

12-2016

# A Robust System for Local Reuse Detection of Arabic Text on the Web

Leena Mahmoud Ahmed Lulu

Follow this and additional works at: [https://scholarworks.uaeu.ac.ae/all\\_theses](https://scholarworks.uaeu.ac.ae/all_theses)

Part of the [Computer Sciences Commons](#)

---

## Recommended Citation

Ahmed Lulu, Leena Mahmoud, "A Robust System for Local Reuse Detection of Arabic Text on the Web" (2016). *Theses*. 646.  
[https://scholarworks.uaeu.ac.ae/all\\_theses/646](https://scholarworks.uaeu.ac.ae/all_theses/646)

This Dissertation is brought to you for free and open access by the Electronic Theses and Dissertations at Scholarworks@UAEU. It has been accepted for inclusion in Theses by an authorized administrator of Scholarworks@UAEU. For more information, please contact [fadl.musa@uaeu.ac.ae](mailto:fadl.musa@uaeu.ac.ae).



جامعة الإمارات العربية المتحدة  
United Arab Emirates University

United Arab Emirates University

College of Information Technology

A ROBUST SYSTEM FOR LOCAL REUSE DETECTION OF  
ARABIC TEXT ON THE WEB

Leena Mahmoud Ahmed Lulu

This dissertation is submitted in partial fulfilment of the requirements for the degree  
of Doctor of Philosophy

Under the Supervision of Professor Boumediene Belkhouche

December 2016

## Declaration of Original Work

I, Leena Mahmoud Ahmed Lulu, the undersigned, a graduate student at the United Arab Emirates University (UAEU), and the author of this dissertation entitled "*A Robust System for Local Reuse Detection of Arabic Text on the Web*", hereby, solemnly declare that this dissertation is my own original research work that has been done and prepared by me under the supervision of Professor Boumediene Belkhouche, in the College of Information Technology at UAEU. This work has not previously been presented or published, or formed the basis for the award of any academic degree, diploma or a similar title at this or any other university. Any materials borrowed from other sources (whether published or unpublished) and relied upon or included in my dissertation have been properly cited and acknowledged in accordance with appropriate academic conventions. I further declare that there is no potential conflict of interest with respect to the research, data collection, authorship, presentation and/or publication of this dissertation.

Student's Signature: 

Date: 5/1/2017

Copyright © 2016 Leena Mahmoud Ahmed Lulu  
All Rights Reserved



## Advisory Committee

1) Advisor: Boumediene Belkhouche

Title: Professor

Department of Computer Science and Software Engineering

College of Information Technology

2) Co-advisor: Saad Harous

Title: Associate Professor

Department of Computer Science and Software Engineering

College of Information Technology

3) Member: Liren Zhang

Title: Professor

Department of Computer and Network Engineering

College of Information Technology

## Approval of the Doctorate Dissertation

This Doctorate Dissertation is approved by the following Examining Committee Members:

1) Advisor (Committee Chair): Boumediene Belkhouche

Title: Professor

Department of Computer Science and Software Engineering

College of Information Technology

Signature 

Date Dec 5, 2016

2) Member: Salah Bouktif

Title: Associate Professor

Department of Computer Science and Software Engineering

College of Information Technology

Signature 

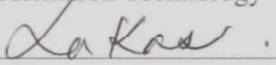
Date 05-12-2016

3) Member: Abderrahmane Lakas

Title: Associate Professor

Department of Computer and Network Engineering

College of Information Technology

Signature 

Date 05-12-2016

4) Member (External Examiner): Nizar Habash

Title: Associate Professor

Department of Computer Science


Institution: New York University- Abu Dhabi

Signature 

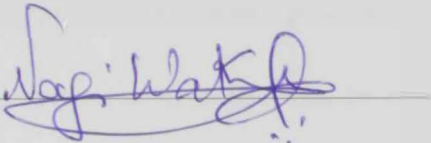
Date Dec 5, 2016

This Doctorate Dissertation is accepted by:

Dean of the College of Information Technology: Professor Omar El-Gayar

Signature  Date Jan 5, 2017

Dean of the College of Graduate Studies: Professor Nagi T. Wakim

Signature  Date 10/1/2017

Copy 5 of 7

## Abstract

We developed techniques for finding local text reuse on the Web, with an emphasis on the Arabic language. That is, our objective is to develop text reuse detection methods that can detect alternative versions of the same information and focus on exploring the feasibility of employing text reuse detection methods on the Web. The results of this research can be thought of as rich tools to information analysts for corporate and intelligence applications. Such tools will become essential parts in validating and assessing information coming from uncertain origins. These tools will prove useful for detecting reuse in scientific literature too. It is also the time for ordinary Web users to become Fact Inspectors by providing a tool that allows people to quickly check the validity and originality of statements and their sources, so they will be given the opportunity to perform their own assessment of information quality.

Local text reuse detection can be divided into two major subtasks: the first subtask is the retrieval of candidate documents that are likely to be the original sources of a given document in a collection of documents and then performing an extensive pairwise comparison between the given document and each of the possible sources of text reuse that have been retrieved. For this purpose, we develop a new technique to address the challenging problem of candidate documents retrieval from the Web. Given an input document  $d$ , the problem of local text reuse detection is to detect from a given documents collection, all the possible reused passages between  $d$  and the other documents. Comparing the passages of document  $d$  with the passages of every other document in the collection is obviously infeasible especially with large collections such as the Web. Therefore, selecting a subset of the documents that potentially contains reused text with  $d$  becomes a major step in the detection problem. In the setting of the Web, the search for such candidate source documents is usually performed through limited query interface. We developed a new efficient approach of query formulation to retrieve Arabic-based candidate source documents from the Web. The candidate documents are then fed to a local text reuse detection system for detailed similarity evaluation with  $d$ . We consider the candidate source document retrieval problem as an essential step in the detection of text reuse.

Several techniques have been previously proposed for detecting text reuse, however, these techniques have been designed for relatively small and homogeneous collections. Furthermore, we are not aware of any actual previous work on Arabic text reuse detection on the Web. This is due to complexity of the Arabic language as well as the heterogeneity of the information contained on the Web and its large scale that makes the task of text reuse detection on the Web much more difficult than in relatively small and homogeneous collections. We evaluated the work using a collection of documents especially constructed and downloaded from the Web for the evaluation of Web documents retrieval in particular and the detailed text reuse detection in general. Our work to a certain degree is exploratory rather than definitive, in that this problem has not been investigated before for Arabic documents at the Web scale. However, our results show that the methods we described are applicable for Arabic-based reuse detection in practice. The experiments show that around 80% of the Web documents used in the reused cases were successfully retrieved. As for the detailed similarity analysis, the system achieved an overall score of 97.2% based on the precision and recall evaluation metrics.

**Keywords:** Information Retrieval, Text Mining, Local Text Reuse Detection, Candidate Document Retrieval, Fingerprinting, Web Queries Formulation, Query Expansion, Arabic Text Processing.

## Title and Abstract (in Arabic)

### نظام فعّال لكشف اقتباس النصوص العربية عبر الشبكة العنكبوتية

#### الملخص

في هذه الأطروحة، قمنا بتطوير تقنيات للعثور على مواضع اقتباس جزء من النص عبر الشبكة العنكبوتية مع التركيز على اللغة العربية. فالهدف من هذا البحث هو تطوير طرق الكشف للنص المقتبس والتي يمكن أن تكشف عن إصدارات بديلة لنفس المعلومات والتركيز على استكشاف جدوى استخدام طرق الكشف للنص المقتبس على شبكة الانترنت. نتائج هذا البحث يمكن اعتبارها أدوات غنية لمحللي المعلومات لتطبيقات الشركات والاستخبارات. وسوف تصبح هذه الأدوات من الأجزاء الأساسية لوسائل التأكد من المعلومات الواردة من مصادر غير مؤكدة وتقييم صحتها. ستثبت هذه الأدوات فاعليتها للكشف عن اقتباس النصوص في البحوث العلمية أيضا. بل هو أيضا الوقت لمستخدمي الانترنت العاديين ليصبحوا مفتشين عن صحة الحقائق من خلال توفير الأدوات التي تسمح للناس للتحقق بسرعة من صحة وأصالة البيانات ومصادرهما، ولذلك ستعطى لهم الفرصة لأداء تقييمهم الخاص بهم لجودة المعلومات التي ترددهم.

يمكن تقسيم الكشف عن إعادة استخدام جزء من النص من مستند ما إلى قسمين رئيسيين: المهمة الفرعية الأولى هي استرجاع الوثائق المرشحة التي من المحتمل أن المصادر الأصلية للمستند وزعت في مجموعة من الوثائق ومن ثم إجراء عملية مقارنة واسعة بين المعني، و كل المصادر الممكنة لإعادة استخدام النص التي تم استردادها الغرض، طو في هذه الأطروحة تقنية جديدة لمعالجة هذه المشكلة الصعبة المتمثلة استرجاع المستندات المرشحة من الشبكة العنكبوتية (الويب)، والتي يمكن تعريفها كالتالي نظرا لمستند الإدخال (د)، فإن مشكلة الكشف عن النص المقتبس في جزء من المستند فقط في الكشف من مجموعة مستندات معينة، عن كل النصوص المقتبسة والمشاركة بين المستند المدخل (د) وغيرها من هذه المستندات. من الجلي كون المقارنة لمقاطع النص من المستند (د) مع مقاطع من كل وثيقة أخرى في مجموعة للمستندات غير مجد خصوصا مع مجموعات كبيرة مثل شبكة الإنترنت. لذلك، فإن اختيار مجموعة فرعية من المستندات التي يُحتمل أن تحتوي على نص مقتبس مع (د) أصبح خطوة رئيسية في مشكلة الكشف المأمولة. في مرحلة الإعداد للويب، يتم إجراء عملية البحث عن هذه المستندات المرشحة عادة من خلال واجهة استعلام بحثية محدودة.



لقد قمنا بتطوير نهج جديد فعال لصياغة الاستعلام لاسترجاع المستندات العربية المرشحة لتكون مصادر للنصوص المقتبسة في المستند المُدخل (د) من الويب، ومن ثم يتم استخدام هذه المستندات لنظام الكشف عن النص المقتبس لتقييم مفصل مع المستند (د). ونحن نعتبر أن مشكلة إيجاد واسترجاع المصادر المرشحة كخطوة أساسية في نجاح نظام الكشف عن النص المقتبس.

لقد تم تقديم العديد من التقنيات سابقا للكشف عن إعادة استخدام النص المقتبس للغات اللاتينية كالإنجليزية، ومع ذلك، فقد تم تصميم هذه التقنيات لمجموعات صغيرة نسبيا ومتجانسة. وعلاوة على ذلك، نحن لسنا على علم بأي عمل فعلي سابق للكشف عن إعادة استخدام النص مصممة خصيصا للمستندات العربية عبر شبكة الويب. يرجع السبب لذلك إلى كون اللغة العربية لغة معقدة مقارنة بغيرها من اللغات بالإضافة إلى عدم تجانس المعلومات الواردة على شبكة الانترنت وحجمها الكبير الذي يجعل مهمة الكشف عن النص المقتبس عبر شبكة الانترنت أكثر صعوبة مما هو عليه في المجموعات الصغيرة نسبيا. تم تقييم العمل باستخدام مجموعة من المستندات تم إعدادها خصيصا وتحملها من شبكة الويب من أجل تقييم أداء مهمة استرجاع المستندات من شبكة الويب بشكل خاص و من ثم تقييم نظام الكشف عن النص المقتبس بشكل عام. يعد هذا البحث لدرجة ما استكشافياً بدلاً من كونه محققاً حيث إنه لم يتم التحقيق في هذه المشكلة من قبل على المستندات العربية في نطاق الويب. ومع ذلك، تظهر نتائجنا أن الخوارزميات التي طورناها في هذا البحث قابلة للتطبيق في الممارسة العملية لكشف النصوص المقتبسة في أجزاء من المستندات ذات الأصول العربية. تبين التجارب أن ما يقارب 80% من مستندات الويب المستخدمة في تمثيل حالات الاقتباس استخدمها تم تحديدها واسترجاعها بنجاح. أما بالنسبة للتحليل المفصل لنسبة التشابه في النصوص، حقق النظام على النتيجة الإجمالية 97,2% بالنسبة إلى مقاييس التقييم المتعارف عليها كالدقة

**مفاهيم البحث الرئيسية:** استرجاع المعلومات، استغلال النصوص، كشف تكرار استعمال النصوص المحلية، استرداد المستندات المرشحة، البصمة الإلكترونية لفهرسة المستندات، صياغة الاستعلامات الشبكية، توسيع نطاق الاستعلام، معالجة النص العربي.

## Acknowledgements

I would like to express my special appreciation and thanks to my advisers Professor Boumediene Belkhouche and Dr. Saad Harous for their great support, patience, and encourage throughout my PhD study. I thank them for continuing to believe in me even at times that I lost believing in myself. Their guidance, advice and support have been priceless. I would also like to thank the members of my advisory committee, Professor Liren Zhang and Dr. Yacine Atif and the members of the examining committee, Dr. Abderrahmane Lakas, Dr. Salah Bouktif, and Dr. Nizar Habash. I am grateful to them for their feedback and advice.

I would like to thank my parents, my husband, my family, and all of those around me who supported me unconditionally throughout my studies. No word can express my gratitude and appreciation to my parents for all the unmatched love, patience, and sacrifices they have made for me. Your love and prayers were what sustained me thus far. I am forever indebted to my mother for her great help in taking care of my children and for the sleepless nights during the time of my writing of this thesis. This thesis would not have been completed without her support. I would like to thank my husband Ashraf and my children, Ghalya, Yahya, Yazan and the little sweet baby Malak. This work would not have been possible without Ashraf's love, patience, and advice. I am forever grateful to Ashraf for his support and encouragement. To my children, I am very overwhelmed for the every-night prayers, and for the love, joy, and happiness that they bring, and will bring, to our lives. Finally, I would like to thank all my brothers Ahmed, Mohammed, Abdallah, and Ayman, and my sisters Deena and Hayat and all their families for their spiritual support, encouragement and prayers. I would like to acknowledge the efforts of my brother Mohammed and thank him for always being there for me whenever needed. I offer my sincere gratitude to my beloved Aunt. Hayat for her continuous encouragement and love. I extend my thanks to my mother-in-law and my sisters-in-law Hanan, Madleen, and Nedaa for their prayers and support. Special thanks to Mrs. Layla, Om Mohammed, the friend of the family, for her unforgettable help and love to me and my children and to Mrs. Majeda, Om Raafat, for her spiritual support and prayers.

A special thanks goes to my friend and colleague Rita for all the stimulating discus-



sions and for our passionate conversations we had during our study. I would like to express my gratitude to all my supporting friends, Maitha, Amna, Noha, Manal, and Shehtaj Sultana.

Finally, I would like to thank the United Arab Emirates University for offering me the scholarship that helped me pursue my PhD study.

## Dedication

*To*

*my dearest parents,*

*my beloved husband,*

*& my adorable children*

*Ghalya, Yahya, Yazan, & Malak*

## Table of Contents

Title . . . . .	i
Declaration of Original Work . . . . .	ii
Copyright . . . . .	iii
Advisory Committee . . . . .	iv
Approval of the Doctorate Dissertation . . . . .	v
Abstract . . . . .	vii
Title and Abstract (in Arabic) . . . . .	ix
Acknowledgements . . . . .	xi
Dedication . . . . .	xiii
Table of Contents . . . . .	xiv
List of Tables . . . . .	xvii
List of Figures . . . . .	xviii
Chapter 1: Introduction . . . . .	1
1.1 Motivations . . . . .	1
1.2 Text Reuse Detection . . . . .	2
1.3 Related Research . . . . .	3
1.4 Text Reuse and Plagiarism Detection . . . . .	7
1.4.1 Plagiarism Detection Types . . . . .	7
1.4.2 Plagiarism Detection Classes . . . . .	8
1.5 Text Reuse on the Web . . . . .	9
1.6 Text Reuse Detection for Arabic Text . . . . .	10
1.7 Contributions of this Thesis . . . . .	10
1.8 Objectives . . . . .	11
1.8.1 Major Objectives . . . . .	11
1.8.2 General Objectives . . . . .	12
1.9 Dissertation Outline . . . . .	12
Chapter 2: An Overview of Fingerprinting Techniques . . . . .	14
2.1 Fingerprint Generation . . . . .	15
2.2 Fingerprint Matching . . . . .	15
2.3 Fingerprinting Approaches . . . . .	16
2.3.1 Overlap Fingerprinting Methods . . . . .	16
2.3.2 Non-Overlap Fingerprinting Methods . . . . .	19
2.3.3 Other Approaches . . . . .	20
2.4 Evaluation of Fingerprinting Approaches . . . . .	22

2.4.1 Information Retrieval Evaluation . . . . .	22
2.4.2 Text Reuse and Plagiarism Detection Corpora . . . . .	26
2.4.3 Performance Evaluation of Fingerprinting techniques . . . . .	27
Chapter 3: System Architecture and Text Analysis for Reuse Detection . . . . .	30
3.1 Introduction . . . . .	30
3.2 Web-based Candidate Documents Retrieval . . . . .	31
3.3 Text Pre-processing and Representation . . . . .	31
3.4 Arabic Language Characteristics . . . . .	32
3.5 Text Preprocessing Techniques . . . . .	35
3.5.1 Tokenization . . . . .	36
3.5.2 Character Normalization and Diacritics Removal . . . . .	36
3.5.3 Stemming . . . . .	37
3.5.4 Stop-words Filtering . . . . .	39
3.5.5 Sentence Identification and Segmentation . . . . .	40
3.5.6 Punctuation Removal . . . . .	41
3.6 Text Representation Techniques . . . . .	41
3.6.1 Bag of Words Representation . . . . .	41
3.6.2 <i>N</i> -Grams Representation Model . . . . .	42
3.6.3 Hash Model . . . . .	43
3.7 Local Text Reuse Detection and Text Alignment . . . . .	44
3.8 Post-processing and Presentation . . . . .	44
3.9 Implementation Details . . . . .	44
3.10 Chapter Summary . . . . .	49
Chapter 4: Candidate Document Retrieval from the Web . . . . .	50
4.1 Introduction . . . . .	50
4.2 Related Literature . . . . .	52
4.3 Our Proposed Document Retrieval Model . . . . .	54
4.3.1 Notation and Problem Definitions . . . . .	54
4.3.2 Model Specification . . . . .	56
4.4 Experimental Results . . . . .	60
4.4.1 A Case Study . . . . .	60
4.4.2 Construction of Candidate Document Retrieval Test Cases . . . . .	61
4.4.3 Best Query Length . . . . .	67
4.4.4 Quality of the Formulated Queries . . . . .	67
4.5 Chapter Summary . . . . .	68
Chapter 5: Fingerprinting-based Detection and Detailed Analysis of Text Reuse . . . . .	69
5.1 Introduction . . . . .	69
5.2 Fingerprinting Approaches . . . . .	70
5.2.1 Hailstorm Fingerprinting Method . . . . .	70
5.2.2 Winnowing Fingerprinting Method . . . . .	71
5.3 A Framework for Text Reuse Detection Using A Hybrid Approach . . . . .	72
5.3.1 Candidate Document Retrieval Model . . . . .	76
5.3.2 Pairwise Comparison Model (Text Alignment) . . . . .	83
5.3.3 Reused Text Presentation . . . . .	85
5.4 Evaluation . . . . .	86

5.4.1 Performance Measures . . . . .	86
5.4.2 Experimental Setup . . . . .	88
5.4.3 Experimental Results and Analysis . . . . .	88
5.4.4 Comparison to Other Fingerprinting Approaches . . . . .	95
5.5 Statistical Analysis . . . . .	98
5.5.1 Experiment Definition . . . . .	98
5.5.2 Experiment Planning . . . . .	98
5.5.3 Experiment Operation . . . . .	99
5.5.4 Experiment Interpretation . . . . .	105
5.6 Chapter Summary . . . . .	106
Chapter 6: Conclusions . . . . .	108
6.1 Dissertation Summary . . . . .	108
6.1.1 Dissertation Contributions . . . . .	109
6.2 Future Research . . . . .	110
References . . . . .	111

## List of Tables

Table 2.1:	Comparison of fingerprinting techniques based on their selection heuristics. Note that $d$ refers to a document, $c$ and $C$ are a chunk and the set of chunks, respectively . . . . .	29
Table 4.1:	Characteristics of the collection for Web documents retrieval . . . . .	66
Table 5.1:	Characteristics of the corpus used to evaluate the system [1] . . . . .	89
Table 5.2:	Micro and macro measures of the DETRA with different values of $n$ for $n$ -grams and sentence size=25 . . . . .	90
Table 5.3:	Micro and macro measures of DETRA with different values of $n$ for $n$ -grams and sentence size=30 . . . . .	92
Table 5.4:	Micro and macro measures of DETRA with different values of $n$ for $n$ -grams and sentence size=35 . . . . .	93
Table 5.5:	Macro measures of the winnowing approach in terms of the F-measure and the overall TRDS compared with our method for the different values of $n = 3 \dots 7$ of $n$ -grams, window size = 10, and sentence size=25 .	96
Table 5.6:	Macro measures of the winnowing approach in terms of the F-measure and the overall TRDS compared with our method for the different values of $n = 3 \dots 7$ of $n$ -grams, window size = 10, and sentence size=30 .	97
Table 5.7:	The statistical measures computed for the four different models based on the number of reused source documents and the overall similarity percentage . . . . .	106

## List of Figures

Figure 1.1:	The general form of text reuse . . . . .	3
Figure 1.2:	The similarity spectrum as described in [2] . . . . .	4
Figure 1.3:	Transition relation in near-duplicate text detection does not apply to local text reuse detection . . . . .	5
Figure 2.1:	The evaluation of 5 fingerprinting algorithms in terms of precision and recall values as reported in [3] . . . . .	26
Figure 2.2:	The performance evaluation of 5 fingerprinting algorithms in terms of F-measure value using TREC newswire, [4] . . . . .	28
Figure 3.1:	Framework for Web-based Arabic Text Reuse Detection . . . . .	30
Figure 3.2:	Different valid word reordering of the same sentence in Arabic . . . . .	32
Figure 3.3:	Two similar words in Arabic شجر (shown in red) that share the same root. However, the two words have different meanings based on the context. The first means dispute, while the second means trees . . . . .	33
Figure 3.4:	Different representations of the same word علم (ilm - knowledge) based on the presence or absence of diacritics . . . . .	34
Figure 3.5:	A complete sentence formed in Arabic by binding clitics to the verb. Unbinding some of the clitics produces longer sentences as in the second example . . . . .	34
Figure 3.6:	Text preprocessing scheme applied before text reuse detection . . . . .	35
Figure 3.7:	The set of prefixes and suffixes in the Arabic language, which are stripped off by the stemming algorithms . . . . .	39
Figure 3.8:	The stop-words list used in the stop-words removal preprocessing operation . . . . .	40
Figure 3.9:	Word and character $n$ -grams on Arabic text for different values of $n$ . . . . .	42
Figure 3.10:	A sample of the visual output presenting a reused text case (enclosed in a box) . . . . .	45
Figure 3.11:	The graphical user interface of the proposed system . . . . .	48
Figure 3.12:	An example of text reuse detection of a 2-pages report for a set of input documents. Page (1) displays a list of the input documents examined for text reuse instances. The second page (2) shows a detailed sample of reused text instances for input-document0055.txt and the corresponding source document from which it shares reused text . . . . .	48
Figure 4.1:	The proposed framework of a Web-based text reuse detection for Arabic documents (DETRA). Phase 1 in the DETRA is shaded in blue . . . . .	51
Figure 4.2:	The building blocks of candidate document retrieval from the Web . . . . .	56
Figure 4.3:	The part of DETRA's system interface for Web documents retrieval task . . . . .	60
Figure 4.4:	An example of a document with text copied from other Web documents. Each reused text is displayed in a bounding box. The different colors indicate different source documents. Note that the representative fingerprints of the document are displayed in red . . . . .	62
Figure 4.5:	The set of randomly generated queries for the sample input document in Figure 4.4. The queries are encoded into UTF-8 characters format in order to accept Arabic queries via the Web search engine interface . . . . .	63



Figure 4.6:	The top ranked Web documents' URL addresses returned by each query	64
Figure 4.7:	The XML output of the detection task identifying the Web URL addresses of the documents that share text with the input document with high similarity rate . . . . .	65
Figure 4.8:	The visual output of the detection task presenting part of the reused text, which is displayed in red . . . . .	65
Figure 4.9:	Comparing the different query lengths. The best result obtained with $w = 10$ terms per query . . . . .	67
Figure 4.10:	The quality of document retrieval mechanism (precision) . . . . .	68
Figure 5.1:	Hailstorm fingerprinting applied on a sample text . . . . .	72
Figure 5.2:	Winnowing fingerprinting applied on a sample text . . . . .	73
Figure 5.3:	The main processes for detecting local text reuse . . . . .	73
Figure 5.4:	An example of a text with selected fingerprints based on Hailstorm method . . . . .	74
Figure 5.5:	The selected fingerprints by our modified Hailstorm method . . . . .	75
Figure 5.6:	The building blocks of source documents retrieval from a given local documents collection . . . . .	76
Figure 5.7:	The meta information about the reused text cases detected as pairs between the input and the source document . . . . .	85
Figure 5.8:	A sample of the visual output presenting a reused text case (enclosed in a box) between the displayed document with a source document named: source_document00401 . . . . .	86
Figure 5.9:	Macro text reuse detection scores (TRDS) based on sentences length	89
Figure 5.10:	The micro and macro evaluation measures of our DETRA for documents with sentence size= 25 . . . . .	91
Figure 5.11:	The overall text reuse detection score (TRDS) of our proposed DETRA for documents with sentence size= 25 . . . . .	91
Figure 5.12:	The micro and macro evaluation measures of DETRA for documents with sentence size= 30 . . . . .	92
Figure 5.13:	The overall text reuse detection score of DETRA for documents with sentence size= 30 . . . . .	93
Figure 5.14:	The micro and macro evaluation measures of DETRA for documents with sentence size= 35 . . . . .	94
Figure 5.15:	The overall text reuse detection score of DETRA for documents with sentence size= 35 . . . . .	94
Figure 5.16:	An example of selecting a common phrase as a fingerprint, which may cause false positive cases . . . . .	94
Figure 5.17:	Another example of selecting a common phrase as a fingerprint, which may affect precision/recall scores . . . . .	95
Figure 5.18:	Comparison between the winnowing method and our DETRA based on the F-measure for sentences sizes= 25 and 30. DETRA outperforms the winnowing-based system . . . . .	97
Figure 5.19:	Comparison between the winnowing-based system and DETRA in terms of the overall text reuse detection scores (Macro TRDS) for sentences sizes 25 and 30. The proposed system DETRA outperforms the winnowing algorithm . . . . .	98



Figure 5.20: The exact fit model (model E), shown in black, and model D of DE-TRA system with both stemming and character normalization applied on the input text, shown in blue. The overall accuracy of the system is 92% . . . . . 101

Figure 5.21: The actual similarity percentages in model E compared with the similarity percentages computed in model C of the system, where character normalization is applied on the text and stemming is bypassed. The overall accuracy of this model is 87% . . . . . 102

Figure 5.22: The exact model E, shown in black, compared with model B without the character normalization process. The overall accuracy of the model is 86% . . . . . 103

Figure 5.23: The worst performance of the system in model A, with disregarding of both stemming and character normalization processes. The overall accuracy is 84% . . . . . 104

Figure 5.24: The combined four different models of the system with the exact model . . . . . 105

## Chapter 1: Introduction

### 1.1 Motivations

Text reuse stands for the act of using existing documents in creating new ones. It occurs in various forms based on the amount and method of reuse. Some documents are entirely copied and stored in many different locations. Some other documents are partially reused using one of the different forms of text reuse such as quotations, translations, rewording, summaries, or plagiarism. For example, people could quote text from other's emails in their replies. One could create various versions of a research paper, each of which is likely to have a significant amount of common text that makes them closely related to each other. Many authors may reuse considerable amount of text from their published work in conference papers to prepare more detailed journal publications of their work. Events or topics may be presented in various ways for different readers. Such presentations may be a copy of one another with few modifications. Therefore, the history of a given topic of interest can be identified given a sufficiently large archive.

Basically, *local* text reuse occurs when only small part of text in a document, *e.g.* sentences, facts, or passages, are copied and modified (see Figure 1.1). The scope of the term *local* includes a wide range of text transformations. For example, a typical text reuse from the Web, given the huge amount of text on the Web and the easy access to them, may happen by using Web search engines to look for a relevant source, then part of the text from that source is copied and possibly modified. More new statements or facts may be added to the original text, and some parts may be deleted, or partially rewritten, producing a similar document to the original source. Obviously, the level of similarity between such documents would vary substantially depending on the amount of editing performed from minor edits to complete modification [2]. In this thesis, we are interested in identifying such reused text in Arabic documents.

In general, detecting similar statements, facts, or passages is hard using the existing Web search engines. Several techniques have been previously proposed for detecting text

reuse; however, these techniques have been designed for relatively small and homogeneous collections. Our investigation of the state of the art reveals the lack of actual work on text reuse detection on the Web. This is due to the heterogeneity of the information contained on the Web and its large scale that make the task of text reuse detection on the Web much more difficult than in relatively small and homogeneous collections. However, the setting of the Web provides much more interesting and challenging properties to text reuse detection such as source detection and tracking event evolution. That is, given a sentence or a passage of interest, the goal of source detection is to find the documents from which the discussed topic originated. The results of this research will have a significant influence on the development of tools that can be used to validate information that comes from various sources of differing reliability. Such a tool would be valuable in many applications in education, scientific research, social networks, and national security.

More objectives and contributions of the thesis are addressed in details later in this chapter. In the next section, the basics of detecting text reuse are presented.

## **1.2 Text Reuse Detection**

Before we start reviewing researchers' approaches for local text reuse detection algorithms, it is necessary for us to have an elementary understanding of text reuse detection and information retrieval.

Text reuse detection has recently received significant attention of many researchers from different fields such as text data mining and information retrieval. Detecting text reuse has many applications that stem from it. For instance, duplicate or near-duplicate detection, which mainly detects documents that are almost similar to the original document except for very few changes, plays an important role in Web search and information retrieval, where it is used to filter the search results list. Plagiarism detection is another interesting application that has attracted the attention of many researchers. It occurs when an author of a document reuses text from one or more documents without acknowledging that reuse.

Local text reuse takes place when only small portion of a text in a document are copied

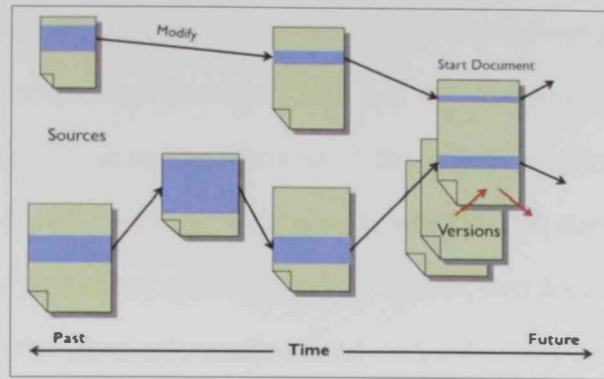


Figure 1.1: The general form of text reuse

and edited. Therefore, given a document, the detection problem of local text reuse can be defined as identifying all reused statements within a given document [4]. This task can be accomplished by first looking for candidate documents that are likely to be the original sources to the given document in a collection of documents and then performing an extensive pairwise comparison between the given document and each of the possible sources of text reuse that have been retrieved. Basically, this means comparing the passages of the given document one by one to the passages in the candidate sources, searching for conspicuous similarities. Intuitively, confining the search space to a reasonable and carefully selected number of candidate documents rather than searching all the available documents in the collection would certainly minimize the complexity of the search as well as enhance the efficiency of the system. This step becomes crucial in the setting of the Web. The detection process of text reuse can then be divided into two main sub-processes:

1. Candidate source documents retrieval.
2. Pairwise similarity comparison within documents.

The next section provides an overview of the related literature for the local text reuse detection problem.

### 1.3 Related Research

The degree at which passages of text are considered similar to each other (*i.e.*, local text reuse detection) can be placed somewhere in the middle range of the Similarity spec-

trum [2] (Figure 1.2). At one extreme of this spectrum is the highest degree of similarity. That is, two documents are considered identical. Much of the research that focused on duplicate detection and plagiarism is located at this end of the spectrum, identifying nearly identical documents [5, 6, 7, 8, 9, 10]. The topical similarity, which is the standard task of information retrieval lies at the other end of the spectrum, which means, two documents are a match if they are topically-related to the same information need or query. This traditional area of research has also been largely focused. Little prior research, however, deals with the intermediate form of similarity on the similarity spectrum, which is where text reuse is located.

The objective of this research is to develop methods for detecting the reuse of facts and concepts at the passage or sentence level. This degree of semantic similarity is a stronger form of topical relevance, but does not impose the syntactic similarity constraints typical of copy detection systems [11].

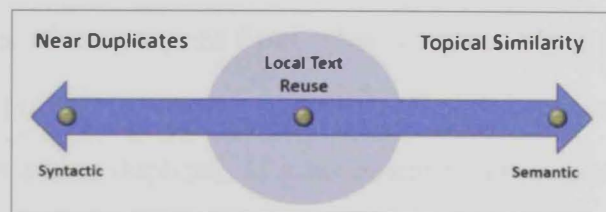


Figure 1.2: The similarity spectrum as described in [2]

As a research field, text reuse has received much attention. Researchers from the University of Sheffield initiated the research on a project named METER (MEasuring TEXT Reuse). The aim of this project was the automated detection of text reuse within specific domain of journalism [12, 13]. A follow up of this project was made by researchers from the University of Lancaster as they used the technologies developed in METER to analyze the text reuse in journalism from the 17th century [14]. A third project has started at the University of Massachusetts Amherst to explore methods for text reuse on the Web [2]. It is worth noting that, although plagiarism detection is considered a special case of the general field of text reuse detection, this particular application has been widely studied in the literature and many techniques have been proposed [3]. The following are surveys about text-based plagiarism detection [15, 16, 17] and surveys about code-based plagiarism detection techniques can be



found in [18, 19].

Detecting exact copies is fairly simple and has been widely studied [10, 20, 21]. Detecting partial copies, *i.e.*, looking for documents that are almost typical to the original documents except for few changes such as the insertion or paraphrasing of a few words (and well-known as near-duplicate or near-similarity detection), is complex and has been a major focus of researchers [22, 23, 24, 25]. It is even more complex when the reused text is modified or paraphrased as it is the case with local text reuse. Detecting this kind of local reuse can be used as a basis of new and powerful tools ranging from document management and information retrieval to a new information seeking behavior of the Web search such as the retrieval of related documents, or detecting the origin of text segments [26, 27].

Generally, the algorithms used for near-duplicate document detection do not work well for the local text reuse detection [4, 28]. This is due to the nature of the local text reuse, which reuses only small parts of a document from other sources. More precisely, near-duplicate document detection algorithms assume a transitive relation between documents. Precisely, document A should be a near duplicate of a document C, given that document A is a near-duplicate of a document B, which is a near-duplicate of document C. Figure 1.3 shows how this assumption is violated in the case of local text reuse detection. Note that  $\Delta$  in the figure is an indicator function that evaluates to 1 if two documents have text reuse relationship [4].

