



A pruning strategy to improve pairwise comparison-based near-duplicate detection

Roya Hassanian-esfahani¹ · Mohammad-javad Kargar²

Received: 15 September 2017 / Revised: 6 October 2018 / Accepted: 28 November 2018 /
Published online: 3 January 2019
© Springer-Verlag London Ltd., part of Springer Nature 2019

Abstract

Efficient and accurate near-duplicate detection is a trending topic of research. Complications arise from the great time and space complexities of existing algorithms. This study proposes a novel pruning strategy to improve pairwise comparison-based near-duplicate detection methods. After parsing the documents into punctuation-delimited blocks called chunks, it decides between the categories of “near duplicate,” “non-duplicate” or “suspicious” by applying certain filtering rules. This early decision makes it possible to disregard many of the non-necessary computations—on average 92.95% of them. Then, for the suspicious pairs, common chunks and short chunks are removed and the remaining subsets are reserved for near-duplicate detection. Size of the remaining subsets is on average 4.42% of the original corpus size. Evaluation results show that near-duplicate detection with the proposed strategy in its best configuration (CHT = 8, $\tau = 0.1$) has F -measure = 87.22% (precision = 86.91% and recall = 87.54%). Its F -measure is comparable with the SpotSig method with less execution time. In addition, applying the proposed strategy in a near-duplicate detection process eliminates the need for preprocessing. It is also tunable to achieve the intended levels of near duplication and noise suppression.

Keywords Near-duplicate detection · Pruning strategy · Similarity

1 Introduction

Developing efficient and accurate near-duplicate detection (NDD) algorithms for different forms of content including text, image, voice and video is a trending research topic. NDD in documents or near-duplicate documents detection (NDDD) is applied in a wide range of contexts including news articles, scientific articles, literature and written works, databases with text records, patents, and emails. It finds a significant role in several IR applications, especially in search engines where the most important objective is to index near-duplicate

✉ Roya Hassanian-esfahani
hasanian@ictrc.ac.ir

¹ Research Institute for Information and Communications Technologies, Academic Center for Education, Culture and Research, Tehran, Iran

² Department of Computer Engineering, University of Science and Culture, Tehran, Iran

documents once, and return only the most proper version in the search results in an accurate and efficient way.

Near duplication covers a diverse range of relationships, ranging from documents with identical content, but different HTML codes to almost the same documents that contain certain content-level differences. Its definition happens to be vague and application dependent. Implicitly, two documents are considered near duplicate, which is also referred to as near replica [26] or roughly the same [4] if a substantial percentage of their contents are shared. Near duplication is a document-to-document (pairwise) relationship. It is not necessarily transitive, i.e., if $a \approx b$ and $b \approx c$, the relation $a \approx c$ is not guaranteed [2]. The near duplication of web documents is a static property, which means that two near-duplicate web documents will probably remain near duplicate over the time, with a high degree of confidence [3]. This phenomenon may occur both intentionally and unintentionally and is widespread over the web. Some origins of near duplication on the web include “www” versus “non-www” versions of the same page, paginations, pages that differ only by a parameter (especially sort order), web pages that have URLs with different parameter orders, such as “?a = 1&b = 2” and “?b = 1&a = 2” or URLs that convey users’ session ID, mirrored websites that only differ in the header or the footnote areas (denoted by the website’s URL and update time), document copies kept on different directories or different servers, different formats of the same document such as HTML and PDF, documents published under different portals or websites, documents rendered in different templates with different advertisements, versioned documents, and finally noised or altered versions of documents, for instance, OCR versions [16, 33, 37].

This research is interested in examining the use of NDD in retrieval systems for web news articles. It specifically focuses on designing a ranking module for a search engine capable of efficiently and accurately detecting reposted versions of news articles. In the web news industry, a news article is reposted by hundreds—and sometimes thousands—of reposting websites in the form of near-duplicate versions. News search engines are strongly dependent on NDD methods to avoid indexing near-duplicate web pages; NDD is also directly connected to their business performance [39]. In an experimental study in 2009, it was found that around 80% of web news articles are reposted versions and only 20% of them are original [41].

Web news article includes distinct zones such as title and body, and fields such as publishing time and date. Reposted versions of web news articles have the following characteristics: fields are usually different on each reposting website and zones are nearly duplicated [12, 27]. In the zones, title is more likely to be changed than the body. Unlike many other disciplines, publishing near-duplicate news articles is an accepted practice in the profession of journalism. It is not considered as negative moral behavior since spreading news in the shortest possible time is one of the main objectives of the players in the journalism ecosystem. Therefore, news publishers do not try to hide the practice of near duplication by intelligent modification of the content. In addition, precise referencing style in its scientific format is not common in the context of news repost composition and occurs rarely. Finally, reposted (near-duplicated) zones get published in different websites’ containers, including banners, logos, advertisements, navigation bars, menus, etc. An example of two near-duplicate web news articles is provided in Fig. 1.

In most of the previous works, the similarity scores are evaluated to decide whether a document is a duplicate of another document. However, the acceptable level of similarity is defined slightly differently in each of them. Near duplication is related and at the same time different from reused, contained, plagiarized, derived, co-derived or copied

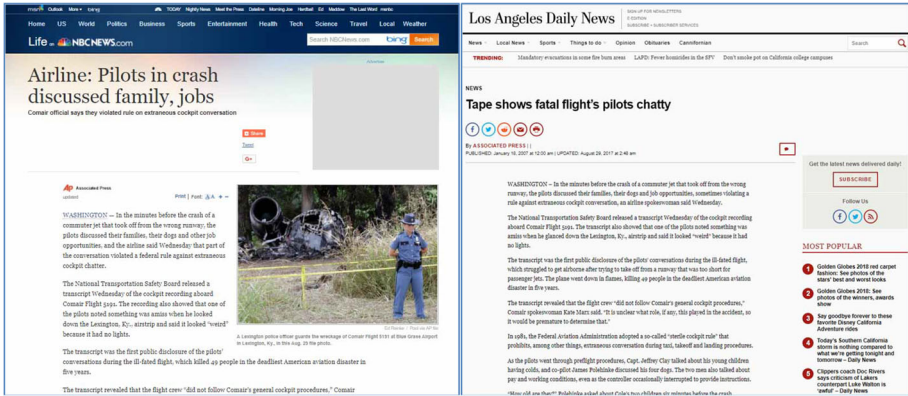


Fig. 1 Two near-duplicate web news articles published on two different websites (available on [http://www.nbcnews.com/id/16671549/ns/us_news-life/t/airline-pilots-crash-discussed-family-jobs/#.WIRsM6iWY2w] and [<https://www.dailynews.com/2007/01/18/tape-shows-fatal-flights-pilots-chatty/>])

documents in the way that they are generated and the type of relationships that exist between them. Consequently, the detection methods are different in the aforementioned situations.

NDD methods are divided into three main approaches: (1) pairwise comparison-based approach including methods such as Bag-of-Words (BOW) Inverted Indexing [34], Filtering [42], Shingling [8], I-Match [11] and SpotSig [37]; (2) signature-based approach including methods dependent on fingerprinting the documents such as MinHashing [6] and SimHashing [9]; and (3) learning methods are based mostly on weighted zone scoring, where zone refers to document blocks such as title, abstract, body, etc. NDD methods can be learned by setting up a classification model by use of linear combination of zone-based feature vectors [18], learning each zone of the document distinctly [5] or hierarchical decision on the basis of zone similarity extents [38]. However, the third approach is out of the scope of this study as it is usually applied in special situations and requires higher time and memory complexities than the two other approaches. Generally, pairwise comparison-based approach provides more acceptable levels of accuracy, but is not efficient and requires a large amount of memory and time. In contrast, signature-based approach sacrifices accuracy in favor of efficiency. In the third category, a small number of studies are conducted.

The focus of this study is on pairwise comparison-based NDD. Pairwise comparison-based NDD methods can be differentiated by the way they represent the documents, the pruning strategy that they apply and the way they calculate the similarity extent. The main consideration of these methods is efficiency since they result in large Shingle sets. For instance, k -Shingling in a document with l tokens lead to a Shingle set with the cardinality of $l - k + 1$. Therefore, the pruning strategies are employed to reduce the number of comparisons the algorithm must perform to be able to deal with large-scale and high-dimensional document collections. Existing strategies are usually based on a fixed-sized subset selection or a proportional subset selection. These strategies increase efficiency, but decrease the accuracy because no semantic insight is used. The main contribution of this work is to develop a novel pruning strategy that is capable of simultaneously reducing the number of comparisons considered and the size of the texts to be compared. The proposed strategy is based on intuitions on the intrinsic properties of near duplication in news articles. It does not select blindly, but prunes the text based on identifying the sources of near duplication. It also defines the new

space parameters based on the existing ones by the use of theoretical analysis. The proposed pruning strategy has linear computational complexity. In addition, it has less space complexity and time complexity with an acceptable level of quality in comparison with the existing methods. To the best of our knowledge, such a pruning strategy has not been previously proposed.

This paper is organized as follows: In Sect. 2, previous works on relevant topics is reviewed in brief. Section 3 introduces the proposed strategy in three subsections, including concepts and notions, main steps, and theoretical analysis. Section 4 gives an overview of the methodology of the research and the evaluation. The results and discussion are provided in Sect. 5. The conclusion is provided in Sect. 6.

2 Previous works

2.1 Comparison-based NDD methods

This section provides a brief review of previous works on pairwise comparison-based NDD. Near duplication of two documents, d_i and $d_j \in D$ where $D = \{d_1 \dots d_n\}$ in a pairwise comparison-based approach, is detected by applying a similarity function $\varphi(.) : D \times D \rightarrow [0, 1]$ on the two documents and deciding whether they match (exactly or approximately) or not based on a predefined similarity threshold. In this definition, d_i and d_j are near duplicate if:

$$\varphi(d_i, d_j) \geq 1 - \varepsilon, \quad 0 < \varepsilon \ll 1. \tag{1}$$

where $\tau = 1 - \varepsilon$ is the similarity threshold that is set manually or learned by certain machine learning (ML) algorithms. The $\varphi(.)$ can be any similarity function. Although the solution is straightforward, but with the theoretical runtime of $O(N^2)$, a direct application of similarity functions is not feasible for large-scale corpuses. Some special methods are, therefore, proposed in three main steps of: (1) changing the document representation, (2) pruning the problem space and (3) pairwise comparison. A summarization of previous works on pairwise comparison-based NDD methods is provided in Table 1.

The digital syntactic clustering (DSC) algorithm, also called **Shingling** [6], is one of the earliest techniques proposed for NDD, which converts the problem of similarity calculation into the problem of set operations. In Shingling, unique contiguous overlapping sequences of k tokens as k -grams are generated, where k is a positive integer. K -grams are generated assuming that the sequence of tokens reserve positional information and are more representative than single tokens. K -grams are hashed and called Shingles since storing and comparing hash values would be more efficient. In this process, each document is converted into amore similar the articles. In this definition Shingle set. The Kronecker delta operator $\delta(.)$ defines the pairwise similarity score between the Shingles of two Shingle sets, where

$$\delta(\text{Shingle}_m, \text{Shingle}_n) = \begin{cases} 0 & \text{if } \text{Shingle}_m = \text{Shingle}_n \\ 1 & \text{if } \text{Shingle}_m \neq \text{Shingle}_n \end{cases}. \tag{2}$$

The similarity of two Shingle sets, $S(d_i)$ and $S(d_j)$, is calculated by computing the Jaccard coefficient [22]—an old but popular set similarity measure that is the ratio of the cardinality of intersection to the cardinality of union of the two sets; this is expressed as follows:

$$\varphi(d_i, d_j) \equiv J(S(d_i), S(d_j)) = \frac{|S(d_i) \cap S(d_j)|}{|S(d_i) \cup S(d_j)|} \tag{3}$$

