Applied Computer Systems
ISSN 2255-8691 (online)
ISSN 2255-8683 (print)
May 2020, vol. 25, no. 1, pp. 11–18
https://doi.org/10.2478/acss-2020-0002
https://content.sciendo.com

Evaluation of Fingerprint Selection Algorithms for Local Text Reuse Detection

Gints Jēkabsons*
Riga Technical University, Riga, Latvia

***Abstract* – Detection of local text reuse is central to a variety of applications, including plagiarism detection, origin detection, and information flow analysis. This paper evaluates and compares effectiveness of fingerprint selection algorithms for the source retrieval stage of local text reuse detection. In total, six algorithms are compared – *Every p-th*, *0 mod p*, *Winnowing*, *Hailstorm*, *Frequency-biased Winnowing* (*FBW*), as well as the proposed modified version of *FBW* (*MFBW*).**

Most of the previously published studies in local text reuse detection are based on datasets having either artificially generated, long-sized, or unobfuscated text reuse. In this study, to evaluate performance of the algorithms, a new dataset has been built containing real text reuse cases from Bachelor and Master Theses (written in English in the field of computer science) where about half of the cases involve less than 1 % of document text while about two-thirds of the cases involve paraphrasing.

In the performed experiments, the overall best detection quality is reached by *Winnowing*, *0 mod p*, and *MFBW*. The proposed *MFBW* algorithm is a considerable improvement over *FBW* and becomes one of the best performing algorithms.

The software developed for this study is freely available at the author's website http://www.cs.rtu.lv/jekabsons/.

***Keywords* – Document fingerprinting, fingerprint selection, local text reuse detection, plagiarism detection.**

I. Introduction

Local text reuse occurs when parts of a document, such as a paragraph or a sentence, are reused in another document. The reused text may also be modified by inserting, removing, replacing, or rearranging words or sentences, as well as interleaving text from one source with a text from another source. Detection of such reuse is central to a variety of applications, including plagiarism detection, origin detection, and information flow analysis.

Generally, the process of local text reuse detection can be divided into two main stages: source retrieval and text alignment [1]. At the source retrieval stage, given a query document and a large collection of stored documents, the task is to retrieve all potential sources from which the query document might have "borrowed" the text. At the text alignment stage, the task is to perform a more detailed analysis to identify contiguous passages of reused text between the query document and the retrieved sources.

One of the established approaches for source retrieval is document fingerprinting [2]–[13]. In this approach, fingerprints are extracted from a document through hashing sequences of consecutive words or characters. Two documents sharing one or more fingerprints indicates possible text reuse.

Fingerprinting methods can be broadly categorized into two groups: overlap and non-overlap methods, depending on whether the hashed sequences are overlapping or non-overlapping [5]. The present paper focuses on the first category.

Storing and handling large sets of fingerprints can often be impractical; thus, various algorithms for fingerprint selection have been proposed [3], [4], [7], [8], [10], [12], [13]. Some subsets of these fingerprint selection algorithms have been empirically evaluated and compared on text reuse detection in news stories [5], [14], detection of (artificially inserted) plagiarism in books [3], and origin detection in blogs and other websites [4]. However, there is still insufficient research done in investigating and comparing performance of these algorithms on real text reuse/plagiarism involving possibly obfuscated local reuse.

In this study, the effectiveness of six fingerprint selection algorithms is evaluated using a dataset that contains real text reuse cases from Bachelor and Master Theses written in English in the field of computer science. The reused text passages are of variable length and include paraphrasing. Simple modification to one of the existing algorithms is also proposed and it is shown that the modification considerably improves its performance. In fact, it becomes one of the best performing algorithms for the dataset.

The rest of the paper is organized as follows: Section II briefly describes the fingerprint extraction process, overviews the compared fingerprint selection algorithms, and proposes modification to one of them. Section III describes experimental setup for comparing the algorithms and presents the experimental results. Finally, Section IV concludes the paper.

II. Fingerprint Extraction and Selection

To extract a sequence of fingerprints from a document's text, first, some or all of the following pre-processing steps are performed: special characters such as punctuation marks are removed, the text is tokenized into separate words, the words

* Corresponding author's e-mail: gints.jekabsons@rtu.lv

are then case-folded and stop words are removed. Next, the words can be stemmed or lemmatized to reduce inflectional and derivationally related forms of a word to a common base form – either stem or lemma.

Result of the pre-processing is a clean normalized sequence of words which is then divided into overlapping sequences of n consecutive words, called n-grams or shingles.

Finally, each n-gram is converted into an integer using some hashing algorithm. Popular choices are MD5 [15] and Rabin [16]; however, it is not very important which particular hashing algorithm is used as long as the algorithm displays some basic properties – it needs to be reproducible, have a distribution close to uniform, and be fast [7].

