Fingerprint-based Similarity Search and its Applications

Benno Stein
Sven Meyer zu Eissen
Bauhaus-Universität Weimar

Abstract

This paper introduces a new technology and tools from the field of text-based information retrieval. The authors have developed

- a fingerprint-based method for a highly efficient near similarity search, and
- an application of this method to identify plagiarized passages in large document collections.

The contribution of our work is twofold. Firstly, it is a search technology that enables a new quality for the comparative analysis of complex and large scientific texts. Secondly, this technology gives rise to a new class of tools for plagiarism analysis, since the comparison of entire books becomes computationally feasible.

The paper is organized as follows. Section 1 gives an introduction to plagiarism delicts and related detection methods, Section 2 outlines the method of fuzzy-fingerprints as a means for near similarity search, and Section 3 shows our methods in action: It gives examples for near similarity search as well as plagiarism detection and discusses results from a comprehensive performance analyses.

1 Plagiarism Analysis

Plagiarism is the act of claiming to be the author of material that someone else actually wrote (Encyclopædia Britannica 2005), and, with the ubiquitousness

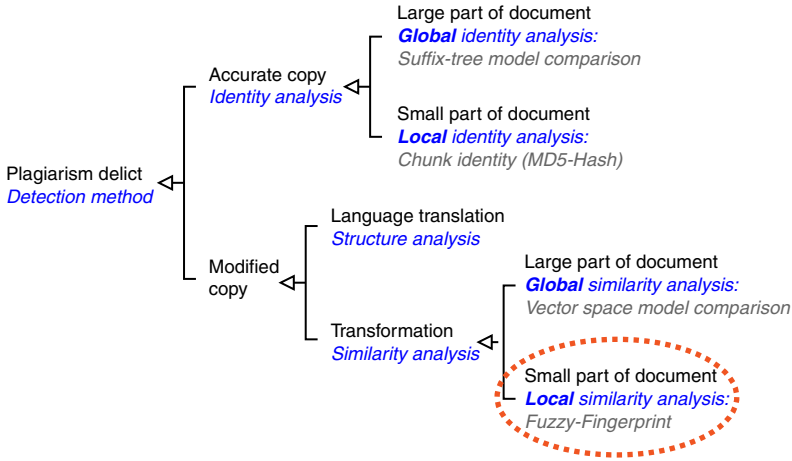

Fig. 1: A taxonomy of plagiarism delicts and analysis methods (Stein & Meyer zu Eißen 2006). The encircled part indicates the most common delict, which can be discovered with fuzzy-fingerprints.

of the World Wide Web it became more common (McCabe 2005). Plagiarism in text documents occurs in several forms: passages are copied one-to-one, passages are modified to a greater or lesser extent, or they are even translated. Clearly, a question of central importance is whether the detection of such and similar delicts can be automated. Figure 1, which is taken from (Stein & Meyer zu Eißen 2006), shows a taxonomy of plagiarism delicts along with possible detection methods. The by far most common plagiarism delict is the extraction of small parts of other authors' documents and their use in a more or less modified form within the own text (shown encircled).

Several techniques for plagiarism analysis have been proposed in the past—most of them rely on one of the following ideas:

Substring Matching. Substring matching approaches try to identify maximal matches in pairs of strings (Gusfield 1997), which then are used as plagiarism indicators. Typically, the substrings are represented in suffix trees, and graph-based measures are employed to capture the fraction of the plagiarized sections (Baker 1993, Monostori, Finkel, Zaslavsky, HodÃąsz & Pataki 2002, Monostori, Zaslavsky & Schmidt 2000). However, Finkel, Zaslavsky, Monostori & Schmidt as well as Baker propose the use of text compression algorithms to identify matches (2002, 1993).

Keyword Similarity. The idea here is to extract and to weight topic-identifying keywords from a document and to compare them to the keywords of other

documents. If the similarity exceeds a threshold, the candidate documents are divided into smaller pieces, which then are compared recursively (Si, Leong & Lau 1997, Fullam & Park 2002). Note that this approach assumes that plagiarism usually happens in topically similar documents.

