$N°$ d'ordre : 77/2017-C/INF

République Algérienne Démocratique et Populaire
Ministère de L'Enseignement Supérieur et de la Recherche Scientifique
Université des Sciences et de la Technologie HOUARI BOUMEDIENE

Faculté d'électronique et d'informatique

Thèse

Présentée pour l'obtention du **diplôme** de **DOCTORAT** $3^{ème}$ cycle (LMD)

En : INFORMATIQUE

Spécialité : Intelligence Artificielle

Par : BELKEBIR Riadh

Sujet

Résumé automatique de la langue Arabe par abstraction

Soutenue publiquement, le 05/11/2017, devant le jury composé de :

M.	BABA ALI Riadh	Professeur	à l'USTHB	Président
M.	GUESSOUM Ahmed	Professeur	à l'USTHB	Directeur de thèse
Mme.	KHELLAF Faiza	Professeur	à l'USTHB	Examinatrice
Mme.	BELKREDIM F.Z	Maître de Conférences A	à l'U.Chlef	Examinatrice
M.	AZZOUNE Hamid	Maître de Conférences A	à l'USTHB	Examinateur
M.	ABBAS Mourad	Maître de Recherches A	au CRSTDLA	Examinateur

I would like to dedicate this thesis to my loving parents ...

Acknowledgements

First and foremost, I would like to thank God Almighty for giving me the opportunity, strength, and knowledge to undertake this Ph.D thesis and to complete it satisfactorily. Without his guidance, this achievement would not have been possible.

I would like to express my sincere gratitude to my advisor, professor Ahmed Guessoum, for the continuous support of my Ph.D thesis, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. Besides my advisor, I would like to thank the members of my thesis committee: Prof. Riadh Baba Ali, Prof. Faiza Khellaf, Dr. Fatma Zohra Belkredim, Dr. Hamid Azzoune and Prof. Mourad Abbas for generously offering their time, support, guidance and good will throughout the preparation and review of this document.

Last but not the least, I would like to thank my family for supporting me throughout writing this thesis.

Abstract

This work addresses the task of automatic Arabic text summarization. It puts forward a global summarization framework (TALAA-ATSF) which allows the selection of the suitable operation to be used on each fragment of the source text. This is made possible through the use of machine learning techniques that learn which operation to apply to a given fragment (a subset of sentences) of the source document. Both abstractive and extractive operations have been proposed. We have proposed an extractive text summarization operation that uses AdaBoost to select the most relevant sentences. We have also designed an abstractive text summarization operation that allows the generation of sentences by generalizing and fusing the concepts used in these sentences. Furhermore, we have built a sentence compression corpus (TALAA-ASC) and enhanced this corpus to use it in the global text summarization framework (TALAA-ATSF).

Résumé

Ce travail traite de la tâche de résumé automatique de textes en Arabe. Il propose un cadre global de résumé (TALAA-ATSF) qui permet de sélectionner l'opération la plus appropriée à utiliser sur chaque partie du texte source. Cette approche est rendue possible grâce à l'utilisation de techniques d'apprentissage automatique pour apprendre quelle opération appliquer à un fragment donné (un sous-ensemble de phrases) du document source. Des opérations d'extraction et d'abstraction ont été proposées. Nous avons proposé une opération extractive de résumé de texte qui utilise AdaBoost pour sélectionner les phrases les plus pertinentes. Nous avons également conçu une opération abstractive de résumé de texte qui permet la génération de phrases en généralisant et fusionnant les concepts utilisés dans ces phrases. De plus, nous avons construit un corpus de compression de phrases (TALAA-ASC) et amélioré ce corpus pour l'utiliser dans le cadre global de résumé de textes (TALAA-ATSF).

الملخص

تتناول هذه الرسالة موضوع التلخيص الآلي لنصوص عربية وهذا من خلال اقتراح إطار عام للتلخيص (TALAA-ATSF) يمكن من خلاله اختيار العملية المناسبة التي سيتم استخدامها مع كل جزء من النص الأصلي. والحل المقترح في هذه الرسالة يستخدم تقنيات التعلم الآلي لمعرفة العملية الأفضل التي يحسن استعمالها مع جزء معين من النص المصدر للقيام بعملية التلخيص الآلي. وقد اقترحنا استعمال عمليات تلخيص من النوع التجريدي والاستخراجي. فاقترحنا عملية تلخيص استخراجية تستخدم خوارزمية AdaBoost لانتقاء الجمل الأكثر أهمية. كما قمنا بتصميم أسلوب تجريدي يسمح بتوليد الجمل من خلال تعميم ودمج الكلمات المستخدمة في هذه الجمل. بالإضافة لهذا فإننا قمنا ببناء مدونة (TALAA-ASC) لتقليص الجمل وطورناها بحيث يمكن استخدامها في الإطارالعام لتلخيص النصوص (TALAA-ATSF) .

Declaration

This dissertation is the result of my own work and includes nothing which is the outcome of work done by others. I state that every single definition, sentence, paragraph, etc., taken from any other work is referenced as indicated in the research agreements text, except if the phrase is frequently used and the information is public knowledge. I further state that no part of my dissertation has already been submitted, or, is being concurrently submitted for any such degree, diploma or other qualification at USTHB or any other university or similar institution.