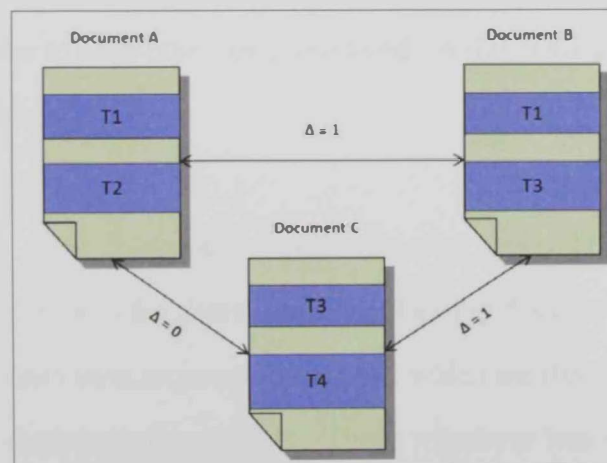


Figure 1.3: Transition relation in near-duplicate text detection does not apply to local text reuse detection

Several techniques for estimating similarity of text passages (or whole documents) to

each other have been proposed, most of them belong to one of the following three approaches as defined by [29]:

**Substring Matching.** In this approach, techniques such as Greedy String Tiling [30] and Local Alignment [31] are used to identify maximum matches in pairs of strings, which then are used as reuse indicators [32]. These strings are represented in suffix trees and graph-based metrics are used to evaluate the amount of text reuse [33]. However, the computational complexity of these methods in terms of time and storage is very high. For example, the worst case complexity of the standard Greedy String Tiling approach is proved to be  $O(n^3)$ , where  $n$  is the length of strings [30]. On the other hand, the complexity of the Local Alignment algorithms given two strings  $A$  and  $B$  is  $O((|A| + |B|)^3)$ .

**Keyword Similarity.** In this approach, representative topically-related keywords are collected and weighted from a given document. These keywords are then compared to the keywords extracted from the other documents. If a match is found between the documents' keywords, the documents with potential similarity are split into smaller portions of text, where these portions are compared recursively. This approach assumes that similarity is found only in documents with similar topics. [34]. Another similar approach compares text passages against their relative frequency of word occurrences [10, 11]. Two passages share similar word frequencies relative to each other are considered similar. This approach, again, assumes documents with similar topics.

**Fingerprint Analysis.** The most popular and successful approach to near-similarity search and local text reuse detection is the detection of overlapping documents fingerprints [20, 35]. Documents are divided into term sequences (chunks), which are then converted into a numeric form representing the document fingerprints. Then, whenever two documents share one or more fingerprints, this most probably indicates a reused text between the two documents while maintaining an acceptable runtime and storage behaviour [10, 21]. Chapter 2 overviews most of the available fingerprinting techniques for local text reuse detection. The next section

describes the general aspects of plagiarism detection.

## **1.4 Text Reuse and Plagiarism Detection**

Plagiarism is the act of unacknowledged re-use of someone's else work. Text plagiarism is an illegal reuse of text from one or more documents without being properly cited [36]. Although plagiarism detection is considered a special case of the general term text-reuse, this particular application has been widely studied in the literature [15, 16, 17, 37].

### **1.4.1 Plagiarism Detection Types**

The main objective of plagiarism detection tools is to detect and prevent the two main types of plagiarism. Namely, plagiarism in programming languages and plagiarism in natural languages (also called textual plagiarism).

One of the earliest models for automated plagiarism detection reported in the literature has focused on detecting plagiarism in students' source code [38]. Detecting plagiarism cases in programming languages is considered less complicated when compared to natural language. The detection involves following the main tools of a given program, such as lines numbers, statements, and methods calls and is well researched in the literature [18, 19]. Examples of plagiarism detection tools for programming codes are MOSS (Measure Of Software Similarity), YAP/YAP3 (Yet Another Plague), and SID (Software Integrity Diagnosis or Share Information Distance).

The detection task becomes more complicated when dealing with documents in natural language as the text may get easily modified or paraphrased. The most researched approach is when a text in a document is a duplicate or a near-duplicate of another document. However, the complexity of the problem increases when paraphrasing takes place. This difficulty of the problem varies from simple word insertions, deletions, or substitution to complete reformulation or even word re-ordering as it is the case for word-free languages. Natural language plagiarism can be classified into two main types: uni-lingual and multi-lingual plagiarism detection. Most of the researchers addressed the uni-lingual plagiarism detection type, which



deals with the automatic identification of textual plagiarism cases in documents with the same language, for instance, Arabic-Arabic. Multi-lingual plagiarism detection, on the other hand, addresses the automatic detection of plagiarism between documents with different languages. For example, English-Arabic.

#### 1.4.2 Plagiarism Detection Classes

Textual Plagiarism detection can also be divided into two major problem classes, namely external plagiarism detection and intrinsic plagiarism detection. In the case of external plagiarism, the process searches for candidate sources (from a large collection of documents) from which a suspicious document may have reused parts of its text [39]. On the other hand, intrinsic plagiarism detection focuses on the changes in the writing style of a given suspicious document by identifying parts of the document that differ significantly from the remaining text within the same document. Note that there is no reference collection used in this type of plagiarism class [40].

Many commercial plagiarism detection applications have been developed in the last decades. Some of these detectors are based on fingerprinting techniques addressed in this research, while other products such as *Academic Plagiarism Checker*, *PlagiarismDetect*, *Turnitin*, and *Grammarly* are for commercial purposes and their underlying technical information are kept hidden.

*Turnitin* is considered one of the most popular plagiarism detection systems. It has been reported that it uses documents collection as well as Web pages and 98% of the UK universities use it for plagiarism detection purposes [41]. On the other hand, its main technology relies on finding string matches of size 8 to 10 words as revealed by the analysis in [42]. Especially, the system performs well only for exact copies and probably not as good in the case of paraphrased plagiarism.

## 1.5 Text Reuse on the Web

In this section, we describe the main issues and challenges that are related to local text reuse detection on the Web.

Several techniques have been proposed for detecting local text reuse on the Web. However, these techniques have been developed to work on Latin languages and on relatively small and topically related datasets that have been created through crawling specific Web pages. To the best of our knowledge, no prior work has been previously conducted on text reuse detection on the Web. Bendersky and Croft [2], however, have recently suggested a framework for detecting English text reuse on the Web. They focused on retrieving and weighting key concepts related to the statement of interest, and defining the time-line of the statement itself and the related Web pages to it. However, the practical applicability of the suggested text reuse detection system on the Web has not been implemented.

Chiu *et. al.* [28] presented a prototype text reuse searching interface based on the architecture proposed in [2]. They investigated the feasibility of a text reuse architecture for the Web. The system first creates an initial document set from the Web and then applies k-gram based fingerprinting methods on the retrieved set. The steps include querying the documents to be used for the reuse detection, downloading them, and finally applying fingerprinting techniques to compute the similarity scores and define the time-line of reuse. Obviously, the performance of the whole system initially depends on which documents are retrieved at the first step, and finally the text reuse detection method. Therefore, the focus should be on new techniques that will advance the current research.

A straightforward approach to find documents on similar topics is to extract keywords from the input document and to retrieve other documents also containing these keywords [43]. Our contribution to this problem is a strategy to formulate queries to a Web search engine in order to retrieve candidate source documents. Using the Web as a source requires techniques for automatic query generation and formulation to identify potential candidates since the Web is the typical place of text reuse.

## 1.6 Text Reuse Detection for Arabic Text

The Arabic language is a rich morphological language that is among the mostly used languages in the world and on the Web as well. However, little or no attention is given to this language in the area of text reuse detection and even plagiarism detection. While the local text reuse detection problem has been mostly studied in the literature for Western languages, it is still one of the biggest challenges in the Arabic language and the research have remained quite limited. While there are few works related to plagiarism detection in Arabic on relatively small collections, we confidently state that there is no available previous work or systems on text reuse detection for Arabic-based documents on the Web.

It is worth mentioning that the only works concerning text reuse, and plagiarism detection in particular, are the works done by Alzahrani et al [44], Jadallah et al. [45], Menai [46], and Jaoua [47]. All of them focused on plagiarism detection of Arabic-based documents in relatively small collections. Besides, all of them are particularly tailored for the external approach of plagiarism detection, while the work done by Bensalem [48] focused on the intrinsic approach, and are, hence, less relevant to our work.

## 1.7 Contributions of this Thesis

In this thesis, we intend to develop techniques for finding text reuse on the Web. That is, our objective is to develop text reuse detection methods that can detect alternative versions of the same information from the Web or using any Arabic-based documents collection. The results of this research can be thought of as rich tools to information analysts to validate and assess information coming from uncertain sources. The system would prove useful for detecting reuse in scientific literature too. It is also the time for ordinary Web users to become “Fact Inspectors” by providing a tool that allows people to quickly check the validity and originality of statements and their sources, so they will be given the opportunity to perform their own assessment of information quality. The main contributions of this thesis are summarized as follows:

- We propose a new queries formulation model for candidate source documents retrieval at the Web scale. The queries are formulated by extracting a set of representative terms from a given document that form as a signature to that document in order to query a Web search engine via a public search API. The set of queries is selected in such a way that ensures a global coverage of the document.
- We propose a new text reuse detection technique based on fingerprinting to detect local text reuse in the setting of the Web. Our technique is hybrid in the sense of applying fingerprinting selection method together with Information Retrieval heuristics to produce better and more robust results and to accommodate with the complexity of the Arabic language.
- We develop a more effective and interactive Web search tool that allows long queries to be used rather than short keywords, which acts as an important sub-task in text reuse detection problem on the Web.

## **1.8 Objectives**

### **1.8.1 Major Objectives**

The major objectives of this dissertation are the following:

1. To design a system for detecting text reuse for Arabic documents on the Web.
2. To focus on developing efficient Web query formulation techniques to retrieve closely related documents from the Web based on the input document being tested.
3. To design models for detecting text reuse that includes paraphrasing, word deletion and words reordering.
4. To analyze the impact of text pre-processing techniques such as stemming, lemmatization, stop-word removal on the overall detection system.
5. To develop techniques for pairwise comparison between pair of documents.

6. To construct a corpus for the evaluation of both sub-systems, the candidate document retrieval from the Web, and the pairwise comparison between pair of similar documents.
7. To evaluate the effectiveness of the text reuse detection model and address the problems that arise in the Web environment.

### **1.8.2 General Objectives**

The general objectives of this research are the following:

- (1) To review the problem of detecting text reuse and its impact on the design of tools for retrieving information and its analysis.
- (2) To review the state of the art technology for text reuse detection, identifying the challenges behind the middle of level of the similarity spectrum.
- (3) To design models for detecting text reuse, capturing some text edits such as synonyms substitution, words deletion and words reordering.
- (4) To review existing fingerprinting techniques for text reuse detection.
- (5) To investigate and test the feasibility of text reuse detection at the Web scale in the setting of Web search and discover the advantages and drawbacks of using the Web as a source.
- (6) To address the challenges that would arise from detecting reuse in Arabic documents.

### **1.9 Dissertation Outline**

This dissertation is structured as follows:

In Chapter 2, we overview several local text reuse detection methods based on fingerprinting techniques. We first define the context of local text reuse and situate it within the general spectrum of information retrieval in order to pinpoint its particular applicability and challenges. After a brief description of the major text reuse detection approaches, we introduce the general principles of fingerprinting algorithms from an information retrieval



perspective. Three classes of fingerprinting methods (overlap, non-overlap, and randomized) are surveyed. Specific algorithms, such as k-gram, winnowing, hailstorm, DCT and hash-breaking, are described. A number of benchmark corpora for Latin documents that are used to evaluate the performance of the text reuse detection system is presented. Finally, the performance measures that are commonly used for evaluation are described and the characteristics of the mentioned algorithms are summarized based on data from the literature.

In Chapter 3, we present the system architecture and describe its phases. The Arabic language characteristics are also presented in this chapter. Text preprocessing and representation techniques used before applying the indexing and detection methods are also discussed. Finally, some implementation details are presented.

In Chapter 4, we define a new method of Web queries formulation for the problem of candidate documents retrieval from the Web. We evaluated the work using a collection of documents especially constructed for the evaluation of Web documents retrieval. The experiments show that, on average, around 80% of the Web documents used in the reused cases were successfully retrieved.

In Chapter 5, we describe in details the general framework to detect text reuse in Arabic texts. A set of experiments were conducted to gain an insight into the effect of the unique features of the Arabic language to the main components of the general framework of local text reuse detection. Query expansion is integrated into the retrieval phase in order to deal with edited text. Queries are expanded with terms deletion, words reordering, and synonyms substitution. We have evaluated the performance of the detection component of our system using a relatively large corpus that has been recently created for the detection of text reuse and plagiarism in Arabic documents. The results demonstrate that the query expansion tools improves the overall performance of the document retrieval phase compared to queries without expansion.

In Chapter 6, we summarize this dissertation and comment on possible future work.

## Chapter 2: An Overview of Fingerprinting Techniques

This chapter provides an in-depth study of a wide range of fingerprinting methods. We first introduce the general principles for any fingerprinting algorithm from an information retrieval perspective and then survey the relevant algorithms from the literature. As mentioned before in Chapter 1, unlike near-duplicate document detection, our focus is on algorithms that detects local text reuse based on parts of documents. For example, a reused text is typically modified to conform with the content of the document. Therefore, technologies to detect such cases of text reuse can not rely on the near-duplicate methods, in which the reused text being close to identical to its original, but rather have to be account for modifications of small parts of the documents. Such local text reuse detection methods are based on a technique called fingerprinting.

A fingerprint  $h(d)$  of a document  $d$  can be considered as a set of encoded substrings taken from  $d$ , which serve as a representation of the document  $d$ . These encoded substrings are usually formed using the  $n$ -gram method, which divides the document into a set of contiguous substrings of length  $n$  (the  $n$ -gram method is described in more details in Section 2.3). Each  $n$ -gram is then hashed and a subset of these hashes is selected to be the document's fingerprints. Note that  $h$  denotes the hash function. Other types of fingerprinting methods are formed by reformulating  $d$  as a whole to obtain a simplified representation of  $d$  while maintaining as much information as possible of the content of the document. Such fingerprints of the document are used for the local text reuse detection. In general, a good fingerprinting technique should satisfy the following properties:

- Accuracy: the fingerprints generated by the fingerprinting technique must be accurate enough to represent the documents.
- Efficiency: the generated fingerprints must be as small as possible.

Considering the above properties, we introduce several fingerprinting methods. However, most of these techniques share the following two operations: fingerprint generation and fingerprint matching.

## 2.1 Fingerprint Generation

Before designing an accurate and efficient fingerprinting technique, there are four areas in the process of creating it that need to be considered [29]:

- (1) Substring Selection. From the original document, substrings (chunks) are extracted according to some selection strategy. Such a strategy may consider positional, frequency-based, or structural information.
- (2) Substring Number. The substring number controls the fingerprint resolution. There is an obvious trade-off between fingerprint quality, computational cost, and space requirements, which must be carefully balanced. The more information of a document is encoded in the fingerprint, the more reliably a possible overlap of two fingerprints can be detected.
- (3) Substring Size. The substring size defines the fingerprint granularity. A fine granularity makes a fingerprint more susceptible to false matches, while with a coarse granularity fingerprinting becomes very sensitive to changes.
- (4) Substring Encoding. The selected substrings are converted into integer numbers. Substring conversion establishes a hash operation where, aside from uniqueness and uniformity, also efficiency is an important issue [49]. In most fingerprinting techniques, the popular MD5 hashing algorithm is often employed [50].

## 2.2 Fingerprint Matching

To estimate the amount of reused text between a pair of documents, a non-symmetric metric measure is used ([12, 26, 4]). Specifically, given two documents  $A$  and  $B$ , the amount of text in document  $A$ , which is shared with document  $B$  is represented as a ratio of the number of shared fingerprints to the number of fingerprints of document  $A$ . The ratio, *containment* of  $A$  in  $B$ , is expressed as follows [35]:



$$C(A, B) = \frac{|F_A \cap F_B|}{|F_A|} \quad (2.1)$$

where  $F_A$  and  $F_B$  are the sets of fingerprints of documents  $A$  and  $B$ , respectively. Note that the shared fingerprint ratio is non-symmetric ( $C(A, B) \neq C(B, A)$ ) since such a measure reflects the differences in the lengths of the documents. Generally, symmetric metrics are used for near-duplicate detection methods [35]. The estimated containment values of documents can then be divided into ranges (e.g., most, considerable, partial) based on the properties and goals of the text reuse application.

## 2.3 Fingerprinting Approaches

This section gives an overview of existing fingerprinting approaches for local text reuse detection. There are broadly two kinds of fingerprinting techniques for text reuse detection: overlap techniques and non-overlap techniques. The following two subsections overview the available methods of the two techniques.

### 2.3.1 Overlap Fingerprinting Methods

Overlap methods use a sliding window that is shifted by a word. The word sequence in the window (or its numeric value) is handled as a chunk. Given a size of the window as  $n$ , the  $i^{th}$  window will contain the  $i^{th}$  words to the  $i + n - 1^{th}$  word in the document. The methods that are based on the sliding window are referred to as *overlap methods* since the adjacent windows overlap each other, i.e.,  $n - 1$  of the words that appear in one window, will also appear in the subsequent window.

The overlap methods proved to produce good results, although they generate many chunks. However, various selection algorithms were proposed ([9, 7, 21, 26]), of which we describe four representatives below.

## ***N*-Gram**

*N*-gram is the first and simplest method of the overlap methods. An *n*-gram is a set of *n* contiguous tokens in the document. These tokens may be characters or words. It uses all the consecutive *n* tokens (also called chunks or shingles) of a document generated by a sliding window with size *n* as fingerprints. This sliding window is shifted by one token at a time. Hence, the total number of fingerprints for a document with total of *M* tokens is calculated as:

$$N_{n\text{-gram}} = M - n + 1 \quad (2.2)$$

Note that the above formula represents an upper bound for the number of fingerprints as the *n*-gram method is usually used in a set-wise approach [51].

Intuitively, the *n*-gram method shows good results since it uses all generated *n*-grams as fingerprints. However, it becomes too expensive and infeasible in big collections. The next overlap methods select a representative subset of the generated *n*-gram fingerprints in order to reduce their number and enhance the efficiency of the method.

## **0 mod *p***

One necessary property for a selection algorithm to work properly is that it must select the same subset of fingerprints for identical documents. Randomly selected chunks would not satisfy this property as it is unpredictable which chunks are selected. The *0 mod p* method selects only chunks with hash values divisible by *p* among all the *n*-gram chunks [9]. Thus, if two documents are identical, the selected chunks by *0 mod p* method in the documents are the same. The average number of fingerprints *0 mod p* selects is then reduced by a factor of *p* as follows:

$$N_{0\text{mod}p} = N_{n\text{-gram}} / p \quad (2.3)$$

The drawbacks of this method is that it does not represent the whole document accurately. For instance, if very common chunks are divisible by  $p$ , this could falsely determine a reuse relationship. On the other hand, it is not guaranteed that if two documents contain similar chunks (at different locations in the documents), that these chunks will be selected in both documents and hence, reuse may not be detected.

### Winnowing

Winnowing is another selection algorithm based on n-gram [26]. It uses a second window with size  $w$  sliding over the chunks derived from the original window in n-gram. Only one chunk is selected as a fingerprint from each winnowing window, which is the chunk with minimum hash value. The rightmost value is selected in case of ties.

Winnowing proved to yield better results than the *Omodp* method in practice and provided a lower bound of the selected fingerprints as:

$$N_{winnowing} \geq \frac{2}{w+1} \cdot N_{n-gram} \quad (2.4)$$

Since the winnowing method selection of the fingerprints depends on the hash values of the chunks within the same winnowing window, this means that even if two documents share a common chunk, winnowing does not necessarily select that chunk's fingerprint in both documents. However, if two documents share a sequence of words, which is at least as large as the winnowing window  $w$ , at least one common word out of this sequence is selected in both documents.

### Hailstorm

The Hailstorm method combines the results of winnowing with a more global coverage of the document. The words of a document are hashed prior to applying winnowing to

the document. Each chunk is then tested and selected as a fingerprint only if the lowest hash value is the leftmost (or the rightmost) token within the chunk [52].

It is required to choose a large window size for the Hailstorm to reach high compression rates. However, the quality of n-gram for the detection of local text reuse decreases for big window sizes. This is due to the fact that the sensitivity of n-gram for small changes increases with an increasing window size.

### 2.3.2 Non-Overlap Fingerprinting Methods

In the non-overlap methods, the document is divided into non-overlapping text segments rather than overlapping chunks. In such methods, the process of dividing text segments is called *breaking*, and the word position where a break occurs is known as *breakpoint*.

#### Hash-breaking

Hash-breaking is similar to the *Omod p* method but without overlapping. The words in the document are hashed and the breakpoints are created at locations where the hash values of the words are divisible by a given parameter  $p$ . The resulting text chunks are then hashed and used as fingerprints of the document.

The expected number of fingerprints for a document  $D$  using the hash-breaking method is  $L(D)/p$ , and thus the average length of text segments is  $p$ . However, in bad cases, the length of the text segment might be much shorter or much longer than expected, depending on the distribution of the hash values. In the worst case, the chunk could be very short and consists only of very common words. In this case, the reuse detection is badly affected. Soe and Croft refined the hash-breaking technique to reduce noisy fingerprints in such a way that it ignores the text segments whose lengths are shorter than  $p$  (revised hash-breaking) [4].

Another weak point of the hash-breaking method is that it is very sensitive to small modifications. A change of one character in a chunk, will lead to a completely different hash values and so the resulting fingerprints of the document. Soe and Croft proposed the use of

Discrete Cosine Transformation (DCT) to overcome the above problem and make it more robust against small changes [4].

### **Discrete Cosine Transformation (DCT)**

The DCT fingerprinting addresses the sensitivity problem of the hash-breaking method. The DCT (as defined in Wikipedia [53]) “expresses a sequence of finitely many data points in terms of a sum of cosine functions oscillating at different frequencies”. Its main characteristic that makes it widely used in numerous applications in science and engineering such as the lossy compression of audio and images is that small high-frequency components can be discarded as they are less important than low-frequency components.

DCT is expected to be more robust against small changes than hash-breaking. However, it can only be tolerant of at most a single word replacement [4].

### **2.3.3 Other Approaches**

#### **Randomized Fingerprinting**

A randomized algorithm or probabilistic algorithm is an algorithm which uses a degree of randomness as part of its logic. In computational geometry, a standard technique to build a structure is to randomly permute the input data and then insert them one by one into the existing structure. The randomization ensures that the expected number of changes to the structure caused by an insertion is small, and so the expected running time of the algorithm can be upper bounded. This technique is known as randomized incremental construction [54]. The same technique is used in [55] for document fingerprinting. This randomized fingerprinting provides a guarantee that a match with more than  $W$  characters will be detected with very high probability. The algorithm uses special data structures called synopsis data structures that are constructed in a randomized fashion for maintaining the distinct items, the most frequent items, and globally frequent items each on a separate data structure. It also uses an algorithm called count-sketch for estimating the frequency of items using a limited space. These two



techniques are generally used for processing data-streams.

### **Fuzzy Fingerprinting**

In the previously mentioned fingerprinting approaches, the fingerprint technology is used to identify text reuse between two documents using pairwise similarity comparison. Specifically, comparing the generated fingerprints for the two documents and checking whether they have a substantial overlap. In the case of fuzzy fingerprinting approach, a compact document model  $\mathbf{d}$  based on the vector space is constructed for a given document  $d$  and a special type of similarity comparison between  $\mathbf{d}$  and the fingerprints of all other documents in a collection is performed ([8, 56, 29]). According to [56], the algorithm for constructing fuzzy fingerprints includes the following steps: 1) Extracting a set of word terms (chunks) for document  $d$  by computing its vector space model  $\mathbf{d}$ ; 2) computing the vector of relative frequencies  $\mathbf{pf}$  of the prefix classes for the document's index words; 3) Normalizing the vector  $\mathbf{pf}$  and computing the vector of relative deviations  $\Delta_{pf}$  to the expected prefix class distribution; 4) Fuzzification of  $\Delta_{pf}$  by abstracting the exact deviations to one fuzzy deviation scheme, and then computing the fuzzy hash value for each word. The main advantage of the fuzzy fingerprinting is that it does not use the computationally expensive hashing function used by other fingerprinting techniques. It also produces better results in terms of the chunk size and the precision of shared fingerprints.

### **SVD-Based Detection**

Singular Value Decomposition (SVD) is a well-known factorization of a rectangular matrix in linear algebra. It is a tool for Latent Semantic Analysis (LSA) to deduce the latent semantic associations between two different entities [57]. In other words, LSA incorporates SVD to analyze relationships between a set of documents and their terms. The reuse detection method described in [58] employed LSA framework to infer the latent semantic associations and subsequently determine the document similarity while processing the selected documents.



The SVD matrix ( $M$ ) is composed of several vectors  $[M_1, M_2, \dots, M_n]$ , where vector  $M_i$  includes terms occurring in document  $i$  with their frequencies. These terms are mainly phrases of  $n$ -grams, which hence helps identify similarities between documents. However, since every document contains only a small subset of all phrases, this makes matrix  $M$  hugely sparse. Therefore, the computational cost of the decomposition in SVD will be very high if the phrase-space is too large.

## 2.4 Evaluation of Fingerprinting Approaches

### 2.4.1 Information Retrieval Evaluation

This section provides a formal framework of information retrieval in general, defining its terminology and briefly introducing the various measures used for evaluating the performance of information retrieval systems before proceeding with the evaluation of the fingerprinting techniques.

Information retrieval is a multidisciplinary field in Computer Science which aims at developing systems that helps obtaining information resources relevant to an information need from a collection of information resources [59]. It is also defined as a term conventionally applied to the type of computer activity that informs the user on the availability (or non-availability) of the resources (*i.e.*, documents) relating to his request [60].

In order for an information retrieval system to operate and retrieve information, that information must first be stored in the computer. Generally, the information will usually have been in the form of documents. However, a document representative is produced from a stored document, either manually or automatically, in which the computer can process. An information retrieval process begins when a user enters his query  $q$  into the system. Queries, denoted by  $Q$ , are formal statements of information needs, for instance a list of keywords to search in Web search engines. In information retrieval, a query  $q$  does not uniquely identify a single object in the collection. Instead, several objects may match the query, perhaps with different degrees of relevancy. This collection of objects is denoted by  $D$  and is usually organized in

documents. Therefore, information retrieval can be summarized as identifying a document  $d \in D$  that is relevant to  $q$ .

Most information retrieval systems compute a numeric value on how relevant each document in the collection of documents to the query. Retrieved documents are ranked according to this value. The top ranking documents are then shown to the user [59]. The relevance of a document  $d$  to a query  $q$  is measured by means of a retrieval model  $M = \langle \alpha, \beta, \gamma \rangle$ , where  $\alpha, \beta$ , and  $\gamma$  are representation functions. That is,  $\alpha : D \rightarrow \mathbf{D}$  that maps a document  $d \in D$  onto a machine-readable abstraction  $\mathbf{d}$ . The second function  $\beta : Q \rightarrow \mathbf{Q}$  maps the queries  $q \in Q$  onto a certain representation  $\mathbf{q}$  that represents its human-readable counterpart of  $q$  in the computation. The relevance function  $\gamma : \mathbf{D} \times \mathbf{Q} \rightarrow \mathbf{M}$  maps a pair of representations  $(\mathbf{d}, \mathbf{q})$  onto a real value that quantifies the relevance of  $d$  to  $q$ . Intuitively, the larger the relevance value of  $d$  to  $q$ , the more relevant it is.

Relevance is a subjective notion. Different users may have different views about the relevance or non-relevance of particular documents to given queries. Therefore, a relevance threshold  $\tau$  is specified that divides  $D$  into two sets of relevant documents  $D^+ = \{d | d \in D \text{ and } \gamma(\mathbf{d}, \mathbf{q}) \geq \tau\}$  and non-relevant documents  $D^- = D/D^+$ . This generally outlines the formal framework of an information retrieval system. In practice, retrieval models differ from one application to another in order to solve particular retrieval tasks.

Much effort and research has focused on solving the problem of evaluation of information retrieval systems. Six main measurable quantities have been studied as described in [61]:

- (1) The coverage of the collection, i.e., the extent to which the system includes relevant matter;
- (2) the time lag, which is, the average interval between the time the search request is made and the time an answer is given;
- (3) the form of presentation of the result;
- (4) the user's effort in obtaining answers to his queries;

- (5) the recall of the retrieval model, specifically, the proportion of relevant material actually retrieved in answer to a query;
- (6) the precision of the retrieval model, which is, the proportion of retrieved material that is actually relevant.

In order to evaluate the performance of an information retrieval model  $M$  of a given system, various measures for evaluating the effectiveness of retrieval models have been proposed [62, 63]. However, recall and precision are the most commonly applied measures in information retrieval. The measures require a collection of documents  $D$  and queries  $Q$ , such that for every query  $q \in Q$  there exists a subset of documents  $D^* \subset D$  that are considered relevant to  $q$ . The performance measures quantify the success of a retrieval model  $M$  in retrieving  $D^*$  for a given query  $q$  by comparing it with the obtained retrieval results  $D^+$ .

### Precision

Precision is the fraction of the documents retrieved that are relevant to the user's information need. It measures the purity of the set of retrieved documents  $D^+$ .

$$\text{precision}(D^+, D^*) = \frac{|\text{retrieved documents} \cap \text{relevant documents}|}{|\text{retrieved documents}|} = \frac{|D^* \cap D^+|}{|D^+|} \quad (2.5)$$

### Recall

Recall measures the completeness of the retrieved documents with respect to the relevant documents, which means, it can be viewed as the probability that a relevant document is retrieved by the query.

$$\text{recall}(D^+, D^*) = \frac{|\text{retrieved documents} \cap \text{relevant documents}|}{|\text{relevant documents}|} = \frac{|D^* \cap D^+|}{|D^*|} \quad (2.6)$$

Note that a 100% recall is returning all the documents in response to a query. Specif-

ically, by relaxing the threshold  $\tau$  so that all the documents in a collection are considered relevant (*i.e.*  $D^+ = D$ ). Therefore, using the recall measure alone doesn't give an accurate view of the performance of a retrieval model and the set of non-relevant documents needs to be computed too by computing the precision. On another note, an estimation of the recall value can be almost impossible. For example, in the case of Web information retrieval, where the amount of the relevant documents for a given query is unknown [64]. However, this can be solved by considering only the top  $n$  retrieved documents for evaluation purposes resulting in a new recall value named "recall at  $n$ " [65].

### F-measure

The precision and recall measures are combined into a single performance value using a weighted harmonic mean of the two measures called the F-measure:

$$F_b(D^+, D^*) = \frac{(1 + b^2) \cdot \text{precision}(D^+, D^*) \cdot \text{recall}(D^+, D^*)}{b^2 \cdot \text{precision}(D^+, D^*) + \text{recall}(D^+, D^*)} \quad (2.7)$$

where  $b \in [0, \infty)$ . The F-measure measures the effectiveness of retrieval with respect to a user who gives  $b$  times as much importance to recall as precision. That is, the precision measure is emphasized more if  $b < 1$  and  $b > 1$  means the recall is more emphasized.

There are various other measures that can be used such as average precision, r-precision, discounted cumulative gain, and many more [65]. However, the three mentioned measures above are considered the most common approaches to measure the performance and effectiveness of retrieval models.

The above outlines the formal framework that describes an information retrieval system in general. Note that not all users' needs are the same and thus retrieval models may differ from one application to another. In fact, there is no one "overall" theory of information retrieval. Nevertheless, the field is very strongly pragmatic: it is driven by practical problems

and considerations and assessed by practical criteria [66].

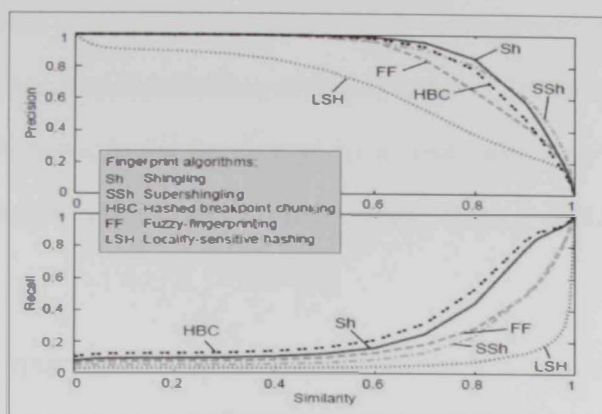


Figure 2.1: The evaluation of 5 fingerprinting algorithms in terms of precision and recall values as reported in [3]

## 2.4.2 Text Reuse and Plagiarism Detection Corpora

One of the main issues in text reuse detection is the lack of a unified document collection (corpus) for its evaluation. Various fingerprinting techniques have been proposed for text reuse detection, however, hardly any evidence can be found to name the best algorithm or the best system for text reuse detection. This is due to the fact that there was no one controlled evaluation benchmark for this field [3]. As in many information retrieval systems, a data collection and a set of measures form the evaluation framework necessary to compare between different approaches under a common setting. In the case of fingerprinting techniques, a number of corpora have been constructed and used to evaluate different fingerprinting methods such as:

1. METER corpus [67]: An early project exploring the automatic detection of text reuse and plagiarism is the METER project (MEasuring Text Reuse), the aim of which was the automated detection of text reuse within the specific domain of journalism ([12]). The METER technologies were subsequently used to analyze text reuse in journalism from the 17th century [2]. The METER corpus provides 445 text reuse cases among 1716 news articles. Although the corpus can be used for text reuse detection, its structure does not support all types of detection techniques.



2. TREC newswire collection or Reuters Corpus Volume 1 [68]: TREC and Reuters are considered standard corpora that are used for the assessment of many similarity-based algorithms, however, the distribution of highly similar documents is very low in these corpora when compared to the number of documents with very low similarity. This fact affects the evaluation in terms of the recall metric since it depends on very few pairs of documents.
3. Co-derivatives corpus [5]: The co-derivatives corpus is generated on the basis of Wikipedia articles revisions. Wikipedia forms a suitable environment for the evaluation of text reuse detection algorithms in general, since it publishes periodically revisions of its articles that undergo constant editing and revisions are often very similar to one another which provides a meaningful evaluation of text reuse detection techniques in terms of precision and recall.
4. PAN-PC [69]: A series of corpora have been created with test cases generated both automatically and manually to analyze reuse detection techniques for Latin languages in the framework of plagiarism detection competition. The PAN-PC corpus provides real, simulated and artificial cases to detect plagiarism. It may be considered as an evaluation framework for plagiarism detection, which consists of a large-scale plagiarism corpus and detection quality measures. This framework may serve as a unified test environment to compare future plagiarism detection research. However, the corpus is built only for Western languages such as English [70]. It is worth mentioning that a very recent shared task called ArPlagDet has been organized in December 2015 to address text reuse and plagiarism detection in Arabic text [1].
5. Other Corpora that are employed in the evaluation of near-duplicate detection techniques that are generally custom-built and not publicly available ([8, 25, 4]).