Exact Fingerprint Match. The documents are partitioned into term sequences, called chunks, from which digital digests are computed that form the document's fingerprint. When the digests are inserted into a hash table, collisions indicate matching sequences. For the fingerprint computation a standard hashing approach such as MD5 hashing is employed, which suffers from two severe problems: (1) it is computationally expensive, (2) a small chunk size (3-10 words) must be chosen to identify matching passages, which additionally increases the effort for fingerprint computation, fingerprint comparison, and fingerprint storage. Recent work that describes details and variants of this approach are given in (Brin, Davis & Garcia-Molina 1995, Shivakumar & Garcia-Molina 1996, Finkel, Zaslavsky, Monostori & Schmidt 2002).

Our approach of fuzzy-fingerprinting overcomes these limitations; it is ideally suited to discover copied and slightly modified passages in large document collections. The technology was firstly published in (Stein 2005) and successfully applied to plagiarism analysis in (Stein & Meyer zu Eißen 2006). However, to understand different intentions for similarity search and its application we first introduce the distinction of local and global similarity. In fact, fuzzy-fingerprints can be understood as a combination of both paradigms, where the parameter "chunk size" (substring size) controls the degree of locality.

2 Similarity Search with Fuzzy-Fingerprints

In the context of information retrieval a fingerprint $h(d)$ of a document d can be considered as a set of encoded substrings taken from d, which serve to identify d uniquely.[1] Following Hoad & Zobel, the process of creating a fingerprint comprises four areas that need consideration (2003).

1. *Substring Selection.* From the original document substrings (chunks) are extracted according to some selection strategy. Such a strategy may consider positional, frequency-based, or structural information.
2. *Substring Number.* The substring number defines the fingerprint resolution. Obviously, there is a trade-off between fingerprint quality, processing effort, and storage requirements, which must be carefully balanced. The more information of a document is encoded in the fingerprint, the more reliably a possible collision of two fingerprints can be interpreted.

[1] The term "signature" is sometimes also used in this connection.

3. *Substring Size.* The substring size defines the fingerprint granularity. A fine granularity makes a fingerprint more susceptible to false matches, while with a coarse granularity fingerprinting becomes very sensitive to changes.
4. *Substring Encoding.* The selected substrings are mapped onto integer numbers. Substring encoding establishes a hash operation where—aside from uniqueness and uniformity—also efficiency is an important issue (Ramakrishna & Zobel 1997). For this, the popular MD5 hashing algorithm is often employed (Rivest 1992).

If the main issue is similarity analysis and not unique identification, the entire document d is used during the substring formation step—i. e., the union of all chunks covers the entire document. The total set of integer numbers represents the fingerprint $h(d)$. Note that the chunks may not be of uniform length but should be formed with the analysis task in mind.

2.1 Local and Global Similarity Analysis

For two documents A and B let $h(A)$ and $h(B)$ be their fingerprints with the respective cardinalities $|h(A)|$ and $|h(B)|$. A similarity analysis between A and B that is based on $h(A)$ and $h(B)$ measures the portion of the fingerprint intersection (Finkel et al. 2002):

$$\varphi_{local}(A, B) = \frac{|h(A) \cap h(B)|}{|h(A) \cup h(B)|}$$

We call such a kind of similarity measure *local similarity* or *overlap similarity*, because it directly relates to the number of identical regions. By contrast, the vector space model along with the cosine measure does not depend on identical regions: Two documents may have a similarity of 1 without sharing any 2-gram. The vector space model along with the cosine measure assesses a global characteristic because it quantifies the term frequency of the entire document; in particular, the model neglects word order. Figure 2 contrasts the principles of local and global similarity analysis pictorially.