Riadh Belkebir

Table of contents

List of figures xv

List of tables xvii

1 General Introduction 1

1.1 Automatic Summarization . . . 1
1.2 Problem Statement . . . 2
1.3 Contributions . . . 3
1.4 Thesis Overview . . . 4
1.5 Published Work . . . 5

2 State of the Art in Arabic Text Summarization 7

2.1 Introduction . . . 7
2.2 Summarization Techniques . . . 8
2.2.1 Extractive Text Summarization . . . 8
2.2.2 Abstrative Text Summarization . . . 15
2.3 Analytical Study . . . 18
2.3.1 Understanding Text Summarization . . . 18
2.3.2 Text Summarization Taxonomies . . . 24
2.4 Arabic Text Summarization . . . 26
2.4.1 Discussion . . . 31
2.5 Text Summarization with Other Applications . . . 32
2.5.1 Text Categorization . . . 32
2.5.2 Opinion Mining . . . 32
2.5.3 Information Retrieval . . . 32
2.6 Conclusion . . . 33

3 A Supervised Approach to Arabic Text Summarization Using AdaBoost 35
3.1 Introduction . . . 35
3.2 The AdaBoost-based summarizer . . . 36
3.2.1 The AdaBoost algorithm . . . 36
3.2.2 System design . . . 38
3.3 Implementation and test . . . 40
3.3.1 Corpus . . . 40
3.3.2 Pretreatment and Normalization of the Arabic Text . . . 41
3.3.3 Comparison . . . 41
3.3.4 Discussion . . . 42
3.4 Conclusion . . . 43

4 Concept Generalization and Fusion for Abstractive Sentence Generation 45
4.1 Introduction . . . 45
4.2 Problem Statement . . . 46
4.2.1 Definitions . . . 47
4.3 System Design . . . 51
4.3.1 Dependency Parsing . . . 51
4.3.2 Detection of Generalizable Sentences . . . 52
4.3.3 Generation and Reduction of the Space of Generalization Versions 52
4.3.4 Heuristics to Select Acceptable Versions from a Given SGV . . . 54
4.3.5 Generation of the Compressed Sentence . . . 59
4.4 Experimentation and Evaluation . . . 61
4.4.1 Evaluation Methodology . . . 61
4.4.2 Evaluation of the Selection of Acceptable Versions Based on the Algorithm of Appearance of Concepts within the Same Context 62
4.4.3 Evaluation of the Ml-Based Approach . . . 63
4.4.4 Comparison . . . 64
4.5 A Running Example . . . 65
4.6 Discussion . . . 69
4.7 Conclusion . . . 73

5 TALAA-ASC: A Corpus for Arabic Sentence Compression 75
5.1 Introduction . . . 75
5.2 The TALAA-ASC corpus . . . 77

5.2.1 Annotation Guidelines . . . 77
5.2.2 Corpus structuring using XML . . . 78
5.3 Statistics . . . 78
5.3.1 Size of the TALAA-ASC corpus . . . 78
5.3.2 Distribution of sentences according to their length . . . 78
5.3.3 Percentage of Drops . . . 81
5.3.4 Compression rate . . . 81
5.3.5 Scatter plots of the lengths of source sentences against compression rate for the TALAA-ASC corpus . . . 82
5.3.6 Graph bars for the distribution of compression rates for the TALAA-ASC corpus . . . 82
5.3.7 Why do some sentences remain uncompressed? . . . 85
5.4 Conclusion . . . 88

6 TALAA-ATSF: A Global Operation-Based Arabic Text Summarization Framework 91
6.1 Introduction . . . 91
6.2 Problem Statement . . . 92
6.2.1 Definitions . . . 93
6.3 System Design . . . 97
6.3.1 General idea . . . 97
6.3.2 Why a multi-layer graph? . . . 98
6.3.3 The Process of Summary Generation . . . 100
6.4 Evaluation of the Global Summarization System . . . 104
6.4.1 Evaluation of the Algorithm for Affectation of Operations to Document Partitions . . . 104
6.4.2 Evaluation of the Summarization Framework . . . 107
6.5 Extension of the framework: TALAA-ATSF with BSO-CHI2-SVM . . . 108
6.6 Discussion . . . 111
6.7 Conclusion . . . 112

7 Conclusions 113
7.1 Main Contributions . . . 113
7.2 Further Research . . . 114

List of figures

1.1 General architecture of the proposed global text summarization framework 6

3.1 AdaBoost summarization system in the global summarization framework 36
3.2 Stage one: Building the learning model 38
3.3 Stage two: summary construction 39

4.1 Concept generalization and fusion in the global summarization framework 46
4.2 Architecture of the Concept Generalization and Fusion System 51
4.3 Space of generalization versions 53
4.4 F1-Score of the system as a function of the threshold of the selection of Acceptable Versions using the Algorithm of Appearance of Concepts within the same Context 63

5.1 The summarization corpus in the global summarization framework . . . 76
5.2 Example of a document from the TALAA-ASC corpus 79
5.3 Distribution of sentences according to their length in the TALAA-ASC corpus. 81
5.4 Relation between the length of a sentence and its compression for the five categories of the TALAA-ASC corpus 83
5.5 Relation between the length of a sentence and its compression for the TALAA-ASC corpus 84
5.6 Compression rate distribution for the “management and work” category in the TALAA-ASC corpus 85
5.7 Compression rate distribution for the “medicine” category in the TALAA-ASC corpus 86
5.8 Compression rate distribution for the “news and politics” category in the TALAA-ASC corpus 86
5.9 Compression rate distribution for the “sports” category in the TALAA-ASC corpus 87