### 2.4.3 Performance Evaluation of Fingerprinting techniques

The standard information retrieval metrics are used to evaluate the performance of fingerprinting algorithms namely: precision, recall, and F-measure. A study of a Selective



number of fingerprinting algorithms have been carried out using Wikipedia as an evaluation framework in order to assess their performance in terms of precision, and recall. The test includes about 6 million articles along with their revisions. The experiment included a selective number of near-duplicate detection techniques based on fingerprinting such as n-gram, hash-breaking, fuzzy fingerprinting, and randomized fingerprinting. They were analyzed by comparing a given document  $d_q$  with all its revisions, which will probably provide high similarity results and hence influences the computed recall values. The given document is also compared to its successor article in order to evaluate the precision values. Figure 2.1 shows the similarity results in terms of precision and recall values [3]. The measures were computed by comparing the set of identified similar pairs of documents by a particular fingerprinting technique to a specified threshold. The study shows that fingerprinting techniques based on n-gram perform better than randomized techniques, with the hash-breaking method performs best. However, n-gram and hash-breaking methods produced much more fingerprints to represent a document than the fuzzy and randomized fingerprinting techniques [3].

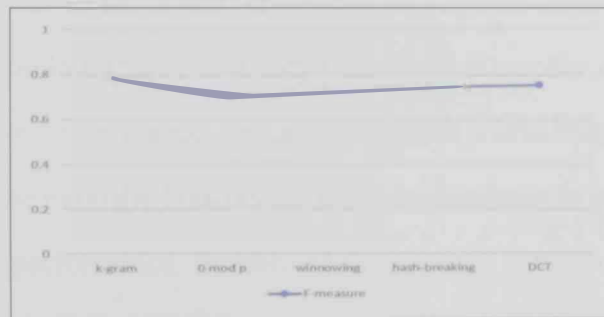


Figure 2.2: The performance evaluation of 5 fingerprinting algorithms in terms of F-measure value using TREC newswire, [4]

Another custom-based experiment was developed to evaluate a number of fingerprinting techniques for local text reuse detection. The fingerprinting methods: n-gram,  $0 \bmod p$ , winnowing, hash-breaking, and DCT fingerprinting were tested on TREC newswire and compared the detected document pairs from each method. The evaluation metric F-measure was used, which combines both values of precision and recall measures in this evaluation to assess the efficiency of the methods [4]. The overall performance of the selected fingerprinting methods is depicted in Figure 2.2.

Algorithm	Selection heuristic	Performance	Drawback
n-gram	All $c$ in each sliding window	Accurate	Too many fingerprints
$0 \bmod p$	$chunk \bmod p = 0$ , $p$ is fixed	Size is $1/p$ of all chunks	No matching guarantee
Winnowing	$h(c)$ is minimum in a window sliding over $d$	Few fingerprints	No match $<$ noise threshold
Hash-breaking	$h(w) \bmod p = 0$	Few fingerprints	Too sensitive to small text changes
DCT	DCT on $h(c) \forall c \in C$	Few fingerprints	Inefficient with many text changes
Fuzzy	Reformulates $d$ by means of fuzzification	Reduced dimension of $d$	Compromised recall quality
Randomized	Randomized construction of hash functions	Reduced dimension of $d$	Compromised recall quality

Table 2.1: Comparison of fingerprinting techniques based on their selection heuristics. Note that  $d$  refers to a document,  $c$  and  $C$  are a chunk and the set of chunks, respectively

Intuitively, the n-gram method outperforms the other methods in terms of accuracy, however, it is the worst in terms of the number of fingerprints generated. The DCT fingerprinting was rated second in terms of accuracy with smaller number of fingerprints. The winnowing algorithm showed a very good performance given a relatively small window size [4]. Table 2.1 summarizes the published results about the main characteristics of the fingerprinting techniques in terms of performance, number of fingerprints generated, and their drawbacks. Note that in the table,  $d$  refers to a document,  $c$  and  $C$  are single chunk and set of chunks, respectively.

## Chapter 3: System Architecture and Text Analysis for Reuse Detection

### 3.1 Introduction

In this chapter, we present a high level overview that outlines the general framework of DETRA, our proposed system for the **D**etection of **T**ext **R**euse of Arabic documents on the Web. The detection process of text reuse can be divided into two major sub-processes. Namely: candidate source documents retrieval, and pairwise similarity comparison between the input document and the candidate source documents. More processes are required in order to achieve the desired results. The general problem of finding text reuse on the Web can be accomplished in four main phases: Web-based candidate documents retrieval, text pre-processing and representation, fingerprinting-based detection, and results presentation. The framework is depicted in Figure 3.1. In what follows, we describe each phase in detail.

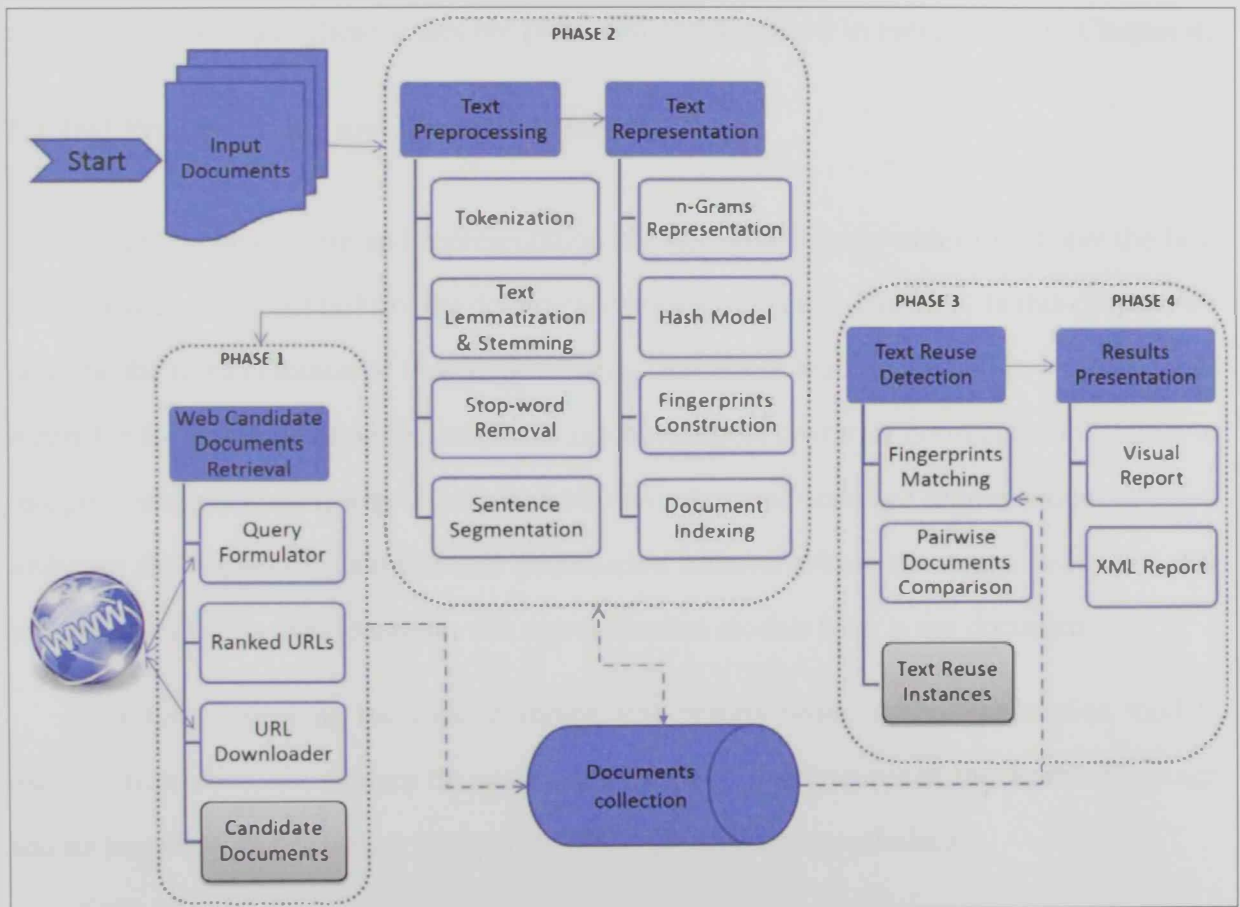


Figure 3.1: Framework for Web-based Arabic Text Reuse Detection

### 3.2 Web-based Candidate Documents Retrieval

In phase one, a document is given as input. The document undergoes a number of preprocessing and representation operations in order to issue a set of queries that globally cover the text in the document. In response to the given set of queries, the system (DETRA) retrieves an initial set of candidate documents from the Web.

The queries are formulated through the query formulator module in order to acquire a promising set of candidate documents that may include text reuse instances. These Web documents are ranked based on their containments of the queries terms. The resulting URLs of the retrieved Web documents are downloaded from the Web to form as documents collection necessary for the next phases of the reuse detection problem. The main stages of this phase include document's signature generation, query formulation, and query submission and documents download. These stages are presented and described in more details in Chapter 4.

### 3.3 Text Pre-processing and Representation

Text preprocessing and representation are essential steps in order to achieve the best performance in both subtasks of the document retrieval and reuse detection. In this chapter, we describe the most common of text preprocessing techniques that are applied to prepare documents for the detection of reused text such as tokenization, character normalization, synonym recognition/replacement, stop-word removal, stemming, and sentence segmentation. We also look into the impact of diacritics and punctuation removal within the Arabic text. We also address some of the most common text representation models for a given document.

Before describing the most common text preprocessing and representation models used for text reuse, we explore the main characteristics and features of the Arabic language and its impact on the accuracy and quality of the preprocessing techniques.

### 3.4 Arabic Language Characteristics

The Arabic language is a rich morphological language that is among the mostly used languages in the world. However, very little attention is given to this language in the area of text reuse detection. While the local text reuse and plagiarism detection problems have been mostly studied in the literature for Western languages using local corpora, very few publications studied the text reuse and plagiarism detection problem using Arabic documents. Furthermore, we confidently state that there is no available previous work on text reuse detection for Arabic-based documents on the Web. However, few research works ([45, 44, 46, 71]) focused on Arabic-based documents on relatively small collections and are particularly tailored for plagiarism detection.

The Arabic language has high specificity than other languages that makes one of the most complicated languages. One of the vital differences of the Arabic language when compared to Western languages is that it is classified as a VSO (Verb Subject Object) language [72], with respect of word order. In fact, the Arabic language may be considered as a free word-order language, i.e., the word order of the grammar can be easily rearranged without affecting the correctness and the completeness of the sentence. On the other hand, Arabic has a rich morphology and word derivations. As for word derivation in Arabic, many words with different meanings may stem from a single Arabic word. The richness of word derivations of Arabic as well as the unique characteristic of being a free word-order language requires more attention when applying text reuse detection techniques on Arabic text. Below, we list some of the main features of the Arabic language as in [73] and how they may affect text reuse detection techniques.

- |                 |   |
|-----------------|---|
| أكل الذئب الغنم | • |
| أكل الغنم الذئب | • |
| الذئب أكل الغنم | • |
| الغنم أكل الذئب | • |
| الغنم الذئب أكل | • |

Figure 3.2: Different valid word reordering of the same sentence in Arabic



1. Arabic consists of 28 letters and more than 30 other characters that represent diacritics, punctuation marks and numbers. On the other hand, Arabic letters has to be linked together in a such a way that depends on the words they form. These features are still not well-handled by some platforms that are not fully Arabic enabled. For example, some systems may display Arabic words with un-connected letters, which makes it difficult to read. This problem may affect text reuse detection technique in the pre-processing stage, if the proper coding system is not used to render Arabic text.
2. The writing direction of the Arabic text is the opposite of the Latin text. Particularly, it starts from right to left. Adjusting the writing direction of the text to the right side has been resolved in most computer platforms, however, we still experience incorrect representation of the Arabic text in many applications. This issue might affect defining the correct bounds of passages/paragraphs in Arabic documents and makes it complicated to later analyze these passages/paragraphs for text reuse detection.
3. Arabic is very flexible in terms of words order. This particular feature of the Arabic language may pose a major problem in the text reuse detection techniques, since none of the known text reuse techniques handle this issue of word reordering and may easily fail in detecting a match between documents with possible text reuse. See Figure 3.2 that shows different reordering of the same sentence, which are all valid sentences in Arabic. The list shows the following equivalent orderings: VSO, VOS, SVO, OVS, and OSV.

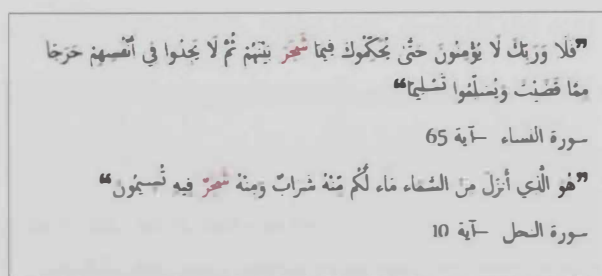


Figure 3.3: Two similar words in Arabic شجر (shown in red) that share the same root. However, the two words have different meanings based on the context. The first means dispute, while the second means trees

4. Arabic is a heavily inflectional language. This makes it among the complicated lan-



guages when analysing it morphologically (see Figure 3.3). Arabic words are created from roots rather than stems. Many words may share the same root and differ only in the type of diacritics used, which gives it a totally different meaning. However, diacritics in Arabic are optional and are usually omitted in the Arabic text, which causes a big ambiguity in differentiating between such words. Consequently, a word with length  $n$  may have at least  $2^n$  different forms with and without diacritics [1] (e.g., the Arabic word علم in Figure 3.4), which, again, may cause ambiguity and failure in detecting a match using text reuse detection techniques if no language-dependent tools are applied in the pre-processing phase such as stemming and text normalization.

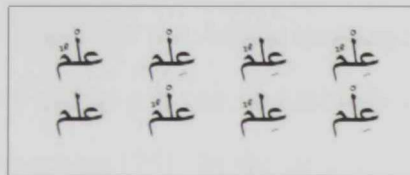


Figure 3.4: Different representations of the same word علم (ilm - knowledge) based on the presence or absence of diacritics

- In Arabic, there are morphemes that resemble a word but are bound to other words. In grammar, they are referred to as clitics [74]. For example, the Arabic word أطعمتمونيها in Figure 3.5 can be considered as a complete sentence. Unbinding clitics produces a longer sentence as shown in the second example (أطعمتم إياي الفاكهة) of the figure. This feature of the Arabic language may pose great difficulty in finding a match between two words with very simple edits in the text such as binding a pronoun to a verb instead of being as a separate word. Note that, most of the text reuse techniques require text normalization and stemming prior to fingerprinting, which will eventually omit the bound pronoun from the text causing a missing part in the sentence and hence failure to find a match for text reuse in some cases.

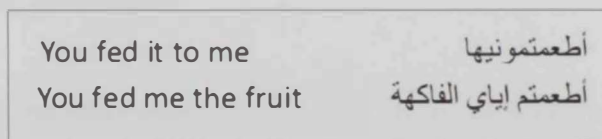


Figure 3.5: A complete sentence formed in Arabic by binding clitics to the verb. Unbinding some of the clitics produces longer sentences as in the second example

6. The inconsistent use of punctuation marks in the Arabic text presents another challenge when defining sentence boundaries. In fact, it is kind of acceptable in Arabic to write an entire paragraph that may consist of more than 80 words without any punctuation marks that indicates the end of that paragraph [75].

Further information on the Arabic language structure, characteristics and preprocessing challenges is found in [73, 75, 76]. Given these major characteristics of the Arabic language, the question that may arise is how well the available fingerprinting techniques will perform on Arabic text? It is obvious that the efficiency of these techniques may be affected by these unique features of the Arabic language and different heuristics must be applied to accommodate with the high ambiguity of the Arabic language. It is worth mentioning that the degree of ambiguity to properly define a token in Arabic is eight times higher than in other languages as estimated by researchers [75]. In the next section, we describe the main techniques used to normalize the Arabic text and prepare it for the reuse detection task. Figure 3.6 shows the step by step preprocessing operations.

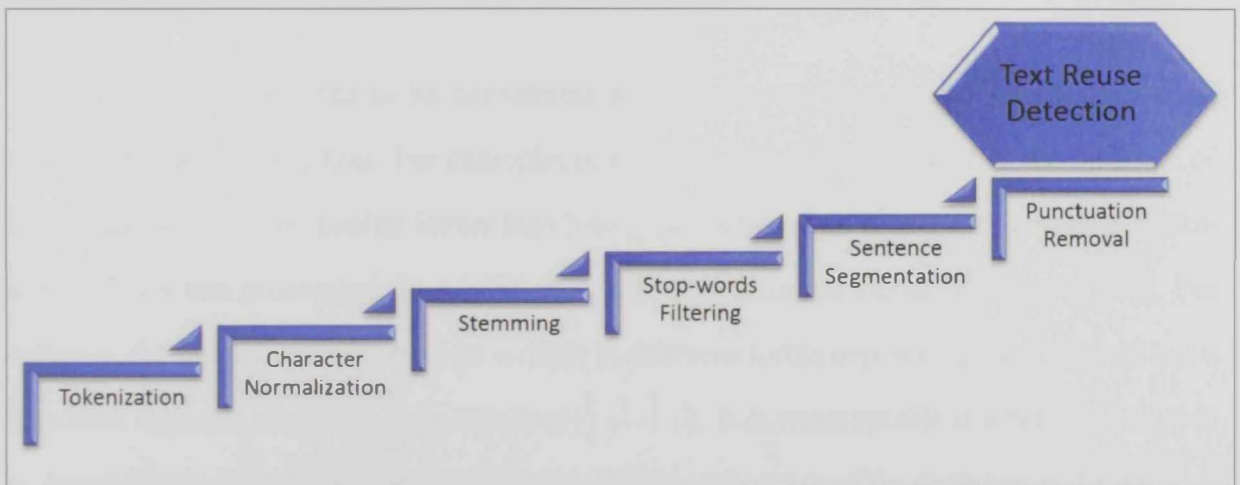


Figure 3.6: Text preprocessing scheme applied before text reuse detection

### 3.5 Text Preprocessing Techniques

Text preprocessing techniques have a significant impact on the efficiency of many Natural Language Processing (NLP) tasks in general, and text reuse detection task in particular. Some preprocessing techniques may improve the accuracy of the systems, while other

techniques may reduce the dimensionality of the problem in terms of decreasing the number of features that need to be handled and hence enhancing the time requirement. Many existing pre-processing techniques may be adopted before actually detecting for cases of text reuse. In what follows, describe the most common preprocessing techniques applied for text reuse detection and describe the most common text representation models for a given document.

### 3.5.1 Tokenization

According to Grefenstette et al. [77], a token is identified as a number, punctuation, date or a word. It is the smallest meaningful unit in a text that can be morphologically analyzed. The tokenization process is concerned with dividing the text into tokens. As mentioned earlier in Chapter 1, The only challenge in this task is differentiating between some punctuation marks whether they are part of a word or not such as the period (.) in the abbreviated words (e.g., .د) and the end of sentence marker as in (جامعة الإمارات.).

### 3.5.2 Character Normalization and Diacritics Removal

Some aspects need to be considered at the character level in order to eliminate insignificant features of a text. For example, in English, case folding, which is the opposite of capitalization, converts capital letters into lower-case letters and it is a very common operation in IR for text processing. In Arabic, character normalization works slightly different. For instance, the letter (أ - "alif") may be written in different forms depending on whether it has a hamza or a madda above or a hamza below (ا، آ، إ). It is unacceptable if a system is unable to determine that أجمل and أجمل are the same word just because of the presence or the absence of the hamza (ء). Another example is the taa-marbouta letter (ة - 'taa-marbouta') at the end of a given word (غالية), in which its two dots above may be omitted as in (غاليه), while they both refer to the same word. Therefore, determining one general form of letters is a needed process in order to reduce the ambiguity caused by the inconsistent use marks [76]. For instance, converting all forms of the letter alif (ا، آ، إ) into one form (ا). Another aspect in character normalization in Arabic text is diacritics. As mentioned before, diacritics help in the correct

pronunciation of Arabic words as well as in differentiating the meanings of words that have the same form. Removing diacritics is a common practice IR for processing Arabic text. A risk exists though after diacritics removal when considering two different words as the same such as the word (سحب) after diacritics removal from سَحَبَ (means pulled) and سُحُبٌ (means clouds). Both words have the same form, while are totally different in meanings, which can not be distinguished without diacritics and may require to look at the adjacent words in order to identify the word's meaning. Nevertheless, since analyzing a document for reuse considers words within their context, It is safe to normalize the Arabic letters by discarding diacritics.

### 3.5.3 Stemming

Stemming is the process of removing the affixes from a given word. Note that two different words could end up having the same stem, which might lead to conflicting results, however, the impact of stemming lies on the reduction of the resulting word-space used for the text reuse analysis. For many IR related problems, it is better to use stemmed words rather than considering its entire form. The reason is that suffixes are considered irrelevant and need to be discarded [78].

Researches have analyzed the effect of stemming in IR differently. For instance, According to a study addressed by Krovetz [79], stemming proved to enhance the results with relatively small documents, and the performance degrades with larger documents. On a different study done by Baeza-Yates et al. [80], they claim that stemming may lead to some conflicting results and the overall impact on performance is small. However, there is no evidence of a negative effect on the retrieval performance when using a reasonable stemmer [81]. After all, when text reuse detection is considered based on our empirical study, stemming performs well and is might be a necessary step especially in the case of applying synonym recognition and substitution as will be discussed later in this thesis.

When Arabic language is concerned, a number of Arabic stemming algorithms have been developed in the literature [82, 83]. One of the earliest works on Arabic stemmers was the one developed in 1994 by Al-Kharashi and Evens [84]. The created a stemmer that maps

Arabic vocabulary terms to their corresponding stems/roots using a manually constructed lexical dictionary. However, this approach is obviously impractical for large sized corpora. Another effective stemmer proposed by S. Khoja in 1999 [85], which automatically strips off Arabic words from their affixes and then maps the truncated words to roots dictionary to find their stems/roots. It also identifies a set of words as Arabic stop-words (see the list in Figure 3.8). However, it has a limitation with stemming names and some modern Arabized words, which leaves them unmodified. Figure 3.7 list the set of prefixes and suffixes in the Arabic language. It is worth mentioning that most of the available Arabic stemmers are root-based stemmers. Explicitly, it returns the word to its base root, mostly trilateral form. However, few works have focused on light stemming, which might be more efficient than root-based stemmers depending on the application used, especially when no prior morphological processing is carried out. To mention some, the works proposed by Aljlal et al. [86], Darwish et al. [87], and Larkey et al. [88]. Light stemming maintains words' infixes while removing the prefixes and suffixes. However, most of the available Arabic light stemmers are not reliable enough and have high errors rate. Other tools such as ElixirFM [89] and MADAMIRA [90] have been recently developed that incorporates Arabic morphological analysis to produce better results for a set of valuable Arabic features including stemming. The later application provides linguistic features such as tokenization, diacritization, lemmatization, part-of-speech tagging that can help in the analysis of the Arabic text depending on the application being used. Tim Buckwalter's morphological analyzer [91] is another morphological analyser but different from the others in that it returns stems rather than roots. It is based on a set of lexicons of Arabic stems, prefixes, and suffixes, with truth tables identifying their valid combinations.

When text reuse detection is concerned, using stemming proved useful in enhancing the detection process as considering a stem of a word is better than considering its entire form [92]. However, the choice of using heavy or light stemmer might not have a significant effect on the overall detection process, since analysing a document for reuse implies considering words within their context. Therefore, we chose to adopt Khoja's stemmer as it produces good enough result for the detection process. Besides, it is developed in JAVA and freely



available for research purposes.

Arabic suffixes list			Arabic prefixes list
ا	كم	هما	ا
ي	كن	هم	ل
ان	نا	هن	لل
ات	تا	ه	و
ها	ما	ة	سي
وا	ته	ون	ب
	تي	ين	فا
	ني	تما	ي
	ن	تم	ن
	ك	تن	م
	ت	كما	ت

Figure 3.7: The set of prefixes and suffixes in the Arabic language, which are stripped off by the stemming algorithms

### 3.5.4 Stop-words Filtering

Stop-words filtering is defined as the elimination of non-informative, high frequency words with little meaning that are considered irrelevant from the documents' representations [78]. The removal is mainly performed using a stop-words list (also called negative dictionary). The main reasons behind stop-words removal are: (1) Stop-words have little or no semantic values, which makes them impractical to include in any query. (2) Eliminating stop-words, being the most frequent words in a document, has an important effect on the dimensionality reduction of the space required to represent documents since stop-words may occupy nearly one third of the document's size and discarding them removes approximately half the words in a document [92]. This certainly enhances the overall performance of the system.

Hiemstra [93] differentiates between two types of stop lists based on whether the words are non-informative regardless of the frequency value they carry or words that have high frequency values but are considered practically useless to keep. Another aspect in stop-words removal is that some words are domain independent and some may be domain dependent. It is a language dependant preprocessing operation similar to stemming and considered an important tool in IR related applications such as similarity detection tasks.



In the text reuse detection task, which is the focus of this thesis, stop-words filtering has a great impact in reducing the index size in the fingerprinting process as well as optimizing queries with more informative terms. A special benefit of stop-word removal is experience in queries formulation for documents retrieval on the Web (detailed in Chapter 4). This is due to the fact that researchers are only allowed for a limited number terms per query to retrieve candidate documents through a Web search engine via a public search API. We adopted the stop-words list that was suggested by [85] (see the list in Figure 3.8).

أن	وهو	كذلك	عن	ليست	لكنه	أنه	كان	فيه	أل
بعد	يكون	التي	الذي	وكانت	ولكن	تكون	لهم	ذلك	ومع
ضد	به	وبين	والتي	أي	له	قد	لأن	لو	فقد
بلى	وليس	فيها	منذ	ما	هذا	بين	اليوم	عدد	بل
إلى	أحد	عليها	أما	عنه	فقط	جدا	لم	الذين	منه
في	على	إن	حين	حول	ليس	لن	هؤلاء	كل	بها
من	وكان	وعلى	ومن	تكون	ثم	نحو	مساء	بد	وفي
حتى	تلك	لكن	لا	مع	هذه	هو	فإن	لدى	فيهو
تحت	عليه	هنا	كانت	يوم	هل	ما	ما زال	ما برح	صار
لها	لدي	وقد	هناك	منها	حيث	لا	أسمى	ما فتى	وليست
أو	كما	لذلك	قبل	إلى	هي	ما يزال	أضحى	ما انفك	إن
إذا	كيف	أمام	سعه	إذا	و	أصبح	ظل	بات	وهذا
كان	لاسيما	ولا يزال	ضمن	وله	أي	إليها	الذين	وإن	لهذا
ليت	لعل	الحالي	أول	ذات	بدلا	إنه	فاته	وأبو	الا
فكان	ستكون	معا	أبو	بين	وإن	إليه	لك	يمكن	بهذا

Figure 3.8: The stop-words list used in the stop-words removal preprocessing operation

### 3.5.5 Sentence Identification and Segmentation

One of the main issues in the text reuse detection task is how long a text should be in order to consider it as a reused text between two documents? Most of the text reuse detection methods deal with a sentence as a relevant unit for the detection process. Therefore, sentence identification and segmentation are generally recommended as one the early steps of the detection process [92]. Sentence identification (also referred to as sentence boundary detection) is the process of defining sentences boundaries within a subsequence set of words in a given document. Sentence segmentation, on the other hand, is the operation of splitting the text into the set of coherent sentences. This particular preprocessing operation is essential in many IR and NLP related applications such as parsing, document summarization, translation, and obviously text reuse detection. In this thesis, recognizing a sentence boundary does not only depend on the usual sentence delimiters (such as question mark (?), exclamation mark (!) or a

period .), but also a sentence should not exceed a certain threshold. It has been noted by NLP researchers that 37 words is the average length of an Arabic sentence, which is a high value when compared to the other languages [73]. As for English, for instance, the average sentence length is about 22 words per sentence [94]. Therefore, in the implementation of this work, we addressed the problem of sentence identification by using a fixed sentence length approach, which divides a document into a set of equal-sized sentences, in which the best sentence length can be determined empirically (see Chapter 5 for more details). This approach is the simplest in terms of complexity. However, one limitation of this approach is that the sentences may be segmented at different locations than the actual sentences boundaries. Nevertheless, this limitation is neglected since we are using an overlapping fingerprinting technique in representing the document which overcomes this limitation as will be discussed in Chapter 5.

### **3.5.6 Punctuation Removal**

Discarding punctuation marks is a very common preprocessing procedure in IR and NLP. However, in very rare cases, punctuation may play a role in text reuse detection. More precisely, in author's style recognition in the intrinsic type of plagiarism detection, where authors may use punctuation at certain patterns in their writing styles [78].

## **3.6 Text Representation Techniques**

After pre-processing, the documents' texts have to be represented in a certain way that aids the efficient detection of text reuse. Below, we describe a number of the most commonly used text representation models in IR in general, and text reuse detection in particular.

### **3.6.1 Bag of Words Representation**

The bag of words (BoW) is one of the most frequently used representation models in IR. It considers a document as a bag of all the words it contains, neglecting any syntactic information that comes with the composition of phrases and sentences in the document [95]. It is worth mentioning that this simple representation offers good results when dealing with

topical similarity as in general IR. However, higher levels of representations are required for the detection of text reuse such  $n$ -Grams representation.

Word n-gram				n
المتحدة	العربية	الإمارات	جامعة	1
العربية المتحدة	الإمارات العربية	جامعة الإمارات		2
الإمارات العربية المتحدة	جامعة الإمارات العربية			3
جامعة الإمارات العربية المتحدة				4
Character n-gram				n
ج	ا	م	ع	1
جا	ام	مع	عة	2
جام	امع	معة	عتا	3
جامع	امعة	معتا	عتال	4
جامعة	امعتا	معتال	عتال	5
جامعة	امعتا	معتال	عتال	6

Figure 3.9: Word and character  $n$ -grams on Arabic text for different values of  $n$

### 3.6.2 $N$ -Grams Representation Model

An  $n$ -gram is a well-known model in the fields of computational linguistics and probability. It is basically composed from a contiguous sequence of overlapping  $n$  units from a given text. These units can be phonemes, characters or words depending on the application [95]. The overlapping of units is performed in such a way that the last element in an  $n$ -gram is the first one in the following element. An  $n$ -gram model is typically used for predicting the next item in a sequence in the form of a  $(n - 1)$ -order Markov model [78]. In the case of text reuse,  $n$ -grams can be considered at word or character level. For instance, the units in word  $n$ -grams are the tokens in the text. See Figure 3.9 for an example of word  $n$ -grams applied on the Arabic text (جامعة الإمارات العربية المتحدة). On the other hand, character  $n$ -gram considers  $n$  contiguous characters to be the units to combine. See Figure 3.9 for an example of character  $n$ -gram on the Arabic phrase (جامعة الإمارات). There are two main features of this model, the first is its simplicity, and the second is its ability to efficiently scale up by simply increasing  $n$ .

It is worth noting that the BoW model described earlier is an instance of a word  $n$ -gram with  $n = 1$ , which provides good results for topic similarity applications. Low values of word  $n$ -grams provides more flexibility in the comparison and in capturing the syntactic information of the text, which may help detect cases of reuse with high rewriting levels. However, this comes with the cost of retrieving too many false negatives. On the other hand, high levels of word  $n$ -grams may be ideal for detecting cases of verbatim copies. However, the cost is that they are very sensitive and missing cases with very few modifications [92]. In Chapter 5, we address this particular aspect in more details in the carried out empirical study of selecting a value for  $n$  and its impact on the evaluation of the detection task in general.

### 3.6.3 Hash Model

In order to efficiently perform the text reuse detection and avoid the direct comparison between strings, which is computationally expensive, documents' texts are represented in a numerical form using specially designed hash models. The purpose of a hash model is to map a string unit in a document (for instance, a word, a sentence, or  $n$ -gram) into a numerical value using a hash function. This numerical value is stored in a hash table along with the other numerically converted strings of the document to allow for the efficient search for similar contents between documents. When looking for text reuse cases, the input document (or passage) is converted by means of the same hash model and queried against the hash table. If a match occurs, two exact text units (a word, a sentence,  $n$ -gram, ...etc) have been discovered. The most commonly used hashing functions are MD5 [50] and Rabin fingerprinting [96].

The MD5 hashing function is a hexadecimal function that produces long hash values. The hash values are significantly changing with a change of a single character between two strings, which makes it very suitable for searching for exact duplicates. Besides, the resulting values are very long that might not be the best choice for hashing short  $n$ -grams. On the other hand, Karp-Rabin hashing function is a more solid decimal model and more adjusted to text strings, and yet ensures a low collision probability. Therefore, it is highly popular for documents processing methods [92]. The Karp-Rabin hashing function is the one adopted in

the implementation of this work.

Intuitively, the hash models can also be applied to one of the above mentioned text representations in order to lower the space required for comparison and to speed up the process.  $N$ -grams, sentences, or text segments from an input document can be hashed and saved into a hash table. The candidate source documents that are suspected to have similar content with the input document are also hashed using the same hash model and then the hash table is queried to look for a match and hence indicate similarity between the input and the suspected document.

### **3.7 Local Text Reuse Detection and Text Alignment**

In the third phase, a sentence-level retrieval by means of fingerprinting is performed on the segmented documents set. Similarity measures are applied at this phase in order to identify instances of text reuse between the query document  $d_q$  and the indexed source documents collection. Fingerprinting techniques and similarity measures were discussed in details in Chapter 2. The output of this phase consists of a list of sentences/passages weighted by their similarity scores along with references to the documents from which the similar sentence/passage was located.

### **3.8 Post-processing and Presentation**

The presentation is the last phase in DETRA system. The resulting similar instances detected in the text reuse detection phase along with the documents from which they were located both form a rich presentation for the user to track the reuse flow. One possible presentation is a list of the matching instances with the input document colored differently to distinguish the reuse text based on their similarity scores. See Figure 3.10.

### **3.9 Implementation Details**

In this section, we describe the implementation aspects of this research.



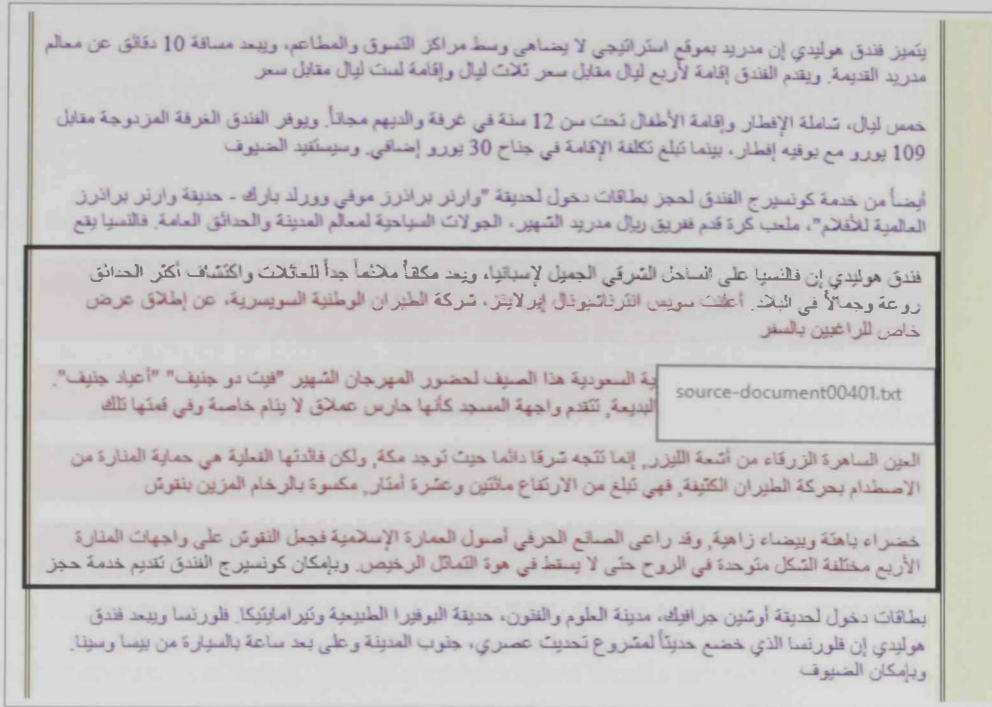


Figure 3.10: A sample of the visual output presenting a reused text case (enclosed in a box)

The implementation of the developed system (DETRA) had been done using JAVA as a programming language. Several libraries have been used in the development of DETRA in order to perform the desired tasks. As an input, the user is allowed to query a single document or a set of documents at once for text reuse detection. Our system DETRA accepts various file formats such as .txt, .doc, .docx, .ppt, .pptx, .pdf, .htm, .html, and .xml as input documents. All of which are converted using a special converter module into a plain text with .txt file format. We used the following external JAVA libraries to perform this task:

- **org.apache.poi** library for reading .doc, .docx, .ppt, .pptx file formats.
- **org.apache.pdfbox** library for reading and extracting the text from .pdf file formats.
- **org.jsoup.Jsoup** library for parsing and extracting text from .htm and .html file formats.
- **javax.xml.stream.XMLInputFactory** and **javax.xml.stream.XMLEventReader** to strip off and extract meaningful text from .xml file format.