Basically, a fingerprint $h(d)$ of a document d is a special document model of d. In this sense, every information retrieval task that is based on a standard document model can also be operationalized with fingerprints. However, fingerprint methods are more flexible since they can be targeted specifically towards one of the following objectives:

1. compactness—with respected to the document length
2. fidelity—with respected to a local similarity analysis

It is difficult to argue whether a fingerprint should be preferred to a standard document model in order to tackle a given retrieval task. To better understand this problem of choosing an adequate document model consider again

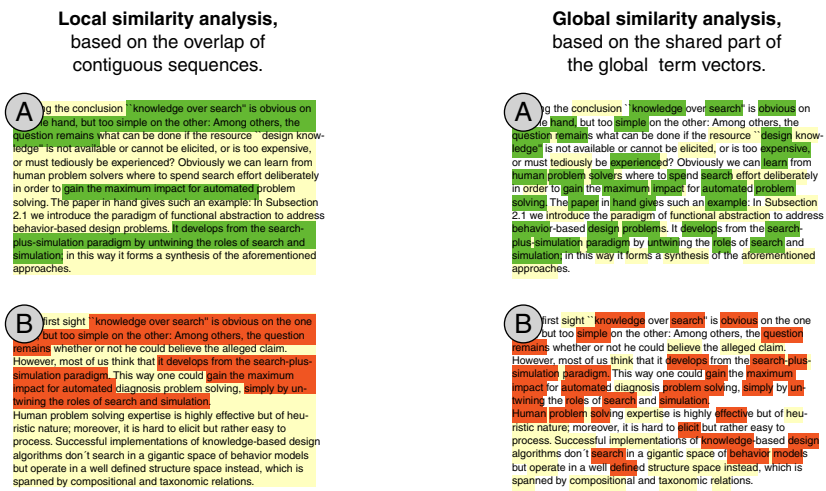

Fig. 2: Two documents A and B which are analyzed with respect to their similarity. The left-hand side illustrates a measure of local similarity: All matching contiguous sequences (chunks) with a length ≥ 5 words are highlighted. The right-hand side illustrates a measure of global similarity: Here the common word stems (without stop words) of document A and B are highlighted. Observe that both similarity analyses may lead to the same similarity assessment.

the taxonomy shown in Figure 1: here we have divided the analysis methods into local and global strategies. Note that in the literature on the subject local plagiarism analysis methods are encountered more often than global analysis methods. Among the shown approaches, the chunk identity analysis—usually operationalized with the MD5 hashing algorithm—is the most popular approach to plagiarism analysis. As already noted, such or similar methods have inherent disadvantages that can only be countered, if the chunk size is drastically increased. This, however, requires some kind of fingerprints that operationalize a "relaxed" comparison concept.

The following subsection addresses this problem. It introduces a fuzzy-fingerprint, h_φ, which is specifically tailored to text documents and which provides the desired feature: an efficient means for near similarity analysis.

2.2 *Fingerprints that Capture Near Similarity*

While most fingerprint approaches rely on the original document d, from which substrings are selected and given to a mathematical function, our approach can be developed simplest from a document's vector space model $\mathbf{d}$.

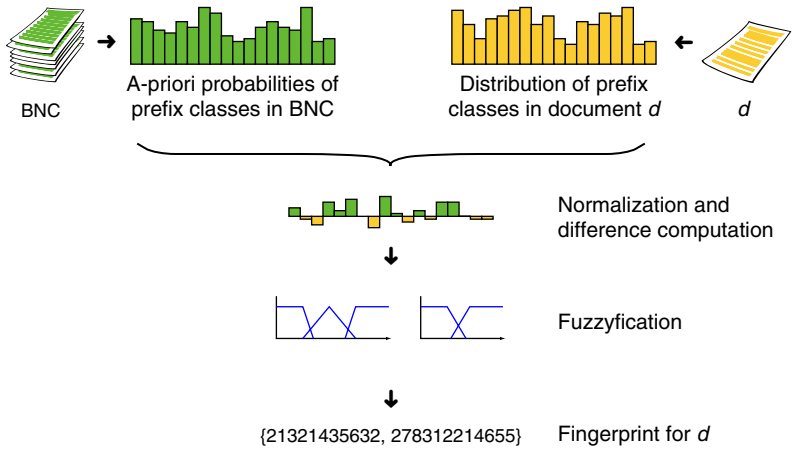

Fig. 3: Pictorial overview of the fuzzy-fingerprint construction process.