5.10 Compression rate distribution for the “technology” category in the TALAA-ASC corpus 87
5.11 Compression rate distribution for the TALAA-ASC corpus 88

6.1 Similarity graph of a document 103
6.2 Example of a document from the annotated TALAA-ASC corpus . . . 106
6.3 Architecture of BSO-CHI2-SVM 110

List of tables

2.1 Extractive Text Summarization Techniques 9
2.2 Text summarization operations 21
2.3 The taxonomy of Jones et al. (1999) 24
2.4 The taxonomy of Mani and Maybury (1999) 25
2.5 The taxonomy of Lloret and Palomar (2012) 27

3.1 Comparaison between AdaBoost, J48 and MLP in terms of F1 score . . 42

4.1 List of highly abstractive concepts in WordNet 50
4.2 Confusion Matrix 61
4.3 Comparison of the Ml-based techniques 64
4.4 Comparison between the approaches that filter the set of acceptable generalization versions 65
4.5 Passage from the file *"chesterton-ball.txt"* of the *"gutenberg"* corpus . . 65
4.6 Paths of the concept *"biscuits"* 66
4.7 Paths of the concept *"meat"* 66
4.8 Paths of the concept *"milk"* 66
4.9 Merged paths for the concept *"biscuits"* 67
4.10 Merged paths for the concept *"meat"* 67
4.11 Merged paths for the concept *"milk"* 67
4.12 Sample from the space of generalization versions of the generalizable sentence *"I have the biscuits and the tinned meat, and the milk"* 68

5.1 Number of documents in each category in TALAA-ASC 78
5.2 Number of sentences in each category in TALAA-ASC 78
5.3 Number of words in each category in TALAA-ASC 78

5.4 Percentage of part-of-speech (POS) tags dropped for the TALAA-ASC corpus. #drop refers to the frequency the POS tag was dropped from the source data. % drop is the percentage of times the POS tag was dropped in forming the compression. 80
5.5 Compression in each category in TALAA-ASC 82
5.6 Compression range for the five categories of the TALAA-ASC corpus . 82
5.7 Percentage of uncompressed sentences in the different categories of the TALAA-ASC corpus 88

6.1 Evaluation of the Algorithm for Affectation of Operations 107
6.2 Evaluation of the Summarization Framework 108

Chapter 1

General Introduction

1.1 Automatic Summarization

Text summarization is one of the most difficult, though promising, applications of Artificial Intelligence (AI) in general, and Natural Language Processing (NLP) more specifically. Various prestigious conferences and organizations have paid special attention to this field. One can mention the Association for the Advancement of Artificial Intelligence (AAAI[1]), the Document Understanding Conferences (DUC[2]) and, the Text Analysis Conference (TAC[3]). Various definitions of text summarization are given in the literature. Hovy and Marcu (2005) define a summary as a text which is produced from one or more texts, which contains a significant portion of the original text(s) information, and which is no longer than half of the original text(s). Mani and Maybury (1999) define the text summarization task as the process of finding the important contents in the original text and presenting them as a concise text in a predefined template. In the literature, various techniques have been proposed to tackle this problem.

Text summarization today is still far from achieving its ultimate goal which consists in generating summaries similar to the ones produced by humans. As we will show later, there is a lot of work that has to be done to produce summarization systems that can emulate the cognitive effort performed by humans when generating summaries. Various techniques have been applied in the field as an attempt to build summarization systems similar to the ones produced by humans. The approaches that have been proposed range from statistics, syntactic analysis, and some of them rely on some

[1] http://www.aaai.org/
[2] http://duc.nist.gov/
[3] http://www.nist.gov/tac/

semantic resources by trying to understand the meaning of the source text and giving it some form of representation.

One of the difficulties in text summarization is the evaluation of the generated summary which is still subjective. This task has been found difficult even for human assessors because it is not clear, what type of information a summary should contain as shown in previous studies (Nenkova, 2006). In fact, if a given summary is provided to different judges, they might asses it differently. We mention that a lot of effort has been done on the evaluation of text summarization and some metrics have been proposed like ROUGE (Lin, 2004).

In this work, we have reached several achievements. We have proposed an approach that selects the most relevant sentences from a document using AdaBoost (Belkebir and Guessoum, 2015a). We have also built a sentence compression corpus (Belkebir and Guessoum, 2015b). We have also developed an abstractive approach that generalizes and fuses the concepts used in these sentences (Belkebir and Guessoum, 2016). Although several text summarization approaches have been proposed in the literature, we have not found a summarization framework that uses different summarization operations to generate the summary in a flexible manner. One of our contributions consists in proposing such a framework that has the ability to combine several operations in a compositional manner (Belkebir and Guessoum, 2017).