The developed system DETRA has the flexibility to allow the user to detect text reuse over a selective set of documents, or check for text reuse over an already constructed corpus



of indexed documents, or surf the Web looking for candidate source documents. The source documents selected (or downloaded from the Web) for the detection task are preprocessed and indexed according to the fingerprinting technique proposed in this research. Indexing the source documents into the corpus, and later on searching that index are carried out using Apache Lucene API [97], which is a rich and powerful full-text search library that easily allows for the scalability of DETRA to accommodate large documents collection such as the Web. Lucene indexer is used for constructing an index of the documents collection, and then Lucene search engine can be used to lookup the pre-built index and answer the issued queries. It is worth noting that we have tuned Lucene search engine indexer to suit the fingerprinting-based text reuse detection task in that it accepts a set of  $n$  words (a fingerprint) as an index term rather than just one word. These  $n$  consecutive words are normalized using Karp-Rabin hashing model. On the other hand, to accomplish the “local” notion of text reuse, the search engine retrieves the matching documents at the sentence-level rather than just associating each index with its corresponding documents that contain this index.

Figure 3.11 displays the graphical user interface (GUI) of the developed DETRA system for text reuse detection of Arabic text. The main components of the interface are numbered from 1 to 6 to left of each component. These component are:

1. Input Folders: the input document(s) that we wish to detect for text reuse are defined at this location by choosing a single document file, or a set of input documents. An optional feature of selecting a different documents collection to detect reused text over them is provided in this current component.
2. Text Reuse Indexing and Detection: by pressing on the Indexing button, the currently selected documents collection will be preprocessed and indexed using Lucene Indexer. If no selected documents collection has been provided, the available corpus is used instead by default. The selected documents collection will go through preprocessing operations and text representations prior to indexing. The operations in component 5 dictates whether to include the selected documents collection to the local corpus or not. The Text Reuse Detection Button performs the required sub-tasks required to perform

the detection process and produce the output that shows any reused text in the input document(s) as well as the computed similarity measure of reuse.

3. Web Retrieval, Indexing and Detection: this component is concerned with the Web documents retrieval to construct a Web-based documents collection for the detection task. The upper bound parameters provided are the query length, the number of queries per document, and the number of Web documents retrieved for each query.
4. Preprocessing and Query Related Parameters: this component contains the required settings for the applied fingerprinting technique. These include the size of the  $n$ -gram, the window size, and the sentence length. The implemented fingerprinting methods DETRA are the winnowing, hailstorm, and our proposed method.
5. Corpus Options: This component of DETRA allows for adding the selected documents collection to the available corpus, search for reuse in the selected collection only, or reconstruct the index of the corpus.
6. System Status: This part of the interface shows the current status of DETRA at each stage.

Each input document, and before it the indexed documents in the corpus, undergo a number of text preprocessing operations as described in Section 3.5. We used Khoja's stemmer for discarding the word's affixes and the suggested stop-words list by [85]. As for the sentence segmentation process, we used a fixed sentence length approach, which divides a document into a set of equal-sized sentences. The text representation models selected are word  $n$ -grams and Karp-Rabin hashing function.

The result of the detection process is a list of the input documents along with the percentage of text reuse in each document using the similarity measure described in Chapter 2. Figure 3.12 displays an example of the output report generated for each queried document. Each document's name is linked to a page that displays the detailed similarity scores, highlighting the reused text and providing the details of the source document that shares the common text with.

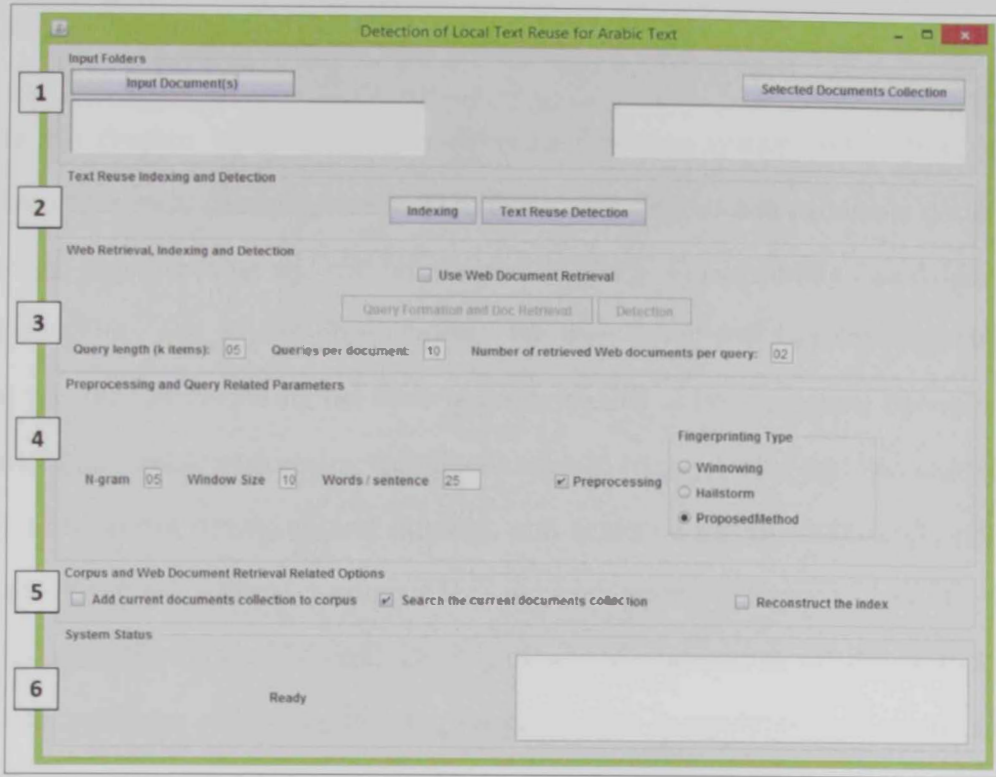


Figure 3.11: The graphical user interface of the proposed system

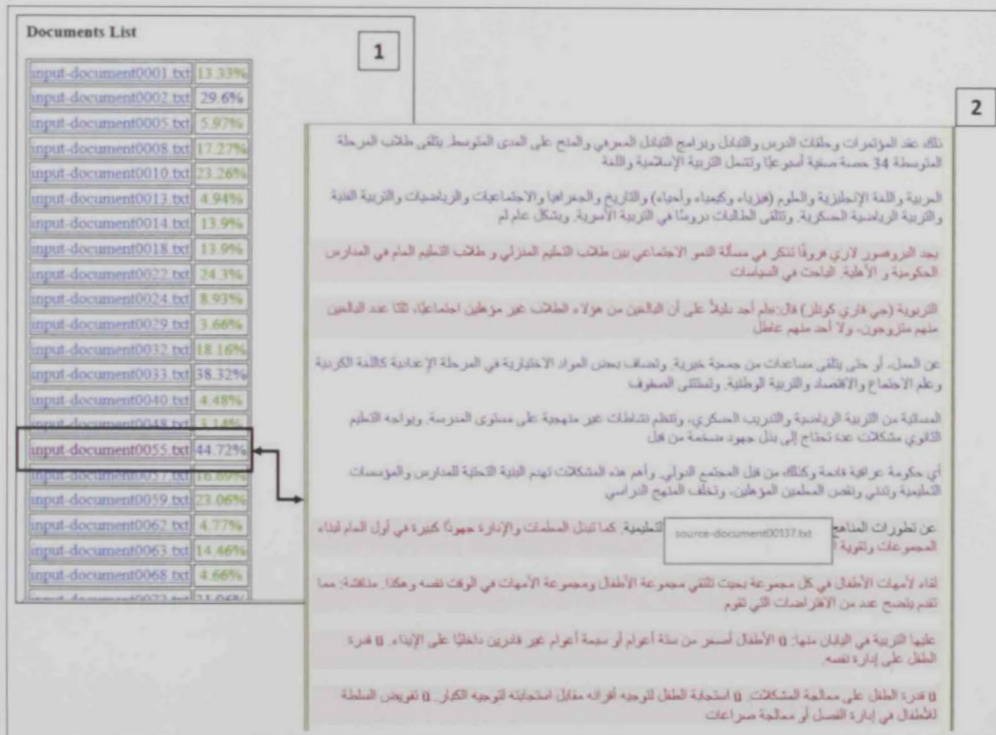


Figure 3.12: An example of text reuse detection of a 2-pages report for a set of input documents. Page (1) displays a list of the input documents examined for text reuse instances. The second page (2) shows a detailed sample of reused text instances for input-document0055.txt and the corresponding source document from which it shares reused text

### 3.10 Chapter Summary

In this chapter, we described the text reuse detection system architecture, which can be split into four main generic phases. The phases are: Web-based candidate documents retrieval phase, preprocessing and text representation phase, fingerprinting based detection and comparison phase, and results presentation. We then paved the way for the indexing and detection methods by preparing the documents by means of preprocessing operations. First, we discussed the text preprocessing techniques applied on the Arabic text including character normalization, words stemming and filtering, and sentence identification and segmentation operations. Next, we defined a number text representation strategies to identify the document's representative terms. This includes bag of words, character or word  $n$ -gram and finally the efficient technique of hashing that helps speed up the computation and can be applied on the different representations mentioned in the chapter along with some implementation details of the proposed system.

## Chapter 4: Candidate Document Retrieval from the Web

### 4.1 Introduction

Given an input document  $d$ , the problem of local text reuse detection is to detect from a given documents collection, all the possible reused passages between  $d$  and the other documents. Comparing the passages of document  $d$  with the passages of every other document in the collection is obviously infeasible especially with large collections such as the Web. Therefore, selecting a subset of the documents that potentially contains reused text with  $d$  becomes a major step in the detection problem. In the setting of the Web, the search for such candidate source documents is usually performed through limited query interface. Moreover, using the current search engines querying interface is perhaps the only available way for researchers to implement text reuse detection against the web, whereas certain fees are charged by the search companies for limited and controlled usage [98]. This chapter describes a new efficient approach of query formulation to retrieve Arabic-based candidate source documents from the Web. The candidate documents are then fed to a local text reuse detection phase for detailed similarity evaluation with  $d$ . We consider the candidate source document retrieval problem as an essential step in the detection of text reuse, which is the focus of this chapter. We evaluated the work using a collection of documents especially constructed for the evaluation of Web documents retrieval in particular and text reuse detection in general. The experiments show that on average, 79.97% of the Web documents used in the reused cases were successfully retrieved.

In this chapter, we present an efficient strategy to formulate queries by extracting a set of representative consecutive terms from a given document that form as a signature (or fingerprint) to that document in order to query a Web search engine via a public search API. The set of queries are selected in such a way that ensures a global coverage of the document. The resulted list of Web documents identified by their URL addresses are then downloaded from the Web and pass through a pre-processed step to prepare them for the detection task. The purpose of this phase is to construct a relatively small but useful dataset that contains



most of the similar instances with the given document. This phase comprises phase one in the overall DETRA system as shown in Figure 4.1.

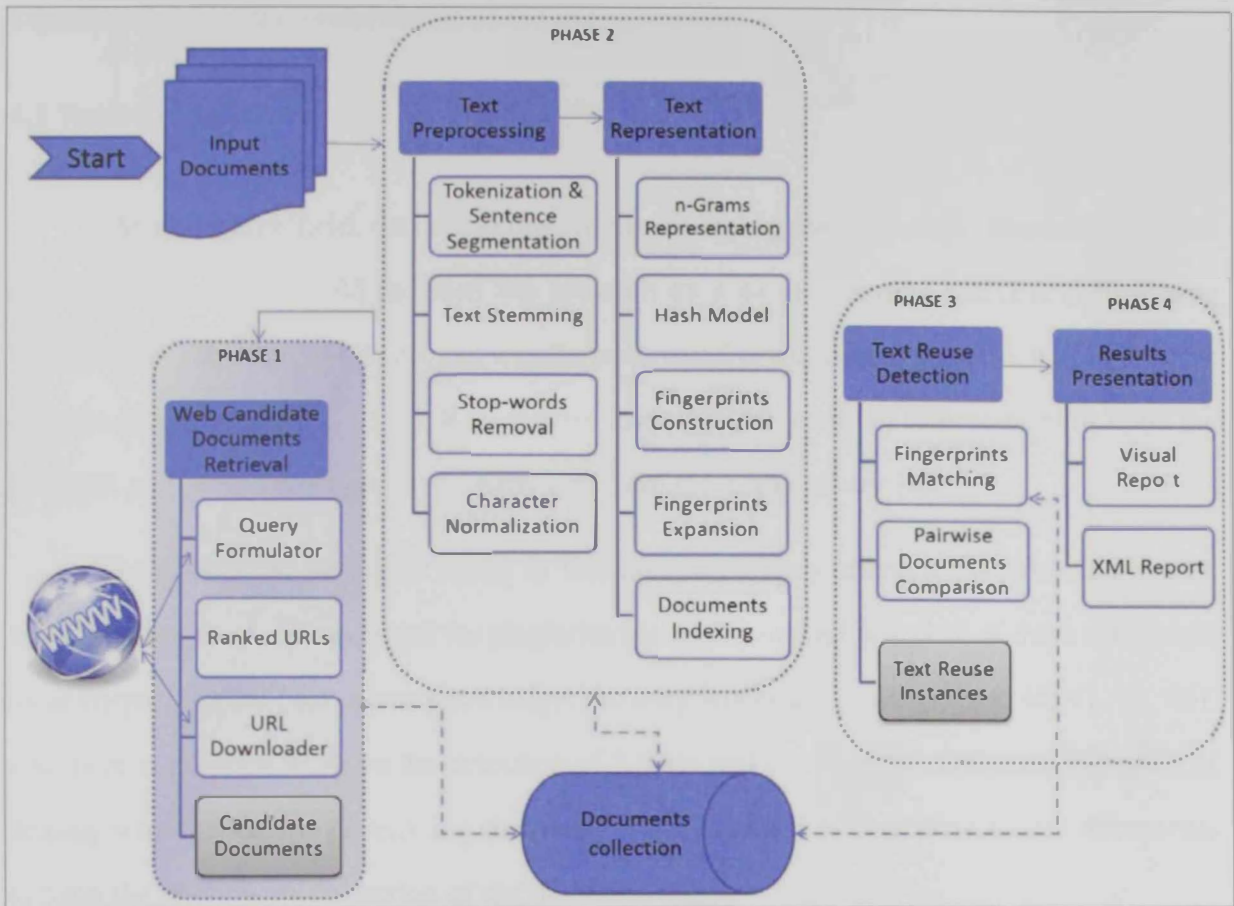


Figure 4.1: The proposed framework of a Web-based text reuse detection for Arabic documents (DETRA). Phase 1 in the DETRA is shaded in blue

Our focus is on generating query signatures for a given document for Web querying using a standard search engine query interface to identify potential similar documents as the Web has become the typical place of text reuse. However, text reuse detection on the Web raises many challenges such as the scale of the Web and the heterogeneity of the documents in terms of the data formats and languages. Another vital challenge is the very limited or no access to a Web search engine's index. Moreover, queries are not issued for free as Web search engines entail different costs and limits on the number of search queries to be issued at a certain period of time [98].

In the next section of this chapter, we introduce the related research of Web document retrieval in the literature. In Section 4.3, we describe the Arabic-based candidate Web docu-



ment retrieval model for the text reuse detection problem. Section 4.4 shows the experiments of two different results. The first empirically computes the best query length and the second experiment shows the performance of the overall model.

## 4.2 Related Literature

As a research field, text reuse has received considerable attention. Researchers from the University of Sheffield initiated the research on a project named METER (MEasuring TExt Reuse). The aim of this project was the automated detection of text reuse within specific domain of journalism ([12, 13]). Researchers from the University of Lancaster built upon the METER project to detect reuse in journalism from the 17th century [2].

Very recently, very few works in this area have been oriented towards Arabic text, however, they were all conducted for plagiarism detection in Arabic and all of them used small local corpus. To the best of our knowledge, the only works are those of ([44, 45, 47, 48, 46]) and there is no prior work on the detection of Arabic text on the Web. The main approach of dealing with the Web is developing methods for the retrieval of candidate source documents to form the documents collection of the detection task.

In what follows, we review the related works on query formulation for candidate document retrieval on the Web for English text. One of the initial works on query formulation is the work of Pôssas et al. [99]. They used maximal termset query formulation bounded by the number of returned results. Bendersky and Croft [2] have recently suggested a framework for detecting text reuse on the Web for English documents. Their candidate document retrieval phase focused on selecting and weighting key concepts related to the statement of interest. However, the practical applicability of the suggested text reuse detection system on the Web has not been implemented. On the other hand, their focus is on Web querying based on single sentences and not on whole documents, as it is the case in our approach, and hence their strategy is quite trivial.

Dasdan et al.'s developed a strong query maker model to find similar documents through a search engine interface [100]. They considered the coverage testing problem that

tests whether a document is contained in a search engine's index or not. They proposed a randomized approach for query generation based on the shingling approach proposed by Broder et al. [6] and proved that their approach finds similar documents.

Hagen et al. [101] proposed a capacity-constrained query formulation heuristic, which is inspired by the maximal termset formulation strategy developed in [99]. However, they produce larger collections of documents when compared to Dasdan et al.'s strategy who retrieve fewer similar results for a given document.

Some other strategies that involve the computation of term frequencies or tf-idf weights have been developed in the literature for query formulation or other related problems and can be found in ([102, 103, 104]). Note that tf-idf score is a standard metric measure that corresponds to the frequency of a term in a given document over its frequency in all the documents of a given corpus [105].

Our method is equivalent to Dasdan et al.'s work in such a way that it takes advantage of the shingling approach and the proximity of terms feature in the query formulation strategy. However, our proposed strategy is further improved by providing a global coverage to the terms in the input document with fewer number of queries. Moreover, our technique is tuned to Web querying Arabic-based documents since the available public APIs does not directly operate with Arabic text and requires different preprocessing of the text. The details of the proposed strategy are described in the next Section 4.3.

To the best of our knowledge, the only framework proposed on query generation for Arabic documents on the Web is in [106]. They proposed three heuristics for an Arabic plagiarism detection system that involve frequency based keyword selection, variance in readability, and selecting first words in each paragraph's heading sentence. They empirically evaluated their heuristics using their own private corpus. However, the practical applicability of their suggested plagiarism detection system on the Web has not been implemented.

### 4.3 Our Proposed Document Retrieval Model

#### 4.3.1 Notation and Problem Definitions

Given a document  $d$  and using a query interface for a Web search engine  $QI$ , the set  $Q = \{Q_1, Q_2, \dots, Q_q\}$  of queries, where each query  $Q_i = \{t_1, t_2, \dots, t_w\}$  consists of  $w$  terms. Our proposed shingle-based query formulation strategy for Arabic text reuse detection task is defined as follows:

**Given:**

1. Input document  $d$  with a set  $W$  of  $n$  terms after preprocessing.
2. Query interface for Web search engine  $QI$ .
3. Upper bound  $q$  on the number of queries to be generated.
4. Upper bound  $w$  on the number of terms per query.
5. Upper bound  $k$  on the number of returned documents per query by  $QI$ .

**Task:** Generate and select  $q$  queries, where each query  $Q_i \in Q$  contains  $w \in W$  terms and returns at most  $k$  documents from the Web using a public search engine query interface.

The question that may arise is why do we need query generation from  $d$ ? It is well known that a Web search engine query interface does not accept large text such as entire documents or even passages as a query. Instead, it accepts queries with limited set of keywords, measures the similarity between these keywords and the indexed documents in its Web pages database, and then returns the most similar ones. In this work, we still need to retrieve documents from the Web, however, the query in our case is not keywords, but the entire document. This is why, query formulation for a given document is needed. The query formulation mechanism selects representative set of sentences from the query document as sub-queries and submits them to a search engine interface. The result of this step is a set of documents identified by their URLs that share some text with the queried document and are downloaded from

the Web to form the documents collection necessary for the detailed text reuse detection task.

Intuitively, selecting a set of scattered terms from the document or choosing highly frequent terms within the document would probably lead to low quality results. This is due to the fact that we are after *local* text reuse and not duplicate or near-duplicate candidate documents, where their sets of terms might be almost the same. The selection of the terms in a document for a given query has to be obtained in sequence preserving their proximity as it will be described in the proposed method.

Note that the query formulation and generation strategy is constrained with a number of parameters imposed by the Web search engine interface. These constraints are:  $w$  and  $q$ . Therefore, our strategy to generate a set of queries  $Q = \{Q_1, Q_2, \dots, Q_q\}$  for a given document  $d$  has to take these parameters into account, while maximizing the global coverage of the text in  $d$ . The proposed query formulation is inspired by shingling and fingerprinting techniques and has the following properties:

1. A query consists of  $w$  consecutive words from  $d$ , which preserves the proximity of the terms.
2. Queries disjointness. The queries are disjoint in such a way that  $Q_i \not\subseteq Q_j$  where  $Q_i$  and  $Q_j$  are members of the set of queries  $Q$  and  $i \neq j$ . This property minimizes the duplications in the results.
3. A query  $Q_i \in Q$  does not return more than  $k$  documents.
4. The Union of all the generated queries provides total coverage to all the terms in  $W \in d$  if possible. Precisely:  $\{\bigcup_{Q_i \in Q} Q_i = W, i \geq 1\}$ .
5. The selected queries from  $Q$  should not exceed the upper bound  $q$  queries.

Our proposed approach is new, efficient, and simple and does not require any required privileged access to the search engine's index. It randomly selects  $q$  queries of  $w$  consecutive terms each from  $d$ . However, in order to insure the generation of the same set of queries for the same document  $d$  each time, a selection strategy is performed (discussed in Section 4.3.2).

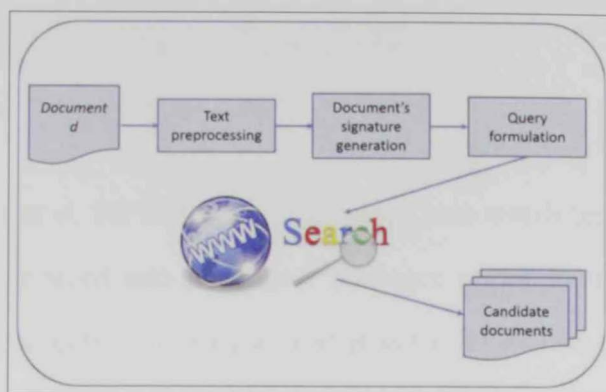


Figure 4.2: The building blocks of candidate document retrieval from the Web

### 4.3.2 Model Specification

Our developed model of the candidate document retrieval from the Web is depicted in Figure 4.2. It consists of the following stages:

1. The text in  $d$  has to undergo a preprocessing step. The original document  $d$  is tokenized, segmented into a set of sentences, and stemmed after omitting stop words and punctuation marks. The Arabic words are further normalized.
2. The second stage is computing the document's signature (fingerprints). The fingerprints of a document  $d$  can be considered as a set of encoded sequence of terms from  $d$  into a numeric form, which serve as a unique representation of the document  $d$ .
3. The third stage formulates  $q$  queries from the generated fingerprints of  $d$ .
4. The Web is then searched, through a public Web search engine interface  $QI$  using the generated set of queries for  $d$ . A set of candidate documents that are likely to contain reused text with  $d$  are collected and retrieved from the Web. The downloaded candidate documents are then prepared for the detailed similarity analysis by the text reuse detection system.

It is worth mentioning that the implemented model extends the basic flow to accept a set of input documents instead of a single document at a time. The model can also operate with



different documents formats.

### **Text Preprocessing**

Given a document  $d$ , the text is first tokenized into words/terms and then each set of consecutive words are grouped into sentences. Sentence segmentation is considered another important process since a sentence can be looked at as a relevant unit for a text reuse detection system. In our system DETRA, each sentence is preprocessed individually and it is further split into a set of overlapping shingles based on a predefined integer value. In this work, the integer value is  $w$ . A  $w$ -shingle as described in [35] is defined as “the set of all contiguous sequences of  $w$  terms in  $d$ ”.

Different text preprocessing models are applied at this phase. Text representation is a key factor when processing a document. This includes diacritics, punctuation and stop-words removal, stemming and letter normalization. Note that in Arabic text, some Arabic letters look almost identical in shape and are only distinguished between each other by adding dots, a hamza or madda above or below the letter. Arabic letter normalization is a needed process and should take place after stemming in order to reduce the ambiguity caused by the inconsistent use of diacritics and marks.

### **Document's signature generation**

For each passage in a given document  $d$ , the shingling approach basically converts every subsequence of  $w$  overlapping terms from  $d$  into a numerical form using especially designed hash model such as Rabin fingerprints [6]. The document's fingerprints are generated and extracted from a selective set of shingles. The rationale of the different selection algorithms of the fingerprinting techniques is to minimize the overall selected set of shingles that serve as representatives of the document  $d$ , and to maximize the chance to retrieve the most promising candidate documents that are likely to share text with  $d$ . This step is probably the most crucial step in the document retrieval problem since the generated and selected

---

**Algorithm 1** Efficient Random Query Formulation Algorithm
 

---

**Input:**  $W : \{t_1, \dots, t_n\}$  set of  $n$  tokens in  $d$ ,  $w$ : query size,  $q$ : number of queries.

**Output:**  $Q = \{Q_1, Q_2, \dots, Q_q\}$ 

```

1:  $i \leftarrow 1$ 
2:  $min \leftarrow 1$ 
3: while ( $i < n$ ) do
4:    $Q_{cand} \leftarrow \{t_i, \dots, t_{i+w-1}\} \subseteq W$ 
5:    $S \leftarrow \bigcup_{t_j \in Q_{cand}} \{h_j : hash(t_j), h_j \subseteq \mathfrak{R}\}$ 
6:    $min \leftarrow find\_min\_index(S)$ 
7:   if ( $(min = i) \text{ or } (min = i + w - 1)$ ) then
8:     if  $Q_{cand}$  is valid then
9:        $Q \leftarrow Q \cup \{Q_{cand}\}$ 
10:       $H \leftarrow H \cup \{rabin\_hash(Q_{cand})\}$ 
11:     end if
12:   end if
13:    $i \leftarrow i + w$  ▷ shift to the next shingle
14: end while
15:  $pick\_min\_queries(Q, H, q)$ 

```

---

fingerprints directly affect the final set of queries.

The proposed fingerprinting-technique provides a global coverage of the words in the document and improves the selection algorithm in such a way that it reduces the generated set of fingerprints. The algorithm guarantees the property of context-freeness as well. Specifically, selecting shingles as fingerprints only depends on the tokens that comprise the shingles and is not affected by the other shingles in the document.

### Query Formulation

Since we are after the formulation of queries in a textual form, we map the shingles in its numeric format back into its corresponding  $w$  consecutive words it was generated from. For this reason, we keep records of the mapping between the numeric shingles and their equivalent sequence of  $w$  terms in the document  $d$ , since Rabin shingling is a one way hash.

The selection of the document's queries is performed by picking the first  $q$  fingerprints after sorting their corresponding hash values in ascending order and then mapping them back to their textual form. This step gives the strategy a random notion in the sense that selecting

the  $q$  fingerprints depends only on their hash values.

A solution to the Random Query Formulation is shown as Algorithm 1. The algorithm first forms a candidate shingle with a subsequence of  $w$  tokens from  $d$ , hashes its tokens separately (Line 5), and then checks if the shingle contains a token with the minimum hash value at the first or last position of the shingle's  $w$  tokens. A shingle is then considered as a valid query if its textual mapping satisfies property-2 mentioned earlier in Section 4.3.1.

The combined family of queries generated by the algorithm provides a global coverage of all  $W \in d$ . Explicitly, it covers all the set  $W$  of terms in  $d$ :  $\{\bigcup_{Q_i \in Q} Q_i = W, i \geq 1\}$ .

To compute the time complexity of the algorithm, the while loop runs at most  $O(n)$  times, where  $n$  is the set of tokens in a document. As for the sort, the merge sort algorithm is known to run in  $O(n \log n)$  time. Therefore, the overall complexity of the algorithm will be of  $O(n \log n)$ .

Note that there may be cases where it is not possible to completely cover  $W$  with the generated set of valid queries (e.g., when the size of the query is small (i.e.,  $w$ ) with a few number of queries ( $q$ ) to be selected). A term  $t \in W$  is covered if and only if there is a valid query  $Q_i \subseteq W$  with  $t \in Q_i$ .

The queries are converted into a different representation other than Arabic characters in order to optimize the retrieval process by returning more relevant results than the original query would have. This is due to the fact that Arabic characters are not properly addressed by most of the public Web search APIs and submitting the query in Arabic may not return proper results. The generated queries are converted into UTF-8 encoding system, which is a readable form by the search engine interface  $QI$ .

### Query Submission and Documents Download

The aim of this stage is to submit the formulated set of queries  $Q$  for the given input document  $d$  to the public Web query interface  $QI$  of the target search engine and retrieve the

returned  $k$  results in order to construct the candidate documents collection for  $d$ . We developed a sub-system using Google's custom search API, which accepts queries as input, and submits them to a query builder module that pre-processes the Arabic queries and converts them to an acceptable query format according to the Web search engine interface. The queries are then returned to a search-hit module, which obtains the list of URL addresses that answers the query. The returned URL addresses are further post-processed to filter out some additional characters and symbols that are generated with Arabic-based Web documents which hinders the proper retrieval and download of the respective Arabic Web documents.

The downloaded set of candidate documents has to be prepared for the detailed analysis phase in the detection system. That is, a clean-up process is needed to filter out unnecessary HTML tags or images and keep only the plain text. The document is also converted into one common file format (UTF-8) that matches the input documents.

## 4.4 Experimental Results

### 4.4.1 A Case Study

Figure 4.3 shows part of the proposed system DETRA, which is dedicated for Web retrieval. The parameters used are the number of terms per query, the number of total queries returned for the given document and the number of Web documents per query.

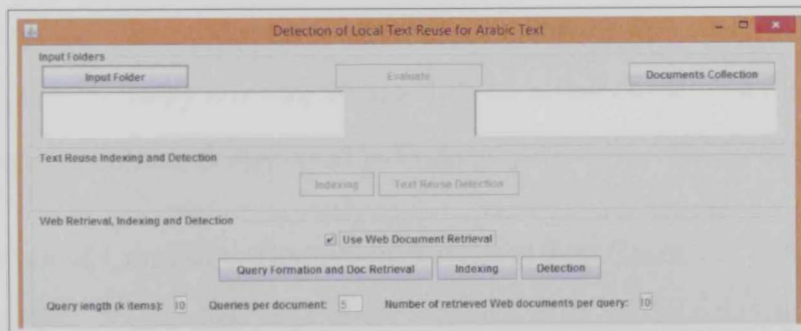


Figure 4.3: The part of DETRA's system interface for Web documents retrieval task

The following example shows the main steps of the proposed Web document retrieval task. Figure 4.4 shows a sample document with some content that has been copied from other

Web documents. Each colored bounding box indicates a different Web document from which the text has been borrowed. Following the steps for the query formulation, and setting the required upper bounds of the total number of queries to 10, and retrieving the top 2 returned documents by each query, we get the following set of queries selected randomly based on their computed hash values as in Figure 4.5. Note that the mapping between the numeric shingles and their equivalent sequence of terms in the document is illustrated in Figure 4.4 in red. First, the document is preprocessed using the different preprocessing operations. Next, the document's signature based on the fingerprinting approach is computed and a selection of the  $q$  minimum fingerprints, as described in Algorithm 1, is performed.

The selected 10 queries from the text are converted to special characters in order to be accepted by the search engine interface, since Arabic text is not well-received by the search engine interface, and hence may not return any results if passed with in the original shape. The encoded queries are shown in Figure 4.5.

Figure 4.6 displays the returned top two ranked Web documents' URL addresses that share the text in the query (or parts of it) with the input document. The returned URL addresses are filtered and repeated URL addresses are removed from the list. Next, the Web documents identified by the URL addresses are acquired from the Web and downloaded. A clean-up process is performed to extract the plain text in the downloaded HTML documents and reformatted into text document to prepare it for the detection task. The result of the detection task shows the URL addresses of the Web documents that share text with the input document with high similarity rate (see Figure 4.7). A visual output that displays the reused content in the input document is displayed in Figure 4.8.

#### **4.4.2 Construction of Candidate Document Retrieval Test Cases**

For the sake of validating the proposed approach, we manually developed a set of test-cases in Arabic for this purpose. The constructed Arabic documents collection is gathered from the Web, and test cases were created by reusing passages from the Web documents. Our collection consists Arabic documents collected from Alkhaleej news, Al Watan news, and



البحوث والدراسات بالخطاطم بالخطاطم بعد المقفمة ذات الاهمية للتأسيس لموضوع الورشة و هدفها و التي قمناها في مقال سابق نلاحظ  
 سعى الامم المتحدة الى خلق الكوادر العلمية لديها لضمان وجود ذراع واق يقي الامة من شرور الفقر والتخلف والحرمان وسيطرة الاجنبي  
 وتحسين شعوبها ضد جميع انواع الامراض الاجتماعية والجنسية والنفسية وغيرها والتي تؤدي الى ضعف الانتاجية وانخفاض معدلات  
 النمو والتنمية الاقتصادية والمستدامة والسماحة باستخدام لغة الارقام كونها لغة الاكوياء فيحسب تقزير المؤسسات الدولية مثل البنك الدولي  
 وبرنامج الامم المتحدة الانماني نخص نجد ان عند العلماء والمهندسين والاختصاصيين والباحثين وفي مجال البحوث والتنمية ز نكل مليون  
 نسمة

فيما (اف ب): طالب الوفد العراقي الذي شارك في الاجتماع الوزاري لمنظمة الدول المصدرة للنفط (اوبك) في فيينا الاربعة بتقديم  
 ضمانات للمستهلكين بشأن التجارب مع هضمهم على النفط وذلك بعد اعلان اوبك الابقاء على قرار تخصيص الانتاج ابتداء من الاول من  
 ابريل. وقال الوفد العراقي في بيان صدر على هامش الاجتماع ان اوبك مدعوة الى تحقيق توازن في السوق ليس فقط على الصعيد المادي  
 بل ايضا على الصعيد النفسي. و اضاف ان هذا يتطلب تقديم ضمانات قوية للمستهلكين بان اوبك ستجواب مع الطلب على النفط وان قرار  
 تخفيض سقف الانتاج مخصص لتخفيض الانتاج الزائد في الاسواق. وتجاهلت منظمة الدول المصدرة للنفط صغوط الولايات المتحدة من  
 اجل اتخاذ قرار يسهل في تخفيض اسعار النفط المرتفعة جدا. وتمسكت بقرار منخذ سابقا ويقضي بتخفيض الانتاج بنسبة مليون برميل يوميا  
 اعتبارا من الاول من ابريل، ليصل الى 5.23 مليون برميل في اليوم. وقال العراق انه ينظر بصورة ايجابية الى هذا القرار ولو انه لا بد  
 من انتظار وضعه موضع التنفيذ، مشيرا الى ان انعكسات اقرار تشير الى نتائج مهمة حتى الان. وراى البيان ان المستهلكين يمكن ان  
 يشعروا بمزيد من الثقة في حال تجنبت المنظمة الادلاء بتصريحات عنيفة حول تغييرات جديدة في سقف الانتاج او رفع الاسعار التي حدثت  
 اوبك هامشها بما بين 22 و28 دولارا للبرميل منذ 1999 ..