Table 1 Summary of previous works on pairwise comparison-based NDD methods

Method	Problem space	NDD condition	Pruning complexity*
Shingling [6]: $d \rightarrow \text{tokens} \xrightarrow{\text{selection}} \text{kgrams} \xrightarrow{\text{hasng}} S(d)$	Shingles	$J(S(d_i), S(d_j)) > \tau$	Random: $O(n)$ TF-based: $O(n \log(n))$
Super-Shingling [7]: (1) $d \rightarrow \text{tokens} \xrightarrow{\text{selection}} \text{kgrams} \xrightarrow{\text{hashing}} S(d)$ 2) $S(d) \xrightarrow{\text{randomized hash functions}} \text{few Super } S(d)$	Super-Shingles	$\text{Supers}(d_i) = \text{Supers}(d_j)$	$O(n)$
I-Match [11]: (1) $D \rightarrow \text{tokens} \xrightarrow{\text{pruning}} L$ (2) $d \rightarrow \text{tokens} \rightarrow U$ (3) $U \cap L \xrightarrow{\text{hashing}} \text{I-Match}(d)$	I-Match signatures	$\text{I-Match}(d_i) = \text{I-Match}(d_j)$	$O(n N \log(nN))$
Generating several I-matches for each document [25]	I-Match signatures	$\text{I-Match}(d_i)_k = \text{I-Match}(d_j)_k$ $K \in \mathbb{N}$	$O(n N \log(nN))$
SpotSig [37]: $d \rightarrow \text{SpotSet}(d) = \{a(d.c)\}$	Signature sets	$J(\text{SpotSet}(d_i), \text{SpotSet}(d_j)) > \tau$	$O(n)$
Selecting based on sentence-level statistics [40]: $d \rightarrow$ sentence level statistics $\xrightarrow{\text{windowing}} \text{kStrings}$	k -strings	Number of shared k -Strings $> \tau$	NA

*Pruning complexity, which is presented as a function of n , is different from the overall computational complexity of the algorithm, while the overall computational complexity of the pairwise comparison-based methods is a fraction $O(N^2)$

** τ , similarity threshold; N , number of documents; n , number of words per document

The Jaccard coefficient is zero when the two sets are disjoint; it is one when they are equal; and strictly between zero and one, otherwise. Therefore, a threshold value close to one is used to determine near duplication. k -Shingling is sensitive to k and changing k results in different Jaccard coefficients. It is worth mentioning that a Jaccard coefficient of one does not necessarily mean that the two documents are the same, but that the documents are permutations of each other having similar distributions of Shingles. In improved versions of Shingling, the occurrence frequency of the Shingles is also taken into account. Although k -Shingling is more accurate and efficient than comparing the documents word by word, it is still inefficient for large document collections because of the large and varying sizes of Shingle sets. It requires $O(kN)$ storage space for unhashed k -grams and $O(N)$ storage space in the case of hashed k -grams, where N is the number of documents. For the time complexity, the complexity of Shingling a single document is $O(1)$ and the pairwise comparison of the Jaccard coefficients on the corpus is a near-quadratic process of at least $O(N^2)$. Improved methods propose filtering and selection of a subset of Shingle sets. Selecting Shingles with the smallest hash values and removing high frequency Shingles [20], choosing at most 400 Shingles for each document by selecting Shingles which their values modulus of 25 are zero [6], and random Shingle selection [30] are among the proposed strategies.

As a more efficient sampling method with a small decrease in precision, the **Super-Shingling** algorithm [7] or DSC-SS is proposed, where each m -dimensional vector (Shingle set) is mapped on to a smaller set of Super-Shingles by the use of different hash functions. In this case, instead of calculating the ratio of the matching Shingles to their union, a single Super-Shingle should match. Some studies propose to improve the quality of the Super-Shingling algorithm by selecting only salient terms as Shingles [16] or filtering the Shingles based on their positions in the document or in the sentence [21]. However, Super-Shingling is not applicable for short texts.

Spot signature or **SpotSig** [37] is another sampling method proposed for Shingling, in which, several localized signatures are generated for each document. SpotSigs are the terms that follow stop words with a predefined distance that is represented by $a(d, c)$, where a stands for antecedents (stop words), d stands for a fixed distance, and c stands for a chain of words following the antecedents with predefined distance. Spot Set $A = \{a_j(d_j, c_j)\}$ is defined as a set of the best SpotSigs. This method is conducted in accompany with an Inverted Indexing and Filtering technique. It is capable of distinction between the main text and noise as noise text rarely conveys stop words. However, in their experiments HTML markups are removed first.

IIT-Match or **I-Match** [11] is a collection-specific fingerprint-based method. Although it is not exactly a Shingling-based method, it uses almost similar definitions proposing a selection strategy that is dependent on document collection properties. They apply a preprocessing step that results in more accuracy and scalability. After tokenizing the documents, an I-Match Lexicon L is constructed as the union of all the tokens of the documents. L is then pruned using a collection statistics measure that is the inverse document frequency (IDF), which is $\log(N/n)$, where N is the number of documents in the collection and n is the number of documents containing the given term. For each document, a set of terms U is built and the more important terms of this set are selected by finding the intersection between L and U , removing terms from U that do not exist in L . Consequently, it maps the resultant set to a single hash value called the I-Match signature by using the SHA1 hash algorithm. In I-Match, two documents are near duplicate if their signatures match. I-Match signatures are rejected for intersection values below a threshold. I-Match in this definition generates many false-positive answers and has a low recall. It also needs tunings on the parameter IDF. In a modified way [25] proposed using several I-Match signatures (instead of one) for each

document by applying a random permutation on the original I-Match Lexicon and thereby creating multiple randomized lexicons. According to this method, the two documents are near duplicate if a certain number of their signatures match. This method is more robust and accurate.

Another pruning strategy is proposed by Wang and Chang [40], which is based on sentence-level statistics of the documents. In their method, the documents are converted into sentence-level statistics, which are a sequence of consecutive numbers representing the number of words in each sentence. Then, ***k*-Strings** are generated by moving a fixed-sized sliding window of size k across the set. The candidate documents will be chosen based on the number of blocks with shared k -Strings.

2.2 Signature-based NDD methods

Locality sensitive hashing (LSH) proposed by Har-Peled et al. [19] is an approximate similarity search framework used for high-dimensional and large-scale NDD. **MinHash** is one of the widely used LSH techniques which was originally designed for NDD in Broder [7]. MinHashing for NDD comprises two main steps—sketching and bucketing. Sketches are randomized summary of the documents [17]. MinHash sketching is a way of embedding data in a reduced dimension matrix through a probabilistic method. Several sketching algorithms have been proposed in recent years to improve the MinHash including the k -mins sketching [13], the bottom- k sketching [14], the b -bit sketching [28], the k -partition sketching [29], the min-max sketching [24], the odd sketching [31] and the BitHash sketching [43]. In banding technique, sequences of r MinHashes are grouped together forming b bands. Further, each band is converted into a band hash value by the use of a regular hash function. The same band hashes group into a fixed number of buckets. This process repeats several times by several hash functions resulting in multiple hash tables. Finally, in a MinHash-LSH, two documents are considered near duplicate if they collide into the same bucket in at least one hash table. Sign normal random projections or **SimHash** [9] is another LSH approximate algorithm to efficiently estimate how two similar documents are. In it, similar documents are hashed to similar hash values.

In addition to the aforementioned methods that focus on NDD, several methods in plagiarism detection are relevant to NDD. Winnowing [35] and its extended version [36] is one of them. Winnowing is a local fingerprinting method that selects fingerprints from hashes of k -grams. Detecting the origin of text segments [1] and text reuse detection [23] are other approaches that are close to NDD.

3 Algorithm

3.1 Concepts and notions

No heavy modifications occur in web news reposting, but differences originate from a limited number of changes in some of the sentences or phrases and the other parts remain intact. The reason for these changes can be any semantic, grammatical, lexical editing. At token level, in two near-duplicate news articles, the token sets of the documents contain a lot of common tokens and a limited number of uncommon tokens. In a boundary situation, in which there is no uncommon token in either of the token sets, the two documents are considered as true

duplicates. True duplicates are either accepted as a near-duplicate status or dropped from the collection in a prior step based on the application requirements.

Definition 1 Let D be a corpus of web news articles and d_i and d_j be two news articles in D . d_i and d_j are considered near duplicates if:

$$\begin{cases} |T(d_i) - (T(d_i) \cap T(d_j))| \ll |T(d_i) \cap T(d_j)| \\ |T(d_j) - (T(d_i) \cap T(d_j))| \ll |T(d_i) \cap T(d_j)| \end{cases} \quad (4)$$

where $T()$ represents the token set of the body of the news articles, and the two conditions take place simultaneously. Thus, the ratio $r = |T(d_i) - (T(d_i) \cap T(d_j))| / (T(d_i) \cap T(d_j)) * 100\%$, which is a ratio of the uncommon tokens of each article to the common tokens, defines the status. It is a positive real number in the range of $[0, \infty)$. The lesser the r , the more similar the articles. In this definition:

- for two true-duplicate articles, $r = 0$
- for two near-duplicate articles, $r \ll 100\%$
- if the number of uncommon terms be exactly twice greater than the number of common terms, $r = 100\%$
- if the number of uncommon terms be greater than $2 * |T(d_i) \cap T(d_j)|$, then $r > 100\%$.

The proper value for r for considering the two documents near-duplicate depends on the application area and the characteristics of the dataset. There is no agreement on a specific value in previous works. It is usually selected experimentally. However, a common understanding is that r should be small enough and the two documents should share substantial percentage of their contents in a way that a potential reader judges them as near-duplicate. One way to find a trustable value for r is to extract it from a manually labeled dataset in the same or similar context.

In the proposed strategy, the documents are parsed into punctuation-delimited blocks called *chunks*. Chunks are contiguous non-overlapping sequences of tokens in a document, which appear in between two punctuation marks such as “;”, “:”, “.”, “!”, and “?”. Chunks come in variable sizes. Two near-duplicate documents have some chunks in common, and each of them may contain some uncommon chunks as well. In Fig. 2, the intended notion is demonstrated on parts of two near-duplicate news articles from a manually labeled NDD corpus, the GoldSet [37]. In the figure, common chunks are bordered and uncommon chunks are highlighted. As the figure shows, common chunks are the same in both documents, whereas uncommon chunks are nearly the same, but not exactly. The uncommon chunks are the sources of near duplication in the two documents.

Referring to near-duplicate documents, four intuitions can be deduced:

1. Two near-duplicate documents contain some common chunks and some uncommon chunks.
2. The number of common chunks in the two near-duplicate documents is greater than the number of uncommon chunks.
3. Contents of the common chunks in a pair of documents convey no useful information in the process of NDD.
4. The number of the common and the uncommon chunks in a pair of documents convey useful information in the process of NDD.

Therefore, each document is converted into a chunk set $\text{Chunk}(d)$. By pairwise comparison of the chunk sets of two documents, the set that conveys the common chunks of $\text{Chunk}(d_i)$ and $\text{Chunk}(d_j)$ which appear in their intersection is called $\text{Chunk}_C(d_i, d_j)$; the sets that convey the uncommon chunks of $\text{Chunk}(d_i)$ and $\text{Chunk}(d_j)$ are called $\text{Chunk}_U(d_i)$ and $\text{Chunk}_U(d_j)$, respectively. The chunk set of each document is the union of the common and uncommon chunk sets.

$$\text{Chunk}(d_i) = \text{Chunk}_C(d_i, d_j) \cup \text{Chunk}_U(d_i) \tag{5}$$

$$\text{Chunk}(d_j) = \text{Chunk}_C(d_i, d_j) \cup \text{Chunk}_U(d_j) \tag{6}$$

where

$$\text{Chunk}_C(d_i, d_j) = \text{Chunk}(d_i) \cap \text{Chunk}(d_j) \tag{7}$$

$$\text{Chunk}_U(d_i) = \text{Chunk}(d_i) - \text{Chunk}_C(d_i, d_j) \tag{8}$$

$$\text{Chunk}_U(d_j) = \text{Chunk}(d_j) - \text{Chunk}_C(d_i, d_j) \tag{9}$$

Essentially, an equality exists between the chunk-level and the Shingle-level representation of the two documents:

$$\begin{aligned} d_i \cup d_j &\equiv \text{Chunk}_C(d_i, d_j) \cup \text{Chunk}(d_i) \cup \text{Chunk}(d_j) \\ &\equiv S(\text{Chunk}_C(d_i, d_j)) \cup S(U_i) \cup S(U_j) \cup S(C) \end{aligned} \tag{10}$$

Equation (10) is true since all the sets are defined as disjoint. The schema is illustrated in Fig. 3 in the form of Venn diagrams.

The notion is to filter the document pairs in a chunk level; then, the intention is to investigate near duplication only on selected structural blocks that are the reason for near duplication of the two documents, i.e., the uncommon chunks. In fact, $\text{Chunk}_C(d_i, d_j)$, which is expected to have greater cardinality in comparison to the two uncommon sets, will be bypassed.

3.2 Main steps

The main steps of the proposed method and its position in a NDD system are provided in Fig. 4.