The total number of n-grams for a document with l words is $m = l – n + 1$. The simplest fingerprinting strategy is to take all n-grams (i.e., technically – their computed hashes) as document's fingerprints. While this "full fingerprinting" strategy is expected to be the best case for finding as many correct text reuse sources as possible, it also requires the largest amount of storage space and the largest amount of fingerprint comparisons during the search. Thus, the strategy is too expensive for use with large collections even if inverted index is used.

Remainder of this section overviews fingerprint selection algorithms that try to take only a representative subset of the whole set of n-grams to strike some trade-off between quality of results, storage requirements, and computational cost.

A. Every p-th

Every p-th algorithm selects every p-th n-gram of a document for some fixed parameter p, selecting m / p n-grams in total.

Unfortunately, *Every p-th* lacks the desirable position independence property [8]: it is very sensitive to changes in n-gram order, insertions, and deletions as shifting an n-gram from position that is divisible by p in any direction by a number that is not divisible by p would result in this n-gram not being selected. *Every p-th* is the only algorithm in this study that does not have the position independence property.

B. 0 mod p

0 mod p algorithm [13] selects n-grams whose hashes are divisible by some fixed parameter p. On average, it selects m / p n-grams.

0 mod p fulfils the so-called context-freeness property [4]: whether an n-gram is selected depends entirely on the n-gram itself (i.e., on the words it is composed of) and not on any other n-gram in the document. This means that any given n-gram is either always selected in all documents containing it or is never selected.

A potential disadvantage of this algorithm is that the maximum gap between two selected n-grams in a document is unbounded, i.e., there can be any number of n-grams in a row that are not divisible by p and therefore not selected. This can create long gaps of unselected n-grams where any matches between documents cannot be detected [8].

C. Winnowing

In *Winnowing* algorithm [8], a window of size w slides over the sequence of document's n-grams and in each step from the w consecutive n-grams it selects the one with the lowest hash value. If the window contains more than one n-gram with the lowest hash value, the rightmost n-gram is chosen. Thus, *Winnowing* guarantees that in each sequence of w n-grams at least one n-gram is selected and if two documents share a sequence of n-grams at least as long as w, it is guaranteed that at least one of those n-grams will be selected as fingerprint for both documents.

Winnowing fulfils the so-called locality property [8]: n-gram selection is completely independent of window's position in the document and any other n-grams outside the window. However, selection depends on other n-grams in the same window. Therefore, the algorithm does not fulfil the context-freeness property.

The lower bound for the number of n-grams selected by *Winnowing* is $2m / (w + 1)$.

D. Hailstorm

Hailstorm algorithm [4] first hashes every word of the document and then selects n-grams for which the rightmost or the leftmost word has the lowest hash value of all word hashes in that n-gram. Notice that hashes of n-grams themselves are not used in making decisions about their selection.

If all words in an n-gram are different, the probability for the n-gram to be selected is $2 / n$ [4]; therefore, the lower bound for the number of n-grams selected by *Hailstorm* is $2m / n$.

Hailstorm, like *0 mod p*, fulfils the context-freeness property but additionally it also guarantees total coverage, i.e., every word in the document exists in at least one of the selected n-grams [4].

A potential disadvantage of this algorithm is that the number of selected n-grams and n-gram size is controlled by the same parameter. This takes away the freedom to choose text reuse detection sensitivity independently from its storage consumption. Discussion of this disadvantage continues in Section III.E.

E. Frequency-Biased Winnowing (FBW)

The working principles for *FBW* algorithm [3] are the same as for *Winnowing* with the exception that, instead of using hashes for n-gram selection, *FBW* uses n-gram collection frequencies, i.e., number of occurrences of each n-gram in the collection documents. If two n-grams have the same frequency, the tie is broken using alphabetical order of the n-grams.

The motivation for using frequencies is to select more representative n-grams that occur in fewer collection documents so that the amount of matches with false sources is reduced, allowing working with smaller n-grams for increased resistance against small changes in text. Additionally, working with rarer n-grams may also reduce search time [3].

F. Modified Frequency-Biased Winnowing (MFBW)

There is a potentially undesirable behaviour of *FBW* algorithm in how it selects n-grams for query documents.

While, for collection documents, the lowest possible frequency for an *n*-gram equals one, for query documents the lowest possible frequency is zero because a non-collection document may contain *n*-grams that do not exist in the collection. Since *FBW* always selects *n*-gram with the lowest frequency, for any window in a query document that has at least one zero-frequency *n*-gram, *FBW* is guaranteed to not find any matches in the collection documents. The risk for a window to have such an *n*-gram becomes higher for larger *w* because longer windows contain more *n*-grams, as well as for larger *n* because larger *n*-grams have a higher probability to not exist in the collection. To try to compensate for this behaviour, *FBW* may have to be used with smaller windows and/or smaller *n*-grams than otherwise necessary. The former would result in more *n*-grams selected, increasing storage consumption and search time, while the latter would increase the number of false matches.