مفط العماني: تشارك المنظمة خلال شهر ابريل الحالي في عدد من الاجتماعات التي ستعقد بالامانة العربية لدول مجلس التعاون لدول  
 الخليج العربية بالرياض. صرح بذلك مصدر اقتصادي ان السلطة ستشارك غدا وبعد غد المقبل بالرياض في اجتماع اللجنة الفنية لمنقشة  
 قانون التأمين الموحد لدول مجلس التعاون مشيرا الى انه سيمثل السلطة في الاجتماع عبدالله الشلبي مدير دائرة شؤون مجلس التعاون  
 وسالم المنطري مدير دائرة التأمين بوزارة التجارة والصناعة. وقال المصدر في تصريح لوكالة الانباء العمانية انه سيتم خلال الاجتماع  
 مناقشة قانون التأمين الاسترشادي واخذ ملاحظات الدول الاعضاء بدول المجلس للتوصل الى قانون تأمين موحد يطبق في دول مجلس  
 التعاون لدول الخليج العربية.

عقدت ندوة تجارة وصناعة عمان صباح امس جلسة المباحثات التجارية بين السلطة وتونس تراس الجانب العماني سماعة الشيخ عبدالله بن  
 سالم الرواس رئيس العرفة فيما تراس الجانب التونسي معالي الصالح بن جمعة رئيس لجنة نادي تونس للمصندين رئيس الوفد. وفي بداية  
 اللقاء رحب سماعة الشيخ عبدالله بن سالم الرواس بالوفد الضيف مشيرا الى ان هذه الزيارة تعتبر فرصة سانحة للتعرف على فرص وحواقر  
 الاستثمار المشتركة المتوفرة في السلطة وتونس حيث تربط البلدين الشقيفين روابط اخوية وثيقة منذ زمن قديم وقد عزز التعاون تأسس  
 اللجنة العمانية التونسية المشتركة التي ساهمت في نمو التعاون في العديد من المجالات الاقتصادية والثقافية والاجتماعية والصحية الا انه  
 يعتقد بان المبادرات التجارية بين الجانبين ما زالت نون الضموح رغم توفر مقومات ازدهارها ونموها و اوضح الشيخ رئيس العرفة بان  
 اسباب تدهن التبادل التجاري بين السلطة وتونس تعود الى عدم وجود خطوط ملاحية مباشرة بين البلدين وقلة وسائل النقل وكذلك قلة  
 المشاركة في المعارض التجارية المتخصصة التي تقام في تونس ومفط اضافة الى ارتفاع التعريفات الجمركية المفروضة على بعض  
 الواردات التونسية. عقدت اللجنة الاقتصادية بمجلس الشورى امس اجتماعها السنوي السادس لدراسة الامتداد السنوي الاول (2003  
 2004م) من الفترة الحامسة برئاسة سماعة بن سعيد الغمامي رئيس اللجنة وبحضور اصحاب استعدادات اعضاء اللجنة والامين العام  
 المساعد للجلسات والتجان. وفي الاجتماع اقرت اللجنة بياتها الخاص بدراسة موضوع الاحتكار التجاري خاصة في قطاع السيارات  
 وخماتها ومستلزماتها على ان تطرح هذا البيان على جلسة المجلس القادمة لاعتماده وقد تضمن اسباب ومبررات اختيار الموضوع من بين  
 عدد من الموضوعات الاقتصادية التي تم طرحها وبحثها في اجتماعات اللجنة السابقة. وقد جاء اختيار اللجنة لهذا الموضوع لعدة اعتبارات  
 منها انه اصبح يشغل بل واهتمام قطاع واسع من المجتمع والربيطه بارتفاع الاسعار في هذا القطاع على وجه الخصوص وشمولية  
 التأثيرات الاجتماعية والاقتصادية على مستوى الاقتصاد الكلي ومختلف شرائح المجتمع.

مفط العماني: تشارك السلطة في الاجتماع السادس والثلاثين للجنة مدافعي مؤسسات النقد والبنوك المركزية بدول مجلس التعاون لدول  
 الخليج العربية الذي سيعقد في تولة الكويت غدا ويتراس وفد السلطة الى الاجتماع سماعة حمود بن سنجور اللزجالي الرئيس التنفيذي للبنك  
 المركزي العماني. وسوف يناقش الاجتماع عددا من المواضيع التي تنطوق بالاشراف والرقابة على الجهاز المصرفي وكذلك موضوع  
 الاتحاد النقدي بين دول المجلس وموضوع يضم المصفوعات

والمنظمة المستمرة من قبل المستشفيات والمرافق الصحية من بداية الحمل وحتى اكتمال سنوات الطفولة، والتحصينات والمراقبة الصحية  
 اعطت نتائجها المباشرة في انخفاض عدد الوفيات بين المواليد، وهي ايضا تعطي تأكيدات ومعاني اخرى الى الوعي والاندراك بالهمية  
 المعادة بين الولادات واعضاء المرأة الراحة والفترة الكافية لرعاية وتربية اطفالها وتضمتهم التنشئة الصحية الصحيحة

Figure 4.4: An example of a document with text copied from other Web documents. Each reused text is displayed in a bounding box. The different colors indicate different source documents. Note that the representative fingerprints of the document are displayed in red

- Query # 1: مناقشة+قانون+التأمين+الموحد+لدول+مجلس+التعاون+مشيرا+سيمتل+السلطنة:
  - UTF-8 Encoding:  
 %D9%84%D9%85%D9%86%D8%A7%D9%82%D8%B4%D8%A9+%D9%82%D8%A7%D9%86%D9%88%D9%86+%D8%A7%D9%84%D8%AA%D8%A3%D9%85%D9%8A%D9%86+%D8%A7%D9%84%D9%85%D9%88%D8%AD%D8%AF+%D9%84%D8%AF%D9%88%D9%84+%D9%85%D8%AC%D9%84%D8%B3+%D8%A7%D9%84%D8%AA%D8%89%D8%A7%D9%88%D9%86+%D9%85%D8%B4%D9%8A%D8%B1%D8%A7+%D8%B3%D9%8A%D9%85%D8%A8%D9%84+%D8%A7%D9%84%D8%B3%D9%84%D8%B7%D9%86%D8%A9
- Query # 2: و اوضح+الشيخ+رئيس+الغرفة+بيان+اسباب+تنسي+التبادل+التجاري+السلطنة:
  - UTF-8 Encoding:  
 %D9%88%D8%A7%D9%88%D8%B6%D8%AD+%D8%A7%D9%84%D8%B4%D9%8A%D8%AE+%D8%B1%D8%A6%D9%8A%D8%B3+%D8%A7%D9%84%D8%8A%D8%B1%D9%81%D8%A9+%D8%A8%D8%A3%D9%86+%D8%A7%D8%B3%D8%A8%D8%A7%D8%A8+%D8%AA%D8%AF%D9%86%D9%8A+%D8%A7%D9%84%D8%AA%D8%A8%D8%A7%D8%AF%D9%84+%D8%A7%D9%84%D8%AA%D8%AC%D8%A7%D8%B1%D9%8A+%D8%A7%D9%84%D8%B3%D9%84%D8%B7%D9%86%D8%A9
- Query # 3: وضعه+موضع+التنفيذ+مشيرا+الاعكاسات+القرار+تتسير+نتائج+مهمة+الآن:
  - UTF-8 Encoding:  
 %D9%88%D8%B6%D8%B9%D9%87+%D9%85%D9%88%D8%B6%D8%B9+%D8%A7%D9%84%D8%AA%D9%86%D9%81%D9%8A%D8%B0+%D9%85%D8%B4%D9%8A%D8%B1%D8%A7+%D8%A7%D9%86%D8%B9%D9%83%D8%A7%D8%B3%D8%A7%D8%AA+%D8%A7%D9%84%D9%82%D8%B1%D8%A7%D8%B1+%D8%AA%D8%B4%D9%8A%D8%B1+%D9%86%D8%AA%D8%A7%D8%AC+%D9%85%D9%87%D9%85%D8%A9+%D8%A7%D9%84%D8%A2%D9%86
- Query # 4: تتعلق+بالاتراف+والرقابة+الجهاز+المصرفي+موضوع+الاتحاد+التقدي+نول+المجلس:
  - UTF-8 Encoding:  
 %D8%AA%D8%AA%D8%B9%D9%84%D9%82+%D8%A8%D8%A7%D9%84%D8%A7%D8%B4%D8%B1%D8%A7%D9%88%D8%A7%D9%84%D8%B1%D9%82%D8%A7%D8%A8%D8%A9+%D8%A7%D9%84%D8%AC%D9%87%D8%A7%D8%B2+%D8%A7%D9%84%D9%85%D8%B5%D8%B1%D9%81%D9%89+%D9%85%D9%88%D8%B9+%D8%A7%D9%84%D8%A7%D8%AA%D8%AD%D8%A7%D8%AF+%D8%A7%D9%84%D9%86%D9%82%D8%AF%D9%89+%D8%AF%D9%88%D9%84+%D8%A7%D9%84%D9%85%D8%AC%D9%84%D8%B3
- Query # 5: ايضا+الصعيد+النفسى+واضاف+يطلب+تقديم+ضمانات+قوية+للمستهلكين+مبن:
  - UTF-8 Encoding:  
 %D8%A7%D9%8A%D8%B6%D8%A7+%D8%A7%D9%84%D8%B5%D8%B9%D9%8A%D8%AF+%D8%A7%D9%84%D9%86%D9%81%D8%B3%D9%8A+%D9%88%D8%A7%D8%B6%D8%A7%D9%81+%D9%8A%D8%AA%D8%B7%D9%84%D8%A8+%D8%AA%D9%82%D8%AF%D9%8A%D9%85+%D8%B6%D9%85%D8%A7%D9%86%D8%A7%D8%AA+%D9%82%D9%88%D9%8A%D8%A9+%D9%84%D9%84%D9%85%D8%B3%D8%AA%D9%87%D9%84%D9%83%D9%8A%D9%86+%D8%A8%D8%A7%D9%86
- Query # 6: السلطنة+خلال+شهر+ابريل+عدد+الاجتماعات+تتعدد+بالاتمامة+العربية+للدول:
  - UTF-8 Encoding:  
 %D8%A7%D9%84%D8%B3%D9%84%D8%B7%D9%86%D8%A9+%D8%AE%D9%84%D8%A7%D9%84+%D8%B4%D9%87%D8%B1+%D8%A7%D8%A8%D8%B1%D9%8A%D9%84+%D8%B9%D8%AF%D8%AF+%D8%A7%D9%84%D8%A7%D8%AC%D8%AA%D9%85%D8%A7%D8%B9%D8%A7%D8%AA+%D8%B3%D8%AA%D8%B9%D9%82%D8%AF+%D8%A8%D8%A7%D9%84%D8%A7%D9%85%D8%A7%D9%86%D8%A9+%D8%A7%D9%84%D8%B9%D8%B1%D8%A8%D9%8A%D8%A9+%D9%84%D8%AF%D9%88%D9%88
- Query # 7: وسقط+اضافة+ارتفاع+التعريف+الجمركية+المفرصة+بعض+الواردات+التونسية+عنفت:
  - UTF-8 Encoding:  
 %D9%88%D9%85%D8%B3%D9%82%D8%B7+%D8%A7%D8%B6%D8%A7%D9%81%D8%A9+%D8%A7%D8%B1%D8%A9%D9%81%D8%A7%D8%B9+%D8%A7%D8%B8%D8%AA%D8%B9%D8%B1%D9%8A%D9%81%D8%A9+%D8%A7%D9%84%D9%85%D9%81%D8%B1%D9%88%D8%B6%D8%A9+%D8%B8%D8%B9%D8%B6+%D8%A7%D9%84%D9%88%D8%A7%D8%B1%D8%AF%D8%A7%D8%AA+%D8%A7%D9%84%D8%AA%D9%88%D9%86%D8%B3%D9%8A%D8%A9+%D8%B9%D9%82%D8%AF%D8%AA
- Query # 8: ام+بالخطب+المقدمة+الاهمية+للتأسيس+لموضوع+الورثة+وهدفيها+تتمناها+مقال:
  - UTF-8 Encoding:  
 %D8%A7%D9%85+%D8%A8%D8%A7%D9%84%D8%AE%D8%B7%D8%A8+%D8%A7%D9%84%D9%85%D9%82%D8%AF%D9%85%D8%A9+%D8%A7%D9%84%D8%A7%D9%87%D9%85%D9%8A%D8%A9+%D9%84%D8%AA%D8%A3%D8%B3%D9%8A%D8%B3+%D9%84%D9%85%D9%88%D8%B6+%D9%85%D9%88%D8%B9+%D8%A7%D9%84%D8%A7%D8%AD%D8%B8%D8%A7%D8%AA+%D8%A7%D9%84%D8%AF%D9%88%D9%84+%D8%A7%D9%84%D8%A7%D8%B9%D8%B6%D8%A7%D8%A1
- Query # 9: خلال+الاجتماع+مناقشة+قوتون+التأمين+الاسترشادي+واخذ+ملاحظات+الدول+الاعضاء:
  - UTF-8 Encoding:  
 %D8%AE%D9%84%D8%A7%D9%84+%D8%A7%D9%84%D8%A7%D8%AC%D8%AA%D9%85%D8%A7%D8%B9+%D9%85%D9%86%D8%A7%D9%82%D8%B4%D8%A9+%D9%82%D8%A7%D9%86%D9%88%D9%86+%D8%A7%D9%84%D8%AA%D8%A3%D9%85%D9%8A%D9%86+%D8%A7%D9%84%D8%A7%D8%B3%D8%AA%D8%B1%D8%B4%D8%A7%D8%AF%D9%8A+%D9%88%D8%A7%D8%AE%D8%B0+%D9%85%D9%84%D8%AD%D8%B8%D8%A7%D8%AA+%D8%A7%D9%84%D8%AF%D9%88%D9%84+%D8%A7%D9%84%D8%A7%D8%B9%D8%B6%D8%A7%D8%A1
- Query # 10: واللجان+الاجتماع+اقرت+اللجنة+مبداها+الخاص+مدراسة+موضوع+الاحتكار+التجاري:
  - UTF-8 Encoding:  
 %D9%88%D8%A7%D9%84%D9%84%D8%AC%D8%A7%D9%86+%D8%A7%D9%84%D8%A7%D8%AC%D8%AA%D9%85%D8%B5%D8%B9+%D8%A7%D9%84%D9%84%D8%AC%D9%86%D8%A9+%D8%A8%D9%8A%D8%A7%D9%86%D9%87%D8%A7+%D8%A7%D9%84%D8%AE%D8%A7%D8%B5+%D8%A8%D8%AF%D8%B1%D8%A7%D8%B3%D8%A9+%D9%85%D9%88%D8%B6%D9%88%D8%B9+%D8%A7%D9%84%D8%A7%D8%AD%D8%AA%D9%83%D8%A7%D8%B1+%D8%A7%D9%84%D8%AA%D8%AC%D8%A7%D8%B1%D9%84

Figure 4.5: The set of randomly generated queries for the sample input document in Figure 4.4. The queries are encoded into UTF-8 characters format in order to accept Arabic queries via the Web search engine interface

- Query # 1:
  - URL # 1: <http://www.alwatan.com/graphics/2004/04apr/3.4/dailyhtml/economy.html>
  - URL # 2: <http://www.alwatan.com/graphics/2010/02feb/24.2/dailyhtml/economy.html>
- Query # 2:
  - URL # 3: <http://www.alwatan.com/graphics/2004/04apr/4.4/dailyhtml/economy.html>
  - URL # 4: <http://www.alwatan.com/graphics/2004/04apr/5.4/dailyhtml/economy.html>
- Query # 3:
  - URL # 5: <http://www.alwasatnews.com/news/383502.html>
  - URL # 6: <http://www.alwatan.com/graphics/2004/04apr/2.4/dailyhtml/economy.html>
- Query # 4:
  - URL # 7: <http://elaph.com/Web/Economics/2009/4/426903.html>
  - URL # 8: <http://www.kuna.net.kw/ArticlePrintPage.aspx?id=1382499&language=ar>
- Query # 5:
  - URL # 9: (Repeated)  
<http://www.alwatan.com/graphics/2004/04apr/2.4/dailyhtml/economy.html>
  - URL # 10: <https://www.alyaum.com/article/1163714>
- Query # 6:
  - URL # 11: <http://omandaily.om/?p=398126>
  - URL # 12: <http://www.bna.bh/portal/news/700274>
- Query # 7:
  - URL # 13: (Repeated)  
<http://www.alwatan.com/graphics/2004/04apr/4.4/dailyhtml/economy.html>
  - URL # 14: (Repeated)  
<http://www.alwatan.com/graphics/2004/04apr/5.4/dailyhtml/economy.html>
- Query # 8:
  - URL # 15: [http://www.dxbpp.gov.ae/downloads/mag/21\\_ar.pdf](http://www.dxbpp.gov.ae/downloads/mag/21_ar.pdf)
  - URL # 16: <http://mubasher.aljazeera.net/shares/20124181132806891.htm>
- Query # 9:
  - URL # 17: <http://www.bna.bh/portal/news/746095>
  - URL # 18: (Repeated)  
<http://www.alwatan.com/graphics/2004/04apr/3.4/dailyhtml/economy.html>
- Query # 10:
  - URL # 19: (Repeated)  
<http://www.alwatan.com/graphics/2004/04apr/5.4/dailyhtml/economy.html>
  - URL # 20: <http://2015.omandaily.om/?p=162221>

Figure 4.6: The top ranked Web documents' URL addresses returned by each query



```

<?xml version="1.0" encoding="UTF-8"?>
<document reference="input-document002.txt">
<feature name="text_reuse" this_offset="633" this_length="1145" source_reference="source-
document0091.txt"
source_url = "http://www.alwatan.com/graphics/2004/04apr/2.4/dailyhtml/economy.html"
source_offset="7" source_length="1145"/>
<feature name="text_reuse" this_offset="1784" this_length="636" source_reference="source-
document0139.txt"
source_url = "http://www.alwatan.com/graphics/2004/04apr/3.4/dailyhtml/economy"
source_offset="6" source_length="636"/>
<feature name="text_reuse" this_offset="2426" this_length="991" source_reference="source-
document0203.txt"
source_url = "http://www.alwatan.com/graphics/2004/04apr/4.4/dailyhtml/economy"
source_offset="6" source_length="991"/>
<feature name="text_reuse" this_offset="3442" this_length="748" source_reference="source-
document273.txt"
source_url = "http://www.alwatan.com/graphics/2004/04apr/5.4/dailyhtml/economy.html"
source_offset="24" source_length="748"/>
</document>

```

Figure 4.7: The XML output of the detection task identifying the Web URL addresses of the documents that share text with the input document with high similarity rate

The screenshot shows a software interface with a menu bar (File, Edit, View, Favorites, Tools, Help) and a toolbar. Below the menu is a 'Documents List' pane containing one entry: 'input-document002.txt' with a similarity percentage of 84.95%. To the right is a 'Document's Text' pane displaying Arabic text. A red rectangular box highlights a paragraph in the text, which is also highlighted in red in the original image. The highlighted text discusses the impact of the Iraq war on the global economy and the role of the United Nations in maintaining peace and stability.

Figure 4.8: The visual output of the detection task presenting part of the reused text, which is displayed in red

Arabic wikisource. The topics range from economy, international and local news, culture, religion, and sports (see Table 4.1). Text reuse instances have been prepared for both Web documents retrieval and in-depth similarity analysis of the overall system. With respect to the Web retrieval task, the reused instances were annotated in a log file and organized as pairs of input documents/URL addresses of the source documents from which we reused the text (see Figure 4.7). Therefore, the retrieval task can be defined as: given an input document and a Web search API, the task is to retrieve all URL addresses of the candidate source documents that share text with the input document.

Corpus Characteristics		
General information	Web documents downloaded	25981
	Text reuse cases	1040
Document length	Very short(<1 p)	16 %
	Short (1-10 p)	82%
	Medium (10-100 p)	2%
Cases per document	Short (1-5 cases)	92%
	Medium (6-15 cases)	8%
Documents sources	AlKhaleej News	22%
	Al Watan	75%
	Arabic Wikisource	3%

Table 4.1: Characteristics of the collection for Web documents retrieval

We evaluated the proposed strategy in two different ways: The best query length, and the percentage of the retrieved documents used for composing the input document. In our first experiment, we empirically studied the best query length by varying the value of the parameter  $w$  that defines the number of terms per query, which should not exceed the upper bound constrained by the search engines. The longest query length considered is 10 terms, which is the maximum limit set by most Web search engines. The results are validated by checking the URL addresses returned by the search engine API and comparing them with the URL addresses of the Web documents from which the input documents reused parts of its content.

The second experiment was performed to measure the quality of the formulated queries. Especially, the ability of the proposed approach to retrieve the documents used for composing the input documents.



### 4.4.3 Best Query Length

The first experiment aimed at finding the best query length in terms of the number of terms per query ( $w$ ). The maximum limit known by most search engines is 10. We used a small portion of the corpus with 278 documents to choose the best query length ( $w$ ) due to the limitation imposed by search engines, which restricts the amount of queries issued per day. We fixed the other parameters, such as the number of queries to be generated per document ( $q$ ) to 10 after experimenting with different values of  $q = 5, 10, 15, 20$ . The other parameter is the number of top Web documents to be downloaded from the returned results of the search engine ( $k$ ). For simplicity, we chose only the top 5 results after experimenting with  $k = 2, 3, 5, 10$ . In the experiment, we started with  $q = 3$  since the lower values of  $q$  are often resulting in high recall and very low precision scores. We observe from the experiment that the best results were obtained with queries consisting of 10 terms as shown in Figure 4.9. Thus, this value has been selected for the next experiment.

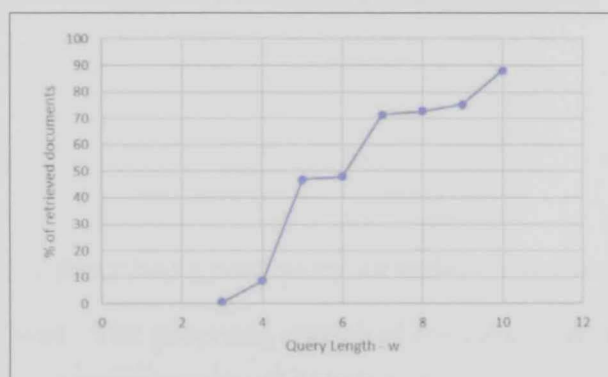


Figure 4.9: Comparing the different query lengths. The best result obtained with  $w = 10$  terms per query

### 4.4.4 Quality of the Formulated Queries

To measure the quality of the formulated queries, the input document's signature (fingerprints) is obtained. Each fingerprint (shingle) selected as a valid query is preprocessed and formulated for submission to the search engine query interface  $QI$  using Google's search engine API. The URL addresses of the returned documents for the submitted queries are compared to the URL addresses of the Web documents used in the composition of the test

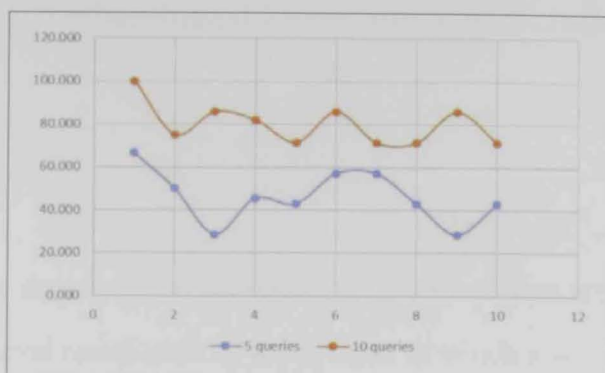


Figure 4.10: The quality of document retrieval mechanism (precision)

cases (input documents). The percentage of the source documents retrieved by the submitted queries is computed. We experimented with two different number of queries ( $k = 5$  and  $k = 10$ ) generated for each input document and used the best query length computed in the previous experiment. Figure 4.10 compares the average percentage of the retrieved source documents. We observe that the proposed document retrieval method performs very well with 10 queries per document than 5 as it covers larger portion of the documents. On average, 79.97% of the source documents that were used in the composition of the input documents were successfully retrieved.

#### 4.5 Chapter Summary

In this chapter, we described a new query formulation approach for Arabic-based text reuse detection on the Web. The proposed model of the candidate document retrieval from the Web consists of four main stages. First, the input document is first preprocessed and then the document's fingerprints are computed. The next stage formulates a specified number of queries from the generated fingerprints of the document and then submitted to the Web through a public Web search engine interface. The performance of the proposed approach was evaluated against a specially designed documents collection, which contains documents that are composed by reused text from Web documents. The experimental results show that queries with 10 terms perform the best and on average, 79.97% of the source documents that were used in the composition of the input documents were successfully retrieved.

## Chapter 5: Fingerprinting-based Detection and Detailed Analysis of Text Reuse

### 5.1 Introduction

In chapter 4, we described our proposed query formulation approach for the candidate source documents retrieval component of the system, in which a set of representative queries are generated from a given input document in order to acquire potential similar documents from the Web via public search engine query interface. The retrieved set of documents are used to construct a relatively small, but topically similar documents collection to a given input document (also called query document). Each document in the constructed documents collection undergoes the necessary preprocessing operations and are represented by sets of hashed fingerprints to create an index of the corpus. Next, the local component of the text reuse detection problem selects the potential source documents that are highly similar with the given input document (exceeds a certain similarity threshold) by means of matching fingerprints and then performs detailed similarity analysis between each suspected source document and the given document. Accurately, the process of comparing each candidate document with the given input document and identifying the similar passages of text. Finally, the extracted passages are filtered, cleaned and presented. In this chapter, we describe the three main tasks mentioned above assuming the documents collection is available.

The remainder of the chapter is structured as follows: in the next section, we will describe the fingerprinting approach used in this research. The proposed detection framework DETRA is detailed in Section 5.3. Section 5.4 presents the evaluation measures, and the experimental results of the system along with a comparison between the proposed system DETRA and the a winnowing-based approach. A statistical analysis on the behaviour of DETRA with respect to Arabic-specific preprocessing operations such as stemming and character normalization is carried out in Section 5.5.

## 5.2 Fingerprinting Approaches

As mentioned earlier in Chapter 2, the fingerprinting approach proved to be more efficient than other approaches when used for local text reuse detection as it enhances the runtime performance in terms of reducing the dimensionality of the search by means of fingerprints. Examples of fingerprinting approaches for document's fingerprints are described in Chapter 2. However, we will revisit the Hailstorm fingerprinting method (Section 2.3.1), since we have modified the method and developed an enhanced and more efficient version of it. The proposed enhanced approach is compared with the winnowing fingerprinting, which is one of the popular fingerprinting methods for local text reuse detection. We assume that the documents collection is already available. Evaluation is carried out using a collection of 1174 Arabic documents including 1725 simulated cases of text reuse (see Section 5.4.3).

This section describes the Hailstorm and Winnowing fingerprinting methods and then the proposed approach is presented in Section 5.3.

Before creating the fingerprints, the given document must go through certain preprocessing steps as described earlier in Chapter 3. The document is then represented by a set of consecutive subsequence of word  $n$ -grams and these shingles are passed into hash functions to be converted into numeric values. A subset of the generated hash values are selected to represent the document and are called the fingerprints of the document. A match is then detected and quantified between any two documents by measuring the amount of shared fingerprints over the total fingerprints generated in the two documents.

### 5.2.1 Hailstorm Fingerprinting Method

The Hailstorm fingerprinting algorithm follows the general steps of any fingerprinting technique. Specifically, it first pre-processes the document's text and then divides the document into a set of  $n$ -grams (also called shingles [35]), which are "sequences of  $n$  consecutive terms extracted from a sliding window with fixed size  $n$  running stepwise over the text" [27]. It then applies a hashing function to convert every token (word) in a shingle into a hash value

and then selects the shingle as part of the document's set of fingerprints only if the minimum hash value of all  $n$  tokens in the shingle occurs at the beginning or the end of that shingle. The selected shingle is then hashed and included in the resulting set of the document's fingerprints. The Hailstorm algorithm is a context-free algorithm, which means the selection of a shingle is independent of any other shingle in the document, which means either it is selected in all the documents containing it or is never selected. This property is very useful especially in the case of text shuffling/reordering. Furthermore, the Hailstorm guarantees a total coverage property. Explicitly, every token in the document is covered by at least one selected shingle (fingerprint). It is worth mentioning that the Hailstorm fingerprinting is unique in terms of being the only fingerprinting algorithm that is both context-free and ensures total coverage [52]. Figure 5.1 shows a simple example of text representation and fingerprinting based on Hailstorm method. In the example we choose  $n$  of the  $n$ -gram to be 3. The text is preprocessed and cleaned by removing the diacritics, punctuation marks (:), stop words (هو ، في) and then applying stemming on the remaining words (Figure 5.1-(b)). The sequence of 3-grams (shingles) derived from the text is depicted in Figure 5.1-(c). A hypothetical set of hashes for each word in a shingle is given in Figure 5.1-(d). Based on the Hailstorm selection algorithm, a shingle is selected only if the minimum hash value of all  $n$  words in the shingle occurs at the beginning or the end of the shingle. The minimum hash value in each shingle is underlined as shown in (d). Figure 5.1-(e) displays the shingles selected by the fingerprinting algorithm. The selected shingles by Hailstorm algorithm are then hashed and stored along with positional information in the document as a set of [fingerprint, position] pairs (starting from base 0) (Figure 5.1-(f)).

### 5.2.2 Winnowing Fingerprinting Method

Winnowing is a well-known fingerprinting technique that proved to be both efficient and insures the detection of a match of a certain length [26]. After representing the text as a set of hashed  $n$ -grams, another fixed-size window (winnowing window) is adopted over the hashed  $n$ -grams and sliding by one chunk at a time. The  $n$ -gram with the minimum hash



الْبَحْثُ الْعِلْمِيُّ: هُوَ أُسْلُوبٌ مُنْتَظَمٌ فِي جَمْعِ الْمَعْلُومَاتِ الْمَوْثُوقَةِ

(a) Some text.

بَحْثُ عِلْمٍ سَلْبٍ نَظْمٍ جَمْعِ عِلْمٍ وَثَقٍ

(b) The text after being preprocessed and stemmed.

(بَحْثُ عِلْمٍ سَلْبٍ)، (عِلْمٍ سَلْبٍ نَظْمٍ)، (سَلْبٍ نَظْمٍ جَمْعٍ)، (نَظْمٍ جَمْعِ عِلْمٍ)، (جَمْعِ عِلْمٍ وَثَقٍ)

(c) The sequence of 3-grams (shingles) derived from the text.  $N=3$  in this example.

(491 ,877 ,072) ,(877 ,072 ,293) ,(072 ,293 ,233) ,(293 ,233 ,877) ,(233 ,877 ,914)

(d) A hypothetical set of hashes for each word in a shingle.

(491 ,877 ,072) ,(072 ,293 ,533) ,(233 ,877 ,914)

(e) Shingles selected by Hailstorm algorithm.

[50066893,4] , [2093546,2] , [13431910 ,0]

(f) Fingerprints recorded with positional information.

Figure 5.1: Hailstorm fingerprinting applied on a sample text

is selected in each winnowing window. If the minimum hash value occurs more than once within a winnowing window, the rightmost occurrence will be selected. The fingerprint of the document will be the set of selected  $n$ -grams with the minimum hash value. Figure 5.2 shows an example of the winnowing selection strategy on a the same text as in Figure 5.1. The text is preprocessed and stemmed (Figure 5.2-(b)). The sequence of 3-grams (shingles) derived from the text is shown in Figure 5.2-(c). Based on the winnowing method, another window (size=3) is used to group each 3-grams and slides by one 3-gram at a time (Figure 5.2-(d)). The generated 3-grams are hashed and the minimum hash value is selected in each window (Figure 5.2-(e)). Finally, the selected fingerprint is stored along with positional information in the document as a set of [fingerprint, position] as in Figure 5.2-(f).

The winnowing method showed better results when compared with other overlap fingerprinting techniques [4]. Therefore, we have selected it for comparison with our DETRA (see Section 5.4.3).

### 5.3 A Framework for Text Reuse Detection Using A Hybrid Approach

Formally, given a document  $d$  and a collection of documents  $D$ , the task of detecting text reuse consists of “identifying pairs of passages  $(p, p')$  from the given document  $d$  and a document  $d'$  from the provided collection of documents ( $d' \in D$ ), respectively such that

الْبَحْثُ الْعِلْمِيُّ: هُوَ أُسْلُوبٌ مُنْتَظَمٌ فِي جَمْعِ الْمَعْلُومَاتِ الْمَوْثُوقَةِ

(a) Some text.

بحث علم سلب نظم جمع علم وثق

(b) The text after being preprocessed and stemmed.

((بحث علم سلب)، (علم سلب نظم)، (سلب نظم جمع)، (نظم جمع علم)، (جمع علم وثق))

(c) The sequence of 3-grams (shingles) derived from the text.  $N=3$  in this example.

((بحث علم سلب)، (علم سلب نظم)، (سلب نظم جمع))، ((علم سلب نظم)، (سلب نظم جمع))، (نظم جمع علم)، ((علم)، (سلب نظم جمع))، (نظم جمع علم)، (جمع علم وثق))

(d) Windows of hashes of size 3 ( $w=3$ ).

((2093546)), ((11594434)), ((2093546)), ((44858753)), ((2093546)), ((44858753)), ((98023127)), ((50066893)), ((11594434))

(e) A hypothetical set of hashes for each word in a shingle.

[2093546 ,2]

(f) Fingerprints selected by Wining algorithm with positional information.