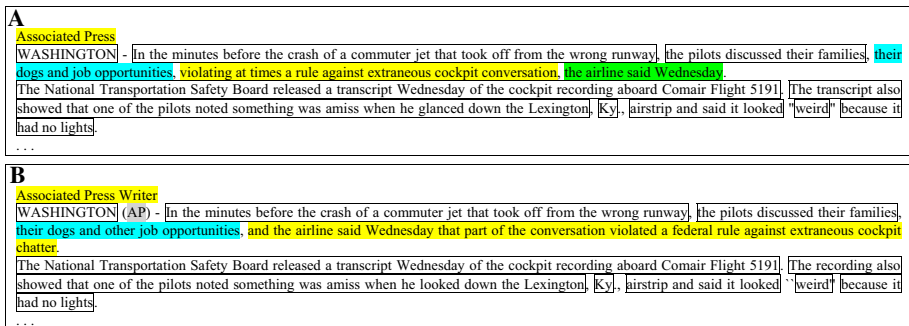


Fig. 2 Common and uncommon chunks demonstrated on parts of two near-duplicate news articles

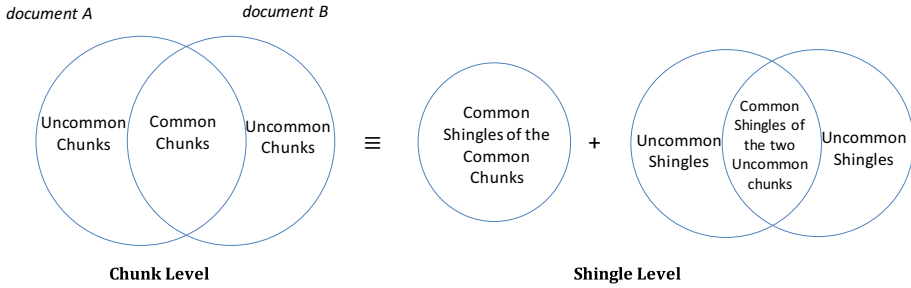


Fig. 3 Scheme of two near-duplicate documents from two different perspectives of chunk level and Shingle level

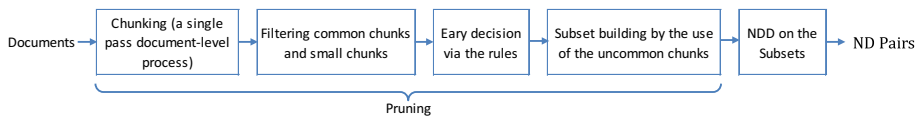


Fig. 4 The main steps of the proposed pruning method and its position in a NDD system

In the **chunking step**, chunks are detected by parsing directly the source code of the two documents, d_i and d_j . To refine this method, some constraints may be placed. For example, the Tab and Newline HTML tags would be considered as punctuation marks. This can be defined for other markup languages such as XML in the same way. In **filtering step**, a chunk threshold (CHT) is defined to drop the chunks with lengths (in tokens) less than the CHT. For instance, $CHT = 5$ means that only the chunks that contain more than five tokens are retained. By this consideration, several noise contents including navigational texts, links, advertisements will be omitted automatically because in most cases, their lengths are less than the threshold value. Therefore, no preprocessing is needed and after applying the proposed steps, only the intended contents would remain. CHT can be set to tune the level of noise suppression. Under this condition, common and uncommon chunks are built by intersecting the two chunk sets. Since common chunks are bitwise similar, being common or uncommon is detected by the use of a simple string compare function with no need for any natural-language preprocessing or processing.

Afterward, in **early decision step**, some consecutive rules are applied. There is a pre-assumption that $Chunk(d_i)$ and $Chunk(d_j)$ are not empty sets.

Rule 1: If the $Chunk_C(d_{i,j})$ is an empty set, which means that no common chunks are detected between the pairs, the Jaccard coefficient would be zero and the pairs are labeled as “non-duplicate” with no need for calculating the Jaccard coefficient in a Shingle-level approach.

Rule 2: Else, if the $Chunk_U$ in both of the documents is empty, the Jaccard coefficient would be one and result is a “true-duplicate” with no need for calculating the Jaccard coefficient in a Shingle-level approach because in addition to having common chunks, no uncommon chunk is detected.

Rule 3: Else, if the ratio of the cardinality of $Chunk_U(d_j)$ to the cardinality of $Chunk_C(d_{i,j})$ is more than a threshold value (Upper-Bound Condition), the pair would be considered “non-duplicate” with no need for calculating the Jaccard coefficient using a Shingle-level approach. Nevertheless, this situation may occur because of some general verbal editing

or substitution of letters, and since in the proposed strategy the preprocessing that includes the normalization task is not applied on the texts prior to this step, the condition may or may not be applied depending on the definition of near duplication in hand and the intended policy of NDD.

Rule 4: Else, if none of the above consecutive conditions is satisfied, the two documents are considered as “suspicious” and they are passed to the next fold for further investigations in a Shingle-level approach.

In this phase, it is anticipated that many pre-decisions have to be made and only a fraction of the document pairs that are labeled as suspicious will enter into the **subset building step**. Now, the common chunks of the two documents are filtered out, as they are the same and contain no extra information with regard to NDD. Only the uncommon chunk sets of the two documents are preserved. As for the example shown in Fig. 2, the two uncommon chunk sets of documents A and B are shown below:

$$\text{Chunk}_U(A) = \{(\text{their dogs and job opportunities}), (\text{violating at times a rule against extraneous cockpit conversation}), (\text{the airline said Wednesday})\}$$

$$\text{Chunk}_U(B) = \{(\text{their dogs and other job opportunities}), (\text{and the airline said Wednesday that part of the conversation violated a federal rule against extraneous cockpit chatter})\}$$

Since some of the logical differences in the two near-duplicate documents may be a result of merging or dividing adjacent chunks, as the example also depicts, the Chunk_U of each document concatenates to form d_i' and d_j' as the two documents' *subsets*.

$$A' = \text{"their dogs and job opportunities violating at times a rule against extraneous cockpit conversation the airline said Wednesday"}$$

$$B' = \text{"their dogs and other job opportunities and the airline said Wednesday that part of the conversation violated a federal rule against extraneous cockpit chatter"}$$

Finally, the two generated subsets, d_i' and d_j' , get through the **NDD on subsets step**. The k -Shingling with k equal to 1 is applied in this step. This selection has been made to theoretically compare the proposed strategy to conventional strategies. However, the robustness of this selection ($k = 1$) has also been examined experimentally. The pseudo-code of the proposed algorithm is presented below:

Algorithm 1: Pseudo-code of the proposed algorithm

INPUT: Document sets $D = \{d_1 \dots d_n\}$

OUTPUT: Result, the status of d_i and d_j against each other

```

1:   Begin
2:   initialize Result
3:   parse all the documents in  $D$ , build chunk sets  $\text{Chunk}(\cdot)$ 
4:   for each  $d_i \in D, 0 < i \leq n$  do
5:       for each  $d_j \in D, i < j \leq n$  do
6:           build  $\text{Chunk}_C(d_i, d_j), \text{Chunk}_U(d_i), \text{Chunk}_U(d_j)$ 
7:           if  $\text{Chunk}_C(d_i, d_j)$  is empty then
8:               Result  $\leftarrow (d_i, d_j, \text{"Non-duplicate"})$ 
9:           else if  $\text{Chunk}_U(d_i)$  is empty AND  $\text{Chunk}_U(d_j)$  is empty then
10:              Result  $\leftarrow (d_i, d_j, \text{"Near-duplicate"})$ 
11:          else if cardinality of  $\text{Chunk}_U(d_j)$  / cardinality of  $\text{Chunk}_C(d_i, d_j) > \text{UpperBoundCondition}$ 
then
12:              Result  $\leftarrow (d_i, d_j, \text{"Non-duplicate"})$ 
13:          else
14:               $d_i^* \leftarrow \text{Concatenated } \text{Chunk}_U(d_i)$ 
15:               $d_j^* \leftarrow \text{Concatenated } \text{Chunk}_U(d_j)$ 
16:              if  $\varphi(d_i^*, d_j^*) < \text{SimilarityThreshold}$  then
17:                  Result  $\leftarrow (d_i, d_j, \text{"Non-duplicate"})$ 
18:              else
19:                  Result  $\leftarrow (d_i, d_j, \text{"Near-duplicate"})$ 
20:              end if
21:          end for
22:      end for
23:      return Result
24:  End

```

In the list above, the Upper-Bound Condition and the Similarity Threshold are the two chunk-level parameters. They are deduced theoretically based on existing parameters and are defined by theoretical analysis presented in Sect. 3.3. Overall, the solution of preprocessing and then applying NDD on the complete parts of all the document pairs of the corpus is reduced to simply, applying the similarity function on the subset of only a fraction of the document pairs. The computational complexity of the proposed selection strategy is $O(1)$. Also, the computational complexity of a NDD algorithm with proposed strategy is $O(cN^2)$, where c is the ratio of pairs labeled as “suspicious” to all the corpus pairs, and N is the number

of documents. The space requirement may also be very small in comparison to the original documents since $|d'| < |d|$.

3.3 Theoretical analysis

The theoretical analysis of the proposed pruning strategy is provided in the form of three theories as presented below. The purpose is to relate the elements of the standard definition of near duplication that are in Shingle level to the presented definition in chunk level.

Theory 1 *Let d_i and d_j be two documents, where $|d_i| \leq |d_j|$. Let $J(S(d_i).S(d_j))$ be the Jaccard coefficient of d_i and d_j and let τ be the similarity threshold. Then,*

$$J(S(d_i).S(d_j)) \geq \tau \Rightarrow \frac{|S(\text{Chunk}_U(d_j))|}{|S(\text{Chunk}_C(d_i.d_j))|} \leq \frac{(1 - \tau)}{\tau} \tag{11}$$

where $S()$ represents the Single set, $J(.)$ represents the Jaccard coefficient, and $\text{Chunk}_U(d_j)$ and $\text{Chunk}_C(d_i.d_j)$ are sets with definitions that are provided above. The equation gives a chunk-level boundary metric that is the Upper-Bound Condition used in the Rule (3). This boundary metric is based on the Shingle-level parameter τ .

Proof Since chunks are defined as unique contiguous non-overlapping sequences of k tokens, the Shingle set of the $\text{Chunk}(d_j)$ is equivalent to the Shingle set of d_j for k equal to one. By replacing $S(d_i)$ by $S(\text{Chunk}(d_i))$ and $S(d_j)$ by $S(\text{Chunk}(d_j))$, referring to the definition of the near-duplicate condition based on the Jaccard coefficient, and by replacing (5) and (6) in the Jaccard definition, the near-duplicate condition based on the Jaccard coefficient in the proposed method is as follows:

$$J(S(d_i).S(d_j)) = \frac{|S(\text{Chunk}(d_i)) \cap S(\text{Chunk}(d_j))|}{|S(\text{Chunk}(d_i)) \cup S(\text{Chunk}(d_j))|} \tag{12}$$

On the other hand, as an obvious condition for any two sets, we have

$$|S(\text{Chunk}(d_i)) \cap S(\text{Chunk}(d_j))| \leq \min\{|\text{Chunk}(d_i)|, |\text{Chunk}(d_j)|\} \tag{13}$$

And

$$|S(\text{Chunk}(d_i)) \cup S(\text{Chunk}(d_j))| \geq \max\{|\text{Chunk}(d_i)|, |\text{Chunk}(d_j)|\} \tag{14}$$

Assuming that $|d_i| \leq |d_j|$, and accordingly, $|\text{Chunk}_U(d_i)| \leq |\text{Chunk}_U(d_j)|$, Eq. (12) can be extended as shown below:

$$\frac{|S(\text{Chunk}(d_i))|}{|S(\text{Chunk}(d_j))|} \geq \frac{|S(\text{Chunk}(d_i)) \cap S(\text{Chunk}(d_j))|}{|S(\text{Chunk}(d_i)) \cup S(\text{Chunk}(d_j))|} \geq \tau \tag{15}$$

By the transitivity property as well as by replacing (5) and (6) in (15), there would be:

$$\frac{|S(\text{Chunk}_C(d_i.d_j)) \cup S(\text{Chunk}_U(d_i))|}{|S(\text{Chunk}_C(d_i.d_j)) \cup S(\text{Chunk}_U(d_j))|} \geq \tau \tag{16}$$

For the numerator $|S(\text{Chunk}_C(d_i.d_j)) \cup S(\text{Chunk}_U(d_i))| = |S(\text{Chunk}_C(d_i.d_j))| + |S(\text{Chunk}_U(d_i))|$, as the worst-case condition, having $|S(\text{Chunk}_U(d_i))| = 0$ leads to the following result:

$$\frac{|S(\text{Chunk}_C(d_i.d_j))|}{|S(\text{Chunk}_C(d_i.d_j))| + |S(\text{Chunk}_U(d_j))|} \geq \tau \tag{17}$$

Finally, solving (17) results in Eq. (11). □

Theory 2 Let d_i and d_j be two documents, where $|d_i| \leq |d_j|$. Let $J(S(d_i).S(d_j))$ be the Jaccard coefficient of d_i and d_j and let τ the similarity threshold. In addition, let $J(S(d'_i).S(d'_j))$ be the Jaccard coefficient of d'_i and d'_j , where d'_i and d'_j are two subsets built on the concatenated strings of $\text{Chunk}_U(d_i) - (\text{Chunk}_U(d_i) \cap \text{Chunk}_U(d_j))$ and $\text{Chunk}_U(d_j) - (\text{Chunk}_U(d_i) \cap \text{Chunk}_U(d_j))$ in a Shingle-level approach. In consequence,

$$J(S(d_i).S(d_j)) \geq J(S(d'_i).S(d'_j)) \tag{18}$$

This theory explores the relationship between the pairwise similarity score of two near-duplicate documents, and the similarity score of their subsets, when the subsets are built upon the above-mentioned instruction. It means that the similarity score of two near-duplicate documents would be greater than or equal to the similarity score of their subsets.