To address this problem, a simple modification to *FBW* is proposed: in a window any zero-frequencies are treated as infinity, i.e., the least preferable frequency for *n*-gram selection. As a result, the most preferable frequency equals one and selecting *n*-grams that do not exist in the collection has the lowest priority. If all *n*-grams in the window have zero frequencies, selection is done using the alphabet as before.

III. Evaluation

A. Description of the Dataset

To evaluate the fingerprint selection algorithms on real text reuse cases, a new dataset was created containing 35 query documents and a source document collection containing 1021 documents, 348 of which are used sources (i.e., their text is reused in query documents) and 673 are unused sources. The query documents are Bachelor and Master Theses submitted for defence at the Institute of Applied Computer Systems, Riga Technical University in the period of 2016–2019. The theses and the sources are written in English on a variety of computer science topics. On average, each thesis contains reused text from about 10 of the sources. For text reuse detection, a total of 35 × 1021 = 35 735 document pairs are to be compared. Table I gives an overview of the dataset.

All sources were manually checked to confirm text reuse. The threshold for registering a text reuse was either one directly copied average-sized sentence or two to three paraphrased average-sized sentences. Fig. 1 shows distribution of text reuse lengths across all 348 reuse cases. As can be seen, in about half of the cases text reuse constitutes less than 1 % of the query document text.

All unused sources were manually selected to be on the same or very similar topics as the query documents.

It should be noted that in order to remove the possibility for finding a source just by matching references given in query documents to titles or headers/footers of sources, reference lists from all query documents were deleted.

TABLE I
Overview of the Dataset

Overall statistics	
Query documents	35
Collection documents	1021
Collection documents used as sources	348
Unused collection documents	673
Total document pairs to be compared	35 735
Text reuse type	
Copy-paste	97 (28 %)
Paraphrased	251 (72 %)
Collection document type	
Scientific papers, technical reports, white papers	626 (61 %)
Websites	272 (27 %)
Theses	52 (5 %)
Entire conference proceedings, books	38 (4 %)
Presentation slides	33 (3 %)
Length of query documents (in words)	
Mean	14 640
Median	14 220
Standard deviation	4708
Length of collection documents (in words)	
Mean	12 684
Median	5098
Standard deviation	28 750

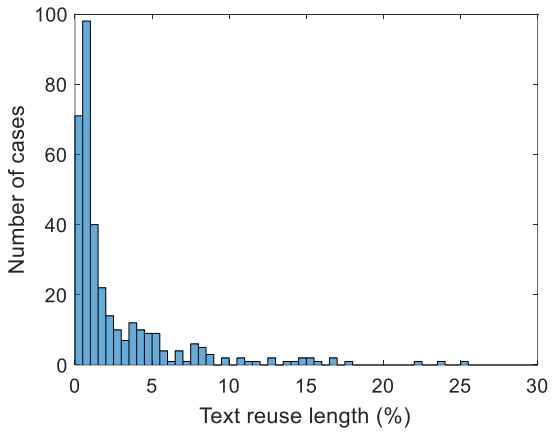

Fig. 1. Distribution of text reuse lengths in the created dataset (in percent of query document length). A single 78 % case is not shown.

B. Parameters

To use fingerprinting for source retrieval, it is necessary to extract fingerprints from each document. As discussed in Section II, it involves text pre-processing, splitting the text into *n*-grams, hashing, and selecting a subset of hashes as fingerprints. Pre-processing is discussed in Section III D. For hashing, the 32-bit FNV-1a variant of the Fowler-Noll-Vo hash function [17] was used.

In total, there are three parameters to be set: First parameter is the *n*-gram size *n*. Smaller *n* generally increases the number

of false matches among documents, while larger *n* makes fingerprinting more sensitive to changes in text since exact matches of less than *n* words in a row cannot be detected. The second parameter is the parameter of a fingerprint selection algorithm controlling the number of selected fingerprints as discussed in Section II. Recall that *Hailstorm* does not have its own parameter – instead the number of selected fingerprints is controlled by *n*. The third parameter is threshold *t* defined as the minimum number of distinct fingerprints matching between two documents to consider one of them as a source for text reuse of the other. In a two-stage text reuse detection system, this parameter would control the final decision whether a potential source would be sent to the alignment stage.

All three parameters were optimized for best retrieval quality vs. the number of selected fingerprints.

C. Evaluation Measures

To evaluate the algorithms and parameter choices, the well-known *precision*, *recall*, and *F-score* measures were used:

$$precision = \frac{|true\ positives|}{|true\ positives| + |false\ positives|}\ , \qquad (1)$$

$$recall = \frac{|true\ positives|}{|true\ positives| + |false\ negatives|}\ , \qquad (2)$$

$$F_\beta = \left(1+\beta^2\right)\frac{precision \cdot recall}{\left(\beta^2 \cdot precision\right) + recall}\ , \qquad (3)$$

where β determines the balance between precision and recall, i.e., it specifies a ratio of how much one is willing to tolerate decrease in precision to increase in recall. At the source retrieval stage of a two-stage local text reuse detection system, high recall is more important than high precision as algorithms of the alignment stage can discard most of the received false positives (at the expense of some computational cost), but the problem of missing sources is unrecoverable. Therefore, in the present study β = 10 is used (meaning recall is weighted 10 times higher than precision).