Figure 5.2: Wining fingerprinting applied on a sample text

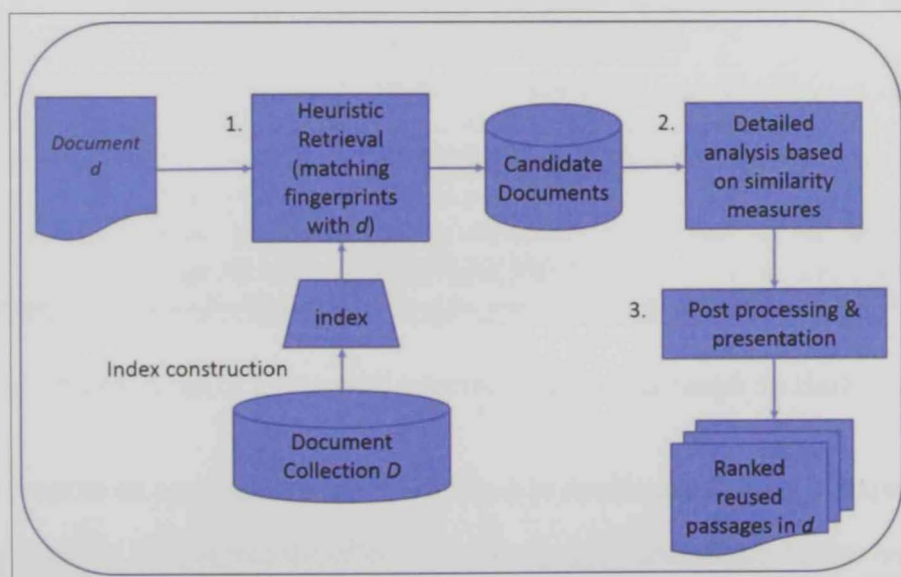


Figure 5.3: The main processes for detecting local text reuse

the detected passages  $p$  and  $p'$  are similar" [1]. This particular problem comprises of three components (see Figure 5.3), which is typically followed by most researchers of text reuse detection field [3]. The first is candidate document retrieval, which focuses on the selection of a small set of documents from a given corpus or documents collection  $D$  that are likely to include text reuse with document  $d$ . The second component is pairwise document comparison between  $d$  and each document  $d'$  in the retrieved set of source documents extracting all the passages of text that are highly similar. This component is also named as text alignment [70]. Finally, a post processing step is performed and used to clean, and filter the extracted passage pairs, and possibly visualize them for later presentation based on the measure of similarity.

<p>البحث العلمي هو أسلوب منظم في جمع المعلومات الموثوقة وتدوين الملاحظات والتحليل الموضوعي لتلك المعلومات باتباع أساليب ومناهج علمية محددة بقصد التأكد من صحتها أو تعديلها أو إضافة الجديد لها، ومن ثم التوصل إلى بعض القوانين والنظريات والتنبؤ بحدوث مثل هذه الظواهر والتحكم في أسبابها.</p> <p>(a) Sample text.</p>				
<p>بحث، علم، سلب، نظم، جمع، علم، وثق، دون، لاحظ، حلل، وضع، علم، تبع، سلب، نهج، علم، حدد، قصد، أكد، صحي، عدل، ضيف، جدد، وصل، بعض، قون، نظر، نبأ، حدث، مثل، ظهر، حكم، سبب</p> <p>(b) Preprocessed text</p>				
بحث علم سلب نظم جمع	علم سلب نظم جمع علم	سلب نظم جمع علم وثق	نظم جمع علم وثق دون	جمع علم وثق دون لاحظ
علم وثق دون لاحظ حلل	وثق دون لاحظ حلل وضع	دون لاحظ حلل وضع علم	لاحظ حلل وضع علم تبع	حلل وضع علم تبع سلب
وضع علم تبع سلب نهج	علم تبع سلب نهج علم	تبع سلب نهج علم حدد	سلب نهج علم حدد قصد	نهج علم حدد قصد أكد
علم حدد قصد أكد صحي	حدد قصد أكد صحي عدل	قصد أكد صحي عدل ضيف	أكد صحي عدل ضيف جدد	صحي عدل ضيف جدد وصل
عدل ضيف جدد وصل بعض	ضيف جدد وصل بعض قون	جدد وصل بعض قون نظر	وصل بعض قون نظرتياً	بعض قون نظرتياً حدث
قون نظر نبأ حدث مثل	نظر نبأ حدث مثل ظهر	نبأ حدث مثل ظهر حكم	حدث مثل ظهر حكم سبب	
<p>(c) The sequence of 5-grams (shingles) derived from the text.</p>				
<b>688 .679 .690 .684 .678</b>	679 .690 .684 .678 .689	690 .684 .678 .689 .681	684 .678 .689 .681 .684	<b>678 .689 .681 .684 .217</b>
685 .677 .684 .691 .679	<b>677 .684 .691 .679 .688</b>	684 .691 .679 .688 .679	691 .679 .688 .679 .690	679 .688 .679 .690 .684
<b>674 .687 .679 .684 .689</b>	687 .679 .684 .689 .685	<b>679 .684 .689 .685 .677</b>	684 .689 .685 .677 .684	689 .685 .677 .684 .691
690 .678 .682 .686 .682	<b>678 .682 .686 .682 .674</b>	682 .686 .682 .674 .687	686 .682 .674 .687 .679	682 .674 .687 .679 .684
679 .130 .689 .688 .676	<b>130 .689 .688 .676 .690</b>	689 .688 .676 .690 .678	688 .676 .690 .678 .682	<b>676 .690 .678 .682 .686</b>
	<b>681 .679 .683 .689 .679</b>	<b>679 .683 .689 .679 .130</b>	683 .689 .679 .130 .689	689 .679 .130 .689 .688
<p>(d) The corresponding hash values for each token in a shingle. The selected fingerprints are displayed in bold.</p>				

Figure 5.4: An example of a text with selected fingerprints based on Hailstorm method

We propose an approach to detect text reuse in Arabic texts. A set of experiments were conducted to gain an insight into the effect of the unique features of the Arabic language to the main components of the general framework of local text reuse detection techniques. We have also evaluated the performance of the detection component of our system using a relatively large corpus that has been recently created for the detection of text reuse and plagiarism in



## Arabic documents [1].

البحث العلمي هو أسلوب منظم في جمع المعلومات الموثوقة وتدوين الملاحظات والتحليل الموضوعي لتلك المعلومات باتّباع أساليب ومناهج علمية محددة بقصد التأكد من صحتها أو تعديلها أو إضافة الجديد لها، ومن ثمّ التوصل إلى بعض القوانين والنظريات والتنبؤ بحدوث مثل هذه الظواهر والتحكم في أسبابها.

(a) Sample text.

بحث، علم، سلب، نظم، جمع، علم، وثق، دون، لحظ، حلل، وضع، علم، تبع، سلب، نهج، علم، حدد، قصد، أكد، صحي، عدل، ضيف، جدد، وصل، بعض، قون، نظر، نبأ، حدث، مثل، ظهر، حكم، سبب

(b) Preprocessed text

بحث علم سلب نظم جمع	علم سلب نظم جمع علم	سلب نظم جمع علم وثق	نظم جمع علم وثق دون	جمع علم وثق دون لحظ
علم وثق دون لحظ حلل	وثق دون لحظ حلل وضع	دون لحظ حلل وضع علم	لحظ حلل وضع علم تبع	حلل وضع علم تبع سلب
وضع علم تبع سلب نهج	علم تبع سلب نهج علم	تبع سلب نهج علم حدد	سلب نهج علم حدد قصد	نهج علم حدد قصد أكد
علم حدد قصد أكد صحي	حدد قصد أكد صحي عدل	قصد أكد صحي عدل ضيف	أكد صحي عدل ضيف جدد	صحي عدل ضيف جدد وصل
عدل ضيف جدد وصل بعض	ضيف جدد وصل بعض قون	جدد وصل بعض قون نظر	وصل بعض قون نظر نبأ	بعض قون نظر نبأ حدث
قون نظر نبأ حدث مثل	نظر نبأ حدث مثل ظهر	نبأ حدث مثل ظهر حكم	حدث مثل ظهر حكم سبب	

(c) The sequence of 5-grams (shingles) derived from the text.

688 .679 .690 .684 .678	679 .690 .684 .678 .689	690 .684 .678 .689 .681	684 .678 .689 .681 .684	<b>678 .689 .681 .684 .217</b>
685 .677 .684 .691 .679	<b>677 .684 .691 .679 .688</b>	684 .691 .679 .688 .679	691 .679 .688 .679 .690	<b>679 .688 .679 .690 .684</b>
<b>674 .687 .679 .684 .689</b>	687 .679 .684 .689 .685	<b>679 .684 .689 .685 .677</b>	684 .689 .685 .677 .684	689 .685 .677 .684 .691
690 .678 .682 .686 .682	<b>678 .682 .686 .682 .674</b>	682 .686 .682 .674 .687	686 .682 .674 .687 .679	682 .674 .687 .679 .684
679 .130 .689 .688 .676	<b>130 .689 .688 .676 .690</b>	689 .688 .676 .690 .678	688 .676 .690 .678 .682	676 .690 .678 .682 .686
	<b>681 .679 .683 .689 .679</b>	679 .683 .689 .679 .130	683 .689 .679 .130 .689	689 .679 .130 .689 .688

(d) The selected fingerprints by our improved method are displayed in bold.

Figure 5.5: The selected fingerprints by our modified Hailstorm method

Our approach integrates both an overlapping-based fingerprinting strategy to represent a document as well as Information Retrieval heuristics to expand the resulting set of fingerprints in order to overcome the limitation of fingerprinting approaches to identify reuse instances in text with minor edits.

The proposed approach is built on an improved version of the Hailstorm fingerprinting, which proved to be the only fingerprinting technique that is both context-free and provides total coverage of the terms in a document [52]. It also combines the results of the winnowing fingerprinting to produce a more robust fingerprinting method. However, the Hailstorm fingerprinting technique may select unnecessary or redundant fingerprints in such a way that all its tokens are covered by other selected fingerprints. We modified the Hailstorm fingerprinting by combining the best features of the winnowing and hailstorm fingerprinting strategies into one more robust method that reduces the amount of selected fingerprints by exploiting the second window of the winnowing algorithm, without affecting the total coverage property

guaranteed by the Hailstorm algorithm. In particular, we use a second window to ensure a shingle that all its terms are contained in other shingles will not be selected in the final set of fingerprints. This would have big impact on the efficiency and robustness of the system as well as the amount of space occupied by the fingerprints, especially with large document. Figure 5.4 depicts an example of a sample text and the corresponding fingerprints based on the original Hailstorm method. Figure 5.5, on the other hand, shows the selected fingerprints by our modified approach.

To deal with cases when parts of the text have been reordered or deleted, the selected fingerprints are expanded with reordered tokens formed from each fingerprint to handle the unique characteristic of the Arabic language being an order-free language. More modified fingerprints are created by omitting words from the shingles that have been selected as fingerprints. More details are given in Section 5.3.1.

### 5.3.1 Candidate Document Retrieval Model

Figure 5.6 displays the building blocks of the source documents retrieval component. In what follows, we describe each block in detail.

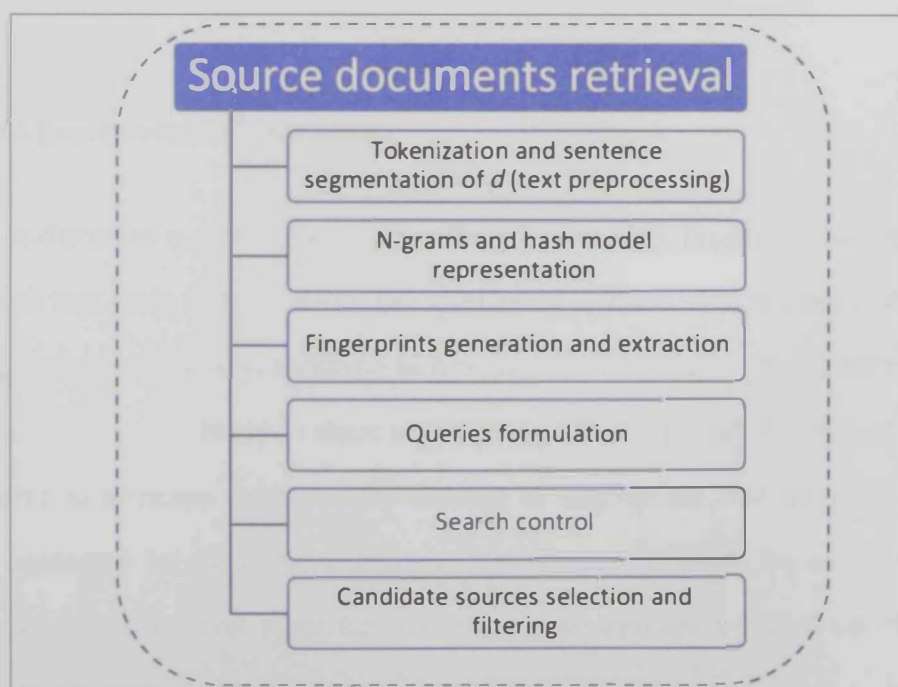


Figure 5.6: The building blocks of source documents retrieval from a given local documents collection



## **Tokenization and sentence segmentation**

Given a document  $d$ , the text is divided into a set of chunks/shingles. The text is first tokenized into words and then each set of consecutive words are grouped into a chunk/shingle. Sentence identification and segmentation is considered an important process since a sentence can be looked at as a relevant unit for an actual text reuse to be detected. In our system, each sentence is processed individually and it is further split into a set of overlapping chunks. The potential of having overlapping sets of chunks ( $n$ -grams in our case) is to reduce the risk of having more than one source in one chunk.

## **N-grams and hash model representation**

For each sentence in a given document  $d$ ,  $n$ -grams are basically composed from a “contiguous sequence of overlapping  $n$  units from a given text” [6]. In our system, the units are the words in the sentence. There are two main features of this model, the first is that it is simple to apply, and the second is that different representations may be formed by simply changing the value of  $n$ .

## **Fingerprints generation and extraction**

Given the set of  $n$ -grams generated in the previous step, fingerprints are generated and extracted from that set in order to formulate queries using them. The rationale of the different selection algorithms of the fingerprinting techniques is to maximize the chance to retrieve the source documents that are likely to share text with a given document. The selection technique may also serve as a means to reduce the amount of fingerprints that are used to formulate the queries necessary for the retrieval process. This step is probably the most crucial step in the overall document retrieval algorithm since the generated and selected set of fingerprints directly affects the overall performance of the system.

The fingerprinting generation and selection technique developed in DETRA exploits

the global coverage provided by the Hailstorm algorithm [52] and improves the selection algorithm in such a way that it reduces the generated set of fingerprints. The improved fingerprinting method guarantees the property of context-freeness, i.e. selecting shingles as fingerprints only depends on the terms that comprise the shingles and is not affected by the other shingles in the document. It also provides a total coverage, which means every word in the document exists in at least one of the selected fingerprints.

---

**Algorithm 2** Improved Hailstorm Fingerprinting Algorithm
 

---

```

1: procedure EFFICIENTHS( $w, n$ )                                ▷  $w$ : window size,  $n$ : shingle size
2:    $hash\_list \leftarrow h[w]$ 
3:    $r \leftarrow 0$                                              ▷ shingle's right end
4:    $l \leftarrow 0$                                              ▷ shingle's left end
5:    $min \leftarrow 0$                                            ▷ index of min hash in a shingle
6:    $h[r] \leftarrow getHash(token[r])$ 
7:   while ( $l < (w - n + 1)$ ) do                                ▷ total shingles in a window
8:      $r \leftarrow r + 1$ 
9:      $h[r] \leftarrow getHash(token[r])$ 
10:    if ( $h[r] \leq h[min]$ ) then
11:       $min \leftarrow r$ 
12:    else if ( $min < l$ ) then                                  ▷  $min$  is outside shingle's range
13:      for ( $j \leftarrow r - 1; j \geq l; j \leftarrow j - 1$ ) do ▷ find the minimum in the current shingle
starting from  $r$ 
14:        if ( $h[j] < h[min]$ ) then
15:           $min \leftarrow j$ 
16:        end if
17:      end for
18:    end if
19:    if ( $r = (n + l - 1)$ ) then
20:      if ( $(min = l) \text{ or } (min = r)$ ) then                    ▷ check if the shingle is accepted as a
fingerprint
21:         $record\_fingerprint(l, r)$ 
22:      end if
23:       $l \leftarrow l + 1$                                      ▷ shift to the next shingle
24:    end if
25:  end while
26: end procedure

```

---

For each sentence in a given document  $d$ , the words in the sentence are hashed separately prior to applying  $n$ -gram to the document. Each  $n$ -gram (also called shingle) within a window of size  $w$  is examined whether the word/token with the lowest hash value lies at the first or the last position within that  $n$ -gram and hence, will be selected as a fingerprint.

Each selected fingerprint is then converted into a numerical form by means of a Karp-Rabin hash model. Algorithm 2 describes an efficient hybrid approach that combines the Hailstorm and WinoWing fingerprinting selection pseudo-code. The algorithm takes advantage of the fact that the minimum hash value of a token in the previous shingle is likely to be within the current shingle as well in a given window. Therefore, it requires only one comparison to test if there is a new minimum in the current shingle since overlapping shingles (or  $n$ -grams) are usually shifted by one token at a time. Note that in the case of choosing between more than one minimal hash, the rightmost minimum is selected (Line 10-Algorithm 2). This ensures that this minimum will remain within the next shingle range and only a single comparison is required for the next shingle and hence, the efficiency of the algorithm is further improved. A re-computation of a new minimum may be necessary by traversing the shingle's tokens only in the case when the minimum hash of the preceding shingle is no longer within the current shingle (Line 13-Algorithm 2). The search for a new minimum in the shingle is carried out starting from the rightmost token to the leftmost one to ensure that the rightmost minimal hash is selected. The selected shingles as fingerprints, which satisfy the criteria that the minimum hash of a token exists either at the rightmost or the leftmost position of the shingle, are saved and hashed along with the local and global positions in the document. A filtering process is carried out after the selection of fingerprints to eliminate redundant fingerprints. A fingerprint is considered redundant if all its tokens may be covered by other selected fingerprints. This process helps reducing the resulting set of fingerprints and improves the overall quality of the fingerprinting algorithm.

When analysing the asymptotic behaviour of Algorithm 2, we observe that the while-loop is the main body of the algorithm and it runs at most  $O(w)$  times. The for-loop inside the while-loop runs only when the minimum hash is not within the current shingle being examined. If we assume that the minimum hash value is given to the token that exists at the leftmost position of all the shingles in a window. In other words, imagine the hash values are sorted in ascending order. Then, at each iteration of the while-loop, The test at Line 10 in the algorithm will not be satisfied and the for-loop is run testing all the hash values of the current

shingle starting from the rightmost token and moving backward to the leftmost token of that shingle (Line 13-Algorithm 2). This step will run  $n - 1$  times to look for the minimum hash value within the shingle being examined. Therefore, at the worst (hypothetical) case scenario, the algorithm will run at most  $O(w * n)$  times to select the fingerprints using Algorithm 2. Since  $n$  is much lesser than the value of the window, we can conclude that the algorithm has a linear time-complexity. Considering all the windows in a given document of size  $N$  words, the overall fingerprinting process may cost  $O(Nw)$ .

As mentioned earlier in Chapter 3, the choice of the value  $n$  would certainly affect the resulting set of fingerprints and hence the precision of text reuse detection, depending on the document's content and domain. More precisely, choosing  $n$  to be sufficiently big such that common Arabic idioms have length shorter than  $n$  and therefore will not be considered as a matching reuse. For instance, it is very common in religious documents to include quotations from Quran and Hadith (sayings of the prophet Mohammed) or expressions that are likely to appear in other documents and considering such expressions and phrases as parts of the queries will lead to many false positive cases in the results, which are incorrectly identified text as reused cases. In the experiments, we examined different values of  $n$  ( $n = 3$  to  $7$ ) and we achieved better results for  $n = 3 \dots 5$ . More discussions are provided in Section 5.4.

---

#### Algorithm 3 Expanded Fingerprints by Deletion Algorithm

---

```

1: procedure DELFINGERPRINTTOKENS(shingles, n) ▷ shingles: list of selected shingles
   as fingerprints, n: number of tokens in a shingle  $w_0, w_1, \dots, w_{n-1}$ 
2:   for ( $j \leftarrow 1; j < n - 2; j \leftarrow j + 1$ ) do
3:     record_fingerprint( $j, n - 1$ ) ▷ delete one token  $w_j$  at a time
4:   end for
5: end procedure

```

---

#### Query formulation

Given the generated set of fingerprints for each individual sentence for a document  $d$ , queries are formulated in such a way that conforms with the search engine being used for indexing the source documents and later on for searching the indexer to find candidate documents with similar fingerprints. In this step, queries are further expanded to accommodate text



reordering and deletion, two common text editing operations in Arabic text. This expansion of queries is essential in order to overcome the limitation of fingerprinting approaches to identify reuse when the original text is modified. We propose techniques for expanding queries with reordered or deleted terms of the selected fingerprints'  $n$ -grams.

Suppose that  $[w_1, w_2, \dots, w_n]$  is the  $n$ -gram (or shingle) of a selected fingerprint. One term of the set  $w_2, \dots, w_{n-1}$  is removed and then hashed creating a modified fingerprint. Note that the first and the last terms are not removed. A single  $n$ -gram will produce  $n - 2$  modified  $n$ -grams. Algorithm 3 shows the steps to generate modified  $n$ -grams that deals with word deletion, a common text editing operation in Arabic text. Following the previous example, the selected  $n$ -gram (بحث علم سلب) will produce one modified fingerprint (بحث سلب) by omitting the term علم.

---

#### Algorithm 4 Expanded Fingerprints by Reordering Algorithm

---

```

1: procedure REORDERFINGERPRINTTOKENS(shingle, n)  $\triangleright$  shingle: a selected shingle as
   a fingerprint, n: number of tokens in the shingle
2:   if (n = 1) then
3:     record_fingerprint(0, n - 1)
4:   else
5:     for (i  $\leftarrow$  0; i < n - 1; i  $\leftarrow$  i + 1) do
6:       ReorderFingerprintTokens(shingle, n - 1)
7:       if (is_even(n)) then
8:         swap(shingle[i], shingle[n - 1])
9:       else
10:        swap(shingle[0], shingle[n - 1])
11:      end if
12:    end for
13:    ReorderFingerprintTokens(shingle, n - 1)
14:  end if
15: end procedure

```

---

More fingerprints may be created by reordering the terms in a selected fingerprint's  $n$ -gram. For instance, given an  $n$ -gram of size=3,  $[w_1, w_2, w_3]$ , the reordered permutations of this  $n$ -gram are:  $[w_1, w_3, w_2]$ ,  $[w_2, w_1, w_3]$ ,  $[w_2, w_3, w_1]$ ,  $[w_3, w_1, w_2]$ ,  $[w_3, w_2, w_1]$ . We used the Heap's algorithm to generate such permutations on the selected shingles as the document's fingerprints in linear time [107]. See Algorithm 4 for details of the process. The modified  $n$ -grams are then hashed and used as expansion of the original query. The modified  $n$ -grams



in the expansion query are intended to improve the retrieval of candidate documents when such editing operations are performed. Continuing with the same previous example, the set of reordered  $n$ -grams for selected  $n$ -gram (بَحْثِ عِلْمِ سَلْبِ) is:

(بَحْثِ سَلْبِ عِلْمِ)، (عِلْمِ بَحْثِ سَلْبِ)، (عِلْمِ سَلْبِ بَحْثِ)، (سَلْبِ بَحْثِ عِلْمِ)، (سَلْبِ عِلْمِ بَحْثِ)

### Search control

Given the set of queries and expansion queries generated in the previous step, these queries are expressed and issued according to the local search engine employed in the system in order to retrieve candidate source documents that have matching fingerprints with the input document  $d$ . A search controller is used to adjust the search based on the results of each submitted query, which may involve excluding subsequent queries that form the different reordering or deletion of words of a given fingerprint if one permutation succeeds in triggering a match in the search results. This filtering technique obviously improves the efficiency of the search by pruning out the queries that are no longer needed. From the implementation point of view, the local search engine used in the system has been implemented using Lucene search API [97], which is a powerful full-text search library that easily allows for the scalability of the system in terms of large documents collection including the Web.

The Lucene search API takes a search query, consisting of the fingerprint of a sentence in the input document, and retrieves a set of documents ranked by the highest similarity scores computed for documents that match certain parts of the query. The resulting candidate documents are ranked based on TF-IDF scores, which stands for term frequency-inverse document frequency: a very common weighting measure to compute the similarity between documents. Term frequency gives more weight to the terms that are likely to appear more in a document. Inverse document frequency, on the other hand, considers the terms that do not appear much in many documents within a collection as more interesting than the ones that are frequent in many documents in the documents collection and are therefore given high weights. This approach filters out common words [97].

It is worth noting that all the documents in the documents collection (or corpus) are already indexed into Lucene Indexer by means of the sets of hashed fingerprints for each sentence in the document along with positional information of the sentences from which the fingerprints were selected within the document. A match is found between any document in the collection and a given input document when similar fingerprints are found between the indexer and the examined document. The positional information stored with the indexed fingerprints, such as the document's ID and the sentence index within the document allows for the direct determination of the local text reuse between documents.

### **Candidate sources selection and filtering**

A filtering technique is applied to the resulting documents retrieved by the local search engine in the previous step. This filtering is essential in order to reduce the resulting set by excluding documents that are not worthwhile to be considered for the next phase of the detailed pairwise documents comparison. At this stage, similarity measures are derived from the number of fingerprints shared between the input and source documents. Following the containment similarity measure introduced in Chapter 2, the similarity measure is computed and the similarity scores that are less than a predefined threshold (empirically tuned 20%) are discarded from the set of candidate source documents.

### **5.3.2 Pairwise Comparison Model (Text Alignment)**

In this component of the system, all passages of reused text are identified between each resulted document from the retrieval phase and the given input document. The name of the component (text alignment) is borrowed from the field of bioinformatics as it is closely analogous to gene sequence alignment. There are three common steps associated with gene sequence alignment and may be followed when looking for similar passages between any pair of documents. Namely: seeding, extension, and filtering [70].

## Seeding

Given an input document, and a potential source document, seeding is concerned with the act of aligning matching seeds between the two documents in order to identify the common passages between them. The matching seeds in our work is basically the matching fingerprints between the input document and the source document. The rationale for this step is to distinguish and locate the substrings that collectively form the anticipated similarity between input and source documents. Considering all the matching seeds (fingerprints) between the sentences of the pair of documents, the following step of extending the seeds into aligned text takes place as will be described next.

## Extension

Given the matching seeds between a pair of input and source documents, they are extended into aligned text between the two documents to consider the whole reused text rather than just fragments of it. In this step, we present a 2-step sentence-based merging approach. It depends on the matching pair of fingerprints between the sentences of the input and source document. It works as follows: (1) In the first step, adjacent sentences that are preceding the current sentence in which the matching seeds were found are inspected for similarity. This is performed by first checking the  $n$ -grams of the preceding sentence pair in the input and source documents respectively and if a match is found within a predefined threshold (no more than 100 characters), then the two consecutive sentences are grouped and aligned. (2) The second step looks at any matching seeds between the sentences that follow the current sentences pair in both the input and source documents and are eventually merged if a match is found within predefined threshold (no more than 100 characters).

## Filtering

Given a set of aligned sentences, the filtering process discards the sentences that do not meet a certain criteria. The main purpose of this process is to exclude very short matching text in order to reduce the rate of false positive cases. In our work, if the detected reused text is less than 100 characters, the aligned sentences are considered very short and not reported for reuse. Otherwise, the aligned pair of sentences in the input and source documents is reported.

### 5.3.3 Reused Text Presentation

The detected pairs of reused passages between a given input document and the source documents is presented in two different forms. The first form is in XML format that would help in the computation of the evaluation measures as will be described in the next section. An example of an XML format reporting a text reuse instance between an input document named “input\_document0001” and a source document named “source\_document00215” is shown in Figure 5.7. The text reuse instance in the figure determines an aligned text of size 610 characters, starting at character offset 2772 in the input document and at character offset 11139 in the source document.

```
<?xml version="1.0" encoding="UTF-8"?>
<document reference="input-document0001.txt">
<feature name="text_reuse" this_offset="2772" this_length="610"
source_reference="source-document00215.txt" source_offset="11139"
source_length="610"/>
</document>
```

Figure 5.7: The meta information about the reused text cases detected as pairs between the input and the source document

The second output form of the reused instances is presented visually as shown in Figure 5.8. The text enclosed in a black rectangle shows a reused text between the given input document and a source document titled: “source\_document00401.txt”.



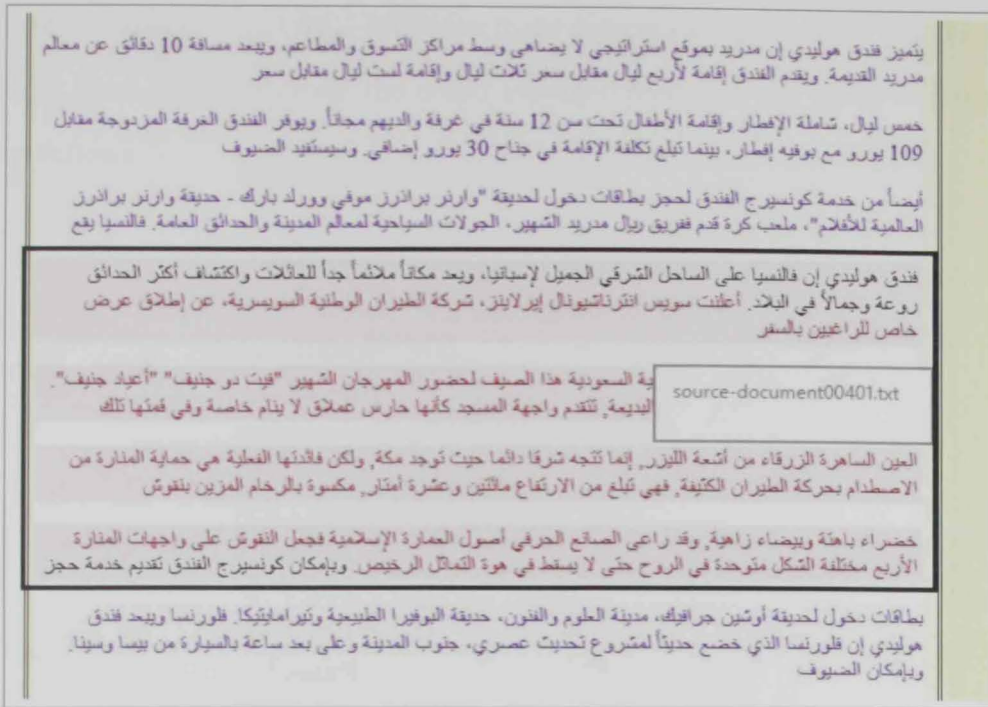


Figure 5.8: A sample of the visual output presenting a reused text case (enclosed in a box) between the displayed document with a source document named: source\_document00401

## 5.4 Evaluation

### 5.4.1 Performance Measures

To assess the performance of the candidate documents retrieval and the detailed detection phases, we used the conventional evaluation measures used in IR. We summarize these measures based on text reuse detection point of view [108]. The methods were evaluated using a micro and macro-averaged precision, recall, and F-measure in addition to the granularity and an overall text reuse detection score that combines these measures into one score. All these measures are computed based on the simulated text reuse instances in the corpus and the detected ones by the system. Note that in formulas 5.1-5.6,  $S$  refers to the set of actual simulated text reuse cases and  $R$  is the set of detected text reuse cases by the system.

A document  $d$  is represented based on the characters indices in the document. A simulated text reuse case in the corpus  $s$  can then be represented as  $s = s_{sim} \cup s_{src}$ , where  $s_{sim} \in d_{sim}$  and  $s_{src} \in d_{src}$ . Following the same representation, a detection case  $r$  by the system



can be represented as  $r = r_{det} \cup r_{src}$ . It follows that  $r$  detects  $s$  iff  $r_{det} \cap s_{sim} \neq \phi$  and  $r_{src} \cap s_{src} \neq \phi$ . Based on these representations, the micro-averaged precision and recall of  $R$  under  $S$  are defined as follows:

$$prec_{micro}(S, R) = \frac{|\cup_{(s,r) \in (S \times R)} (s \cap r)|}{|\cup_{r \in R} r|} \quad (5.1)$$

$$rec_{micro}(S, R) = \frac{|\cup_{(s,r) \in (S \times R)} (s \cap r)|}{|\cup_{s \in S} s|} \quad (5.2)$$

“The precision and recall count the proportion of the true positive part of each detected and actual simulated case respectively” [1]. The  $F1$  measure is the harmonic mean of the computed precision and recall. The length of the reused text instance does not affect the macro-averaged precision and recall measures; they rather characterizes the power of the system to detect all the reused cases and they are defined as follows:

$$prec_{macro}(S, R) = \frac{1}{|R|} \sum_{r \in R} \frac{|\cup_{s \in S} (s \cap r)|}{|r|} \quad (5.3)$$

$$rec_{macro}(S, R) = \frac{1}{|S|} \sum_{s \in S} \frac{|\cup_{r \in R} (s \cap r)|}{|s|} \quad (5.4)$$

$$gran(S, R) = \frac{1}{|S_R|} \sum_{s \in S_R} |R_S| \quad (5.5)$$

$$TRDS(S, R) = \frac{F1}{\log_2(1 + gran(S, R))} \quad (5.6)$$

Note that  $gran(S, R)$  in Equation 5.5 is the granularity of the detected text reuse cases by the system, which averages the number of the separate text fragments detected for each actual case [108]. Obviously, the best granularity score is 1, which means only one complete reuse case has been detected for each actual simulated text reuse case. Equation 5.6 ( $TRDS$ ) which refers to the overall Text Reuse Detection Score combines all the measures into one overall score of the text reuse detection process. In the equation,  $F1$  denotes the F-Measure, i.e., the weighted harmonic mean of precision and recall. The logarithm of the granularity is used in the denominator to decrease its effect on the overall score.

#### 5.4.2 Experimental Setup

The evaluation of the system has been carried out using a newly constructed corpus dedicated for text reuse in Arabic texts. The corpus (as described in [1]) consists of 1174 Arabic documents including 1725 simulated cases of text reuse. The documents lengths vary between short (1-10 pages) and medium (10-100 pages), while the test cases lengths fall between 300 to more than 30000 characters. The general characteristics of the used corpus are summarized in the Table 5.1 .

#### 5.4.3 Experimental Results and Analysis

This section presents the experiments that are carried out to explore the effect of the proposed fingerprinting method on the overall performance of the text reuse detection system.

Corpus Characteristics		
General information	Corpus size	1174
	Text reuse cases	1725
	Source documents	48.30%
Text reuse cases per document	Without text reuse	27.84%
	With text reuse	72.16%
Document length	Very short (<1 p)	22.57 %
	Short (1-10 p)	73.34%
	Medium (10-100 p)	4.09%
Cases length	Very short (<300 char)	21.28%
	Short (300-1000 char)	42.43%
	Medium (1000-3000 char)	28.46%
	Long (3000-30000 char)	7.83%

Table 5.1: Characteristics of the corpus used to evaluate the system [1]

In the experiments, different sentence lengths has been tested to investigate the effect of the sentence length on the fingerprinting-based retrieval model. Note that for all the experiments, the performance measures of the system have been computed for different values of  $n$ -grams varying from 3 to 7. On another note, the best sentence length of the Arabic text has been empirically tested and the results show that sentence lengths of Arabic text above 30 words produce better results. This goes in-line with the reported average length of Arabic sentence as 37 when compared to other languages such as English that has an average of 22 words per sentence (see Section 3.5.5). Figure 5.9 depicts the overall text reuse detection scores TRDS over different sentence lengths (25, 30, 35). Obviously, the TRDS score for sentence length= 25 is much less than the results obtained with sentence sizes 30 and 35, respectively.