The key idea behind h_φ is an analysis and comparison of the distribution of the index terms in $\mathbf{d}$ with respect to their expected term frequency class.[2] We abstract the concept of term frequency classes towards *prefix* frequency classes, by comprising index terms into a small number of equivalence classes such that all terms from the same equivalence class start with a particular prefix. There might be the equivalence class of terms whose first character is from the set {“a”, “A”} or, as the case may be, the equivalence class of terms whose first character is from the set {“x”, “X”, “y”, “Y”, “z”, “Z”}.

Based on large corpora a standard distribution of index term frequencies can be computed and the a-priory probability of a term being member in a certain prefix class be stated. The deviation of a document’s term distribution from these a-priory probabilities forms a document-specific characteristic that can be encoded as a compact fingerprint. The following four steps define the construction of a fuzzy-fingerprint $h_\varphi(d)$ for a document $d \in D$ more precisely; Figure 3 illustrates the procedure.

1. Extraction of the set $\mathbf{d}$ of index terms from d. In connection with Web documents this includes the removal of HTML tags, scripting code, etc.

[2]The term frequency class, also called word frequency class, can be used as an indicator of a word’s customariness. Let $\mathcal{D}$ be a representative text corpus, let $|\mathcal{D}|$ be the number of words (terms) in $\mathcal{D}$, and let $f(w)$ denote the frequency of a word $w \in \mathcal{D}$. In accordance with (University of Leipzig 1995) the word frequency class $c(w)$ of a word $w \in \mathcal{D}$ is $\lfloor \log_2(f(w^*)/f(w)) \rfloor$, where w^* denotes the most frequently used word in $\mathcal{D}$. In the Sydney Morning Herald Corpus (Dennis 1995), w^* denotes the word “the”, which corresponds to the word frequency class 0; the most uncommonly used words within this corpus have a word frequency class of 19.

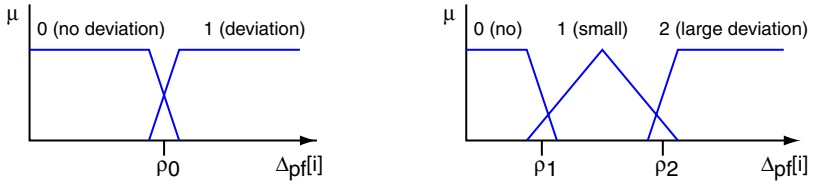

Fig. 4: The two fuzzy deviation schemes that are used for the fingerprint construction.

2. Computation of $\mathbf{pf}$, the vector of relative frequencies of k prefix classes for the index terms in $\mathbf{d}$. Our experiments rely on prefix classes that are characterized by a single alphabetic character, say, typical values for k are between 10 and 30.
3. Normalization of $\mathbf{pf}$ with respect to a reference corpus and computation of Δ_{pf}, the vector of deviations to the expected distribution. Our normalization grounds on the British National Corpus, which is a 100 million word collection of samples of written and spoken language from a wide range of sources (Aston & Burnard 1998).
4. Fuzzification of Δ_{pf} using two fuzzy deviation schemes. We propose the schemes depicted in Figure 4, which means that a deviation either falls in one of two or in one of three intervals.

Remarks. (1) The granularity of the fingerprints is controlled within two dimensions at the following places: In Step 2, by the number k of equivalence classes (= different prefix codes) to be distinguished, and in Step 4, by the resolution of the fuzzy deviation schemes. (2) Since $h_\varphi(d)$ computes a *set* of digital digests for a document d, we agree upon the following understanding of hash collisions:

$$h_\varphi(d) \cap h_\varphi(d') \neq \emptyset \;\Rightarrow\; \varphi(\mathbf{d}, \mathbf{d}') \geq 1 - \varepsilon \qquad (1)$$

(3) Finally, recall that in the vector space model all information about term order is lost. Consequently, the presented fuzzy-fingerprint approach does not encode order information either.

3 Use Cases and Performance Analysis

The purpose of this section is twofold: (1) It comprises use cases that demonstrate the wide application range of our technology, and (2) it presents analysis results that quantify different performance aspects of fuzzy-fingerprints.