Proof Expanding (11) on Shingle-level approach will give:

$$J(S(d_i).S(d_j)) = \frac{|S(d_i) \cap S(d_j)|}{|S(d_i) \cup S(d_j)|} = \frac{|S(\text{Chunk}_C(d_i.d_j))| + |S(C)|}{|S(\text{Chunk}_C(d_i.d_j))| + |S(U_i)| + |S(U_j)| + |S(C)|} \tag{19}$$

where U_i , U_j and C are uncommon and common sets from intersecting $\text{Chunk}_U(d_i)$ and $\text{Chunk}_U(d_j)$ on a Shingle-level operation. Since $|S(\text{Chunk}_C(d_i.d_j))|$ is a positive integer equal to or greater than zero and $|S(C)| < |S(U_i)| + |S(U_j)| + |S(C)|$, subtracting $|S(\text{Chunk}_C(d_i.d_j))|$ from both numerator and denominator of the fraction (19) results in the following smaller fraction:

$$\frac{|S(\text{Chunk}_C(d_i.d_j))| + |S(C)|}{|S(\text{Chunk}_C(d_i.d_j))| + |S(U_i)| + |S(U_j)| + |S(C)|} \geq \frac{|S(C)|}{|S(U_i)| + |S(U_j)| + |S(C)|} \tag{20}$$

On the other hand,

$$\begin{aligned} J(S(d'_i).S(d'_j)) &= J(S(d_i - \text{Chunk}_C(d_i.d_j)).S(d_j - \text{Chunk}_C(d_i.d_j))) \\ &= \frac{|S(C)|}{|S(U_i)| + |S(U_j)| + |S(C)|} \end{aligned} \tag{21}$$

Hence, introducing (20) into (21) results in Eq. (18). □

Theory 3 Let d_i and d_j be two documents, where $|d_i| \leq |d_j|$. Let $J(S(d_i).S(d_j))$ be the Jaccard coefficient of d_i and d_j and let τ be the similarity threshold. In addition, let $J(S(d'_i).S(d'_j))$ be the Jaccard coefficient of d'_i and d'_j , where d'_i and d'_j are two subsets built on the concatenated strings of $\text{Chunk}_U(d_i) - (\text{Chunk}_U(d_i) \cap \text{Chunk}_U(d_j))$ and $\text{Chunk}_U(d_j) - (\text{Chunk}_U(d_i) \cap \text{Chunk}_U(d_j))$ in a Shingle-level approach. In consequence,

$$\begin{aligned}
 1 \geq J(S(d_i).S(d_j)) \geq \tau &\Rightarrow 1 \geq J(S(d'_i).S(d'_j)) \\
 &\geq \tau - \frac{(|S(U_i)| + |S(U_j)|)(|S(\text{Chunk}_C(d_i.d_j))|)}{(|S(U_i)| + |S(U_j)| + |S(C)|)(|S(\text{Chunk}_C(d_i.d_j))| + |S(U_i)| + |S(U_j)| + |S(C)|)}
 \end{aligned} \tag{22}$$

This theory gives the similarity threshold required to perform the last step of the proposed strategy, which is to NDD on the generated subsets. By this, having a similarity threshold, we are able to transform it into the new condition defined in this study and apply it on the generated subsets.

Proof First, we multiply $(|S(U_i)| + |S(U_j)| + |S(C)|)$ with both the numerator and the denominator of $J(S(d_i).S(d_j))$.

$$\begin{aligned}
 J(S(d_i).S(d_j)) &= \frac{(|S(U_i)| + |S(U_j)| + |S(C)|)(|S(\text{Chunk}_C(d_i.d_j))| + |S(C)|)}{(|S(U_i)| + |S(U_j)| + |S(C)|)(|S(\text{Chunk}_C(d_i.d_j))| + |S(U_i)| + |S(U_j)| + |S(C)|)} \\
 &= \frac{(|S(C)|)(|S(\text{Chunk}_C(d_i.d_j))|) + (|S(C)|)^2 + (|S(C)|)(|S(U_i)| + |S(U_j)|) + (|S(\text{Chunk}_C(d_i.d_j))|)(|S(U_i)| + |S(U_j)|)}{(|S(U_i)| + |S(U_j)| + |S(C)|)(|S(\text{Chunk}_C(d_i.d_j))| + |S(U_i)| + |S(U_j)| + |S(C)|)} \\
 &= \frac{(|S(C)|)(|S(\text{Chunk}_C(d_i.d_j))|) + (|S(C)|)^2 + (|S(C)|)(|S(U_i)| + |S(U_j)|)}{(|S(U_i)| + |S(U_j)| + |S(C)|)(|S(\text{Chunk}_C(d_i.d_j))| + |S(U_i)| + |S(U_j)| + |S(C)|)} \\
 &\quad + \frac{(|S(\text{Chunk}_C(d_i.d_j))|)(|S(U_i)| + |S(U_j)|)}{(|S(U_i)| + |S(U_j)| + |S(C)|)(|S(\text{Chunk}_C(d_i.d_j))| + |S(U_i)| + |S(U_j)| + |S(C)|)}.
 \end{aligned} \tag{23}$$

On the other hand, multiplying $(|S(\text{Chunk}_C(d_i.d_j))| + |S(U_i)| + |S(U_j)| + |S(C)|)$ with both the numerator and the denominator of $J(S(d'_i).S(d'_j))$ leads to the following results

$$\begin{aligned}
 J(S(d'_i)..S(d'_j)) &= \frac{(|S(C)|)(|S(\text{Chunk}_C(d_i.d_j))| + |S(U_i)| + |S(U_j)| + |S(C)|)}{(|S(U_i)| + |S(U_j)| + |S(C)|)(|S(\text{Chunk}_C(d_i.d_j))| + |S(U_i)| + |S(U_j)| + |S(C)|)}
 \end{aligned} \tag{24}$$

Now, (24) is equal to the first part of Eq. (23) and can be replaced as follows:

$$\begin{aligned}
 (S(d_i).S(d_j)) &= J(S(d'_i).S(d'_j)) \\
 &\quad + \frac{(|S(\text{Chunk}_C(d_i.d_j))|)(|S(U_i)| + |S(U_j)|)}{(|S(U_i)| + |S(U_j)| + |S(C)|)(|S(\text{Chunk}_C(d_i.d_j))| + |S(U_i)| + |S(U_j)| + |S(C)|)}
 \end{aligned} \tag{25}$$

which makes Eq. (22) true. □

4 Evaluation

First, the proposed method with several configurations was applied on the dataset with the aim of experimental evaluation of the parameter space. Second, the results were compared with the three common pruning strategies. Since the focus of the manuscript is on the selection strategies in NDD, NDD-related evaluation criteria were used. Effectiveness was evaluated by three measures: precision, recall and *F*-measure. Precision is the ratio of the number

Table 2 GoldSet corpus characteristics

Size	99.2 MB (XML files), 6.5 MB (extracted main texts)
Number of near-duplicate groups	68
Number of documents	2168
Number of near-duplicate pairs	82,934
Average number of words per document	2568

of correctly identified near-duplicate pairs divided by the total number of identified near-duplicate pairs. Recall is the ratio of the number of the identified near-duplicate pairs, divided by the number of near-duplicate pairs. *F*-measure is an effectiveness measure based on recall and precision. It summarizes the effectiveness of the method in a single number.

$$\text{Precision} = \frac{\text{Number of correctly identified near-duplicate pairs}}{\text{Number of identified near-duplicate pairs}} \quad (26)$$

$$\text{Recall} = \frac{\text{Number of identified near-duplicate pairs}}{\text{Number of near-duplicate pairs}} \quad (27)$$

$$F\text{-measure} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad (28)$$

To calculate the above-mentioned measures, having a threshold, first, the Algorithm 1 was run and a result was achieved for each pair of documents. Thereafter, the results were compared with the labels. Measures are tested on the GoldSet corpus [37]. Characteristics of the GoldSet corpus are provided in Table 2.

It should be mentioned that in all the experiments of the proposed strategy, the algorithm was applied directly on the web page source codes to investigate the ability of the proposed algorithm to surpass several types of noises as well as in order to possibly eliminate some preprocessing steps. In contrast, for the baseline methods in line with the previous works, all the algorithms were applied on the main body of the news articles extracted from the web pages, which loaded extra complexities on to the system. All the experiments were conducted on the same machine with an Intel® 2.40 GHz CPU and 8 GB RAM.

5 Results and discussion

This section provides experimental results in three parts. First, some exploratory experiments were conducted to investigate the characteristics of the dataset regarding different aspects of the proposed strategy. Second, a group of experiments was designed to fine-tune the parameters of the proposed strategy, including CHT and τ . Finally, the third part of Sect. 5 was for evaluating *F*-measure, and execution of the proposed strategy (with best parameters setting) and some recent and well-known strategies.

5.1 Exploratory experiments

Corpus was analyzed with regard to the proposed notion of near duplication. To evaluate the strength of Definition 1, the ratio r was computed for each near-duplicate pair using the

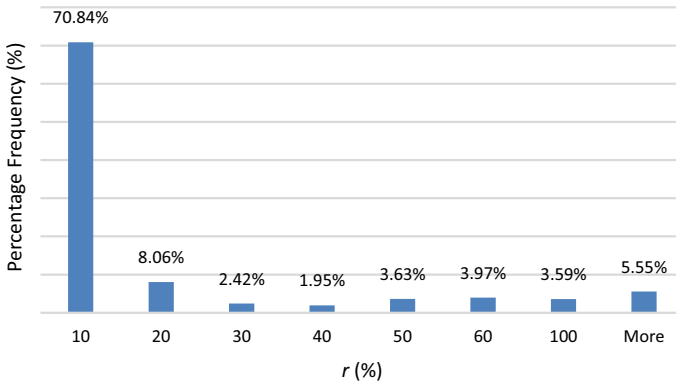


Fig. 5 Percentage frequency of r

dataset labels. It should be mentioned that for calculating $T()$, the body of the news articles were extracted first. As Fig. 5 depicts, in more than 70% of the near-duplicate pairs the ratio achieved a value between zero and 10%. In other words, the ratio of uncommon tokens was 10% (or less) of the common tokens for more than 70% of the near-duplicate pairs. In this bin, around 44% of them (31.48% of total) were true duplicates with $r = 0$ and the rest were near duplicates. After that, in around 8% of them, r had a value between 10 and 20%. The results show that the definitions fit well in most of the near-duplicate cases. Going back to the dataset, most of the pairs with r greater than 50% were found to be related to the in-progress news of some incidents.

Detailed distribution frequencies of the number of chunks in the documents for different CHTs are provided in Appendix 1. The results show that the distribution frequency of the chunks decreases tremendously with an increase in CHT. This was also confirmed by analyzing the statistics of the chunked datasets that apply different CHTs (Table 3). Applying CHT = 4 resulted in documents with average lengths of 573.07 words or 70.49 chunks, whereas applying CHT = 20 resulted in documents with average lengths of 46.33 words or 2.04 chunks. The size of the pruned dataset decreased with an increase in the CHT. Comparing Tables 2 and 3 reveals a significant difference between the average number of words per document (and consequently the dataset size) in the original and the chunked case. The decrease in the length of the documents to be compared results in more accurate and efficient NDD.