Figure 5.9: Macro text reuse detection scores (TRDS) based on sentences length

Table 5.2 shows the detailed results in terms of micro and macro averaged precision,

N	Micro Measures				Macro Measures				Gran.
	Precision	Recall	F-measure	TRDS	Precision	Recall	F-measure	TRDS	
3	0.996	0.988	0.992	0.905	0.979	0.979	0.979	0.893	1.138
4	0.996	0.984	0.99	0.886	0.982	0.973	0.977	0.875	1.17
5	0.997	0.985	0.991	0.841	0.987	0.971	0.979	0.831	1.263
6	0.998	0.974	0.986	0.759	0.992	0.952	0.972	0.749	1.459
7	0.998	0.958	0.978	0.692	0.993	0.921	0.956	0.677	1.663

Table 5.2: Micro and macro measures of the DETRA with different values of  $n$  for  $n$ -grams and sentence size=25

recall, and F1 measures in addition to the overall score of the detection system (TRDS) and granularity measures. The results were obtained for the  $n$ -grams from 3 to 7, while fixing the sentence length to 25. Note that the precision values get improved by increasing  $n$ -gram sizes. This is justified as increasing  $n$  will avoid selecting highly frequent Arabic phrases especially from documents with religious content, which may cause false positive detection cases. As for recall values, it is obvious that increasing  $n$ -gram sizes will affect the sensitivity of the detection and might not retrieve all the potential source documents. In our system, the detected cases that are less than a certain threshold are discarded, which significantly enhanced the performance of the system. The overall TRDS takes into account the granularity of the detected cases whether they are detected as a single case or as smaller adjacent cases. In the results, the granularity is a bit higher than 1 (which is the optimal value), and hence the TRDS scores are affected accordingly. Typically, it is easy to obtain high precision results with fingerprinting techniques especially with our method, which provides a global coverage of the document. However, precision and recall measures are so related that one may be traded for the other. Figure 5.10 presents the micro and macro averaged measures of precision, recall, and F-measure of the DETRA. The micro results show the performance of DETRA at the character-level, whereas macro measures represent the results at the document-level results, in terms of the number of cases detected as a whole.

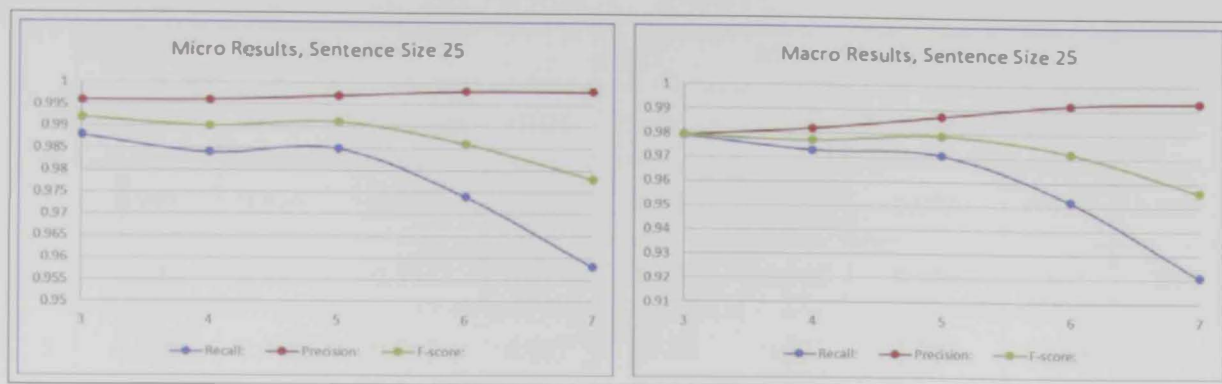


Figure 5.10: The micro and macro evaluation measures of our DETRA for documents with sentence size= 25

Figure 5.11 depicts the text reuse detection measure (TRDS), which combines all the measures, including granularity into one score. The chart shows that for higher values of  $n$ -grams, there is a decrease in the performance of the system. The best overall results are obtained for relatively small  $n$ -grams from 3 to 5.

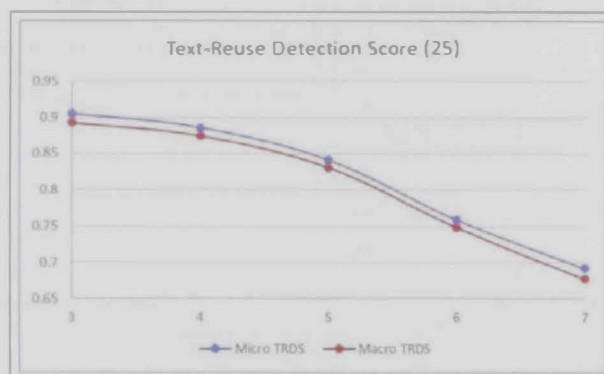


Figure 5.11: The overall text reuse detection score (TRDS) of our proposed DETRA for documents with sentence size= 25

Table 5.3 displays the performance difference of the system for sentences size=30 when compared to the previous results in Table 5.2. Obviously, the adjustment of the sentence size paid off in terms of improved performance. The results were anticipated based on what has been reported by NLP researchers about the Arabic language being much higher than the other languages in terms of the average sentence length (around 37). The granularity and TRDS are positively affected by the increase in the sentence size even for higher  $n$ -grams.

Figure 5.12 displays the micro and macro results of the system for different  $n$ -gram values varying from 3 to 5. Figure 5.13, on the other hand, shows the overall macro TRD



N	Micro Measures				Macro Measures				Gran.
	Precision	Recall	F-measure	TRDS	Precision	Recall	F-measure	TRDS	
3	0.995	0.986	0.991	0.936	0.975	0.971	0.973	0.919	1.083
4	0.995	0.987	0.991	0.929	0.975	0.968	0.972	0.911	1.095
5	0.996	0.986	0.991	0.921	0.981	0.973	0.977	0.908	1.108
6	0.996	0.977	0.986	0.867	0.986	0.958	0.972	0.854	1.201
7	0.997	0.975	0.986	0.819	0.988	0.949	0.968	0.804	1.303

Table 5.3: Micro and macro measures of DETRA with different values of  $n$  for  $n$ -grams and sentence size=30

score of the system.

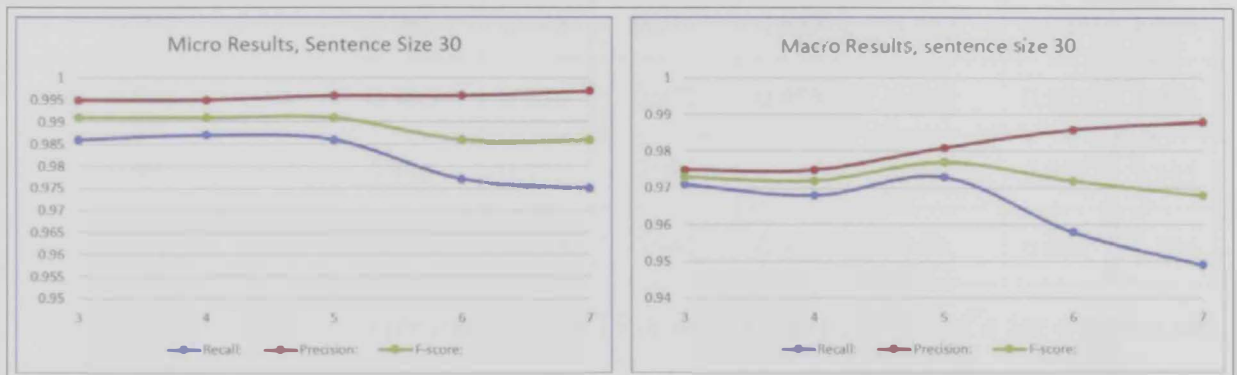


Figure 5.12: The micro and macro evaluation measures of DETRA for documents with sentence size= 30

Table 5.4 shows more results of the system for higher sentences size 35 with different sizes of  $n$ -grams. This allows for the analysis of the performance changes across different runs of the system with different parameters settings. When compared to the previous results of Tables 5.2 and 5.3, the system attains the best results with sentences size=35, which is the closest to the average sentence size of the Arabic language. The granularity score is significantly improved, which hence affects the overall TRD score. The rationale for this is the ability of the system to detect text reuse cases, even with medium or large case lengths.

Figures 5.14 and 5.15 display the micro and macro results of the system for different  $n$ -gram values varying from 3 to 5. It has been noted that, for the  $n$ -gram size = 4, the recall value

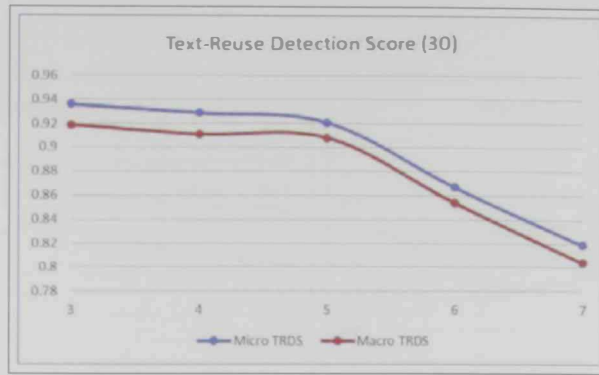


Figure 5.13: The overall text reuse detection score of DETRA for documents with sentence size= 30

N	Micro Measures				Macro Measures				Gran.
	Precision	Recall	F-measure	TRDS	Precision	Recall	F-measure	TRDS	
3	0.994	0.987	0.99	0.951	0.95	0.971	0.97	0.932	1.057
4	0.994	0.976	0.985	0.946	0.971	0.953	0.962	0.923	1.059
5	0.995	0.98	0.988	0.939	0.977	0.959	0.968	0.992	1.073
6	0.995	0.976	0.986	0.918	0.981	0.946	0.963	0.897	1.105
7	0.996	0.972	0.984	0.888	0.987	0.935	0.96	0.866	1.156

Table 5.4: Micro and macro measures of DETRA with different values of  $n$  for  $n$ -grams and sentence size=35

is more decreased when compared to the closer  $n$ -gram sizes 3 and 5. This made us interested to understand the underlying reason behind its low recall score. An investigation of its output showed that most of the false positive cases comes from documents with repeated idioms of size = 4, especially documents with religious text. We further investigated the problematic documents and we found out that a fingerprint with a very common phrase صلى الله عليه وسلم has caused many false positive results since it appears more than 350 times in the documents collection. See Figure 5.16 for an illustration. Another example is presented in Figure 5.17 with a common phrase صندوق تنمية الموارد البشرية when used as a fingerprint.

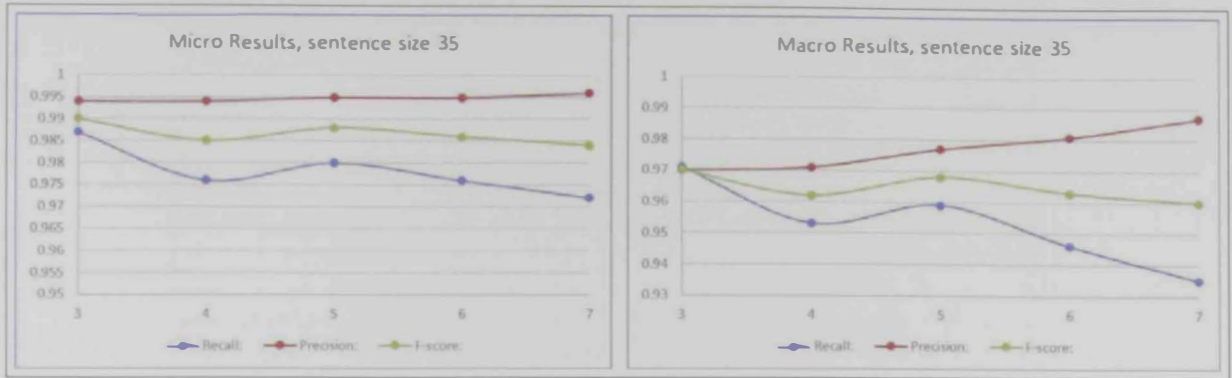


Figure 5.14: The micro and macro evaluation measures of DETRA for documents with sentence size= 35

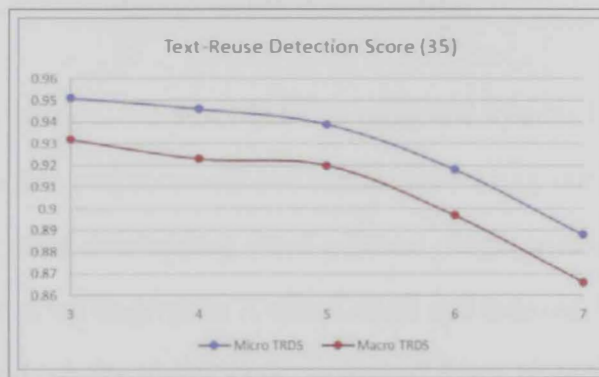


Figure 5.15: The overall text reuse detection score of DETRA for documents with sentence size= 35

**The detected text reuse case in an input document:**

حديث زينب بنت الرسول صلى الله عليه وسلم - يدل على ان المرأة إذا أسلمت وامتنع زوجها من الإسلام، فلها ان تتربص وتنتظر إسلامه، فإذا اختارت أن تقيم منتظرة لإسلامه، فإذا أسلم أقامت معه، فلها ذلك، كما كان النساء يعلنن في عهد النبي صلى الله عليه وسلم كزينب ابنته وغيرها

**Different occurrences of text reuse in the matching source document:**

فكرة الردة اقترنت على عهد النبي صلى الله عليه وسلم بعداوة الإسلام وحره من أمن كان يعمل لنصرته ومن ارتد كان يعمل على حره، أم في عهد الخليفة أبي بكر الصديق فنذكر مقولته المشهورة والله لو منعوني عقالا كانوا يؤدونه للنبي صلى الله عليه وسلم لقاتلتهم عليه

Figure 5.16: An example of selecting a common phrase as a fingerprint, which may cause false positive cases

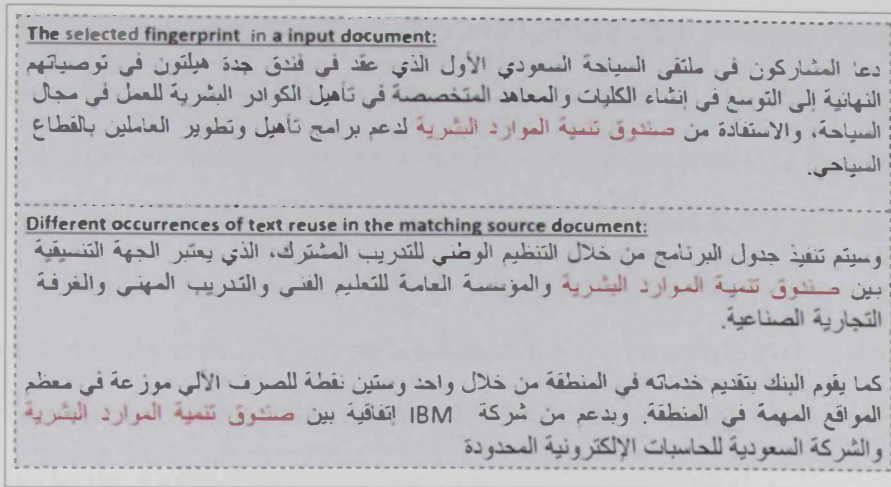


Figure 5.17: Another example of selecting a common phrase as a fingerprint, which may affect precision/recall scores

#### 5.4.4 Comparison to Other Fingerprinting Approaches

In the previous section, we described our proposed system DETRA. The entire documents collection has been preprocessed and represented using our proposed fingerprinting approach as a selected set of overlapping word  $n$ -grams (fingerprints) at the sentence-level of the documents. Each selected fingerprint is then hashed and indexed. Fingerprints of the same length are also generated for any given input document. Each fingerprint is issued as a query in order to retrieve candidate source documents. The source documents with matching fingerprints exceeds a similarity measure of a certain threshold will be chosen for detailed pairwise comparison with the input document. In this section, we compare our fingerprinting-based approach with another popular approach, which is the winnowing fingerprinting. The same documents collection and evaluation measures have been used to compare between the two methods.

Table 5.5 displays the averaged F-measure and TRD evaluation scores at the macro level for both the winnowing and DETRA. We chose macro TRDS to compare between the two methods as this measure is more concerned with the detection of reused cases at the document-level rather than the character-level. It better shows the ability of the method to detect reused cases based on the selected fingerprints in each method. In this experiment,

various values of  $n$  have been investigated ( $n = 3 \dots 5$ ), for the sentence length=25 words. As for the winnowing method, the window size has been set to 10. We chose to compare between the two methods with the sentence size 25, which is the closest to the average English sentence length, since the winnowing method proved to produce better results with English text. Overall, it can be observed that DETRA outperforms the winnowing system in all the different values of  $n$ . As expected, the performance of the winnowing system decreases as the  $n$ -gram size increases. This may also apply to DETRA. However, it is more contained in our method thanks to the global property of the improved fingerprinting method inherited from the Hailstorm fingerprinting.

N	Winnowing		Our method	
	F-measure	TRDS	F-measure	TRDS
3	0.971	0.754	0.979	0.893
4	0.959	0.694	0.977	0.875
5	0.934	0.625	0.979	0.831
6	0.894	0.596	0.972	0.749
7	0.846	0.546	0.956	0.677

Table 5.5: Macro measures of the winnowing approach in terms of the F-measure and the overall TRDS compared with our method for the different values of  $n = 3 \dots 7$  of  $n$ -grams, window size = 10, and sentence size=25

Table 5.6 shows another set of experiments that has been carried out in order to compare between DETRA with the winnowing-based system. Obviously, DETRA performs better than the winnowing system in the detection process. The improvement in the performance of our system with sentence length 30 is significant compared to the results of Table 5.5. Our DETRA shows more robustness in the results with the various sizes of  $n$ -grams in terms of the F-measure and TRDS measures in contrast to the winnowing method that notably decreases by the increase of the  $n$ -gram size. These high TRDS scores indicate the strength and robustness of DETRA in detecting text reuse for Arabic text.



N	Winnowing		Our method	
	F-measure	TRDS	F-measure	TRDS
3	0.972	0.889	0.973	0.919
4	0.967	0.845	0.972	0.911
5	0.955	0.77	0.977	0.908
6	0.945	0.73	0.972	0.854
7	0.905	0.659	0.968	0.804

Table 5.6: Macro measures of the winnowing approach in terms of the F-measure and the overall TRDS compared with our method for the different values of  $n = 3 \dots 7$  of  $n$ -grams, window size = 10, and sentence size=30

Figure 5.18 illustrates the F-measure results of the winnowing-based system and DETRA for the two sets of experiments applied on different  $n$ -gram sizes and sentences lengths 25 and 30.

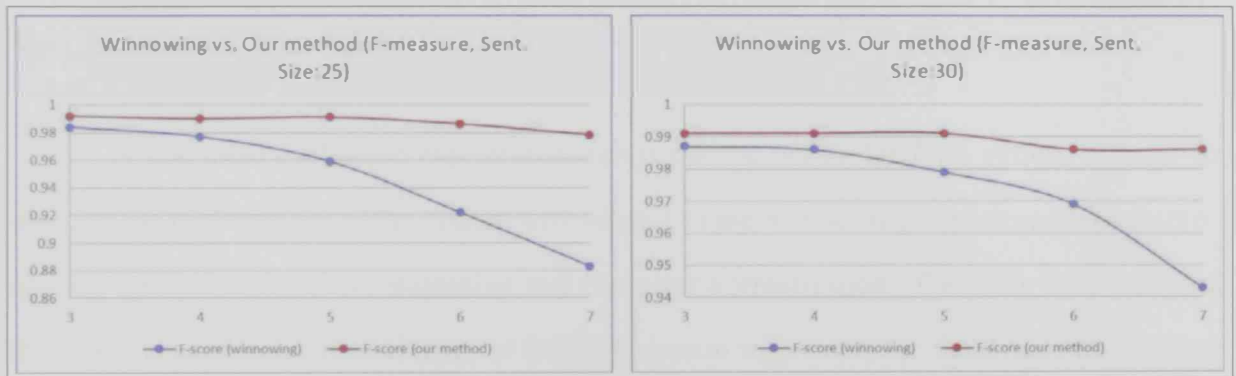


Figure 5.18: Comparison between the winnowing method and our DETRA based on the F-measure for sentences sizes= 25 and 30. DETRA outperforms the winnowing-based system

Figure 5.19 depicts the resulting TRDS scores of DETRA compared to the winnowing-based system for the  $n$ -gram sizes from 3 to 7, and sentence length 30. For winnowing, the window size is set to 10.

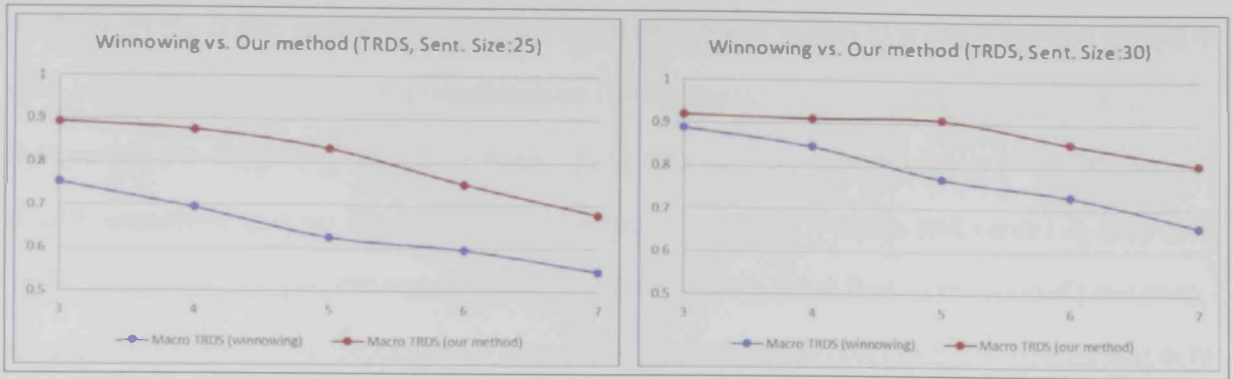


Figure 5.19: Comparison between the winnowing-based system and DETRA in terms of the overall text reuse detection scores (Macro TRDS) for sentences sizes 25 and 30. The proposed system DETRA outperforms the winnowing algorithm

## 5.5 Statistical Analysis

In this section, we follow the general framework for experimentation proposed by [109]. The structured experimentation consists of the following phases: definition, planning, operation, and interpretation.

### 5.5.1 Experiment Definition

A statistical evaluation experimentation is carried out on DETRA system in order to evaluate the performance of the system with respect to the Arabic language-dependent preprocessing operations, namely: stemming and character normalization. The main motivation of this study is to examine the behavior of DETRA system with respect to the Arabic text related preprocessing operations (the object of study) and to assess the effect of these operations in the overall performance of DETRA.

### 5.5.2 Experiment Planning

In this phase, the study planning phase is discussed based on three main aspects: 1) design, 2) criteria, and 3) measurement.

Four different models of the system have been designed and produced along with the exact model. The models are labelled model E to model A, where model E is the exact fit

model with the actual reused cases in each input document, model D is our proposed DETRA with stemming and character normalization (S+N) applied on the input documents, model C bypasses the stemming operation while applying character normalization (wS +N), model B with stemming and no character normalization applied (S + wN), and model A does not apply any of the two preprocessing operations (wS + wN). Note that in the model's notation, w refers to *without*. The experiments have been performed on 168 input documents and 600 source documents and each model has been compared with the exact fit model (model E), which shows the actual reuse between the input documents and the corresponding source documents.

The criteria used in the analysis of each model is accomplished by examining the overall similarity percentage for each input document, which is the ratio of the shared characters over the total characters in each input document. Another criteria is the detected number of source documents with reused text has been computed as well.

A number of statistical measures have been produced for each model in order to validate its performance. These include:

- The maximum, minimum, mean and median of the similarity percentage computed and the total number of source documents detected for each input document.
- The variance, which measures the variability of a model behaviour for a given input document. In other words, the variance is how much the similarity percentage for a given input document vary between different models of the system.
- The bias, which measures the difference between the expected performance of a given model in terms of the similarity percentage and the actual value which we are after.

### 5.5.3 Experiment Operation

The four different scenarios of the whole detection process with the presence or absence of the stemming or character normalization operations have been executed, creating four different models of the system. In this phase, we describe each model and compare it

with the exact model (model E).

Figure 5.20 depicts the exact fit model (model E) with our DETRA system (model D) that applies both preprocessing operations of stemming and character normalization on the Arabic input documents. The chart shows the overall similarity percentage (Y-axis) for each document sorted in ascending order (X-axis). The black curve in the figure illustrates the actual similarity percentage of the reused text in each document, while the blue curve shows the performance of DETRA in the current model D when compared with the exact fit model. The current model nicely fits the exact one with an accuracy of 92%. This is an obvious evidence, which suggests that the influence of stemming and character normalization of Arabic text may have a positive impact on the performance of the system, particularly, in the initial retrieval of the candidate source documents based on the generated fingerprints as well as the detailed comparison between a given input document with each retrieved candidate source document.

The resulting behaviour of model D has a very low bias towards the actual model, which measures the difference between the expected similarity percentage of our model and the correct value which we are trying to detect in model E. As bias measures how far off the model's results are from the exact fit, model D demonstrates the best bias to the exact fit model.

Figure 5.21 plots the chart diagram for model C of the system, which has been executed after applying the preprocessing operations including character normalization, but without stemming (model C - wS + N). The overall accuracy of the system decreases to 87% as more false positive and false negative cases were produced in this model. Moreover, the recall value is affected in this model due to the fact that some source documents were not successfully retrieved by the system as shown in Table 5.7 (Min. similarity % of model C).

Figure 5.22 displays a different scenario of DETRA, where stemming is applied on the documents while character normalization is omitted from the preprocessing phase. In this scenario, the retrieval of the candidate source documents may not be affected much by

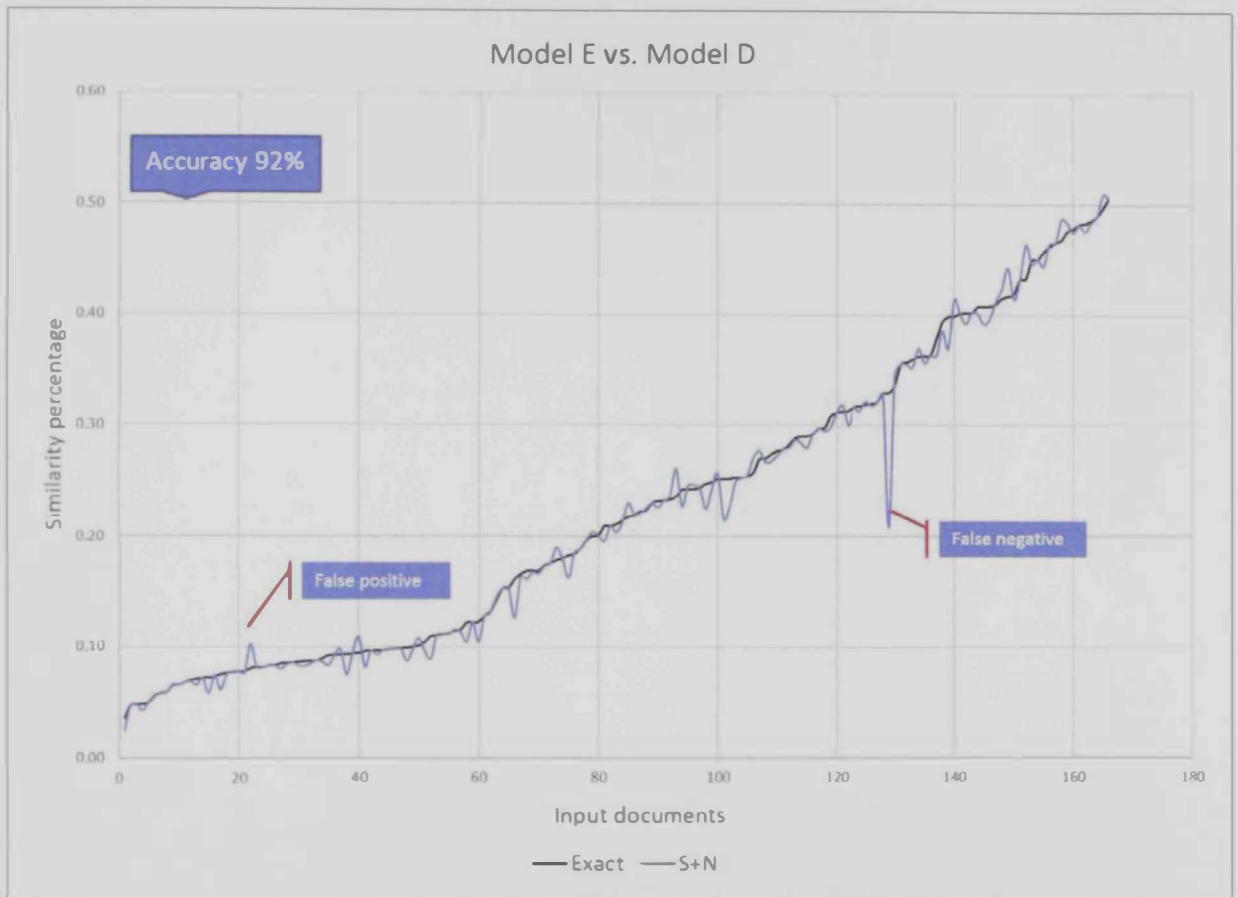


Figure 5.20: The exact fit model (model E), shown in black, and model D of DETRA system with both stemming and character normalization applied on the input text, shown in blue. The overall accuracy of the system is 92%



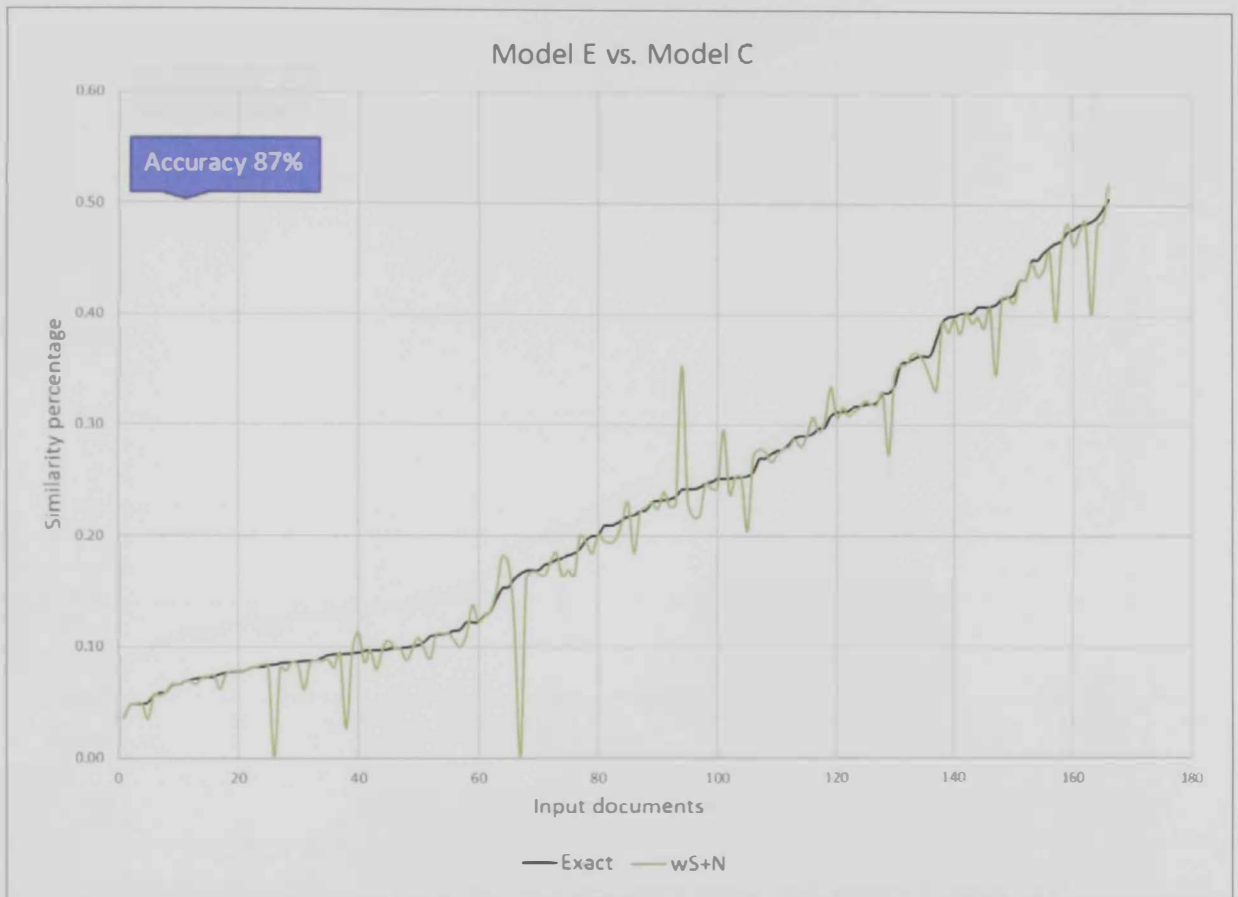


Figure 5.21: The actual similarity percentages in model E compared with the similarity percentages computed in model C of the system, where character normalization is applied on the text and stemming is bypassed. The overall accuracy of this model is 87%

neglecting the character normalization process, since stemming may trim most of the un-normalized characters such as the hamza (أ), taa-marbouta (ة), or yaa-maqsoua (ي) when producing stems of the words. However, the pairwise comparison between the input document and the retrieved source documents will be greatly affected by difference of one character between two, seemingly similar, words such as (احسان and إحسان).

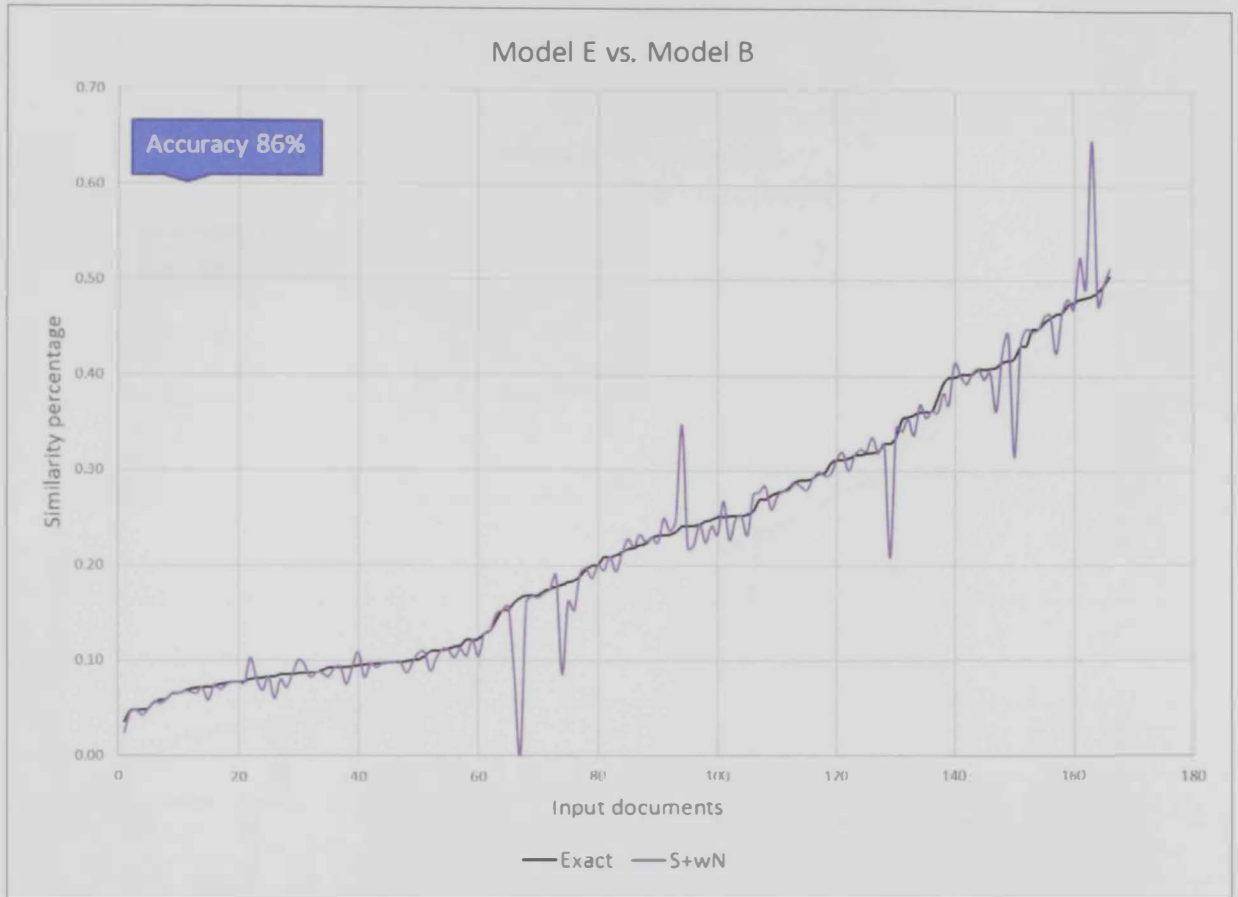


Figure 5.22: The exact model E, shown in black, compared with model B without the character normalization process. The overall accuracy of the model is 86%

The final model, model A, has been produced by the DETRA after bypassing both processes being examined ( $wS + wN$ ). This model obviously shows the worst performance of the system as shown in Figure 5.23. The computed accuracy of this model is the lowest among the other models (84%).



