Preisach, Burkhardt, Schmidt-Thieme, Decker (Eds.):
Data Analysis, Machine Learning and Applications
Selected Papers from the 31th Conf. of the German Classification Society
Berlin, ISBN 978-3-540-78239-1, pp. 601-609, ©Springer 2008.

New Issues in Near-duplicate Detection

Martin Potthast and Benno Stein

Bauhaus University Weimar, 99421 Weimar
{martin.potthast | benno.stein}@medien.uni-weimar.de

Abstract Near-duplicate detection is the task of identifying documents with almost identical content. The respective algorithms are based on fingerprinting; they have attracted considerable attention due to their practical significance for Web retrieval systems, plagiarism analysis, corporate storage maintenance, or social collaboration and interaction in the World Wide Web.

Our paper presents both an integrative view as well as new aspects from the field of near-duplicate detection: (*i*) *Principles and Taxonomy.* Identification and discussion of the principles behind the known algorithms for near-duplicate detection. (*ii*) *Corpus Linguistics.* Presentation of a corpus that is specifically suited for the analysis and evaluation of near-duplicate detection algorithms. The corpus is public and may serve as a starting point for a standardized collection in this field. (*iii*) *Analysis and Evaluation.* Comparison of state-of-the-art algorithms for near-duplicate detection with respect to their retrieval properties. This analysis goes beyond existing surveys and includes recent developments from the field of hash-based search.

1 Introduction

In this paper two documents are considered as near-duplicates if they share a very large part of their vocabulary. Near-Duplicates occur in many document collections, from which the most prominent one is the World Wide Web. Recent studies of Fetterly *et al.* (2003) and Broder *et al.* (2006) show that about 30% of all Web documents are duplicates of others. Zobel and Bernstein (2006) give examples which include mirror sites, revisions and versioned documents, or standard text building blocks such as disclaimers. The negative impact of near-duplicates on Web search engines is threefold: indexes waste storage space, search result listings can be cluttered with almost identical entries, and crawlers have a high probability of exploring pages whose content is already acquired.

Content duplication also happens through text plagiarism, which is the attempt to present other people's text as own work. Note that in the plagiarism situation document content is duplicated at the level of short passages; plagiarized passages can also be modified to a smaller or larger extent in order to obscure the offense.

Aside from deliberate content duplication, copying happens also accidentally: in companies, universities, or public administrations documents are stored multiple times, simply because employees are not aware of already existing previous work (Forman *et al.* (2005)). A similar situation is given for social software such as customer review boards or comment boards, where many users publish their opinion about some topic of interest: users with the same opinion write essentially the same in diverse ways since they read not all existing contributions.

A solution to the outlined problems requires a reliable recognition of near-duplicates—preferably at a high runtime performance. These objectives compete with each other, a compromise in recognition quality entails deficiencies with respect to retrieval precision and retrieval recall. A reliable approach to identify two documents d and d_q as near-duplicates is to represent them under the vector space model, referred to as $\mathbf{d}$ and $\mathbf{d}_q$, and to measure their similarity under the l_2-norm or the enclosed angle. d and d_q are considered as near-duplicates if the following condition holds:

$$\varphi(\mathbf{d}, \mathbf{d}_q) \geq 1 - \varepsilon \quad \text{with } 0 < \varepsilon \ll 1,$$

where φ denotes a similarity function that maps onto the interval $[0, 1]$. To achieve a recall of 1 with this approach, each pair of documents must be analyzed. Likewise, given d_q and a document collection D, the computation of the set D_q, $D_q \subset D$, with all near-duplicates of d_q in D, requires $O(|D|)$, say, linear time in the collection size. The reason lies in the high dimensionality of the document representation $\mathbf{d}$, where "high" means "more than 10": objects represented as high-dimensional vectors cannot be searched efficiently by means of space partitioning methods such as kd-trees, quad-trees, or R-trees but are outperformed by a sequential scan (Weber *et al.* (1998)). By relaxing the retrieval requirements in terms of precision and recall the runtime performance can be significantly improved. Basic idea is to estimate the similarity between d and d_q by means of fingerprinting. A fingerprint, F_d, is a set of k numbers computed from d. If two fingerprints, F_d and F_{d_q}, share at least κ numbers, $\kappa \leq k$, it is assumed that d and d_q are near-duplicates. I. e., their similarity is estimated using the Jaccard coefficient:

$$\frac{|F_d \cap F_{d_q}|}{|F_d \cup F_{d_q}|} \geq \frac{\kappa}{k} \quad \Rightarrow \quad P\big(\varphi(\mathbf{d}, \mathbf{d}_q) \geq 1 - \varepsilon\big) \text{ is close to 1}$$