1.2 Problem Statement

Text summarization is a challenging task in natural language processing. It is either extractive or abstractive. The aim of the extractive approach is to select the most important content (sentences, phrases, words, etc.) verbatim from the source document to generate the summary. This approach has dominated the field due to its simplicity. Many techniques have been proposed to tackle extractive text summarization. For example, we find approaches that use features like the position of sentences in the document, sentence length, keywords, etc. More advanced techniques have also been proposed. In this sense, we mention that methods that use graph representation or semantic knowledge have been proposed (see (Lloret and Palomar, 2012) for more detailed discussion). However, the extractive approach has some limitations with regards to the coherence of the generated summary. This limitations will open the doors to investigate another interesting approach which is the abstractive approach.

In contrast to extractive text summarization, abstractive text summarization builds a semantic representation of the text and uses more advanced natural language

generation techniques to generate the summary. In this approach, not only the material which is present in the source document is copied, but new material could be inserted to generate the summary. Besides, this approach provides a higher degree of text condensation with a higher amount of information compared to the extractive approach. We should point out that, we find just few works on the abstractive text summarization approach. This is probably due to the difficulty of this approach.

We also mention that most of the work that has been presented in the literature has focused on summarization of documents in the English language and some European languages like French and Spanish. A lot of effort has been done for text summarization in some Asiatic languages like Chinese and Japanese. In fact, there are just a few works for the Arabic language. The aim of our work is to contribute to Arabic text summarization, and abstractive text summarization in general.

Various text summarization operations have been proposed in the literature. Among these operations one finds sentence selection/deletion, sentence compression, sentence fusion, etc. We have also observed that there are just a few abstractive text summarization operations. In this thesis, we also aim to contribute to the state of the art by proposing an abstractive text summarization operation (Concept generalization and fusion for abstractive text summarization). We have also observed that we do not have in the literature a system that combine all of these operations in a compositional manner to generate summaries. For this reason, we propose a system that combines all of these operations in a single framework to generate summaries. A more detailed discussion about our contribution is given in the following section.

1.3 Contributions

This thesis contributes to the text summarization task in the following ways:

- We propose a framework that has the ability to integrate different text summarization operations. Then, we propose different operations which are used in the framework. In this approach, we have used different mathematical and computer science theories to develop our framework. First, we use *number theory* and specifically *Bell numbers theory* to generate what we will define later as *partitions of the document*. We also use *graph theory* for the *multigraph* that we define later. The framework sets a strategy to select the most suitable operation for each portion of text (subset of sentences) to be summarized. For this purpose, we have designed a machine learning system that has the ability to assign summarization operations to the subsets of sentences of the source document. This machine

learning system has been trained on the TALAA-ASC enhanced, annotated corpus. This corpus contains a set of documents where for each document, the operations are associated with subsets of sentences of this document.

- We also propose a system that tries to select the best sentences (sentence selection operation) using the AdaBoost machine learning algorithm (Belkebir and Guessoum, 2015a). This system learns to predict whether a sentence should be included in the summary or not. The machine learning system has been trained on an annotated corpus of <source,summary> documents.

- We present and discuss a sentence abstraction approach by means of concept generalization and fusion which goes beyond sentence deletion (Belkebir and Guessoum, 2016) . This approach produces sentences that contain words produced using generalization and fusion operations on a concept taxonomy, and which are not necessarily present in the source sentences.

- We also provide a sentence compression corpus (TAALA-ASC) that we make available to the NLP community (Belkebir and Guessoum, 2015b). To the best of our knowledge, this is the first sentence compression corpus devoted to the Arabic language. This corpus is a parallel corpus that will help both in evaluating and building new sentence compression models.

1.4 Thesis Overview

The remainder of this thesis is structured as follows:

- Chapter 2 introduces and discusses relevant literature review on text summarization. It discusses both the extractive and the abstractive approaches. A special attention is devoted to the work done on Arabic text summarization.

- Chapter 3 presents an approach which is part of the global framework. This approach aims to select important sentences using the AdaBoost machine learning algorithm.

- Chapter 4 introduces an approach to the generation of abstractive sentences using generalization and fusion operations. We present the steps that we followed to generate sentences and discuss the novelty of this approach.

- Chapter 5 presents TALAA-ASC, a sentence compression corpus for the Arabic language. This will be used in the generation of learning models for sentence

compression. It can also be used in the evaluation of the Arabic sentence compression systems.

- Chapter 6 presents a global framework for Arabic text summarization. It shows the process of generation of summaries using several operations. This chapter presents different definitions and algorithms and sets a strategy to combine the different operations used in text summarization.
- Chapter 7 concludes our work and summarizes the major contributions of this thesis. It also discusses future research directions and possible improvements.

Figure 1.1 presents the global architecture of the framework. The summarization operations (concept generalization and fusion for abstractive sentence generation and the supervised approach to Arabic text summarization using AdaBoost) presented in the schema are the components of the summarization framework while the extended TALAA-ASC corpus is used for both training and evaluating the TALAA-ATSF framework.

1.5 Published Work

The material presented in this thesis has been published in various papers as follows:

Chapter 3 is related to the work of Belkebir and Guessoum (2015a). Chapter 4 which presents an approach that allows the generation of abstractive sentences using generalization and fusion operations has been published in (Belkebir and Guessoum, 2016).

Chapter 5 presents TAALA-ASC, which is a sentence compression corpus. It is related to the work presented in (Belkebir and Guessoum, 2015b).