Figure 5.23: The worst performance of the system in model A, with disregarding of both stemming and character normalization processes. The overall accuracy is 84%

### 5.5.4 Experiment Interpretation

The combination of all the different models is depicted in Figure 5.24, which may be useful to assess the bias-variance relationship on the behaviour of the system between the different models. Since the bias measures how far these models similarity measures are from the correct value, model D demonstrates the lowest bias and hence produces more accurate results with the closest fit to the exact model. As for the variance, which measure how much the similarity estimation for a given input document vary between different models of the system, the lowest variance value indicates the “goodness” of the model.

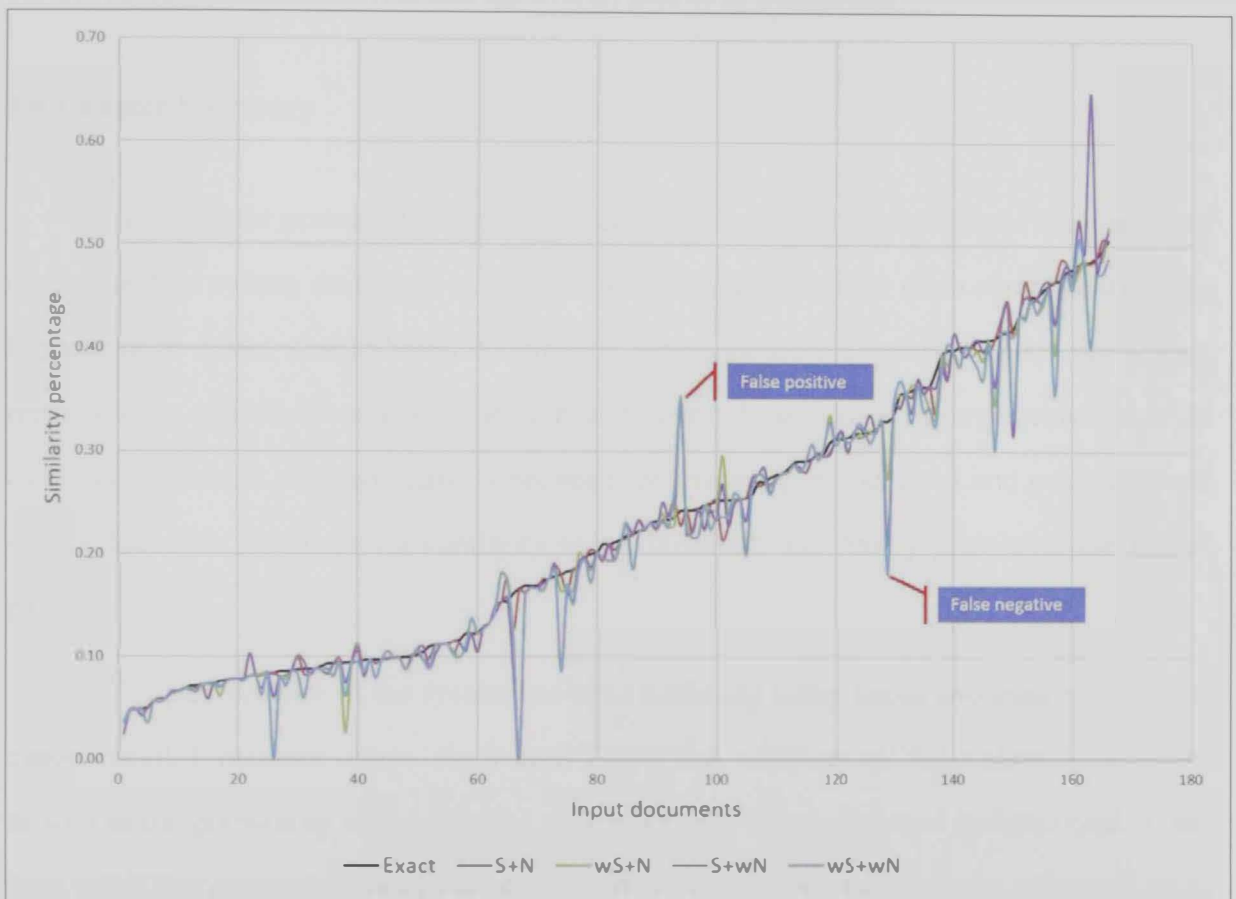


Figure 5.24: The combined four different models of the system with the exact model

Table 5.7 shows various statistical measures of the examined models based on the number of source documents detected with reused text as well as the estimated similarity percentage. The closest variance value to the exact model’s variance confirms the goodness of the proposed DETRA system in model D, with both stemming and character normalization

are applied to the Arabic text. In conclusion to this study, there is a strong evidence that Arabic language-dependent preprocessing operations such as stemming and character normalization have indeed an explicit impact on the overall behaviour of the system.

Measure	Model E (Exact)		Model D (De-TRA)		Model C (wS + N)		Model B0 (S + wN)		Model A (wS+ wN)	
	Sources	Similarity %	Sources	Similarity %	Sources	Similarity %	Sources	Similarity %	Sources	Similarity %
Max	52	51%	56	51%	54	52%	57	65%	54	51%
Min	1	4%	1	2%	1	0%	1	0%	1	0%
Median	3	21%	3	21%	3	20%	3	21%	3	19%
Mean	4.1	22%	4.4	22%	4.7	22%	4.7	22%	4.8	21%
Variance	26.8	1.8%	31.8	1.8%	33.3	1.8%	35.1	1.9%	34.8	1.7%

Table 5.7: The statistical measures computed for the four different models based on the number of reused source documents and the overall similarity percentage

## 5.6 Chapter Summary

This chapter presented the general fingerprinting-based framework of the local text reuse detection system developed in this research. We described the main components of the framework to detect Arabic-based documents, which includes candidate source documents retrieval from a given documents collection using our hybrid method of fingerprinting and IR techniques. Next, a detailed analysis between the given input document and each retrieved source document that passes a similarity score threshold, and finally presenting the reused passages.

The performance of the system has been evaluated using micro and macro based precision, recall, F-measure scores. An overall TRDS that combines all the evaluation measures as well as the granularity of the detected results has also been computed and analyzed. It has been noted that common phrases that stems from religious contents may raise a concern when developing text reuse detection systems for Arabic documents. Therefore, we think these characteristics of the Arabic text has to be taken into account.

The proposed fingerprinting method outperforms the winnowing fingerprinting approach in all the sets of experiments taking into account different  $n$ -gram sizes and sentences lengths. The winnowing method performs very well with small  $n$ -gram sizes, but breaks down



as the  $n$ -gram and sentences sizes increase. Our proposed method proved to be more robust for detecting text reuse particularly when the sentence length increases towards the average sentence length in the Arabic language.

## Chapter 6: Conclusions

In this chapter, we summarize the dissertation and comment on possible future work. The main aim of this research was to orient the research of automatic text reuse detection towards the Arabic language, being one of the mostly used languages in the World and on the Web as well. Yet, the research in this area is still in its infancy. Among the different types of text reuse, plagiarism, which can be defined as the unacknowledged reuse of text, has received a lot of the researchers attention in the past years especially on Western languages. However, very few works related to plagiarism detection in Arabic have been developed. On the other hand, we confidently state that there is no available systems on text reuse detection for Arabic-based documents on the Web.

The focus of this research was on the development of models for automatic text reuse on Arabic text. Special attention was given to the cases of text reuse carried out at the Web scale.

The detection problem of local text reuse was defined as distinguishing all reused passages within a given document. At the Web scale, this task was achieved in two main phases. The first phase was concerned with globally searching the Web for candidate relevant documents for a given set of input documents that are likely to be the original sources to some reused passages of a given document. In the second step, the local component of the text reuse detection problem selects the candidate source documents that are highly similar with a given input document and a detailed comparison between the given document and each of the possible sources of text reuse that have been retrieved was performed. Basically, this means extracting all the pairs of similar passages from the given document and the candidate sources and computing the similarity score between them.

### 6.1 Dissertation Summary

Concerning the main phase of Web documents retrieval, the system first creates an initial documents collection obtained from the Web, and then applies the detection techniques for finding text reuse with a given input document on this collection. For this purpose, we

developed a new efficient approach of queries formulation from a given document, which formulates several queries based on a fingerprinting approach to issue the Web through a public search engine interface in order to retrieve a set of documents. These documents would potentially contain most of the reused text (see Chapter 4). The candidate documents can then be downloaded, preprocessed, and indexed in order to be used by the local detection system for detailed similarity analysis with the given input document. We evaluated the work using a collection of documents especially constructed for the evaluation of Web documents retrieval. The experiments demonstrated that almost 80% of the Web documents used in the simulated reused cases were successfully retrieved. As for the local reuse detection, we developed a hybrid approach that is based on fingerprinting and IR heuristics to select the most relevant source documents to a given input document from the documents collection (see Chapter 5). It was compared with the winnowing fingerprinting method and the evaluation results showed that our proposed approach is more robust and outperformed the winnowing method in all the different sizes of  $n$ -grams. It was also observed that our method performs the best when the sentence length is closer to the reported average sentence length in the Arabic text.

### 6.1.1 Dissertation Contributions

The main contributions of this dissertation are:

1. Development of the first general framework for Arabic-based text reuse detection on the Web.
2. Development of a new model for queries formulation to acquire candidate source documents from the Web.
3. Development of a corpus for evaluating the system for Web retrieval of candidate documents.
4. Incorporation of IR heuristics along with the fingerprinting technique for query expansion to deal with reused cases that can not be detected using fingerprinting alone.

## 6.2 Future Research

This dissertation focused on the development of text reuse detection of Arabic text on the Web. The text reuse detection is an interesting and challenging area. Yet, it has not been given the attention it deserves for addressing the problem on the Arabic language. There are a number of problems that need to be addressed especially when tackling Arabic text, which may be interesting for possible continuation of this research. Possible future research may include the following:

- New approaches for expanding the generated fingerprints. The query expansion approach employed in this research produces more modified fingerprints that address the reordering feature of the Arabic text as well as the deletion of some terms in the text. In this research, all different reordering of a given text is generated as well as the deletion of one term at a time in a given fingerprint. A possible improvement of this approach is to integrate linguistic approaches to prune out the expanded fingerprints that may not be grammatically correct. A possible approach may be by associating lexical information to the words of the modified fingerprints and preserve only the ones that are grammatically valid. We anticipate that this process will possibly reduce the amount of generated fingerprints and hence improve the overall efficiency and efficiency of the system.
- Exploration or Development of special Arabic lexical resource for text reuse detection. It would be useful to carry out a study which explores the available Arabic lexical resources and compare between them. Another interesting direction is to create a thesaurus of paraphrased phrases in Arabic that may not be detected even with synonyms substitution of the words in a fingerprint.
- Incorporating resources for the Holy Quran and the Hadeeth, which is the saying of the prophet, peace be upon him, in order to cope with the characteristic of the Arabic text that usually includes religious content. A development of such module would be useful in order to tag such content and filter them out from the retrieval process.

## References

- [1] Bensalem, I., Boukhalfa, I., Rosso, P., Abouenour, L., Darwish, K., and Chikhi, S., "Overview of the AraPlagDet PAN@ FIRE2015 Shared Task on Arabic Plagiarism Detection," in *Notebook Papers of FIRE 2015, FIRE-2015*, pp. 114–125, 2015.
- [2] Bendersky, M. and Croft, W. B., "Finding Text Reuse on the Web," in *Proceedings of the Second ACM International Conference on Web Search and Data Mining, WSDM '09*, (New York, NY, USA), pp. 262–271, ACM, 2009.
- [3] Martin Potthast, *Technologies for Reusing Text from the Web*. PhD dissertation, Bauhaus-Universitat Weimar, 2011.
- [4] Seo, J. and Croft, B. W., "Local Text Reuse Detection," in *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, (New York, NY, USA), pp. 571–578, ACM, 2008.
- [5] Bernstein, Y. and Zobel, J., "A Scalable System for Identifying Co-derivative Documents," in *String Processing and Information Retrieval* (Apostolico, A. and Melucci, M., eds.), vol. 3246 of *Lecture Notes in Computer Science*, pp. 1–11, Springer Berlin / Heidelberg, 2004.
- [6] Broder, A. Z., Glassman, S. C., Manasse, M. S., and Zweig, G., "Syntactic Clustering of the Web," *Computer Networks and ISDN Systems*, vol. 29, pp. 1157–1166, Sept. 1997.
- [7] Heintze, N., "Scalable Document Fingerprinting," in *Proc. USENIX Workshop on Electronic Commerce*, pp. 1–10, 1996.
- [8] Hoad, T. C. and Zobel, J., "Methods for Identifying Versioned and Plagiarised Documents," *The American Society for Information Science and Technology*, vol. 54, no. 3, pp. 203–215, 2003.
- [9] Manber, U., "Finding Similar Files in a Large File System," in *Proceedings of the USENIX Winter 1994 Technical Conference*, (San Fransisco, CA, USA), pp. 1–10, 1994.
- [10] Shivakumar, N. and Garcia-Molina, H., "A Copy Detection Mechanism for Digital Documents," in *In Proceedings of 2nd International Conference on Theory and Practice of Digital Libraries*, 1995.
- [11] Metzler, D., Bernstein, Y., Croft, W. B., Moffat, A., and Zobel, J., "Similarity Measures for Tracking Information Flow," in *CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management*, (New York, NY, USA), pp. 517–524, ACM, 2005.



- [12] Clough, P., Gaizauskas, R., Piao, S., and Wilks, Y., "METER: MEasuring TExt Reuse," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pp. 152–159, ACM Press, 2002.
- [13] Clough, P., *Measuring Text Reuse*. PhD dissertation, University of Sheffield, 2003.
- [14] McEnery, T., "Looking at Text Re-Use in a Corpus of Seventeenth-Century News Reportage," 2003.
- [15] Clough, P., "Plagiarism in Natural and Programming Languages: An Overview of Current Tools and Technologies," tech. rep., University of Sheffield, 2000, 2000.
- [16] Clough, P., "Old and New Challenges in Automatic Plagiarism Detection. National UK Plagiarism Advisory Service," 2003.
- [17] Maurer, H., Kappe, F., and Zaka, B., "Plagiarism : A Survey," *Journal of Universal Computer Science*, vol. 8, no. 12, pp. 1050–1084, 2006.
- [18] Roy, C. K. and Cordy, J. R., "Scenario-Based Comparison of Clone Detection Techniques," in *Proceedings of the 2008 The 16th IEEE International Conference on Program Comprehension, ICPC '08*, (Washington, DC, USA), pp. 153–162, IEEE Computer Society, 2008.
- [19] Roy, C. K., Cordy, J. R., and Koschke, R., "Comparison and Evaluation of Code Clone Detection Techniques and Tools: A Qualitative Approach," *Science of Computer Programming*, vol. 74, pp. 470–495, May 2009.
- [20] Shivakumar, N. and Garcia-Molina, H., "Building a Scalable and Accurate Copy Detection Mechanism," in *In Proceedings of 1st ACM Conference on Digital Libraries*, pp. 160–168, 1996.
- [21] Brin, S., Davis, J., and Garcia-molina, H., "Copy Detection Mechanisms for Digital Documents," in *Proceedings of the ACM SIGMOD Annual Conference*, pp. 398–409, 1995.
- [22] Broder, A. Z., "Identifying and Filtering Near-Duplicate Documents," in *Proceedings of the 11th Annual Symposium on Combinatorial Pattern Matching, COM '00*, (London, UK, UK), pp. 1–10, Springer-Verlag, 2000.
- [23] Charikar, M., "Similarity Estimation Techniques From Rounding Algorithms," in *Proceedings of the Annual ACM Symposium on Theory of Computing*, pp. 380–388, 2002.
- [24] Bernstein, Y. and Zobel, J., "Accurate Discovery of Co-Derivative Documents via Duplicate Text Detection," *Information Systems*, vol. 31, no. 7, pp. 595–609, 2006.
- [25] Henzinger, M., "Finding Near-Duplicate Web Pages: A Large-Scale Evaluation of Algorithms," in *SIGIR'06: Proceedings of the 29th Annual International ACM SIGIR*

*Conference on Research and Development in Information Retrieval, SIGIR '06*, (New York, NY, USA), pp. 284–291, ACM Press, 2006.

- [26] Schleimer, S., Wilkerson, D. S., and Aiken, A., “Winnowing: Local Algorithms for Document Fingerprinting,” in *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data 2003*, pp. 76–85, ACM Press, 2003.
- [27] Mittelbach, A., Lehmann, L., Rensing, C., and Steinmetz, R., “Automatic Detection of Local Reuse,” in *Sustaining TEL: From Innovation to Learning and Practice* (Wolpers, M., Kirchner, P., Scheffel, M., Lindstaedt, S., and Dimitrova, V., eds.), vol. 6383 of *Lecture Notes in Computer Science*, pp. 229–244, Springer Berlin / Heidelberg, 2010.
- [28] Chiu, S., Uysal, I., and Croft, W. B., “Evaluating Text Reuse Discovery on the Web,” in *IliX 2010*, (New Brunswick, New Jersey, USA), pp. 299–302, ACM, 2010.
- [29] Stein, B. and Eissen, S. Z., “Near Similarity Search and Plagiarism Analysis,” in *From Data and Information Analysis to Knowledge Engineering Studies in Classification, Data Analysis, and Knowledge Organization*, pp. 430–437, 2006.
- [30] Wise, M., “String Similarity via Greedy String Tiling and Running Karp-Rabin Matching. Dept. of CS. University of Sidney,” 1992.
- [31] Burrows, S., Tahaghoghi, S. M. M., and Zobel, J., “Efficient Plagiarism Detection for Large Code Repositories,” *Software: Practice and Experience*, vol. 37, no. 2, pp. 151–175, 2007.
- [32] Gusfield, D., *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge Univ. Press, 1997.
- [33] Monostori, K., Finkel, R. A., Zaslavsky, A. B., Hodszy, G., and Pataki, M., “Comparison of Overlap Detection Techniques,” in *ICCS '02 Proceedings of the International Conference on Computational Science-Part I*, pp. 51–60, Springer-Verlag, 2002.
- [34] Si, A., Leong, H. V., and Lau, R. W. H., “Check: A document Plagiarism Detection System,” in *Proceedings of ACM Symposium for Applied Computing*, pp. 70–77, ACM, 1997.
- [35] Broder, A. Z., “On the Resemblance and Containment of Documents,” in *In Compression and Complexity of Sequences*, pp. 21–29, IEEE Computer Society, 1997.
- [36] M.Kashkur, Parshutin, S., and Borisov, A., “Research Into Plagiarism Cases and Plagiarism Detection Methods,” *Scientific journal of Riga Technical University*, vol. 44, pp. 139–144, 2010.
- [37] Belkhouche, B., Nix, A., and Hassell, J., “Plagiarism Detection in Software Designs,” in *Proceedings of the 42nd Annual Southeast Regional Conference*, pp. 207–211, ACM, 2004.

- [38] Ottenstein, K., "An Algorithm Approach to the Detection and Prevention of Plagiarism," *ACM SIGCSE Bulletin*, vol. 8, no. 4, pp. 30–41, 1976.
- [39] Stein, B., zu Eissen, S. M., and Potthast, M., "Strategies for Retrieving Plagiarized Documents," in *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, (New York, NY, USA), pp. 825–826, ACM, 2007.
- [40] Lalmas, M., MacFarlane, A., Rüger, S. M., Tombros, A., Tsirikika, T., and Yavlingsky, A., eds., *Advances in Information Retrieval, April 10-12*, vol. 3936 of *Lecture Notes in Computer Science*. (London, UK), Springer, 2006.
- [41] BBC, "The Ctrl+C, Ctrl+V boom. BBC News Magazine," 2011.
- [42] Suri, H., "Evaluation of Two Turnitin Trials in the Faculty of Law," tech. rep., Monash University, 2007.
- [43] Hagen, M. and Stein, B., "Candidate Document Retrieval for Web-Scale Text Reuse Detection," in *Proceedings of the 18th International Conference on String Processing and Information Retrieval*, SPIRE'11, (Berlin, Heidelberg), pp. 356–367, Springer-Verlag, 2011.
- [44] Alzahrani, S. and Salim, N., "Fuzzy Semantic-Based String Similarity for Extrinsic Plagiarism Detection Lab Report for PAN at CLEF 2010," 2010.
- [45] Jadalla, A. and Elnagar, A., "Iqtebas 1.0: A Fingerprinting-Based Plagiarism Detection System for Arabic Text-Based Documents," *IJMIA: International Journal on Data Mining and Intelligent Information Technology Applications*, vol. 2, no. 2, pp. 31–43, 2012.
- [46] Menai, M., "Detection of Plagiarism in Arabic Documents," *International Journal of Information Technology and Computer Science*, vol. 10, pp. 80–89, 2012.
- [47] Jaoua, M., Jaoua, F., Belguith, L. H., and Hamadou, A. B., "Automatic Detection of Plagiarism in Arabic Documents Based on Lexical Chains (in Arabic)," *Arab Computer Society Journal*, vol. 4, pp. 1–11, 2011.
- [48] Bensalem, I., Rosso, P., and Chikhi, S., "Intrinsic Plagiarism Detection in Arabic Ttext: Preliminary Experiments," in *Proceedings of the 2nd Spanish Conference on Informantino Retrieval*, (Spain), pp. 325–329, 2012.
- [49] Ramakishna, M. V. and Zobel, J., "Performance in Practice of String Hashing Functions," in *Proceedings of the International Conference on Database Systems for Advanced Applications*, (Australia), pp. 215–223, 1997.
- [50] Rivest, R. L., "The MD5 Message-Digest Algorithm," 1992.

- [51] Lyon, C., Malcolm, J., and Dickerson, B., "Detecting Short Passages of Similar Text in Large Document Collections," in *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, pp. 118–125, 2001.
- [52] Abdel-Hamid, O., Behzadi, B., Christoph, S., and Henzinger, M., "Detecting the Origin of Text Segments Efficiently," in *WWW '09: Proceedings of the 18th International Conference on World Wide Web*, pp. 61–70, 2009.
- [53] Ahmed, N., "How I Came Up With the Discrete Cosine Transform," *Digital Signal Processing*, vol. 1, no. 1, pp. 4–5, 1991.
- [54] Seidel, R., "Backwards Analysis of Randomized Geometric Algorithms," in *Trends in Discrete and Computational Geometry, volume 10 of Algorithms and Combinatorics*, pp. 37–68, Springer-Verlag, 1992.
- [55] Ganguly, S. and Veda, G., "A New Randomized Algorithm for Document Fingerprinting," tech. rep., Indian Institute of Technology, 2000.
- [56] Stein, B., "Fuzzy-Fingerprints for Text-Based Information Retrieval," in *Proceedings of I-KNOW 2005*, pp. 572–579, 2005.
- [57] Landauer, T. K., Foltz, P. W., and Laham, D., "An Introduction to Latent Semantic Analysis," *Discourse Processes*, no. 25, pp. 259–284, 1998.
- [58] Ceska, Z., "Plagiarism Detection Based on Singular Value Decomposition," in *GoTAL*, pp. 108–119, 2008.
- [59] Frakes, W. B., *Information Retrieval Data Structures and Algorithms*. Prentice-Hall, Inc., 1992.
- [60] Lancaster, F. W., *Information Retrieval Systems: Characteristics, Testing and Evaluation*. Wiley, New York, 1968.
- [61] Cleverdon, C. W., Mills, J., and Keen, M., "Factors Determining the Performance of Indexing Systems," in *ASLIB Cranfield project, Cranfield*, pp. 05–14, 1966.
- [62] Makhoul, J., Kubala, F., Schwartz, R., , and Weischedel, R., "Performance Measures for Information Extraction," in *Proceedings of DARPA Broadcast News Workshop*, (Herndon, VA), pp. 249–252, 1999.
- [63] Goffman, W. and Newilly, V., "A Methodology for Test and Evaluation of Information Retrieval Systems," *Journal of Information Storage and Retrieval*, vol. 3, pp. 19–25, 1966.
- [64] Baeza-Yates, R. and Ribeiro-Neto, B., *Modern Information Retrieval*. New York, NY, USA: ACM Press, 1999.
- [65] Manning, C. D., Raghavan, P., and Schtze, H., *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008.



- [66] Robertson, S., "Salton Award Lecture on Theoretical Argument in Information Retrieval," *SIGIR Forum*, vol. 34, pp. 1–10, 2000.
- [67] Clough, P., Gaizauskas, R., and Piao, S., "Building and Annotating a Corpus for the Study of Journalistic Text Reuse," in *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC 2002)*, vol. 5, pp. 1678–1691, European Language Resources Association (ELRA), 2002.
- [68] Rose, T. G., Stevenson, M., and Whitehead, M., "The Reuters Corpus Volume 1 - From Yesterday's News to Tomorrow's Language Resources," in *Proceedings of the Third International Conference on Language Resources and Evaluation*, vol. 3, pp. 827–833, ACM Press, 2002.
- [69] Stein, B. and Meyer zu Eissen, S., "Intrinsic Plagiarism Analysis with Meta Learning," in *SIGIR 2007 Workshop on Plagiarism Analysis, Authorship Identification, and Near-Duplicate Detection (PAN 2007)* (Stein, B., Stamatatos, E., and Koppel, M., eds.), pp. 45–50, 2007.
- [70] Potthast, M., Stein, B., Eiselt, A., universitöt Weimar, B., cede no, A. B., and Rosso, P., "Overview of the 1st International Competition on Plagiarism Detection," in *In: SEPLN 2009 Workshop on Uncovering Plagiarism, Authorship, and Social Software Misuse (PAN 09)*, CEUR-WS.org, pp. 1–9, 2009.
- [71] Hussein, A., "A Plagiarism Detection System for Arabic Documents," in *Intelligent Systems* (Filev, D., Jablkowski, J., Kacprzyk, J., Krwczak, M., Popchev, I., Rutkowski, L., Sgurev, V., Sotirova, E., Szynekarczyk, P., and Zadrozny, S., eds.), pp. 541–552, Springer International Publishing, 2015.
- [72] Al-Muhtaseb, H. and Mellish, C., "Some Differences Between Arabic & English: A Step Towards an Arabic Upper Model," in *Proceedings of the 6th International Conference on Multilingual Computing*, pp. 1–12, 1998.
- [73] Attia, M., *Handling Arabic Morphological and Syntactic Ambiguity within the LFG Framework with a View to Machine Translation*. PhD dissertation, University of Manchester, 2008.
- [74] Crystal, D., *A First Dictionary of Linguistics and Phonetics*. Westview Press, 1980.
- [75] Farghaly, A. and Shaalan, K., "Arabic Natural Language Processing: Challenges and Solutions," vol. 8, pp. 14:1–14:22, Dec. 2009.
- [76] Habash, N., "Introduction to Arabic Natural Language Processing,"
- [77] Grefenstette, G. and Tapanainen, P., "What is a Word, What is a Sentence? Problems of Tokenization," in *Proceedings of the 3rd Conference on Computational Lexicography and Text Research (COMPLEX-94)*, pp. 571–578, 1994.



- [78] Jurafsky, D. and Martin, J. H., *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall, 2000.
- [79] Krovetz, R., "Viewing Morphology as an Inference Process," in *the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Korfhage, R., Rasmussen, E., and Willett, P., eds.), pp. 191–202, ACM Press, 1993.
- [80] Baeza-Yates, R. and Ribeiro-Neto, B., *Modern Information Retrieval*. ACM Press, New York, NY, 1999.
- [81] Hull, D., "Stemming Algorithms: A Case Study for Detailed Evaluation," *Journal of the American Society for Information Science*, vol. 47, no. 1, pp. 70–84, 1996.
- [82] Larkey, L., Ballesteros, L., and Connell, M., "Improving Stemming for Arabic Information Retrieval: Light Stemming and Co-Occurrence Analysis," in *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, (ACM), pp. 282–287, 1999.
- [83] Habash, N., Rambow, O., and Roth, R., "MADA+TOKAN: A Toolkit for Arabic Tokenization, Diacritization, Morphological Disambiguation, POS Tagging, Stemming and Lemmatization," in *Proceedings of the Second International Conference on Arabic Language Resources and Tools* (Choukri, K. and Maegaard, B., eds.), pp. 102–109, The MEDAR Consortium, 2009.
- [84] Al-Kharashi, I. and Evens, M., "Comparing Words, Stems, and Roots as Index Terms in an Arabic Information Retrieval System," *Journal of the American Society for Information Science*, vol. 45, no. 8, p. 560, 1994.
- [85] Khoja, S. and Garside, R., "Stemming Arabic Text. Computing Department, Lancaster University, Lancaster, UK," 1999.
- [86] Aljlal, M. and Frieder, O., "On Arabic Search: Improving the Retrieval Effectiveness via a Light Stemming Approach," in *Proceedings of the eleventh international conference on Information and knowledge management*, pp. 347–352, ACM, 2011.
- [87] Darwish, K. and Oard, D., "CLIR Experiments at Maryland for TREC 2002: Evidence Combination for Arabic-English Retrieval," in *Proceedings of the Eleventh Text REtrieval Conference*, 2002.
- [88] L. Larkey, L. B. and Connell, M., "Light Stemming for Arabic Information Retrieval," *Arabic Computational Morphology*, pp. 221–243, 2007.
- [89] Smrž, O., "ElixirFM: Implementation of Functional Arabic Morphology," in *Proceedings of the 2007 Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources*, pp. 1–8, Association for Computational Linguistics, 2007.

- [90] Pasha, A., Al-Badrashiny, M., Diab, M. T., El Kholly, A., Eskander, R., Habash, N., Pooleery, M., Rambow, O., and Roth, R., "MADAMIRA: A Fast, Comprehensive Tool for Morphological Analysis and Disambiguation of Arabic," in *LREC*, vol. 14, pp. 1094–1101, 2014.
- [91] Buckwalter, T., "Qamus: Arabic Lexicography," 2002.
- [92] Barrón-Cedeño, L. A., *On the Mono- and Cross-Language Detection of Text Re-Use and Plagiarism*. PhD dissertation, Universitat Politècnica de València, 2012.
- [93] Hiemstra, D., "Using language models for information retrieval," tech. rep., University of twente, 2001.
- [94] McDonald, R., Lerman, K., and Pereira, F., "Multilingual Dependency Analysis with a Twostage Discriminative Parser," in *Proceedings of the Tenth Conference on Computational Natural Language Learning*, pp. 216–220, ACM Press, 2006.
- [95] Manning, C. and Schütze, *Foundations of Statistical Natural Language Processing*. The MIT Press, 2002.
- [96] Rabin, M., "Fingerprinting by Random Polynomials. Technical Report TR-CSE-03-01," tech. rep., Center for Research in Computing Technology, Harvard University, Cambridge, MA, 1981.
- [97] LingPipe-Blog, "Natural Language Processing and Text Analytics: Lucene 4 Essentials for Text Search and Indexing," 2014.
- [98] Potthast, M., Hagen, M., Beyer, A., Busse, M., Tippmann, M., Rosso, P., and Stein, B., "Overview of the 6th International Competition on Plagiarism Detection," in *Working Notes Papers of the CLEF 2014 Evaluation Labs* (Cappellato, L., Ferro, N., Halvey, M., and Kraaij, W., eds.), CEUR Workshop Proceedings, CLEF and CEUR-WS.org, Sept. 2014.
- [99] Pôssas, B., Ziviani, N., Ribeiro-Neto, B., and Meira, W., Jr., "Maximal Termsets As a Query Structuring Mechanism," in *Proceedings of the 14th ACM International Conference on Information and Knowledge Management, CIKM '05*, (New York, NY, USA), pp. 287–288, ACM, 2005.
- [100] Dasdan, A., D'Alberto, P., Kolay, S., and Drome, C., "Automatic Retrieval of Similar Content Using Search Engine Query Interface," in *Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November 2-6, 2009*, pp. 701–710, 2009.
- [101] Hagen, M. and Stein, B., "Candidate Document Retrieval for Web-Scale Text Reuse Detection," in *String Processing and Information Retrieval, 18th International Symposium, SPIRE 2011, Pisa, Italy, October 17-21, 2011. Proceedings*, pp. 356–367, 2011.

- [102] Ghani, R., Jones, R., and Mladenic, D., "Automatic Web Search Query Generation to Create Minority Language Corpora," in *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, (New York, NY, USA), pp. 432–433, ACM, 2001.
- [103] Yang, Y., Bansal, N., Dakka, W., Ipeirotis, P., Koudas, N., and Papadias, D., "Query by Document," in *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, WSDM '09, (New York, NY, USA), pp. 34–43, ACM, 2009.
- [104] Álvaro R. Pereira and Ziviani, N., "Retrieving Similar Documents from the Web," *J. Web Eng.*, vol. 2, no. 4, pp. 247–261, 2003.
- [105] Chakrabarti, S., *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan-Kaufman, 2002.
- [106] Khan, I., Siddiqui, M., Jambi, K., Imran, M., and Bagais, A., "Query Optimization in Arabic Plagiarism Detection: An Empirical Study," *International Journal of Intelligent Systems and Applications*, vol. 7, no. 1, pp. 73–79, 2015.
- [107] Heap, B., "Permutations by Interchanges," *The Computer Journal*, vol. 6, no. 3, pp. 293–298, 1963.
- [108] Potthast, M., Stein, B., no, A. B.-C., and Rosso, P., "An Evaluation Framework for Plagiarism Detection," in *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, (Stroudsburg, PA, USA), pp. 997–1005, Association for Computational Linguistics, 2010.
- [109] Basili, V. R., Selby, R., and Hutchens, D. H., "Experimentation in software engineering," *IEEE Transactions on Software Engineering*, vol. 12, no. 7, pp. 733–743, 1986.