Let $F_D = \bigcup_{d \in D} F_d$ denote the union of the fingerprints of all documents in D, let $\mathcal{D}$ be the power set of D, and let $\mu : F_D \to \mathcal{D}, \ \ x \mapsto \mu(x)$, be an inverted file index that maps a number $x \in F_D$ on the set of documents whose fingerprints contain x; $\mu(x)$ is also called the postlist of x. For document d_q with fingerprint F_{d_q} consider now the set $\hat{D}_q \subset D$ of documents that occur in at least κ of the postlists $\mu(x)$, $x \in F_{d_q}$. Put another way, $\hat{D}_q$ consists of documents whose fingerprints share a least κ numbers with F_{d_q}. We use $\hat{D}_q$ as a heuristic approximation of D_q, whereas the retrieval performance, which depends on the finesse of the fingerprint construction, computes as follows:

$$prec = \frac{\hat{D}_q \cap D_q}{\hat{D}_q}, \qquad rec = \frac{\hat{D}_q \cap D_q}{D_q}$$

The remainder of the paper is organized as follows. Section 2 gives an overview of fingerprint construction methods and classifies them in a taxonomy, including so far unconsidered hashing technologies. In particular, different aspects of fingerprint construction are contrasted and a comprehensive view on their retrieval properties is presented. Section 3 deals with evaluation methodologies for near-duplicate detection and proposes a new benchmark corpus of realistic size. The state-of-the-art fingerprint construction methods are subject to an experimental analysis using this corpus, providing new insights into precision and recall performance.

2 Fingerprint Construction

A chunk or an n-gram of a document d is a sequence of n consecutive words found in d.[1] Let C_d be the set of all different chunks of d. Note that C_d is at most of size $|d| - n$ and can be assessed with $O(|d|)$. Let $\mathbf{d}$ be a vector space representation of d where each $c \in C_d$ is used as descriptor of a dimension with a non-zero weight.

According to Stein (2007) the construction of a fingerprint from $\mathbf{d}$ can be understood as a three-step-procedure, consisting of dimensionality reduction, quantization, and encoding:

1. Dimensionality reduction is realized by projecting or by embedding. Algorithms of the former type select dimensions in $\mathbf{d}$ whose values occur unmodified in the reduced vector $\mathbf{d}'$. Algorithms of the latter type reformulate $\mathbf{d}$ as a whole, maintaining as much information as possible.
2. Quantization is the mapping of the elements in $\mathbf{d}'$ onto small integer numbers, obtaining $\mathbf{d}''$.
3. Encoding is the computing of one or several codes from $\mathbf{d}''$, which together form the fingerprint of d.

Fingerprint algorithms differ primarily in the employed dimensionality reduction method. Figure 1 organizes the methods along with the known construction algorithms; the next two subsections provide a short characterization of both.

[1] If the hashed breakpoint chunking strategy of Brin *et al.* (1995) is applied, n can be understood as expected value of the chunk length.

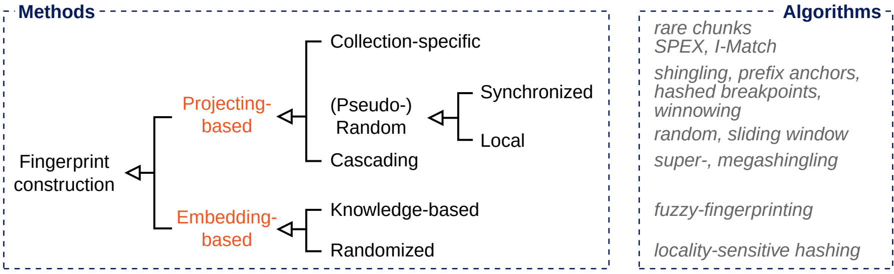

Figure 1. Taxonomy of fingerprint construction methods (left) and algorithms (right).