3.1 Use Cases

Plagiarism in Text

The availability of educational material on the World Wide Web entices students to plagiarize from these sources. Of course, this malpractice is also observed in other situations where people can profit from plagiarized material, e. g. authors who transcribe from other papers or employees who copy

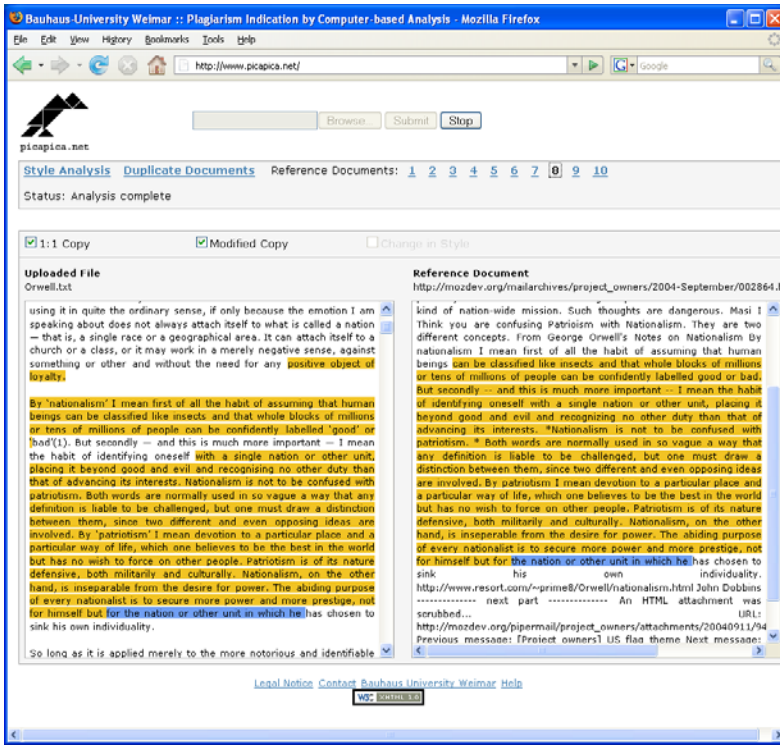

Fig. 5: The Plagiarism Finder takes an input document, automatically extracts meaningful keywords, organizes a focused Web search, and analyzes candidate documents with respect to plagiarized passages.

Fig. 6: The framed areas indicate original and plagiarized code respectively.

for their presentations. The mentioned scenarios have one aspect in common: Typically, short passages from third-party sources are slightly modified and copied into the target document. Hence they cannot be detected by traditional approaches that use cryptographic hashes. Figure 5 shows a snapshot of our Plagiarism Finder,[3] which has been developed with the fuzzy-fingerprint technology and which searches the World Wide Web for plagiarized documents.

Source Code Plagiarism

The recent lawsuit of SCO against major Linux vendors (SCO claimed that parts of Linux program sources were plagiarized from SCO Unix) shows the importance of a technology to automatically detect plagiarism at the level of program code. Since copied program source has to be adapted to fit into an existing framework, a tolerant comparison technology like fuzzy-fingerprints is necessary to identify suspicious code passages. Figure 6 shows code snippets of the same algorithm in a different context, which map onto the same fuzzy-fingerprint.

Identifying Versioned Documents

The development of a technology involves an evolution of its documentation. It is common practice to keep several versions of documents that relate to var-