Chapter 6 which represents the global summarization framework has been accepted for publication and will appear as (Belkebir and Guessoum, 2017). Section 6.5 of chapter 6 which presents the future extension of TALAA-ATSF by using categorization as an initial step in text summarization has been published in (Belkebir and Guessoum, 2013).

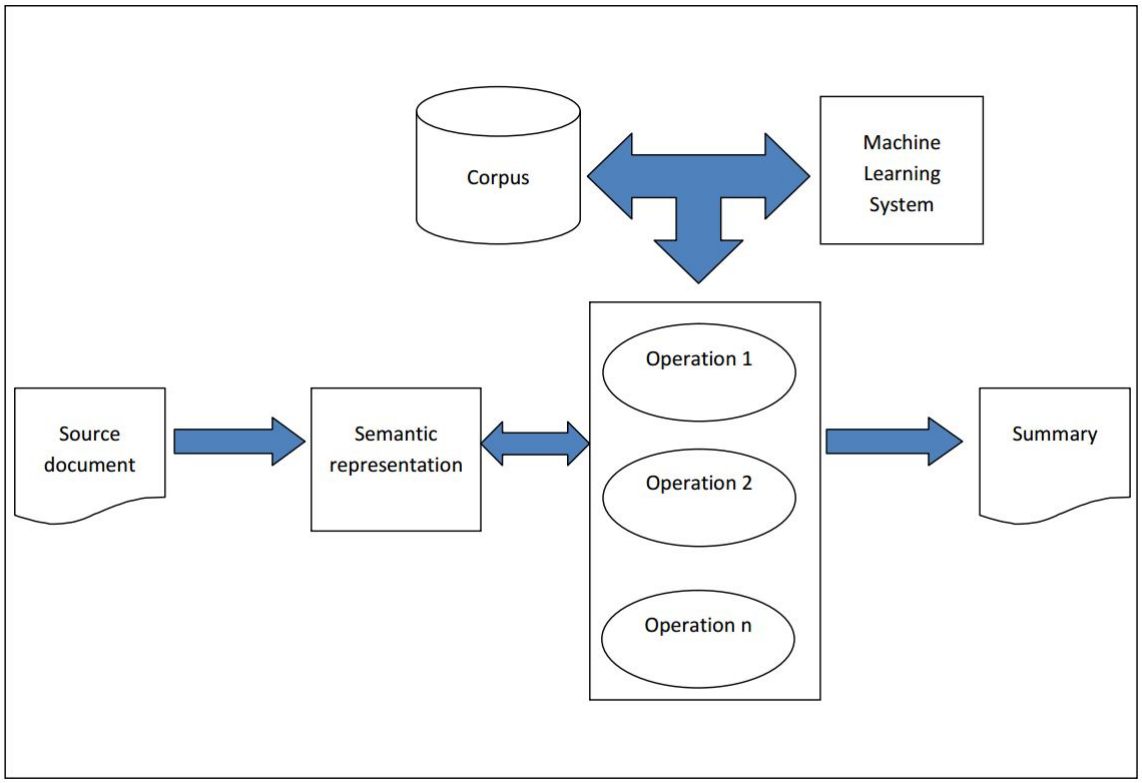

Fig. 1.1 General architecture of the proposed global text summarization framework

Chapter 2

State of the Art in Arabic Text Summarization

2.1 Introduction

Text summarization is not a new discipline; it dates back to the earliest work of Luhn (1958) and Edmundson (1969). Various approaches have been proposed in the literature to tackle the summarization problem. A lot of effort has been done on the English language and other European languages but we found little work on the Arabic language.

Text summarization has been successfully linked to other applications like text categorization and information retrieval. We discuss in this chapter some of the work that has addressed this area of research.

Let us point here that although many operations have been developed to text summarization, there is a lack of full summarization frameworks that have the ability to integrate the various operations within the same framework. One of the main contributions of this thesis is to propose a global framework that has the ability to integrate any kind of summarization operation within a full summarization framework.

In the sequel, we present the main contributions to the field of text summarization as well as an analytical study of this field. We also discuss work done on Arabic text summarization.

2.2 Summarization Techniques

In this section, we give an overview of the main techniques that have been used in the literature. We are not interested here in giving an exhaustive review of the work done on text summarization. In fact, a number of interesting papers have addressed this problem such as, (Mani and Maybury (1999), Hovy and Lin (1999), Spärck Jones (2007), Lloret and Palomar (2012), Nenkova and McKeown (2011) and Saggion and Poibeau (2013)). In this work, however, we try to cover a comprehensive list of works that have addressed the task of summarization by focusing mainly on the operations that have been used in the text summarization systems. We also discuss the main approaches that have been proposed both for the extractive and the abstractive approaches.

2.2.1 Extractive Text Summarization

In extractive text summarization (ETS), the most important pieces of information (words, chunks of sentences, entire sentences, etc.) are selected from a source document without adding any external material. We are interested here in the classification of extractive text summarization works based on the techniques used. These are summarized in the taxonomy presented in Table 2.1. We discuss the taxonomy and detail the related studies below.

Extractive text summarization techniques could be broadly classified into four categories. The scoring-based methods where the aim is to associate a score to each sentence according to a predefined function. The second category aims to build a learning model that could predict whether a new sentence is worth including in the generated summary or not; the model is trained using a parallel corpus which contains pairs of source and summary documents. The graph based techniques generally build a graph where the nodes represent the different sentences and edges represent the strengths of relations between the nodes. The fourth technique uses semantics and discourse to capture the semantic meaning of the source text so as to generate coherent summaries as output.