2.1 Dimensionality Reduction by Projecting

If dimensionality reduction is done by projecting, a fingerprint F_d for document d can be formally defined as follows:

$$F_d = \{h(c) \mid c \in C_d \ \text{ and } \ \sigma(c) = \mathit{true}\},$$

where σ denotes a selection heuristic for dimensionality reduction that becomes true if a chunk fulfills a certain property. h denotes a hash function, such as MD5 or Rabin's hash function, which maps chunks to natural numbers and serves as a means for quantization. Usually the identity mapping is applied as encoding rule. Broder (2000) describes a more intricated encoding rule called supershingling.

The objective of σ is to select chunks to be part of a fingerprint which are best-suited for a reliable near-duplicate identification. Table 1 presents in a consistent way algorithms and the implemented selection heuristics found in the literature, whereas a heuristic is of one of the types denoted in Figure 1.

2.2 Dimensionality Reduction by Embedding

An embedding-based fingerprint F_d for a document d is typically constructed with a technique called "similarity hashing" (Indyk and Motwani (1998)). Unlike standard hash functions, which aim to a minimization of the number of hash collisions, a similarity hash function $h_\varphi : \mathbf{D} \rightarrow U, U \subset \mathbf{N}$, shall produce a collision with a high probability for two objects $\mathbf{d}, \mathbf{d}_q \in \mathbf{D}$, iff $\varphi(\mathbf{d}, \mathbf{d}_q) \geq 1-\varepsilon$. In this way h_φ downgrades a fine-grained similarity relation quantified within φ to the concept "similar or not similar", reflected by the fact whether or not the hashcodes $h_\varphi(\mathbf{d})$ and $h_\varphi(\mathbf{d}_q)$ are identical. To construct a fingerprint F_d for document d a small number of k variants of h_φ are used:

$$F_d = \{h_\varphi^{(i)}(\mathbf{d}) \mid i \in \{1, \ldots, k\}\}$$

Two kinds of similarity hash functions have been proposed, which either compute hashcodes based on knowledge about the domain or which ground on

Algorithm	**(Author)**	**Selection heuristic $\sigma(c)$**
rare chunks	(Heintze (1996))	c occurs once in D
SPEX	(Bernstein and Zobel (2004))	c occurs at least twice in D
I-Match	(Chowdhury *et al.* (2002), Conrad *et al.* (2003), Kołcz *et al.* (2004))	$c = d$; excluding non-discriminant terms of d
shingling	(Broder (2000))	$c \in \{c_1, \ldots, c_k\}, \quad \{c_1, \ldots, c_k\} \subset_{rand} C_d$
prefix anchor	(Manber (1994))	c starts with a particular prefix, or
	(Heintze (1996))	c starts with a prefix which is infrequent in d
hashed breakpoints	(Manber (1994))	$h(c)$'s last byte is 0, or
	(Brin *et al.* (1995))	c's last word's hash value is 0
winnowing	(Schleimer *et al.* (2003))	c minimizes $h(c)$ in a window sliding over d
random	(misc.)	c is part of a local random choice from C_d
one of a sliding window	(misc.)	c starts at word i mod m in d; $1 \leq m \leq \|d\|$
super- / megashingling	(Broder (2000) / Fetterly *et al.* (2003))	c is a combination of hashed chunks which have been selected with shingling

Table 1. Summary of chunk selection heuristics. The rows contain the name of the construction algorithm along with typical constraints that must be fulfilled by the selection heuristic σ.

domain-independent randomization techniques (see again Figure 1). Both similarity hash functions compute hashcodes along the three steps outlined above: An example for the former is fuzzy-fingerprinting developed by Stein (2005), where the embedding step relies on a tailored, low-dimensional document model and where fuzzification is applied as a means for quantization. An example for the latter is locality-sensitive hashing and the variants thereof by Charikar (2002) and Datar *et al.* (2004). Here the embedding relies on the computation of scalar products of $\mathbf{d}$ with random vectors, and the scalar products are mapped on predefined intervals on the real number line as a means for quantization. In both approaches the encoding happens according to a summation rule.