As another test on the dataset, the average Jaccard coefficient (k -Shingling with different k values from 1 to 3) of the subsets of the near-duplicate pairs with different CHTs is shown in Fig. 6. This figure reveals that the pairwise similarity scores of the subsets of near-duplicate pairs for k values greater than 1 are small and do not significantly change from CHT = 4 to CHT = 20. This makes sense because the subsets are small and k -Shingling with k values greater than one does not work well for small-sized texts. Therefore, $k = 1$ was selected for the rest of the experiments. Moreover, for $k = 1$, by increasing the CHT, the Jaccard coefficient, which is the similarity score between the pairs of near-duplicate news articles, increases and the subsets become more coherent. This happens because by increasing the CHT, many of the noise contents are disregarded and the residue is closer to the main contents (which are near duplicate, pairwise). In addition, owing to an intrinsic property of the news articles, smaller chunks—for instance, times and dates, small descriptive phrases, pronouns—are more exposed to slight changes in producing near-duplicate documents.

Table 3 Characteristics of pruned datasets with different values for CHTs ($N = 2168$)

Chunking threshold	Ave. length of chunked documents (words)	Ave. length of chunked documents (chunks)	Size of chunked dataset (MB)
CHT = 4	573.07	70.49	11.5
CHT = 5	497.62	54.05	9.5
CHT = 6	424.14	41.05	8.5
CHT = 7	368.29	32.50	6.5
CHT = 8	325.08	26.84	5.5
CHT = 9	280.89	21.49	5.5
CHT = 10	244.55	17.51	4.5
CHT = 11	207.28	13.80	3.5
CHT = 12	177.69	11.12	3.5
CHT = 13	152.61	9.01	2.5
CHT = 14	136.63	7.75	2.5
CHT = 15	117.55	6.43	2.5
CHT = 16	100.93	5.30	2.5
CHT = 17	85.21	4.30	1.5
CHT = 18	69.05	3.32	1.5
CHT = 19	54.19	2.47	1.5
CHT = 20	46.33	2.04	1.5

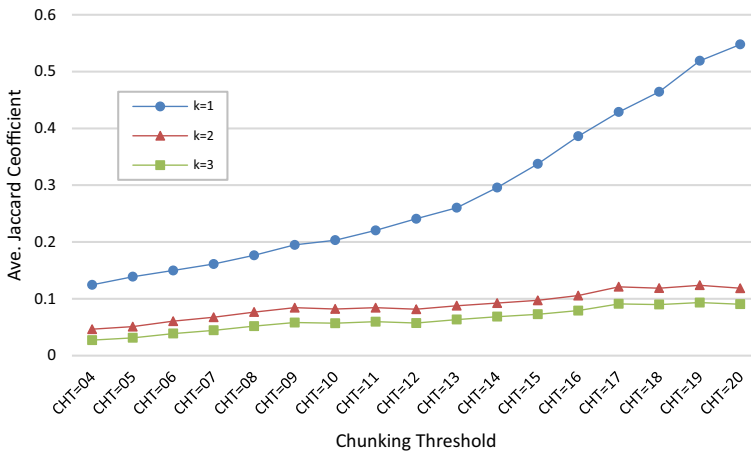


Fig. 6 Average Jaccard coefficient of the subsets in the near-duplicate pairs of the corpus for different CHTs and k values

5.2 Parameter setting

Precision of the proposed strategy with different similarity thresholds is provided in Fig. 7. Since in the proposed strategy only the subsets are evaluated, rather than the whole texts, the similarity threshold discriminating between near duplicate and non-duplicates is different. As the figure demonstrates, for CHTs from 5 to 12, and similarity thresholds between 0.2 and 0.9, the precision takes more acceptable values. In this period, the precision acts independently

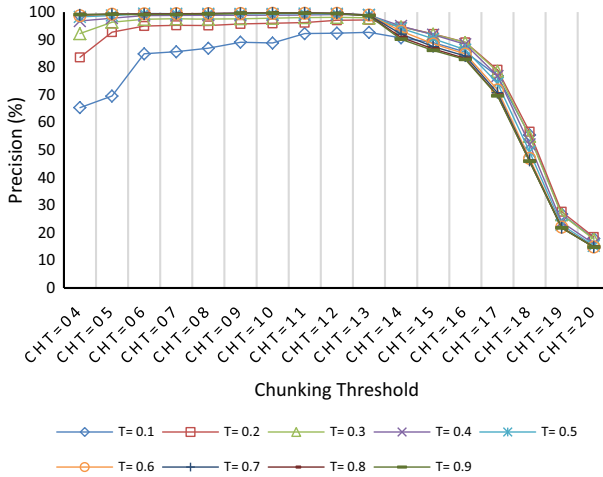


Fig. 7 Precision of NDD with the proposed strategy with different similarity thresholds

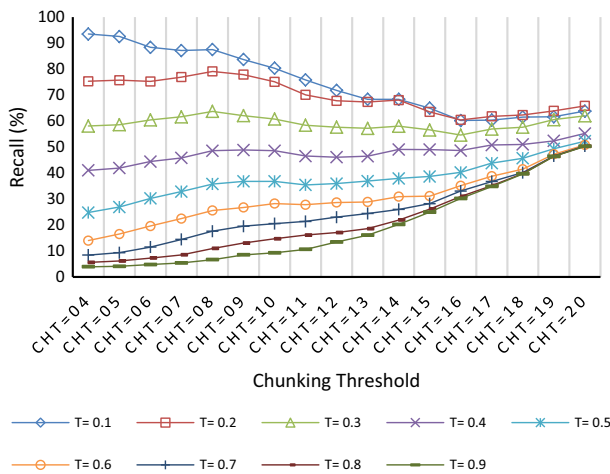


Fig. 8 Recall of NDD with the proposed strategy with different similarity thresholds

from the CHT. In increasing the CHT from 13 to 20, the precision drops. By setting the similarity threshold to 0.1, the algorithm has the lowest precision values in all experiments. However, regardless $\tau = 0.1$, the precision acts almost consistent in all the experiments independent of the similarity threshold. The recall of the proposed strategy with different similarity thresholds is provided in Fig. 8. In contrast with the precision, the recall is greater for similarity thresholds equal to 0.1 and 0.2. The recall of the other settings takes place under these two values. CHTs equal to 4 and 5 have the greatest recalls.

The *F*-measure of the proposed strategy with different similarity thresholds is provided in Fig. 9. This figure confirms that by selecting a similarity threshold equal to 0.1 or 0.2, the best performances are achieved. In addition, CHTs in range of 6–9 give the best performances. Table of details on precision, recall, and *F*-measure values is provided in Appendix 2. The

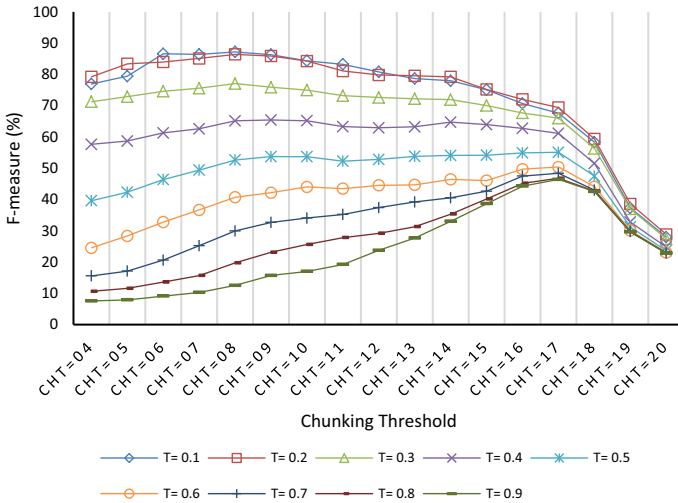


Fig. 9 F-measure of the proposed strategy with different similarity thresholds

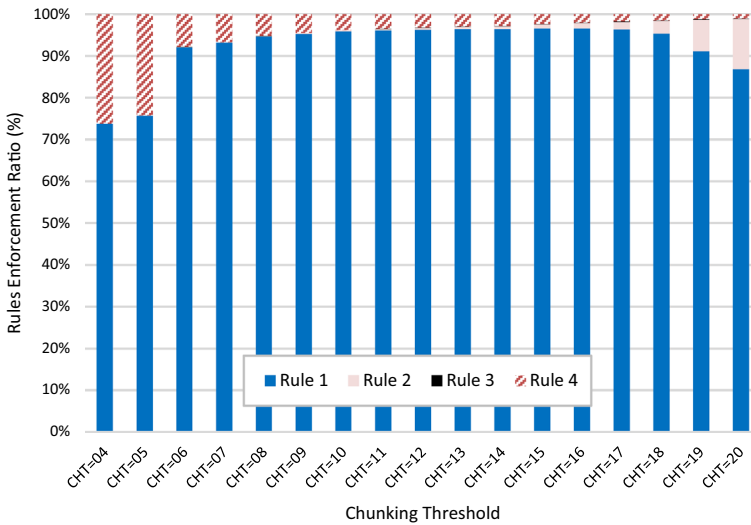


Fig. 10 The enforcement of each filtering rule in making the final decision for different CHTs

similarity threshold advised for the proposed strategy is essentially small (between 0.1 or 0.2), which is revealed by theoretical analysis as well.

By embedding a counter in the algorithm, the enforcement of each proposed filtering rule in making the final decision for different CHTs was counted. As Fig. 10 demonstrates, in all the experiments, most of the pre-decisions were made using Rule 1. Moreover, Rule 4, that is the actual NDD, takes place very rarely in comparison with the three other rules. These early decisions make it possible to disregard many of the non-necessary comparisons—on average 92.95% of them (STD = 9.95%).

In Table 4, the percentage of the suspicious pairs (i.e., the number of the times Rule 4 applies) to all the corpus pairs for the different CHTs are presented. The results indicate that

Table 4 The ratio of suspicious pairs to all the pairs for each CHT

CHT	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
<i>c</i> (%)	26.19	24.22	7.80	6.69	5.20	4.50	3.80	3.46	3.18	2.93	2.73	2.29	1.98	1.72	1.47	1.18	1.06

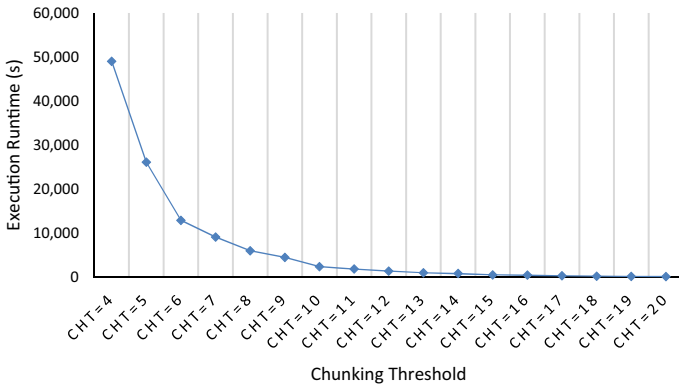


Fig. 11 The execution time of the proposed method for different CHTs

in the worst case, for CHT = 4, only 26.19% of the pairs require precise NDD investigations and the status of the other 73.81% of the pairs are pre-decided by the filtering rules. With an increase in the CHT, the percentage decreases sharply so that for CHT = 20, only 1.06% of the pairs require the application of NDD techniques and the status of the remaining 98.94% of the pairs is pre-decided by the three filtering rules that compare the cardinality of the common and the uncommon chunks together.

The execution time of the proposed method for different CHTs is presented in Fig. 11. The algorithm runs faster by increasing CHT.

Finally, CHT = 8 and $\tau = 0.1$ are selected for the rest of the study. In this configuration, most value for the *F*-measure is achieved. (Detailed results are presented in Appendix 2.) *F*-measure is a general compound metric. However, one may select another configuration that specifically reaches more precision or more recall. The selection is strongly dependent on the area of application.

To evaluate the behavior of the proposed method as the data size increases, experiments with different corpus sizes from 10 to 100% were conducted. The results are presented in Fig. 12.

As shown in Fig. 12, the *F*-measure of the proposed method varies in a range of 8.83% by varying the corpus size from 10 to 100%. The proposed method does not show a significant difference by changing the size of the corpus. The logical explanation is that it does not rely on any corpus-dependent statistics (such as the most frequent terms).