[3] www.picapica.net

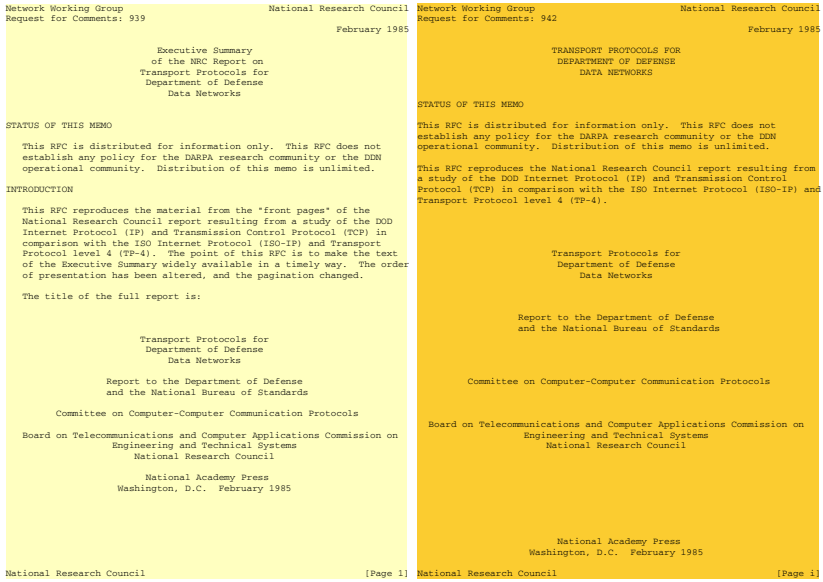

Fig. 7: Pages of two documents that represent different versions of a report about data transmission protocols. With the fingerprinting technology the respective documents are identified as variants of the same technical report.

ious software releases. Other examples for documents that develop over time include project progress reports, discussions in forums, and email threads. Fuzzy-fingerprints have proven to successfully identify versioned documents when the fingerprint is taken at the document level. Figure 7 shows an example from the RFC document collection.

Finding Similar Web Sites and Mirrors

An Internet search using the phrase “Linux Documentation Project” results in several, almost identical Web pages held at different locations, copied from each other, and revised slightly (Hoad & Zobel 2003). Of course, this phenomenon does not only apply to the mentioned project but can be observed in connection with many replicated Web pages and projects.

The identification of such sets of similar pages is useful when searching for mirror sites if a primary source is offline or overloaded. Moreover, the identification of these pages with redundant content is highly useful for search engine providers, who can optimize their storage systems using references and provide a search interface for similar Web sites. In contrast to a sequential scan of a document repository, which is expensive when the underlying

repository is huge, fuzzy-fingerprints identify similar Web sites and mirrors in constant time.

Grouping together Similar Texts

Clustering texts is the state-of-the-art methodology to identify groups of texts that share a similar subject. Fuzzy-fingerprints allow to identify groups of texts in which the pairwise similarity is above a threshold in a natural way: these groups are made up of documents that share the same fingerprint. In contrast to other methods that try to identify such groups, fuzzy-fingerprints are much faster (at least by one order of magnitude) while providing high-quality groupings.

Improvement of Text Compression

One of the most commonly used methods to compress texts is to identify common substrings and to replace them with short references. Text compression algorithms therefore hold dynamic dictionaries of frequently used substrings, which adapt according to occurrence frequency when proceeding within a text stream. Fuzzy-fingerprints can be used to group together similar texts before compressing them. This procedure allows for optimizing dictionaries for a set of texts in advance, resulting in better compression rates.

3.2 Performance Analysis

This section presents performance results of our implementation of h_φ and its application to real world similarity retrieval tasks. The purpose of our experiments is twofold. Firstly, we want to shed light on the practical performance of fuzzy-fingerprints with respect to retrieval accuracy. Secondly, we want to gain evidence on the high runtime performance of fuzzy-fingerprints in comparison with traditional approaches. The parameters of h_φ are given in Table 1.

The retrieval analysis relies on two corpora. One corpus consists of all "Request For Comments" (RFCs), a collection of about 3600 text files on In-

	Number of prefix classes k	Number of deviation intervals r	Number of fuzzification schemes ρ
h_φ	18	3	3

Tab. 1: Parameters of h_φ of the fuzzy-fingerprint. Three variations of the fuzzification scheme shown on the right-hand side in Figure 4 are used. The prefixes that define the equivalence classes are of length one, i. e. one class for each letter where the letters $\{j, k, q, u, v, x, y, z\}$ are discarded since their prefix frequencies in the English language is rather low.