2.3 Discussion

We have analyzed the aforementioned fingerprint construction methods with respect to construction time, retrieval time, and the resulting size of a complete chunk index. Table 2 compiles the results.

The construction of a fingerprint for a document d depends on its length since d has to be parsed at least once, which explains that all methods have the same complexity in this respect. The retrieval of near-duplicates requires a chunk index μ as described at the outset: μ is queried with each number of a query document's fingerprint F_{d_q}, for which the obtained postlists are merged. We assume that both the lookup time and the average length of a postlist can

Algorithm		Runtime Construction	Runtime Retrieval	Chunk length	Finger-print size	Chunk index size
rare chunks		$O(\|d\|)$	$O(\|d\|)$	n	$O(\|d\|)$	$O(\|d\| \cdot \|D\|)$
SPEX	$(0 < r \ll 1)$	$O(\|d\|)$	$O(r \cdot \|d\|)$	n	$O(r \cdot \|d\|)$	$O(r \cdot \|d\| \cdot \|D\|)$
I-Match		$O(\|d\|)$	$O(k)$	$\|d\|$	$O(k)$	$O(k \cdot \|D\|)$
shingling		$O(\|d\|)$	$O(k)$	n	$O(k)$	$O(k \cdot \|D\|)$
prefix anchor		$O(\|d\|)$	$O(\|d\|)$	n	$O(\|d\|)$	$O(\|d\| \cdot \|D\|)$
hashed breakpoints		$O(\|d\|)$	$O(\|d\|)$	$E(\|c\|) = n$	$O(\|d\|)$	$O(\|d\| \cdot \|D\|)$
winnowing		$O(\|d\|)$	$O(\|d\|)$	n	$O(\|d\|)$	$O(\|d\| \cdot \|D\|)$
random		$O(\|d\|)$	$O(k)$	n	$O(k)$	$O(\|d\| \cdot \|D\|)$
one of sliding window		$O(\|d\|)$	$O(\|d\|)$	n	$O(\|d\|)$	$O(\|d\| \cdot \|D\|)$
super- / megashingling		$O(\|d\|)$	$O(k)$	n	$O(k)$	$O(k \cdot \|D\|)$
fuzzy-fingerprinting		$O(\|d\|)$	$O(k)$	$\|d\|$	$O(k)$	$O(k \cdot \|D\|)$
locality-sensitive hashing		$O(\|d\|)$	$O(k)$	$\|d\|$	$O(k)$	$O(k \cdot \|D\|)$

Table 2. Summary of complexities for the construction of a fingerprint, the retrieval, and the size of a tailored chunk index.

be assessed with a constant for either method.[2] Thus the retrieval runtime depends only on the size k of a fingerprint. Observe that the construction methods fall into two groups: methods whose fingerprint's size increases with the length of a document, and methods where k is independent of $|d|$. Similarly, the size of μ is affected. We further differentiate methods with fixed length fingerprints into these which construct small fingerprints where $k \leq 10$ and those where $10 \ll k < 500$. Small fingerprints are constructed by fuzzy-fingerprinting, locality-sensitive hashing, supershingling, and I-Match; these methods outperform the others by orders of magnitude in their chunk index size.

3 Wikipedia as Evaluation Corpus

When evaluating near-duplicate detection methods one faces the problem of choosing a corpus which is representative for the retrieval situation and which provides a realistic basis to measure both retrieval precision and retrieval recall. Today's standard corpora such as the TREC or Reuters collection have deficiencies in this connection: In standard corpora the distribution of similarities decreases exponentially from a very high percentage at low similarity intervals to a very low percentage at high similarity intervals. Figure 2 (right) illustrates this characteristic at the Reuters corpus. This characteristic allows only precision evaluations since the recall performance depends on

[2] We indexed all English Wikipedia articles and found that an increase from 3 to 4 in the chunk length implies a decrease from 2.42 to 1.42 in the average postlist length.

very few pairs of documents. The corpora employed in recent evaluations of Hoad and Zobel (2003), Henzinger (2006), and Ye *et al.* (2006) lack in this respect; moreover, they are custom-built and not publicly available. Conrad and Schriber (2004) attempt to overcome this issue by the artificial construction of a suitable corpus.