D. Pre-Processing Experiments

In this study, all documents were first pre-processed by case-folding and tokenization. Simple tokenization strategy was used, where text was split by non-alphanumeric characters.

To decide on other pre-processing steps, a number of experiments were run with the full fingerprinting strategy. There were three steps to decide on: (a) Whether to remove common stop words; (b) Whether to remove all words shorter than a certain threshold and what this threshold should be; (c) Whether to do stemming (the well-known Porter stemmer was used).

Table II presents the results of the experiments in terms of *n* and *t* (optimized for maximum F_{10}), corresponding precision, recall, and F_{10} values as well as the average number of selected fingerprints *m* for a document in the collection. We experimented with word length thresholds in the range from 2 to 5 and achieved the best results with 3. Thus, we only report results for this value. Such a threshold removes any small numbers, short variable names in mathematical expressions or software source code, "Y"/"N"-type entries in tables etc.

TABLE II

REUSE DETECTION PERFORMANCE FOR EACH PRE-PROCESSING TYPE AND THEIR COMBINATIONS, WITH OPTIMIZED *n* AND *t*

Pre-processing steps	*n*	*t*	Prec. (%)	Rec. (%)	F_{10} (%)	Mean *m*
Initial result	6	2	14.67	89.94	85.60	12 679
a) stop word removal	4	2	9.95	97.99	90.09	8238
b) word length ≥ 3	4	6	13.80	95.11	89.87	9586
c) stemming	6	2	14.18	91.38	86.71	12 679
a + b	4	2	19.04	97.41	93.60	7212
a + b + c	4	3	25.49	97.13	94.50	7072

The results show that removal of stop words and short words reduced the noise level enough to allow using smaller *n* to reach higher recall (*n* dropped from 6 to 4) without lowering precision too much. The effect of stemming is smaller but still overall positive.

The best result is obtained when all pre-processing steps are used together (the row denoted as "a + b + c"). It has the best F_{10} while the average number of fingerprints has reduced by almost half. Thus, in all following experiments the full set of pre-processing steps is used and the evaluation F_{10} = 94.50 % can be considered the best-case result.

E. Algorithm Evaluation Results: F_{10} vs. Number of the Selected Fingerprints

Since the goal is to reduce the number of selected fingerprints while keeping F_{10} at the best possible level, F_{10} is first viewed as a function of the number of fingerprints as shown in Fig. 2. The same results at four different fingerprint selection sizes are shown in Table III. The results are obtained by varying the parameter of each fingerprint selection algorithm (or *n* for *Hailstorm*) and optimizing *n* and *t* (or just *t* for *Hailstorm*) for maximum F_{10}. For example, to select about 50 % of fingerprints, according to the formulas given in Section II, *Every p-th* and *0 mod p* algorithms will have *p* = 2, *Winnowing*, *FBW*, and *MFBW* will have *w* = 3, and *Hailstorm* will have *n* = 4.

Overall, with the exception of *Hailstorm*, it can be seen that in order to maximize F_{10} the reduction in the number of fingerprints is compensated by using a combination of smaller *n*-gram size *n* and threshold *t*. Smaller *n* increases probability of having matching *n*-grams between query documents and their correct sources, while smaller *t* reduces the requirement of how many *n*-grams should match. Lowering those parameters also introduces more false positives as it is evident by smaller precision values. The sharp drop in F_{10} when less than about 5 % of fingerprints are selected is where high recall cannot be easily sustained anymore, *n* is progressively reduced, which in turn decreases precision, and performance of all algorithms collapses.

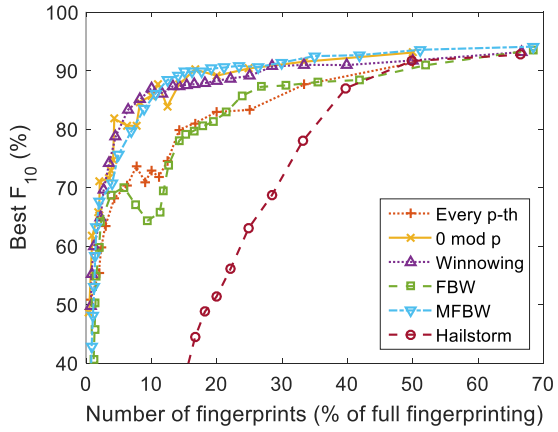

Fig. 2. Best F_{10} value vs. number of selected fingerprints. Full fingerprinting result is not shown – it is located at 100 % fingerprints and F_{10} = 94.50 % as given in Table II.