2.2.1.1 Scoring-based ETS

An ETS approach based on sentence scoring was first proposed in (Luhn, 1958). Luhn (1958) used term frequency counts to generate summaries from scientific documents by trying to find the most salient sentences in the document. The assumption here was that frequent words give an indication about the main topic of the document. The author did not select all the words for calculation. He rather discarded stop words

Category	Description
Scoring-Based ETS	This method associates a score with each sentence. Then, it ranks them and takes highly scored sentences according to a certain retention rate.
Machine learning-Based ETS	It relies on a parallel corpus that contains pairs of <source, summary> documents and tries to construct a learning model that decides whether to include a given sentence in a summary or not.
Graph-Based ETS	It constructs a graph over a document where nodes represent the different sentences and the edges are weighted and represent the strengths of relations between sentences. Different strategies have been proposed to select the best sentences from this graph.
Methods Using Semantics and Discourse ETS	The focus here is on aspects related to semantic level representation of the source document. This semantic representation will allow the selection of the most important content of a document.

Table 2.1 Extractive Text Summarization Techniques

and articles like "a" and "the". A number of other techniques have followed the same assumption. In this sense, Lloret and Palomar (2009) combined the frequency of words with the length of noun phrases to rank sentences according to their importance. The authors reported that this approach outperformed the state-of-the art approaches in single-document summarization on the newswire domain.

Victoria (2004) analyzed many statistical approaches, such as term frequency and inverse document frequency (tf*idf). The tf*idf gives high values to frequent terms in a document in the case they are not highly frequent in the entire collection. This approach has also been used in (Gotti et al., 2007).

The position of sentences in the text was also considered as a feature in text summarization by Barzilay and McKeown (2005) and Christensen et al. (2004). In fact, in documents like news stories, the leading paragraph most of the time contains the most important information in the text and the rest of the text represents details and/or background information about the leading paragraph. For this reason, selecting the leading paragraph in the summary for some kind of documents like news stories is important. However, for another category of documents like scientific texts the introduction usually provides background information, while the main important

content is stated in the conclusion, so the position feature that a summarization system should use needs to be adapted to the category of the text to be summarized. This underlines the importance of the idea of categorization before summarization. Indeed, if the system knows ahead of time the category of document, it can in some cases easily determine the portions of text which are important, thus selecting the best strategy to generate the summary. Comparing the words appearing in sentences with those in the title was also used as a technique to compute the relevance of sentences (Barzilay and McKeown, 2005). Duboue et al. (2003) have considered the occurrence of specific cuewords or expressions in sentences, such as "to summarize", "in conclusion". These features have been applied following a scoring function or by combining these features using machine learning systems.

Other techniques which are based on the use of information gain as a feature have been proposed. Mori (2002) used information gain to determine the weight of document terms. The first step is to build clusters of documents, and then the weight of each word in a cluster is computed. At the final stage, highest scored sentences computed using information gain are selected to generate the summary.

The aforementioned techniques use a different ways of scoring the different terms of a sentence; they then rank the sentences and select those that have the highest scores. Orasan (2004) and Orăsan (2009) have shown that these techniques, even though they are simple and do not require a deep level of semantic knowledge analysis, are suitable for generating good quality summaries. However, Filatova and Hatzivassiloglou (2004) stated that this kind of approach is not sufficient enough to generate high-quality summaries. In this case, additional semantic knowledge such as discourse information, events, topic-based semantics could be more appropriate to produce better quality summaries.

2.2.1.2 Machine Learning-based approaches

The main idea of this approach is to train a machine learning system with pairs of <source, summary> document. In this approach, the focus is on building prediction models that try to predict whether a sentence is worth including in a summary or not. In this case, basic features (keywords, sentence position, etc.) as well as some advanced features like graph features have been used. We distinguish two possible approaches to summarizing documents with this approach. In the first approach, binary classifiers are produced to predict whether a sentence should be selected for the summary or not. Here, we can cite the first machine learning approaches that have been proposed in TS which include binary classifiers (Kupiec et al., 1995), Hidden Markov Models

(Conroy and O'leary, 2001); (Schlesinger et al., 2002), and Bayesian methods (Aone et al., 1998).

The second approach uses a machine learning algorithm to assign weights to the text sentences. For instance, NetSum (Svore et al., 2007) generates extracts from newswire documents using neural networks to give a score to the sentences and select the most salient ones. In the work of Schilder and Kondadadi (2008), a system named FastSum has been developed. This system uses Support Vector Regression (SVR) as a machine learning system and Least Angle Regression for feature selection. In (Li et al., 2007), Support Vector Regression (SVR) has also been used as a machine learning system for summarization. In order to train the summarizer, the authors used features like sentence position, word-, phrase-, semantic-based as well as name entities. Then sentences are scored and ranked using this machine learning system to output the summary.

In (Fuentes et al., 2007), Support Vector Machines (SVM) have been used to select the most important content from a query-focused summary. Supervised and semi-supervised approaches have been used in (Wong et al., 2008) to generate extractive summaries. In this approach, the authors used different types of features like number of words in a sentence, sentence position, centroid and high frequency terms. They have used Support Vector Machines (SVM) as a supervised approach to generate the summaries. In the semi-supervised approach, an SVM and a Naïve Bayesian classifier are co-trained to handle the unlabeled data.