We propose to use the Wikipedia Revision Corpus for near-duplicate detection including all revisions of every Wikipedia article.[3] The table in Figure 2 shows selected order of magnitudes of the corpus. A preliminary analysis shows that an article's revisions are often very similar to each other with an expected similarity of about 0.5 to the first revision. Since the articles of Wikipedia undergo a regular rephrasing, the corpus addresses the particularities of the use cases mentioned at the outset. We analyzed the fingerprinting algorithms with 7 Million pairs of documents, using the following strategy: each article's first revision serves as query document d_q and is compared to all other revisions as well as to the first revision of its immediate successor article. The former ensures a large number of near-duplicates and hence improves the reliability of the recall values; rationale of the latter is to gather sufficient data to evaluate the precision (cf. Figure 2, right-hand side).

Figure 3 presents the results of our experiments in the form of precision-over-similarity curves (left) and recall-over-similarity curves (right). The curves are computed as follows: For a number of similarity thresholds from the interval $[0; 1]$ the set of document pairs whose similarity is above a certain threshold is determined. Each such set is compared to the set of near-duplicates identified by a particular fingerprinting method. From the intersection of these sets then the threshold-specific precision and recall values are computed in the standard way.

As can be seen in the plots, the chunking-based methods perform better than similarity hashing, while hashed breakpoint chunking performs best. Of those with fixed size fingerprints shingling performs best, and of those with

[3] http://en.wikipedia.org/wiki/Wikipedia:Download, last visit on July 29, 2011

Wikipedia corpus:

Property	**Value**
documents	6 Million
revisions	80 Million
size (uncompressed)	1 terabyte

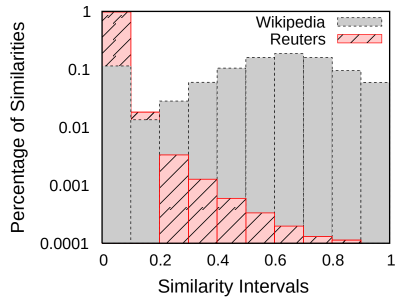

Figure 2. The table (left) shows order of magnitudes of the Wikipedia corpus. The plot contrasts the similarity distribution within the Reuters Corpus Volume 1 and the Wikipedia corpus.

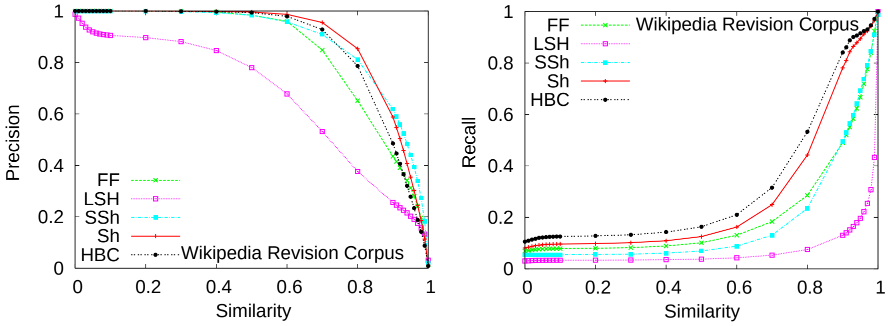

Figure 3. Precision and recall over similarity for fuzzy-fingerprinting (FF), locality-sensitive hashing (LSH), supershingling (SSh), shingling (Sh), and hashed break-point chunking (HBC).

small fixed size fingerprints fuzzy-fingerprinting and supershingling perform similar. Note that the latter had both 50 times smaller fingerprints than shingling which shows the possible impact of theses methods on the size of a chunk index.