The best results are achieved by *MFBW*, *Winnowing*, and *0 mod p*. The unbounded gap size problem of *0 mod p* (as described in Section II B) does not prevent it from giving one of the best results. Evidently, for such text reuse cases as in the dataset under consideration, gaps of *n*-grams with hashes indivisible by *p* are relatively short and rare, even when *p* is set to 20 (which reduces the number of selected fingerprints to 5 %). Specifically, it was found that in the dataset the longest gap for $n = 3$ and $p = 20$ was 244 *n*-grams. For comparison purposes, this is about two orders of magnitude shorter than the longest gap found by Schleimer *et al.* [8] in a collection of websites when they used *0 mod p* with character 50-grams and $p = 50$.

The next group of algorithms with mutually similar performance is *FBW* and *Every p-th*. As Table III shows, to maximize F_{10} while reducing fingerprint size, for each reduction in the number of fingerprints *FBW* has to lower *n* earlier than any other algorithm. This keeps high recall but loses precision as more false positives are introduced. *Every p-th* (as well as *MFBW*) has a similar tendency but not as extreme.

Table IV shows average percentage of query document trigrams selected as fingerprints that were also selected for any of the collection documents (i.e., the collection frequency of the fingerprints is larger than zero). Table V shows average percentage of query document trigrams selected as fingerprints that were also selected for correct sources but for none of the other collection documents (these fingerprints perfectly identify correct sources but not necessarily all of them). Smaller percentages indicate lower consistency in selecting the same fingerprints in query and source documents.

There is a noticeable relationship between these percentages (across algorithms and across fingerprint sizes) and what values are chosen for *n* and *t* as given in Table III, i.e., lower consistency usually means smaller parameter values chosen to compensate for it.

TABLE III

BEST PERFORMANCE OF THE ALGORITHMS AT FOUR DIFFERENT FINGERPRINT SELECTION SIZES (RESULTS FOR FULL FINGERPRINTING ARE GIVEN IN TABLE II)

Algorithm	***n***	***t***	**Prec. (%)**	**Rec. (%)**	**F_{10} (%)**
Number of fingerprints: ~50 % of full fingerprinting					
Every p-th	3	4	15.63	95.98	91.33
0 mod p	3	6	18.09	97.13	93.10
Winnowing	4	2	29.91	93.68	91.74
FBW	3	2	33.20	92.53	90.92
MFBW	4	1	24.65	96.26	93.57
Hailstorm	4	2	22.55	94.54	91.64
Number of fingerprints: ~20 % of full fingerprinting					
Every p-th	3	1	7.31	92.53	82.95
0 mod p	3	3	17.95	92.82	89.14
Winnowing	3	2	13.62	93.39	88.27
FBW	2	1	6.63	91.67	81.34
MFBW	3	1	15.79	94.83	90.35
Hailstorm	10	1	87.25	51.15	51.36
Number of fingerprints: ~10 % of full fingerprinting					
Every p-th	3	1	14.07	76.15	72.96
0 mod p	3	2	19.29	88.79	85.73
Winnowing	3	1	8.63	95.69	87.00
FBW	1	9	2.27	87.36	63.73
MFBW	3	1	24.12	88.22	85.96
Hailstorm	20	1	97.59	23.28	23.45
Number of fingerprints: ~5 % of full fingerprinting					
Every p-th	2	1	3.00	87.36	68.33
0 mod p	3	1	10.61	87.93	82.01
Winnowing	3	1	12.25	85.34	80.58
FBW	1	3	3.71	85.92	70.47
MFBW	2	1	10.71	80.46	75.59
Hailstorm	43	1	92.86	7.47	7.54

As can be seen, both *Every p-th* and *FBW* have the smallest percentages that gradually get smaller as the number of selected fingerprints decreases. For *Every p-th*, this behaviour can be explained by its positional dependency (see Section II A). It has low consistency in selecting the same *n*-grams in different documents because selection depends on their positions and, naturally, the smaller the number of selected *n*-grams the more inconsistent the selection.

Reasons behind inconsistency of *FBW* are different. In contrast to *Every p-th*, it has position independence property but, as discussed in Section II E, it has preference in selecting zero-frequency *n*-grams from query documents. In *MFBW* on the other hand, by discouraging the selection of such *n*-grams, the percentages become much larger and much more stable across different fingerprint sizes as well as much nearer to those of *Winnowing* and *0 mod p*.

TABLE IV

PERCENTAGES OF TRIGRAMS SELECTED IN QUERY DOCUMENTS ALSO SELECTED IN ANY COLLECTION DOCUMENTS. REPORTED AT FOUR DIFFERENT FINGERPRINT SELECTION SIZES. THE RESULT FOR FULL FINGERPRINTING IS 24.03.