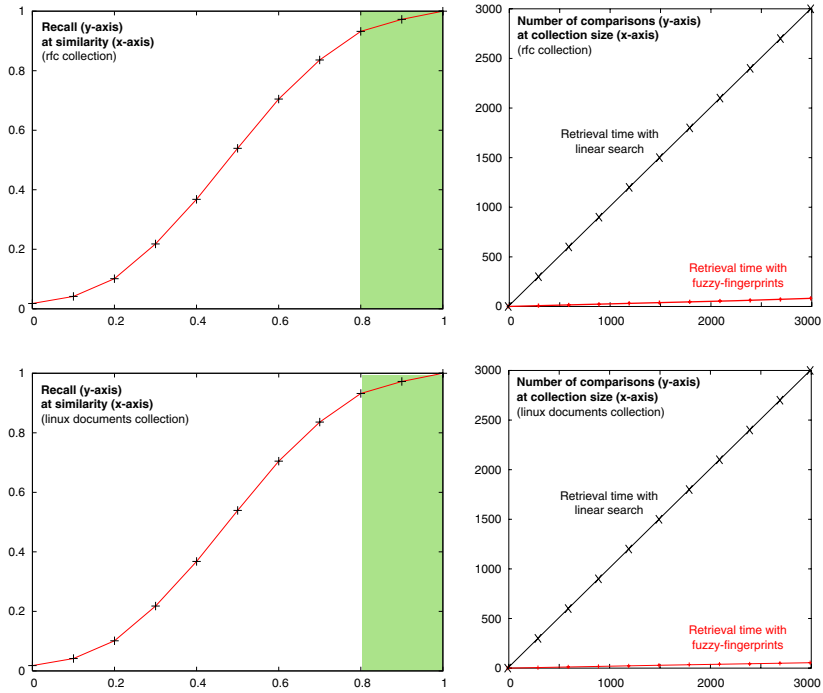

Fig. 8: The plots on the left-hand side show the recall at similarity values that were achieved for the retrieval with fuzzy-fingerprints. The plots on the right-hand side illustrate the retrieval speed up: They contrast the number of comparisons of the standard retrieval process (diagonal line) and of the fuzzy-fingerprint retrieval process.

ternet technology (Postel 2004). Since the documents in the RFC collection are versioned and thus also include updates of documents, the existence of pairs of documents with a high similarity is very likely. A second corpus is made up of about 15000 Internet documents collected with a breadth-first-search crawl starting from the "Linux Documentation Project" (Aznar 2004). For this corpus HTML documents and text documents were downloaded, the visible portion of the HTML documents were extracted, and documents with less than 50 plain words were discarded. To ensure that solely English documents are in this corpus a stopword-based language test was applied.[4] Again, documents of a high similarity are likely to occur within this collection since Linux FAQs etc. are frequently updated and, in particular, mirrored on several sites.

[4]The test is similar to the test that has been used to compile the TREC Web Collections (Text Retrieval Conference 2003).

When using fingerprints in a retrieval process instead of a "rich" document model, completeness cannot be guaranteed: There may be documents that are very similar to each other under, for instance, the vector space model—though their fingerprints are different. Hence, the key question here relates to the quality of recall, i. e., the fraction of similar documents that can be identified as such by means of their fuzzy-fingerprint.

In the experiments we employed the cosine measure along with the vector space model to assess the reference similarity, and we computed the recall values with respect to different cosine similarity thresholds. The plots on the left-hand side in Figure 8 show the resulting *recall at similarity* curves, which look very promising: For the sets of documents that are similar to each other ($> 80\%$, see the shaded area) high recall-values were achieved for queries based on the fuzzy-fingerprint retrieval.

The question of *precision* reads as follows: How many documents that yield the same fuzzy-fingerprint under h_φ are actually similar to each other? Note that the documents whose fingerprints are involved in a collision form candidates for a high similarity, and a subsequent in-depth similarity analysis based on the vector space model must be applied for them. Since with a standard retrieval approach the entire collection is investigated, the ratio between the collection size and the size of the collision set can directly be interpreted as the factor for retrieval speed-up. The plots on the right-hand side in Figure 8 illustrate the sizes of both sets: The diagonal line corresponds to the retrieval time of a sequential scan; the line below, close to the x-axis, shows the average size (and hence the retrieval time) of the collision set for fuzzy fingerprints. Obviously, the use of h_φ leads to a substantial retrieval speed-up.