4 Summary

Algorithms for near-duplicate detection are applied in retrieval situations such as Web mining, plagiarism detection, corporate storage maintenance, and social software. In this paper we developed an integrative view to existing and new technologies for near-duplicate detection. Theoretical considerations and practical evaluations show that shingling, supershingling, and fuzzy-fingerprinting perform best in terms of retrieval recall, retrieval precision, and chunk index size. Moreover, a new, publicly available corpus is proposed, which overcomes weaknesses of the standard corpora when analyzing use cases from the field of near duplicate detection.

References

BERNSTEIN, Y. and ZOBEL, J. (2004): A scalable system for identifying co-derivative documents, *Proc. of SPIRE '04*.

BRIN, S., DAVIS, J. and GARCIA-MOLINA, H. (1995): Copy detection mechanisms for digital documents, *Proc. of SIGMOD '95*.

BRODER, A. (2000): Identifying and filtering near-duplicate documents, *Proc. of COM '00*.

BRODER, A., EIRON, N., FONTOURA, M., HERSCOVICI, M., LEMPEL, R., MCPHERSON, J., QI, R. and SHEKITA, E. (2006): Indexing Shared Content in Information Retrieval Systems, *Proc. of EDBT '06*.

CHARIKAR, M. (2002): Similarity Estimation Techniques from Rounding Algorithms, *Proc. of STOC '02*.

CHOWDHURY, A., FRIEDER, O., GROSSMAN, D. and MCCABE, M. (2002): Collection statistics for fast duplicate document detection, *ACM Trans. Inf. Syst.,20.*

CONRAD, J., GUO, X. and SCHRIBER, C. (2003): Online duplicate document detection: signature reliability in a dynamic retrieval environment, *Proc. of CIKM '03.*

CONRAD, J. and SCHRIBER, C. (2004): Constructing a text corpus for inexact duplicate detection, *Proc. of SIGIR '04.*

DATAR, M., IMMORLICA, N., INDYK, P. and MIRROKNI, V. (2004): Locality-Sensitive Hashing Scheme Based on p-Stable Distributions, *Proc. of SCG '04.*

FETTERLY, D., MANASSE, M. and NAJORK, M. (2003): On the Evolution of Clusters of Near-Duplicate Web Pages, *Proc. of LA-WEB '03.*

FORMAN, G., ESHGHI, K. and CHIOCCHETTI, S. (2005): Finding similar files in large document repositories, *Proc. of KDD '05.*

HEINTZE, N. (1996): Scalable document fingerprinting, *Proc. of USENIX-EC '96.*

HENZINGER, M. (2006): Finding Near-Duplicate Web Pages: a Large-Scale Evaluation of Algorithms, *Proc. of SIGIR '06.*

HOAD, T. and ZOBEL, J. (2003): Methods for Identifying Versioned and Plagiarised Documents, *Jour. of ASIST, 54.*

INDYK, P. and MOTWANI, R. (1998): Approximate Nearest Neighbor—Towards Removing the Curse of Dimensionality, *Proc. of STOC '98.*

KOŁCZ, A., CHOWDHURY, A. and ALSPECTOR, J. (2004): Improved robustness of signature-based near-replica detection via lexicon randomization, *Proc. of KDD '04.*

MANBER, U. (1994): Finding similar files in a large file system, *Proc. of USENIX-TC '94.*

SCHLEIMER, S., WILKERSON, D. and AIKEN, A. (2003): Winnowing: local algorithms for document fingerprinting, *Proc. of SIGMOD '03.*

STEIN, B. (2005): Fuzzy-Fingerprints for Text-based Information Retrieval, *Proc. of I-KNOW '05.*

STEIN, B. (2007): Principles of Hash-based Text Retrieval, *Proc. of SIGIR '07.*

WEBER, R., SCHEK, H. and BLOTT, S. (1998): A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces, *Proc. of VLDB '98.*

YE, S., WEN, J. and MA, W. (2006): A Systematic Study of Parameter Correlations in Large Scale Duplicate Document Detection, *Proc. of PAKDD '06.*

ZOBEL, J. and BERNSTEIN, Y. (2006): The case of the duplicate documents: Measurement, search, and science, *Proc. of APWeb '06.*