Text summarization has successfuly been tackled using machine learning approaches. However, one of the problems that has been faced when doing summarization following this approach is the difficulty to find large training corpora. In fact, the generation of this kind of corpus is very consuming in terms of time and resources.

2.2.1.3 Graph-based ETS

Graph-based extractive summarizers have been proposed in the literature. In this approach, nodes represent sentences and edges represent the relation between these sentences. These relations include semantic relations, semantic distance between sentences, etc. The assumption here is that the topology of the graph could provide interesting insights into the most important content of the text. Among the most relevant approaches in this category, we mention LexRank (Erkan and Radev, 2004) which is a multi-document summarization system. In this system, sentences are represented by graph nodes. A predefined threshold is defined upon which a decision is made whether two sentences should be connected or not. After constructing the graph,

a random walk algorithm is performed to find the most central sentences in the text and hence to generate the summary.

Some graph-based algorithms have been analyzed for text summarization in (Mihalcea, 2004). Wan and Xiao (2007) proposed an approach that uses affinity graphs, for both generic and query focused multi-document text summarization. The idea of this work is to select sentences by penalizing redundant information. Character and word n-grams have also been proposed in (Giannakopoulos et al., 2008) to select important information from a set of documents. WordNet concepts have been used in (Miller and Fellbaum, 1998), and the graph has been built using the is-a relation between concepts. This approach has been successful in domains such as biomedical documents, newswire and image captions.

Qazvinian and Radev (2008) combined word frequency along with sentence clustering methods using graph approaches to generate the summary. It was also shown in (Erkan and Radev, 2004; Mihalcea and Tarau, 2004), that graph-based approaches worked well for both single and multi-document summarization. Chali and Joty (2008) stated that the use of syntactic and semantic information in constructing the graph gives better results than with the TF*IDF cosine similarity. To generate coherent texts, Salton et al. (1997) proposed to select paragraphs instead of sentences. They have shown that selecting well-connected paragraphs (by cosine similarity) yields good summaries. Leskovec et al. (2005) have used machine learning methods to detect which portions of the generated graph should be included in the summary. They have concluded that PageRank weights were the best class of features to generate the summary.

2.2.1.4 Methods using Semantics and Discourse

Text summarization has also been addressed from a linguistic point of view. To this end, discourse based methods have been proposed. In the sequel, we present the main approaches in this field.

Rhetorical Structure Theory (RST) Rhetorical structure theory (Mann and Thompson, 1988) has gained a lot of interest in text summarization. This approach analyzes the input text and provides a tree representation of this input. In the tree representation provided by RST, the elementary discourse units (EDUs) are the smallest units of the text. The different units are combined recursively together forming a hierarchical structure of the entire text. The different units participating in the tree are represented by a nucleus or satellite status. The nucleus is considered as having the most important information of the text. Marcu (1999) has extended the rhetorical

relations and used this discourse representation to select the most important content units in a document. To enrich the summarization system, Khan et al. (2005) have combined RST with a generic summarizer. The authors claimed that combining RST with a generic summarizer does not improve the results obtained by the generic summarizer alone. The authors claimed that this was caused by the RST parser which was not able to detect all the RST relations. It is also worth mentioning that RST has given good results for single document summarization of news and Scientific American article (Marcu, 1997, 1998, 2000).

To improve the coherence and cohesion of summaries, Da Cunha et al. (2007) have combined statistical and linguistic techniques. They have shown that combining these two types of approaches were more beneficial than using one single technique.

Lexical or coreference chains Another approach is to use lexical or coreference chains. This approach has been first used in (Baldwin and Morton, 1998). The idea of this approach is to select the longest coreference chain. The assumption here is that the longest coreference chain contains the main topic of the document while shorter chains indicate subtopics. In order to generate coherent summaries, this approach selects the longest coreference chain.

Referential cohesion is one of the problems that is difficult to tackle in text summarization. To solve this issue, Gonçalves et al. (2008) have used coreference chains. A post-processing step is defined in their system to rewrite referential expressions in a more coherent manner. This was applied after the generation of the summary. The authors claimed that their approach improved the coherence of the generated summary.

The lexical-chains-based approach aims to determine sequences of words which are semantically related to each other (for example, synonyms). It is assumed that the main topic could be detected just by selecting the longest chain. In this sense, Barzilay and Elhadad (1999), Medelyan (2007) and Ercan and Cicekli (2008) have used this technique to generate summaries.

The ability of this approach to identify all the connected entities inside the document prevents the summarization systems from the common dangling anaphora phenomenon and hence allow the generation of coherent summaries (Elsner and Charniak (2008)). Dangling anaphora is the problem consisting in having pronouns in the text but with incorrect antecedents in the summary. For instance, if the summary contains the word “he” but its antecedent (eg. the teacher) is ommited in the summary , the reader will not be able to understand the summary. To tackle this problem, some anaphora resolution approaches have been proposed (Mitkov et al., 2007; Orasan, 2004).