5.3 Evaluation

Figure 13 presents a comparison between *F*-measure of the proposed strategy and the three existing strategies including:

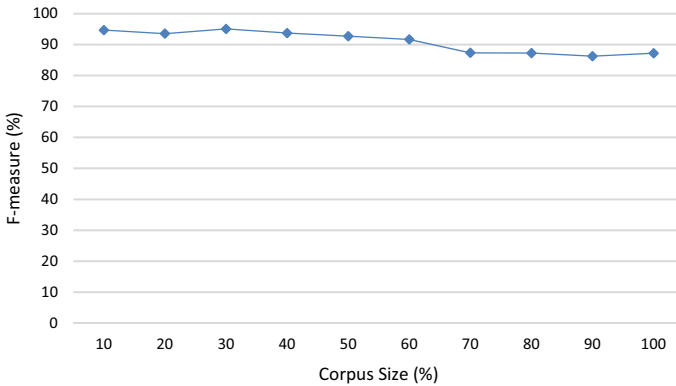


Fig. 12 *F*-measure of the proposed method at different corpus sizes

- *Selecting the Shingles randomly* 1-Shingling algorithm was applied, 200 random Shingles were selected, and NDD with $\tau = 0.6$ was conducted. The experiment was repeated 10 times, the average *F*-measure was put into the figure, and standard deviation was 1.4.
- *Selecting the most frequent Shingles* 1-Shingling algorithm was applied, 200 most frequent Shingles were selected, and NDD with $\tau = 0.6$ was conducted.
- *SpotSig pruning method* The original SpotSig deduplication method [37] benefited from three main steps including extracting Spot signatures, partitioning the dataset by applying a filtering rule based on a length threshold and inverted index pruning that is indexing documents by their Spot signatures rather than their Shingles. In the first step, which is a kind of Shingle-based pruning strategy, it provides a semantic preselection of Shingles. It wisely selects some frequent words of the text in accompany with a chain of words after each one, called Spot signatures. Significantly, only their proposed pruning strategy, which is the focus of this study and not the complete method, was applied in the experiment. The pruning strategy used in SpotSig is similar to the proposed strategy. Both of them reduce the corpus size semantically resulting into a much smaller corpus. Then, an NDD method, which is the calculation of the pairwise Jaccard coefficient, is applied on the reduced texts. On this note, the two Spot signatures (*SpotSig1* and *SpotSig2*) which had shown the best performance in the original article were selected as antecedents that are {it, a, there, was, said, the, is} and {a, there, was, said, the, is} with $d = 2$ and $c = 3$. These choices are valid since the same dataset is used in both studies.
- *SpotSig complete method* The original method with all the steps (antecedents: {a, there, was, said, the, is}, $d = 2$ and $c = 3$) was applied to provide a more thorough analysis of strengths and weaknesses of the proposed method in comparison to state-of-the-art methods.

For the first two baseline methods, parameters were selected in line with the previous observations and results of a set of experiments. The results of these experiments are provided in Appendix 3.

As the figure depicts, the proposed strategy has the second best *F*-measure (87.22%) among all the experiment results—with a small drop—after the complete SpotSig method with *F*-measure 90.12%. Thus, its *F*-measure is comparable with the SpotSig method.

In the following, results of the comparison of execution times for the proposed strategy against the baselines are presented in Fig. 14. As expressed before, for some of the baseline

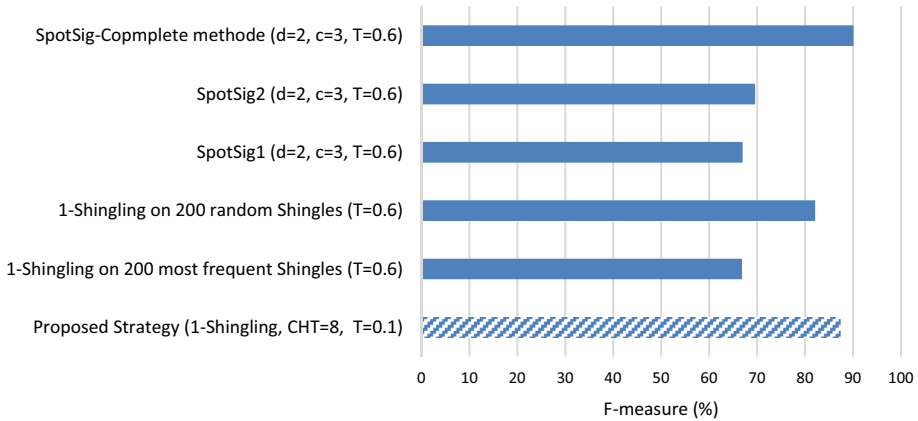


Fig. 13 Evaluation results

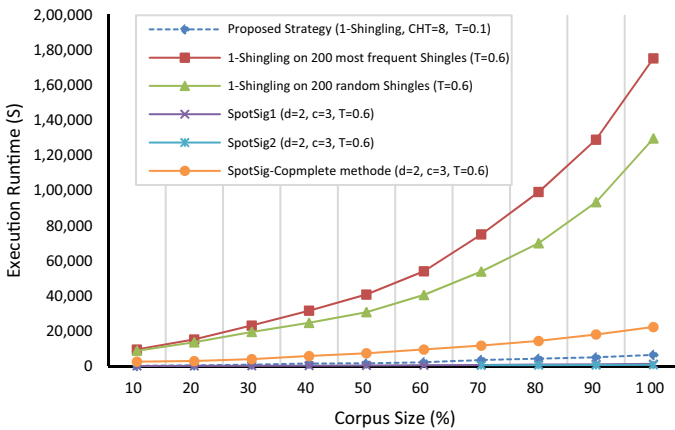


Fig. 14 Efficiency comparison of the proposed method against baseline for different corpus size

methods, the execution times include an extra process for bodies' extraction and preprocessing, while the proposed strategy does not convey any extraction process.

Results for *SpotSig1* and *SpotSig2* were found to be very close to each other. The runtime of the proposed strategy is comparable with *SpotSig1* and *SpotSig2*; however, it gives better *F*-measure regarding Fig. 13.

Finally, the proposed strategy outperforms *SpotSig* in terms of efficiency. However, it shows a smaller but still competitive effectiveness to *SpotSig*, with a 2.9% drop in *F*-measure. Here forward, some explanations on the origin of these results and further expectations are provided. It should be mentioned that both strategies need only a single pass over incoming text streams. In addition, both of them skip the text extraction phase. They are capable of filtering natural-language text passages out of noisy web page components through the application of wise preselection methods. Nevertheless, the proposed strategy possesses some privileges over *SpotSig*.

a) From effectiveness point of view:

Firstly, SpotSig needs three kinds of parameters to define an effective set of Spot signature, $A = \{a_j(d_j, c_j)\}$. Among them the stopword antecedents, a_j , is itself a well-defined subset of stopwords and there may exist several values for d_j and c_j . The original manuscript proposes nine Spot signatures and evaluates their effectiveness. However, firstly many other signatures are left unchecked. One may find different results examining other combinations, and the best options are unclear. Secondly, this approach is language dependent. Not the best English Spot signature can be applied to texts in other languages; nor the effectiveness of the method on other languages, can be approved by having tested the method on English texts.

Secondly, the proposed strategy is completely developed based on the definition of near duplication. It first applies some filtering rules. Thereafter, it focuses on the uncommon chunks of the remaining texts to evaluate the situation through some similarity calculation methods. Thus, it draws a meaningful systematic approach so that in application, one have options to better it by defining more precise filtering rules, or more efficient similarity calculation methods. This is an important feature when dealing with different definitions and/or different levels of near duplication that have been mentioned in previous studies. In contrast, SpotSig proposes an innovative pruning method with no logical connection to the different definitions or different levels of near duplication. It works well on the examined dataset, but no further improvement or extension can be easily envisaged for different situations.

Thirdly, this study evaluates both methods on the same dataset. However, comparative results might be different for different near duplication situations as explained above. The original manuscript strictly advises SpotSig for detecting web pages having “identical core content with different framing and banners.”

b) From efficiency point of view:

While the main steps of the proposed method are provided in Fig. 4, the SpotSig method consists of four sequential steps: (1) Spot signature extraction, (2) partitioning based on some boundary rules, (3) inverted indexing and (4) applying Jaccard similarity measure on the candidate Spot signatures. Among these, the third one makes this method more time-consuming against the proposed method. In the third step, by converting the corpus into auxiliary inverted indexes and storing a list of pointers to all document vectors that contain the Spot signatures, the inverted index is build in $O(NL)$ time where L is the average number of Spot signatures in a document. Thus, the proposed method would always outperform SpotSig in terms of efficiency.

6 Conclusion

This study proposes a language independent substring selection and filtering strategy for improving pairwise comparison-based NDD methods. The excellence of the proposed strategy owes to three aspects: the filtering strategy that cuts down the number of required comparisons, the selection strategy that is linear in the number of words in the document and the subset building that results to much smaller strings that are the reason for near duplication. The proposed strategy is based on intelligent pruning of the problem space by moving from a token-level to a chunk-level approach. The proposed strategy also avoids the biasing problem that is common in Shingling methods; a prior filtering is applied; and only the uncommon chunks of the suspicious pairs are investigated. Since the average length of the uncommon chunks for pairs labeled as suspicious is roughly the same (limited by an Upper-Bound Condition), the algorithm applied on the subsets will not face this biasing problem.

The proposed strategy can be applied on raw web page source codes directly without the need for any preprocessing steps. This eliminates the need for efficient content extraction techniques. The achievements are on the intuition regarding near duplication, which takes place in news articles. It is more effective than the previous selection strategies including the selection of the more influential token or the more frequent tokens. In addition, in contrast with some previous works, the proposed strategy is independent from document collection statistics and the availability of all the documents is not required at the time of NDD. This property makes it proper for performing NDD on the documents online as they come.

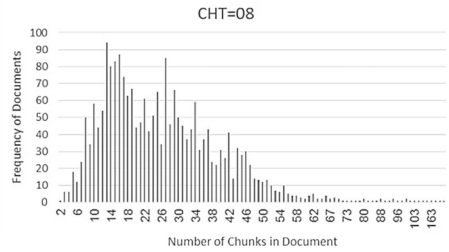
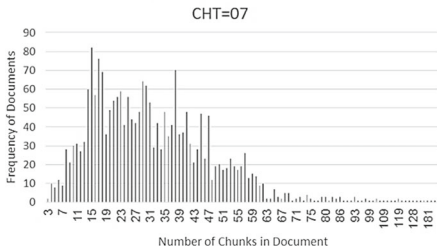
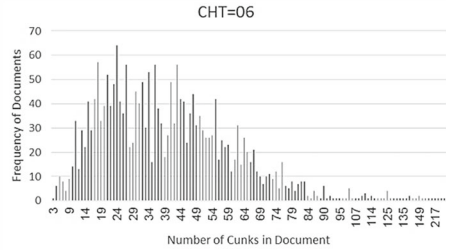
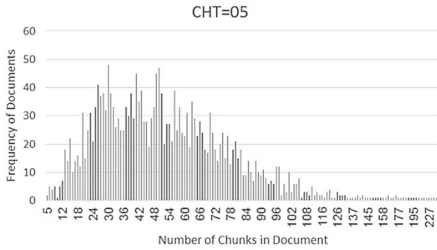
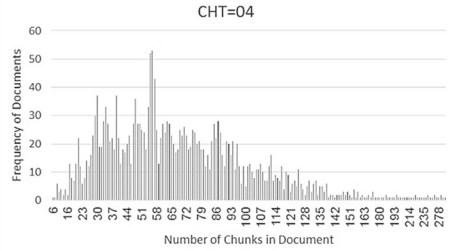
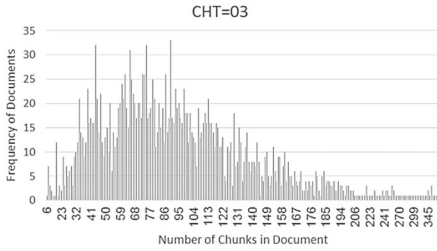
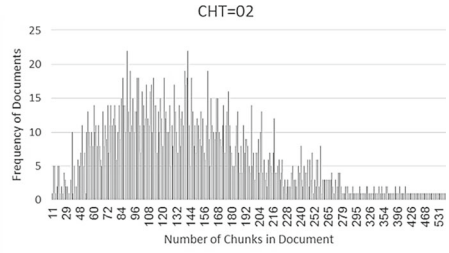
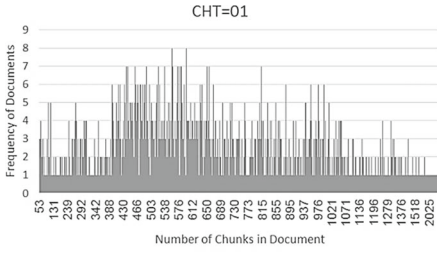
The aim of this paper was to present an improved pruning strategy for NDD tasks. This was explored by details for a vast parameter space. However, in the last step, the plain k -Shingling with $k = 1$ technique was applied on the generated subsets. Although the k -Shingling is an accurate and widely used technique, applying other similarity search methods reviewed in Sect. 2, such as MinHash, on the last step of the proposed strategy would probably result in more efficiency.