Algorithm	**~50 % selected**	**~20 % selected**	**~10 % selected**	**~5 % selected**
Every p-th	16.81	10.39	7.24	5.06
0 mod p	24.05	24.35	24.80	24.92
Winnowing	21.78	20.07	19.60	19.24
FBW	12.54	6.77	4.73	2.89
MFBW	21.96	17.58	16.73	15.85

TABLE V

PERCENTAGES OF TRIGRAMS SELECTED IN QUERY DOCUMENTS SELECTED ONLY IN CORRECT SOURCES. REPORTED AT FOUR DIFFERENT FINGERPRINT SELECTION SIZES. THE RESULT FOR FULL FINGERPRINTING IS 14.19.

Algorithm	**~50 % selected**	**~20 % selected**	**~10 % selected**	**~5 % selected**
Every p-th	9.64	5.55	3.92	2.50
0 mod p	14.16	14.40	14.34	14.35
Winnowing	13.38	12.60	12.22	11.96
FBW	10.90	6.71	4.71	2.89
MFBW	16.24	14.32	13.77	13.00

The most stable percentages in Tables IV and V are those for *0 mod p* and *Winnowing*. This is in line with Table III showing that even when only 5 % of fingerprints are selected both algorithms still use trigrams. *0 mod p* is the most consistent algorithm – the proportion of selected useful and non-zero frequency *n*-grams stays approximately the same irrespective of the total number of selected *n*-grams. This can be explained by the algorithm's context-freeness property (see Section II B) – *n*-grams do not "compete" for selection in a local window where selection inconsistencies may arise if even just one word is modified.

Finally, some remarks about *Hailstorm*. Recall that this algorithm has a fundamental difference from the other algorithms: the number of selected *n*-grams and *n*-gram size cannot be controlled separately. Moreover, the algorithm is designed so that the number of selected fingerprints is inversely proportional to *n*, i.e., the algorithm extracts either smaller amount of larger *n*-grams or larger amount of smaller *n*-grams. This is the opposite of what is required and it makes a considerable disadvantage of the algorithm because, to maintain high recall, reduction in the number of selected fingerprints (which reduces recall) actually requires to be compensated by using smaller *n*-grams, especially if the reused text passages are very short and contain paraphrases (like the case with the dataset under consideration).

As a result, *Hailstorm* is competitive when the number of selected fingerprints is above 40–45 %, otherwise it has by far the worst F_{10} score.

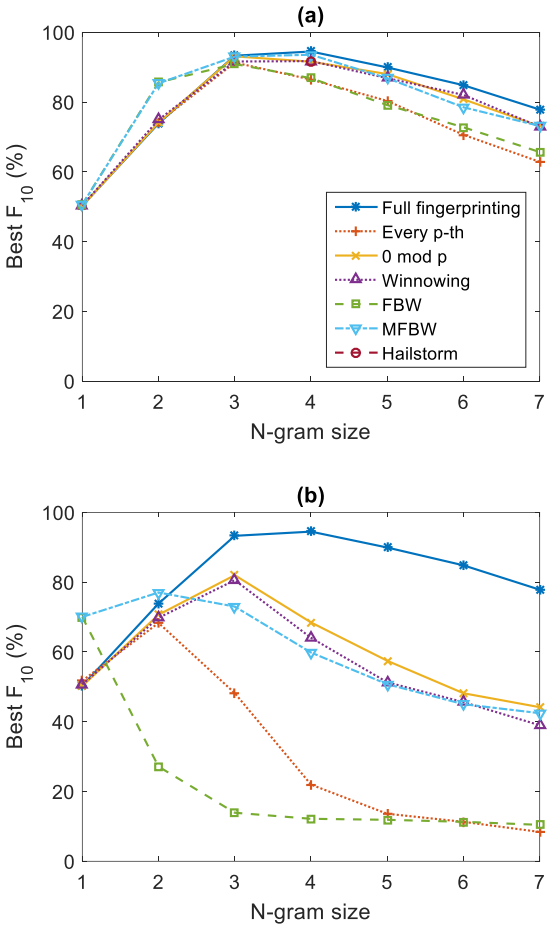

Fig. 3. Best F_{10} value vs. *n*-gram size when (a) 50 % and (b) 5 % of fingerprints are selected. Full fingerprinting is shown for reference only since it always selects 100 % of fingerprints.

F. Algorithm Evaluation Results: F_{10} vs. n-Gram Size

Fig. 3 shows F_{10} as a function of *n*-gram size when 50 % and 5 % of fingerprints are selected. In the two plots, one can see the performance for each *n*-gram size from which the best result for each algorithm was chosen for the corresponding fingerprint sizes in Table III. For instance, in Fig. 3b the best result for *Every p-th* is when $n = 2$, for *0 mod p* when $n = 3$ and so on.

Overall, from these results it can be concluded that if one has a reason to use unigrams or bigrams, then *MFBW* (or even *FBW*) might be the best choice. As discussed in Section II E, this bias towards smaller *n*-grams was the intention of the authors of *FBW*. However, for larger *n*-grams *Winnowing* and *0 mod p* give similar or even better results (and without the need to know the *n*-gram frequencies), especially for smaller fingerprint sizes where the number of distinct *n*-grams becomes large while their frequencies become 1 or 0 and *FBW*/*MFBW* loses its benefit from using frequency information.

