of large document collections. A simple approach could be to build a fingerprint for each document by selecting the set of words S for it as described at the beginning of Section 4. This will result in fingerprints for each different document based on a different S. The fingerprints can now be compared to find identical documents. This approach is unlikely to be successful for similar documents that are not identical. We have experimented with this approach and have found that it works satisfactorily for small collections.

We are studying another approach that involves selecting the representative set of words S from a more generic collection of English words. An Oxford dictionary site (askoxford.com) provides a variety of lists of English words including the most commonly used 1000 words and most commonly used nouns, verbs and adjectives. We propose to use the same S which consists of 75 most frequently used nouns, verbs and adjectives to build a fingerprint for each document in the collection we are analysing. To make it faster to compare documents, the fingerprints so obtained may be hashed to a large hash table (say 2^{32} for a table of almost a billion). A new document can then be similarly fingerprinted and hashed and compared to all the documents that are hashed to that address.

6. Conclusions and Further Work

We believe the two contributions of this paper are the method of selecting representative words from a document and representing the sequence of their occurrence in the document and in other documents by a string which is each document's fingerprint. This new fingerprinting technique SSWF has been shown

to be effective in determining similarity by using 70 words with frequency of at least three. The technique is robust since shorter fingerprints using 25 words also work well. We have also shown that SSWF performs well for small documents (say, one page) as well as larger documents (we have tested documents only up to 100 pages long). The weakness of the SSWF approach is that although it is suitable for *1*-to-*n* document comparisons, it currently is not suitable for *n*-to-*n* similarity analysis. We are studying and evaluating a number of different ways to modify the basic SSWF approach to make it effective for *n*-to-*n* similarity detection.

Acknowledgements

The authors wish to thank Professor B. Srinivasan and Professor H. Schmidt who read an earlier version of this paper and made a number of useful suggestions.

References

S. Brin , J. Davis and H. García-Molina, Copy detection mechanisms for digital documents, Proceedings of the ACM SIGMOD Conference on Management of Data, San Jose, California, pp 398–409, May 1995.

A. Broder, On the resemblance and containment of documents, in Proceedings of the Compression and Complexity of Sequences, p.21–29, June 1997.

A. Broder, Identifying and filtering near-duplicate documents, in Combinatorial Pattern Matching, 11th Annual Symposium, CPM 2000, Montreal, Canada, June 21-23, 2000, R. Giancarlo, D. Sankoff (Eds.), Lecture Notes in Computer Science 1848 Springer 2000, pp 1-10.

A. Broder. Filtering Near-duplicate Documents. Proceedings of FUN 98, FUN with Algorithms, An International Conference, June 18-20, 1998, Elba Island, Italy.

A. Broder , S. C. Glassman, M. S. Manasse and G. Zweig, Syntactic clustering of the Web, Selected Papers from the Sixth International World Wide Web Conference, *Computer Networks and ISDN Systems*, Vol 29, No 8–13, pp 1157–1166, 1997.

J. Cho, N. Shivakumar and H. García-Molina, Finding replicated Web collections, Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 355–366, 2000.

A. Chowdhury, O. Frieder, D. Grossman, and M. C. McCabe, Collection statistics for fast duplicate document detection, *ACM Transactions on Information Systems (TOIS)*, Vol 20, No 2, pp 171–191, 2002.

N. Heintze, Scalable document fingerprinting, USENIX Workshop on Electronic Commerce, 1996.

T. C. Hoad and J. Zobel, Methods for identifying versioned and plagiarized documents, *Journal of the American Society for Information Science and Technology*, Vol 54, No 3, pp 203–215, 2003

U. Manber, Finding similar files in a large file system, in Proceedings of the Winter 1994 USENIX Technical Conference, San Francisco, pp 1–10, 1994.

S. Schleimer, D. S. Wilkerson and A. Aiken, Winnowing: local algorithms for document fingerprinting, Proceedings of the ACM SIGMOD Conference on Management of Data,pp 76–85, June 2003.

N. Shivakumar and H. García-Molina, SCAM: a copy detection mechanism for digital documents, in Proceedings of the Second International Conference in Theory and Practice of Digital Libraries (DL'95), Austin, Texas, June 1995.

N. Shivakumar and H. García-Molina, Building a scalable and accurate copy detection mechanism, in Proceedings of the First ACM Conference on Digital Libraries (DL'96), Bethesda, Maryland, pp 160–168, March 1996.

R. A. Wagner and M. J. Fischer, The string-to-string correction problem, *Journal of the ACM*, Vol 21, No 1, pp 168–173, 1974.

I. H. Witten, A. Moffat, and T. C. Bell, Managing gigabytes: compressing and indexing documents and images, Morgan Kaufmann Publishing, San Francisco, 1999.