References

Aston, G. & Burnard, L. (1998). The BNC Handbook, `http://www.natcorp.ox.ac.uk`.

Aznar, G. (2004). The Linux Documentation Project, `http://www.tldp.org`.

Baker, B. S. (1993). On finding duplication in strings and software, `http://cm.bell-labs.com/cm/cs/papers.html`.

Brin, S., Davis, J. & Garcia-Molina, H. (1995). Copy detection mechanisms for digital documents, *SIGMOD '95*, ACM Press, New York, NY, USA, pp. 398–409.

Dennis, S. (1995). The sydney morning herald word database, `http://www2.psy.uq.edu.au/CogPsych/Noetica/OpenForumIssue4/SMH.html`.

Encyclopædia Britannica (2005). New Frontiers in Cheating, `http://www.britannica.com/eb/article?tocId=228894`.

Finkel, R., Zaslavsky, A., Monostori, K. & Schmidt, H. (2002). Signature Extraction for Overlap Detection in Documents, *Proceedings of the 25th Australian conference on Computer science*, Australian Computer Society, Inc., pp. 59–64.

Fullam, K. & Park, J. (2002). Improvements for scalable and accurate plagiarism detection in digital documents, `http://www.lips.utexas.edu/~kfullam/pdf/DataMiningReport.pdf`.

Gusfield, D. (1997). *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*, Cambridge University Press.

Hoad, T. & Zobel, J. (2003). Methods for Identifying Versioned and Plagiarised Documents, *American Society for Information Science and Technology* **54**(3): 203–215.

McCabe, D. (2005). Research Report of the Center for Academic Integrity, `http://www.academicintegrity.org`.

Monostori, K., Finkel, R., Zaslavsky, A., HodÃąsz, G. & Pataki, M. (2002). Comparison of overlap detection techniques, *Lecture Notes in Computer Science*, Vol. 2329, pp. 51–60.

Monostori, K., Zaslavsky, A. & Schmidt, H. (2000). Document overlap detection system for distributed digital libraries, *DL '00: Proceedings of the fifth ACM conference on Digital libraries*, ACM Press, New York, NY, USA, pp. 226–227.

Postel, J. (2004). RFC (Request For Comments) Collection, `http://www.rfc-editor.org`.

Ramakrishna, M. & Zobel, J. (1997). Performance in Practice of String Hashing Functions, *Proceedings of the International Conference on Database Systems for Advanced Applications, Melbourne, Australia*, pp. 215–223.

Rivest, R. L. (1992). The md5 message-digest algorithm, `http://theory.lcs.mit.edu/řivest/rfc1321.txt`.

Shivakumar, N. & Garcia-Molina, H. (1996). Building a scalable and accurate copy detection mechanism, *DL '96: Proceedings of the first ACM international conference on Digital libraries*, ACM Press, New York, NY, USA, pp. 160–168.

Si, A., Leong, H. V. & Lau, R. W. H. (1997). Check: a document plagiarism detection system, *SAC '97: Proceedings of the 1997 ACM symposium on Applied computing*, ACM Press, New York, NY, USA, pp. 70–77.

Stein, B. (2005). Fuzzy-Fingerprints for Text-Based Information Retrieval, *in* K. Tochtermann & H. Maurer (eds), *Proceedings of the 5th International Conference on Knowledge Management (I-KNOW 05), Graz*, Journal of Universal Computer Science, Know-Center, pp. 572–579.

Stein, B. & Meyer zu Eißen, S. (2006). Near Similarity Search and Plagiarism Analysis, *in* M. Spiliopoulou, R. Kruse, C. Borgelt, A. Nürnberger & W. Gaul (eds), *From Data and Information Analysis to Knowledge Engineering*, Springer, pp. 430–437.

Text Retrieval Conference (2003). The TREC Web Document Collection, `http://trec.nist.gov`.

University of Leipzig (1995). Wortschatz, `http://wortschatz.uni-leipzig.de`.