G. Algorithm Evaluation Results: Precision vs. Recall

Fig. 4 shows precision-recall curves when 50 % and 5 % of fingerprints are selected. Overall, we can make similar conclusions as in Section III E but, for more balanced precision-

recall ranges than those relevant to F_{10}, the difference in performance between the algorithms is even greater.

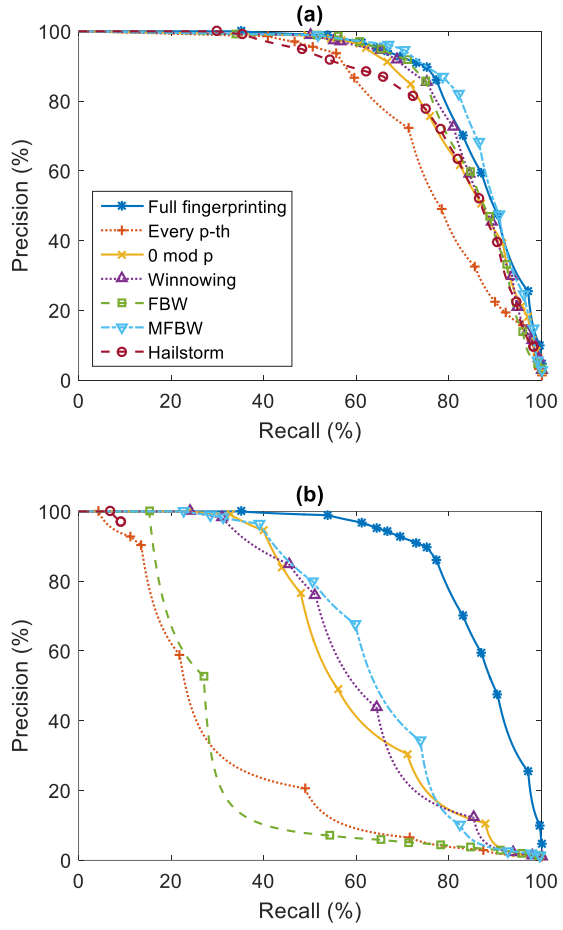

Fig. 4. Precision-recall curves when (a) 50 % and (b) 5 % of fingerprints are selected. Full fingerprinting is shown for reference only since it always selects 100 % of fingerprints.

Overall, *MFBW* is the best, followed by *Winnowing* and *0 mod p*. In Fig. 4a, we can see that in some ranges *MFBW* is even able to outperform full fingerprinting despite using only 50 % of fingerprints. In Fig. 4b, it can be observed how *FBW* and *Every p-th* have to sacrifice most of their precision to get any useful recall (which is achieved by reducing *n* and *t*, as already discussed in Section III E). Once again, it can be seen how at small fingerprint sizes *Hailstorm* cannot reach any useful recall at all.

IV. Conclusion

Six different fingerprint selection algorithms for local text reuse detection have been evaluated and compared. Overall, *Winnowing*, *0 mod p*, and *MFBW* algorithms gave the best results for all fingerprint sizes. With *Winnowing* and *0 mod p*, F_{10} smoothly reduces from about 94 %, when all fingerprints are used, to about 81–82 %, when only 5 % of the fingerprints are used. Further reduction in the number of retained fingerprints starts to degrade the reuse detection quality very fast.

The proposed *MFBW* algorithm is a considerable improvement over *FBW* and becomes one of the best performing algorithms for the dataset under consideration. Nevertheless, for smaller number of selected fingerprints, its selection strategy is not as effective as that of *Winnowing* and *0 mod p* – when fewer than about 5–10 % of fingerprints are selected, its detection quality degrades slightly faster.

One of the disadvantages of *FBW* and *MFBW* is the requirement to create *n*-gram frequency table before fingerprinting can be performed (i.e., it can be said that fingerprinting of the document collection is actually done twice – first, full fingerprinting is performed to compute frequencies, and then selective fingerprinting is carried out using those frequencies). For applications where this requirement is not a serious obstacle, *MFBW* may be a good choice, otherwise *Winnowing* and *0 mod p* are better choices. A possible further research direction might be to investigate more easily obtainable approximations or replacements for the frequencies, e.g., taking *n*-gram frequencies from a fixed existing representative text corpus.

The worst performing algorithm in the present study is *Hailstorm*. The results indicate that the lack of freedom to choose *n*-gram size independently from the number of selected fingerprints does not allow *Hailstorm* to be competitive with other algorithms in local text reuse detection. Originally, *Hailstorm* was developed for origin detection where relatively long unchanged text passages are copied between documents [4]. This is not the case for the dataset under consideration where the length of reused passages is generally much smaller and many passages are paraphrased.

Finally, because the text reuse detection experiments in this study involve source retrieval stage but do not involve text alignment stage, another worthwhile further research direction is experimentation with a system that includes both stages.