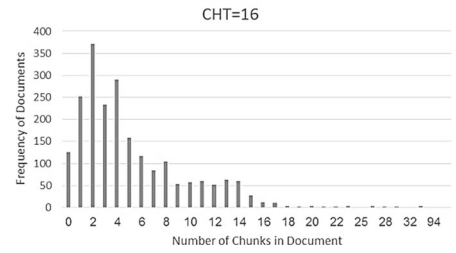
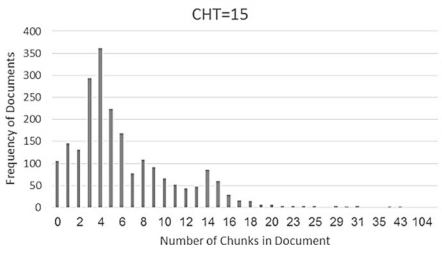
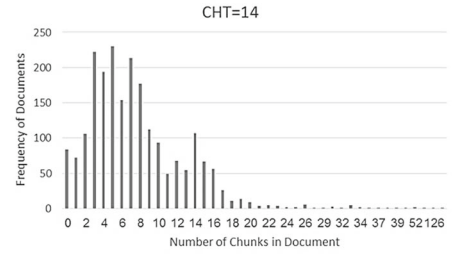
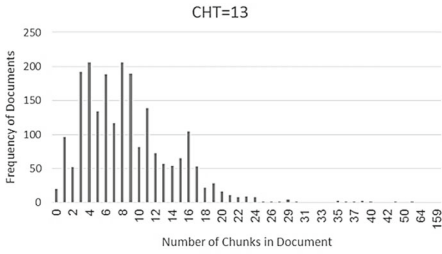
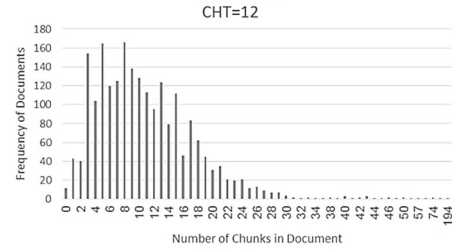
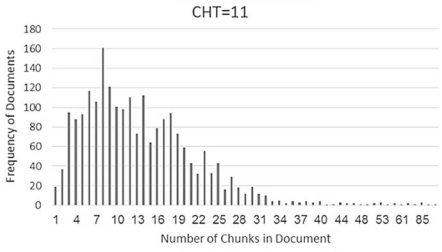
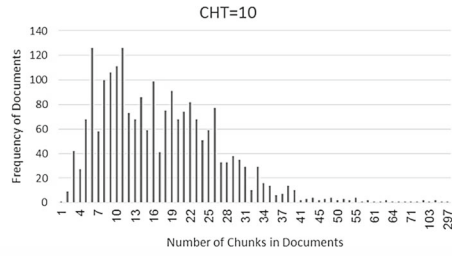
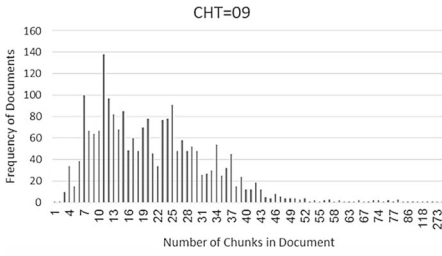
The selected parameters seem valid for other datasets since the characteristics of web news articles are statistically the same over various news websites and over a period of time. This issue is described using a concept called news writing style, which is explained in the journalism's body of knowledge. Inverted Pyramid Style of news writing is the most frequent writing style used for web news articles. It is described as "the child of technology, commerce and history" (Scanlan 2003). However, this claim should be evaluated through experiments. In addition, behavior of the proposed method against scaling issues and its behavior on much larger datasets should be investigated in further studies.

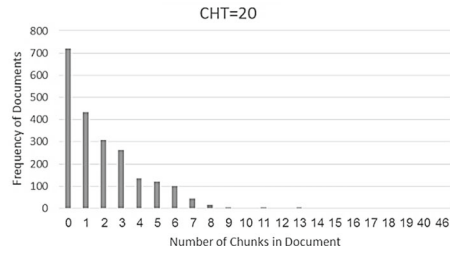
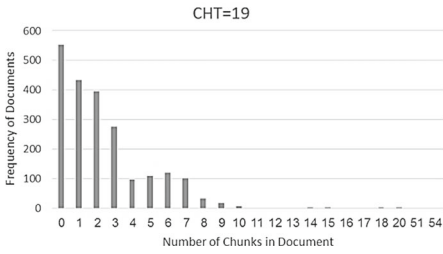
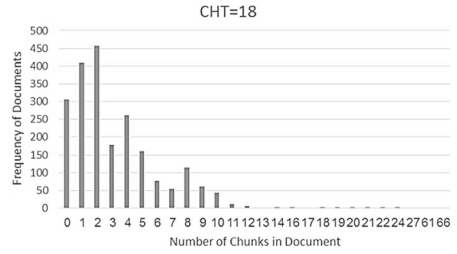
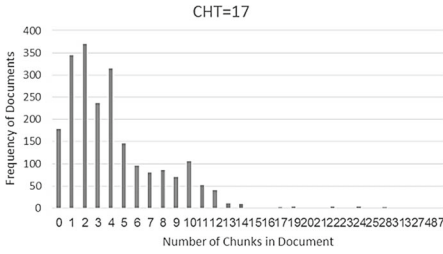
As Conrad et al. [15] propose, near duplication in news can be viewed differently. Examining the effectiveness of the proposed strategy in various types of near duplication is proposed as a suggestion for further study. Near duplication also takes place differently in different domains of news articles, such as sports and entertainment. Another suggestion for further study is to examine the proposed strategy scrupulously on each category: this may reveal important issues regarding the capabilities of the strategy in different situations. Testing the validity of the proposed strategy for special cases such as biomedical news articles [10] and articles containing tables or images will also be relevant.

Appendix 1

Distribution frequency of the number of chunks in the documents for different CHTs:







Appendix 2

See Table 5.

Table 5 Effectiveness evaluation by precision, recall and *F*-measure for different CHT and τ

	Precision								
	$T = 0.1$	$T = 0.2$	$T = 0.3$	$T = 0.4$	$T = 0.5$	$T = 0.6$	$T = 0.7$	$T = 0.8$	$T = 0.9$
CHT = 4	65.38	83.62	92.24	96.79	98.30	98.74	99.03	99.15	99.12
CHT = 5	69.60	92.75	96.37	97.82	98.78	99.32	99.27	99.20	99.22
CHT = 6	84.90	94.98	97.41	98.78	99.50	99.41	99.28	99.31	99.13
CHT = 7	85.64	95.25	97.61	98.81	99.44	99.47	99.45	99.30	99.12
CHT = 8	86.91	95.17	97.48	98.89	99.40	99.50	99.52	99.42	99.34
CHT = 9	89.11	95.75	97.63	98.85	99.37	99.53	99.58	99.62	99.61
CHT = 10	88.81	95.96	97.76	98.92	99.37	99.60	99.74	99.67	99.60
CHT = 11	92.24	96.20	98.05	98.92	99.46	99.62	99.72	99.72	99.64
CHT = 12	92.38	97.05	98.11	99.12	99.39	99.59	99.58	99.51	99.46
CHT = 13	92.68	97.18	97.92	98.94	99.04	98.97	98.96	98.76	98.71
CHT = 14	90.67	94.71	94.64	94.98	94.15	93.12	92.05	90.85	90.19
CHT = 15	88.94	92.09	92.00	91.81	90.35	88.51	87.50	86.68	86.13
CHT = 16	85.56	89.08	89.00	88.33	86.38	84.83	84.07	83.19	82.83
CHT = 17	76.83	79.12	78.54	76.87	74.22	71.88	70.78	69.87	69.65
CHT = 18	55.47	56.68	55.03	52.07	49.43	47.02	46.23	45.99	45.91
CHT = 19	26.75	27.65	26.61	23.88	22.84	22.08	21.84	21.80	21.77
CHT = 20	17.94	18.48	17.61	16.00	15.27	14.92	14.83	14.79	14.78

Table 5 continued

	Recall								
	<i>T</i> = 0.1	<i>T</i> = 0.2	<i>T</i> = 0.3	<i>T</i> = 0.4	<i>T</i> = 0.5	<i>T</i> = 0.6	<i>T</i> = 0.7	<i>T</i> = 0.8	<i>T</i> = 0.9
CHT = 4	93.51	75.33	58.11	41.05	24.81	14.03	8.47	5.62	3.94
CHT = 5	92.58	75.76	58.63	41.93	26.95	16.56	9.37	6.16	4.12
CHT = 6	88.41	75.28	60.47	44.44	30.25	19.64	11.52	7.31	4.79
CHT = 7	87.15	76.94	61.68	45.83	32.89	22.49	14.46	8.53	5.43
CHT = 8	87.54	79.14	63.75	48.61	35.81	25.61	17.65	10.97	6.69
CHT = 9	83.71	77.90	62.13	48.90	36.82	26.77	19.56	13.09	8.53
CHT = 10	80.35	75.13	60.82	48.62	36.81	28.29	20.56	14.71	9.31
CHT = 11	75.84	70.12	58.42	46.56	35.45	27.82	21.40	16.16	10.66
CHT = 12	71.83	67.80	57.67	46.12	35.97	28.70	23.05	17.10	13.50
CHT = 13	68.39	67.37	57.24	46.49	36.94	28.88	24.46	18.61	16.10
CHT = 14	68.39	68.11	58.05	49.11	37.96	30.96	26.01	21.93	20.24
CHT = 15	64.97	63.55	56.60	49.06	38.69	31.14	28.23	26.16	24.95
CHT = 16	60.24	60.47	54.62	48.68	40.28	35.16	33.11	31.02	30.24
CHT = 17	60.41	61.84	56.94	50.82	43.82	38.81	36.76	35.19	34.82
CHT = 18	61.55	62.34	57.67	51.02	45.71	41.50	40.21	39.81	39.69
CHT = 19	61.68	63.97	60.59	52.39	49.36	47.27	46.61	46.49	46.41
CHT = 20	63.84	65.86	62.04	55.25	52.26	50.82	50.47	50.32	50.29
	<i>F</i> -measure								
	<i>T</i> = 0.1	<i>T</i> = 0.2	<i>T</i> = 0.3	<i>T</i> = 0.4	<i>T</i> = 0.5	<i>T</i> = 0.6	<i>T</i> = 0.7	<i>T</i> = 0.8	<i>T</i> = 0.9
CHT = 4	76.95	79.26	71.30	57.65	39.62	24.57	15.60	10.64	7.58
CHT = 5	79.46	83.40	72.91	58.70	42.35	28.38	17.12	11.60	7.92
CHT = 6	86.62	83.99	74.62	61.30	46.40	32.80	20.64	13.61	9.14
CHT = 7	86.39	85.12	75.59	62.62	49.43	36.69	25.25	15.71	10.30
CHT = 8	87.22	86.42	77.09	65.18	52.65	40.74	29.98	19.76	12.54
CHT = 9	86.33	85.91	75.93	65.43	53.73	42.20	32.70	23.13	15.72
CHT = 10	84.36	84.28	74.99	65.19	53.72	44.06	34.09	25.64	17.03
CHT = 11	83.24	81.11	73.21	63.32	52.27	43.50	35.24	27.81	19.25
CHT = 12	80.82	79.83	72.64	62.95	52.83	44.57	37.44	29.19	23.77
CHT = 13	78.71	79.58	72.24	63.26	53.81	44.71	39.22	31.32	27.68
CHT = 14	77.97	79.24	71.96	64.74	54.11	46.47	40.56	35.34	33.06
CHT = 15	75.09	75.21	70.08	63.95	54.18	46.07	42.68	40.19	38.69
CHT = 16	70.70	72.04	67.70	62.77	54.94	49.72	47.51	45.19	44.30
CHT = 17	67.64	69.42	66.02	61.19	55.10	50.40	48.39	46.80	46.43
CHT = 18	58.35	59.37	56.32	51.54	47.50	44.09	43.01	42.68	42.58
CHT = 19	37.32	38.61	36.98	32.81	31.23	30.10	29.74	29.68	29.64
CHT = 20	28.01	28.86	27.43	24.81	23.64	23.07	22.93	22.86	22.85

The bold value (*F*-measure = 87.22%) is the best result achieved

Appendix 3

Setting the similarity threshold

Results of the experiments with the aim of finding the best value for the similarity threshold are provided below. In line with previous studies [32, 37], the results show that selecting $\tau = 0.6$ (between the two turning points of 0.5 and 0.7) would lead to an acceptable performance (Fig. 15).

k-Shingling on *m* random Shingles

Several experiments were conducted by having *k* from 1 to 3 and *m* from 200 to 500. The results are provided in Fig. 16. As shown by the results, the best *F*-measure is achieved by having (*k* = 1 and *m* = 200) or (*k* = 1 and *m* = 300) or (*k* = 1 and *m* = 400) or (*k* = 1 and *m* = 500). Selecting each of the mentioned settings would result in *F*-measure = 82.11%.