References

[1] M. Potthast, M. Hagen, A. Beyer, M. Busse, M. Tippmann, P. Rosso, and B. Stein, "Overview of the 6th International Competition on Plagiarism Detection," in *CEUR Workshop Proceedings*, vol. 1180, 2014, pp. 845–876.

[2] D. T. Citron and P. Ginsparg, "Patterns of text reuse in a scientific corpus," in *Proceedings of the National Academy of Sciences*, Jan 2015, 112, no. 1, pp. 25–30. https://doi.org/10.1073/pnas.1415135111

[3] Y. Sun, J. Qin, and W. Wang, "Near Duplicate Text Detection Using Frequency-Biased Signatures," in *Web Information Systems Engineering (WISE 2013), Lecture Notes in Computer Science*, vol. 8180. Springer, Berlin, Heidelberg, 2013, pp. 277–291. https://doi.org/10.1007/978-3-642-41230-1_24

[4] O. Abdel-Hamid, B. Behzadi, S. Christoph, and M. Henzinger, "Detecting the origin of text segments efficiently," in *WWW'09: Proceedings of the 18th international conference on World wide web*, ACM, New York, NY, USA, 2009, pp. 61–70. https://doi.org/10.1145/1526709.1526719

[5] J. Seo and W.B. Croft. "Local text reuse detection," in *Proceedings of SIGIR'08*, Singapore. ACM, ACM Press, July 2008, pp. 571–578. https://doi.org/10.1145/1390334.1390432

[6] D. Sorokina, J. Gehrke, S. Warner, and P. Ginsparg, "Plagiarism detection in arXiv," Cornell University, Ithaca, NY, USA, Tech. Rep. TR2006-2046, 2006. https://doi.org/10.1109/ICDM.2006.126

[7] T. C. Hoad and J. Zobel, "Methods for identifying versioned and plagiarized documents," *Journal of the American Society for Information Science and Technology*, vol. 54, no. 3, 2003, pp. 203–215. https://doi.org/10.1002/asi.10170

[8] S. Schleimer, D. S. Wilkerson, and A. Aiken, "Winnowing: local algorithms for document fingerprinting," in *Proceedings of SIGMOD'03*, 2003, pp. 76–85. https://doi.org/10.1145/872757.872770
[9] R. A. Finkel, A.B. Zaslavsky, K. Monostori, and H. W. Schmidt. "Signature extraction for overlap detection in documents," in *Proceedings of the 25th Australasian Computer Science Conference, Conferences in Research and Practice in Information Technology*, vol 4, Melbourne, Australia: Australian Computer Society Inc., 2002, pp. 59–64.
[10] N. Heintze, "Scalable document fingerprinting," in *1996 USENIX Workshop on Electronic Commerce*, 1996.
[11] N. Shivakumar and H. Garcia-Molina, "SCAM: A copy detection mechanism for digital documents," in *Proceedings of the 2nd Annual Conference on the Theory and Practice of Digital Libraries*, 1995.
[12] S. Brin, J. Davis, and H. Garcia-Molina, "Copy detection mechanisms for digital documents," in *Proceedings of ACM SIGMOD'95*, 1995, pp. 398–409. https://doi.org/10.1145/568271.223855
[13] U. Manber, "Finding similar files in a large file system," in *WTEC'94: Proceedings of the USENIX Winter 1994 Technical Conference*, USENIX Association, Berkeley, CA, USA, 1994, pp. 1–10.
[14] A. Mittelbach, L. Lehmann, C. Rensing, and R. Steinmetz, "Automatic Detection of Local Reuse," in *Sustaining TEL: From Innovation to Learning and Practice - Proceedings of the 5th European Conference on Technology Enhanced Learning, EC-TEL 2010*, no. LNCS 6383, Springer Verlag, September 2010, pp. 229–244. https://doi.org/10.1007/978-3-642-16020-2_16
[15] R. Rivest, "The MD5 Message-Digest Algorithm," *RFC 1321*, April 1992. https://doi.org/10.17487/rfc1321
[16] M. O. Rabin, "Fingerprinting by random polynomials," Harvard University, Cambridge, MA, USA, Tech. Rep. TR-15-81, 1981.
[17] G. Fowler, L. C. Noll, K.-P. Vo, D. Eastlake, and T. Hansen, "The FNV non-cryptographic hash algorithm," Internet Engineering Task Force, Internet-Draft, 2019. [Online]. Available: https://tools.ietf.org/html/draft-eastlake-fnv-17 [Accessed: Feb. 24, 2020].

Gints Jēkabsons received his Doctoral degree in 2009 from Riga Technical University. At present, he is an Assistant Professor and Researcher at the Department of Artificial Intelligence and Systems Engineering of Riga Technical University. His current research interests include information retrieval, machine learning, and natural language processing.
E-mail: gints.jekabsons@rtu.lv
ORCID iD: https://orcid.org/0000-0002-9575-2488