Fig. 15 Results of applying different similarity thresholds on 500 most frequent selection strategies

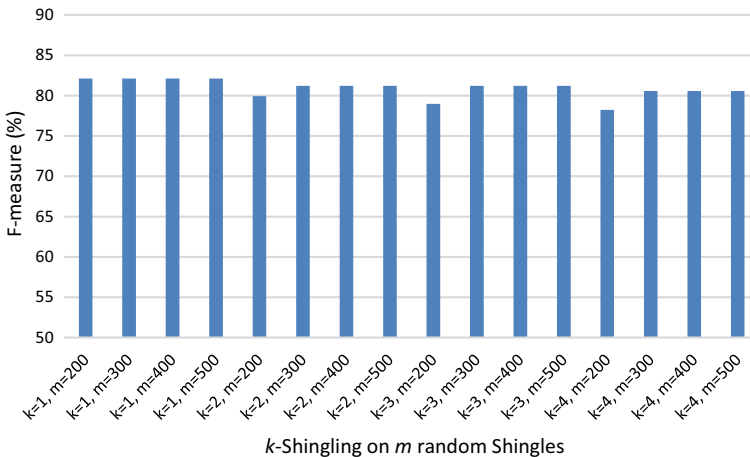
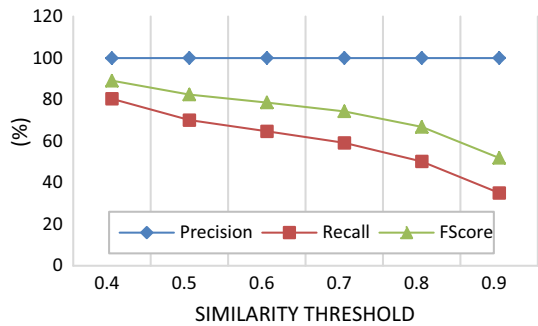


Fig. 16 *F*-measure of *k*-Shingling on *m* random Shingles in several settings of *k* and *m*

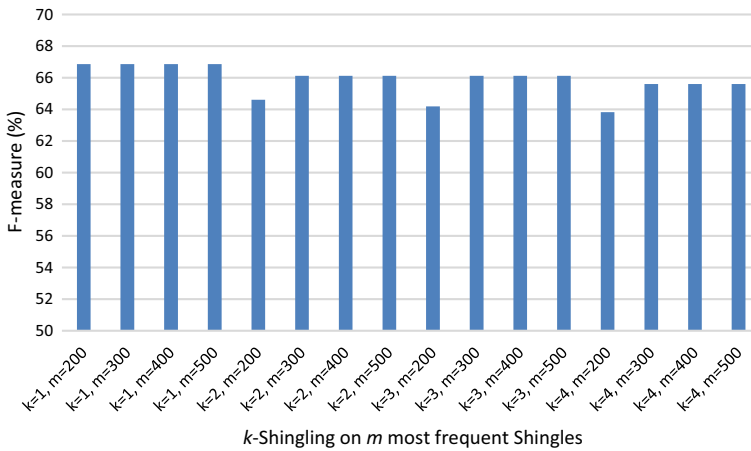


Fig. 17 *F*-measure of *k*-Shingling on *m* most frequent Shingles in several settings of *k* and *m*

k-Shingling on *m* most frequent Shingles

Several experiments were conducted by having *k* from 1 to 3 and *m* from 200 to 500. The results are provided in Fig. 17. As shown by the results, the best *F*-measure is achieved by having ($k = 1$ and $m = 200$) or ($k = 1$ and $m = 300$) or ($k = 1$ and $m = 400$) or ($k = 1$ and $m = 500$). Selecting each of the mentioned settings would result in *F*-measure = 66.86%.

References

1. Abdel Hamid O, Behzadi B, Christoph S, Henzinger M (2009) Detecting the origin of text segments efficiently. In Proceedings of the 18th international conference on World Wide Web
2. Alonso O, Fetterly D, Manasse M (2013) Duplicate news story detection revisited. In Asia information retrieval symposium. Springer, Berlin, pp 203–214
3. Bernstein Y, Shokouhi M, Zobel J (2006) Compact features for detection of near-duplicates in distributed retrieval. In International symposium on string processing and information retrieval. Springer, Berlin, pp 110–121
4. Bhimireddy M, Gandhi KP, Hicks R, Veeramachaneni BR (2015) A survey to fix the threshold and implementation for detecting duplicate web documents. All Capstone Projects, Paper 155
5. Bilenko M, Mooney RJ (2003) Adaptive duplicate detection using learnable string similarity measures. In: Proceedings of the ninth ACM SIGKDD international conference on knowledge discovery and data mining, pp. 39–48
6. Broder AZ (1997) On the resemblance and containment of documents. In: Proceedings of the international conference on compression and complexity of sequences. IEEE, pp 21–29
7. Broder AZ (2000) Identifying and filtering near-duplicate documents. In: Annual symposium on combinatorial pattern matching. Springer, Berlin, pp 1–10
8. Broder AZ, Glassman SC, Manasse MS, Zweig G (1997) Syntactic clustering of the web. *J Comput Netw ISDN Syst* 29(8):1157–1166
9. Charikar MS (2002) Similarity estimation techniques from rounding algorithms. In: Proceedings of the thirty-fourth annual ACM symposium on theory of computing. ACM, pp 380–388
10. Chen Q, Zobel J, Verspoor K (2017) Duplicates, redundancies and inconsistencies in the primary nucleotide databases: a descriptive study. *Database* 1:baw163. <https://doi.org/10.1093/database/baw163>
11. Chowdhury A, Frieder O, Grossman D, McCabe MC (2002) Collection statistics for fast duplicate document detection. *ACM Trans Inf Syst (TOIS)* 20(2):171–191
12. Clough PD (2003) Measuring text reuse. Department of Computer Science, University of Sheffield, Sheffield

13. Cohen E, Datar M, Fujiwara S, Gionis A, Indyk P, Motwani R et al (2001) Finding interesting associations without support pruning. *IEEE Trans Knowl Data Eng* 13(1):64–78
14. Cohen E, Kaplan H (2007) Bottom-k sketches: better and more efficient estimation of aggregates. In: *ACM SIGMETRICS performance evaluation*
15. Conrad JG, Guo XS, Schriber CP (2003) Online duplicate document detection: signature reliability in a dynamic retrieval environment. In *Proceedings of the twelfth international conference on Information and knowledge management*. ACM, pp 443–452
16. Cooper JW, Coden AR, Brown EW (2002) A novel method for detecting similar documents. In *HICSS. Proceedings of the 35th annual Hawaii international conference on system sciences, 2002*. IEEE, pp 1153–1159
17. Dobra A, Garofalakis M, Gehrke J, Rastogi R (2009) Multi-query optimization for sketch-based estimation. *Inf Syst* 34(2):209–230
18. Hajishirzi H, Yih W, Kolcz A (2010) Adaptive near-duplicate detection via similarity learning. In: *Proceedings of the 33rd international ACM SIGIR conference on research and development in information retrieval*, pp 419–426
19. Har-Peled S, Indyk P, Motwani R (2012) Approximate nearest neighbor: towards removing the curse of dimensionality. *Theory Comput* 8(1):321–350
20. Heintze N (1996) Scalable document fingerprinting. In: *1996 USENIX workshop on electronic commerce*, vol 3
21. Hoad TC, Zobel J (2003) Methods for identifying versioned and plagiarized documents. *J Am Soc Inf Sci Technol* 54(3):203–215
22. Jaccard P (1901) *Distribution de la Flore Alpine: dans le Bassin des dranses et dans quelques régions voisines*. Rouge
23. Jangwon SEO, Croft WB (2008) Local text reuse detection. In: *Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval*. ACM, pp 571–578. <http://dl.acm.org/citation.cfm?id=1390432>
24. Ji J, Li J, Yan S, Tian Q, Zhang B (2013) Min-max hash for Jaccard similarity. In: *The 13th international conference on data mining (ICDM)*. IEEE, pp 301–309
25. Kolcz A, Chowdhury A (2008) Lexicon randomization for near-duplicate detection with I-Match. *J Supercomput* 45(3):255–276
26. Kolcz A, Chowdhury A, Alspector J (2004) Improved robustness of signature-based near-replica detection via lexicon randomization. In: *Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, pp 605–610
27. Leskovec J, Backstrom L, Kleinberg J (2009) Meme-tracking and the dynamics of the news cycle. In: *15th ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, pp 497–506
28. Li P, König C (2010) b-Bit minwise hashing. In: *The 19th international conference on World Wide Web (WWW'10)*. ACM Press, New York, p 671
29. Li P, Owen A, Zhang C-H (2012) One permutation hashing. In: Pereira F, Burges CJC, Bottou L, Weinberger KQ (eds) *Advances in neural information processing systems (Proceeding of the neural information processing systems conference)*, pp 3113–3121
30. Lo GS, Dembele S (2015) Probabilistic, statistical and algorithmic aspects of the similarity of texts and application to Gospels comparison. *arXiv preprint arXiv:1508.03772*
31. Mitzenmacher M, Pagh R, Pham N (2014) Efficient estimation for high similarities using odd sketches. In: *Proceedings of the 23rd international World Wide Web Conference Committee (IW3C2)*
32. Montanari D, Puglisi PL (2012) Near duplicate document detection for large information flows. In: *International conference on availability*, p 16. http://link.springer.com/chapter/10.1007/978-3-642-32498-7_16
33. Pamulaparty L, Rao CVG, Rao MS (2014) A near-duplicate detection algorithm to facilitate document clustering. *Int J Data Min Knowl Manag Process* 4(6):39
34. Sarawagi S, Kirpal A (2004) Efficient set joins on similarity predicates. In: *Proceedings of the 2004 ACM SIGMOD international conference on management of data*. ACM, pp 743–754
35. Schleimer S, Wilkerson DS, Aiken A (2003). Winnowing: local algorithms for document fingerprinting. In: *Proceedings of the 2003 ACM SIGMOD international conference on management of data*. ACM, pp 76–85
36. Sun Y, Qin J, Wang W (2013) Near duplicate text detection using frequency-biased signatures. *WISE* 1:277–291
37. Theobald M, Siddharth J, Paepcke A (2008) Spotsigs: robust and efficient near duplicate detection in large web collections. In: *Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval*. ACM, pp 563–570

38. Van Bezu R, Borst S, Rijkse R, Verhagen J (2015) Multi-component similarity method for web product duplicate detection. In: Proceedings of the 30th annual ACM symposium on applied computing
39. Vaughan L (2014) Discovering business information from search engine query data. *Int J Online Inf Rev* 38(4):562–574
40. Wang J, Chang H (2014) Exploiting near-duplicate relations in organizing news archives. *Int J Intell Syst* 29(7):597–614
41. Wang Y, Zeng D, Zheng X, Wang F (2009) Propagation of online news: dynamic patterns. In: IEEE international conference on intelligence and security informatics, ISI'09. IEEE, pp 257–259
42. Xiao C, Wang W, Lin X, Yu JX, Wang G (2011) Efficient similarity joins for near-duplicate detection. *ACM Trans Database Syst (TODS)* 36(3):15
43. Zhang W, Ji J, Zhu J, Li J, Xu H, Zhang B (2016) BitHash: an efficient bitwise Locality Sensitive Hashing method with applications. *Int J Knowl Based Syst* 97:40–47

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Roya Hassanian-esfahani received her BS in Electronic Engineering and MSc in Information Technology Engineering from Shiraz University, Iran. She received her PhD in Information Technology Engineering from Research Institute for Information and Communication Technology, Academic Center for Education, Culture and Research, Tehran, Iran. Her dominant research interests include web mining and web information retrieval.



Mohammad-javad Kargar is an Assistant Professor at the Department of Computer Engineering in University of Science and Culture, Tehran, Iran. He is the founder of Web Research center, International Conference on Web Research and International Journal of Web Research in Iran. His research interest is web research and mining, data science and distributed systems.