First, documents are processed to resolve anaphoric pronouns, then the summarization system is run. The authors claimed that the performance of the systems did not improve, which was explained by the poor precision of the anaphora resolution systems. Orasan (2007) have tested the approach with an ideal anaphora resolution system where the anaphoric relations were performed manually. The authors claimed that, in this situation, the summaries have improved considerably.

Steinberger et al. (2007) stated that the quality of the anaphoric resolution system is not the only criterion in generating better quality summaries. In fact, the manner in which the anaphoric systems are used is a factor that improve the quality.

The lexical-chains-based approach has also been used using semantic resources. The intuition here is that the lexical chains approach considers that topics are expressed using different related words (Nenkova and McKeown, 2011). The approach mainly relies on WordNet (Miller et al., 1990), which is a lexical resource. WordNet allows to identify synonymy, antonymy, general-specific and part-whole relations.

Barzilay et al. (1999) proposed an approach where the input document is segmented; they also identified lexical chains within segments and across segments. The different lexical chains are scored, and finally one sentence of each highly scored lexical chain is selected.

Nenkova and McKeown (2011) stated that the success of the lexical chains approaches mainly rely on the coverage provided by WordNet. For this reason, approaches like Latent Semantic Analysis could propose a solution to the issue since it gives an approximate semantic interpretation of the input without heavily relying on a semantic resource like WordNet.

Latent Semantic Analysis Latent semantic analysis (LSA) (Deerwester et al., 1990) has been proposed to be used in text summarization. It allows a semantic representation of the text document based on the co-occurrence of words. In (Gong and Liu, 2001), LSA has been proposed to tackle both single and multi-document summarization of news. This approach represents a document as a matrix where each row represents the word of the document while a column represents a sentence. Each cell a_{ij} of the matrix represents the weight of the word i in the document j. In case where the document does not contain the word, the weight is zero, otherwise it has the TF*IDF score for this word. Then, singular-value decomposition (SVD) techniques are applied to the matrix A. This matrix is represented as a product of three matrices: $A = U\Sigma V^T$. Where U is an m $\times$ n column-orthonormal matrix whose columns are called left singular vectors, Σ is an n $\times$ n diagonal matrix whose diagonal elements are

non-negative singular values sorted in descending order, and V is an n $\times$ n orthonormal matrix whose columns are called right singular vectors. It was suggested in (Gong and Liu, 2001) that each row of V^T can be considered as mutually independent topics of the source document. In this case, they just select each row of V^T to generate the summary. We mention also that Steinberger et al. (2007) have also analyzed several variations of the methods of Gong and Liu (2001) and achieved an improvement over the basic method of Gong and Liu (2001).

Hachey et al. (2006) have proposed another method to use singular value decomposition. Initially, they followed the basic LSA approach and built the matrix A. They built this matrix taking as a basis a large collection of documents. The performance of this approach has been tested with TF*IDF and with and without SVD. They claimed that SVD improved the performance over a co-occurrence approach but the performance was not significant over the TF*IDF summarizer.

Coreference Information Coreference resolution has also been used in text summarization. The coreference resolution process aims to find all the references of an entity in the document. Early results (Baldwin and Morton, 1998; Boguraev and Kennedy, 1999) using this approach have concluded that it was not beneficial for text summarization. Then, this technique has been used in (Steinberger et al., 2007) for generic single document summarization of news that rely on word frequency features and proved that this approach has led to important improvement of the quality of the generated summary. An LSA-driven summarizer has been improved as well using anaphora resolution in (Gong and Liu, 2001). In a first experiment, the references to the same entity were replaced by its first mention in the document. The resulting input document is then provided to an LSA summarizer. In the second experiment, they used the presence of an entity in a sentence as an additional feature helping to determine the importance of the sentence. In this case, the references to the entity remained unchanged. The authors stated that the first approach has led to a decrease in the performance of the LSA-based summarizer. However, the second approach has given a significant improvement.

2.2.2 Abstrative Text Summarization

Abstractive summarization allows an internal representation of the source document so as to produce a faithful summary of the source. In this case, external text can be inserted into the generated summary. Probably due to the complexity of the abstractive

approach, the vast majority of work in text summarization has adopted an extractive approach.

An interesting efforts have been done on abstractive summarization. Khan et al. (2015) have used sematic role labeling to represent documents. They have proposed an abstractive summarization framework for multi-document summarization. The content of the summary document is selected based on a ranking of the predicate argument structures. Pighin et al. (2014) have used abstractive text summarization for news events. They represent the events as a collection of pattern clusters. In this representation, each cluster represents an event (e.g., marriage) and every pattern represents a manner of expressing this event (e.g., X married Y, X and Y tied the knot). In order to extract the event, they have compared three methods, a heuristic-based, a memory-based and a compression-based method. They have concluded that the memory-based method generates significantly more grammatical and informative sentences. Phrase selection and merging have also been applied for abstractive text summarization. In (Bing et al., 2015), new sentences can be constructed by exploiting more fine-grained syntactic units than sentences, namely noun/verb phrases. From the original document, this approach constructs concepts and facts represented by phrases. The authors use integer linear optimization to simultaneously select and merge the different phrases. In (Boudin et al., 2015), the notion of Basic Semantic Unit (BSU) has been coined to describe the semantics of an event or action. The authors have also defined a semantic link network taking into account the BSUs to capture semantic information of the text documents. The structure of the summary is generated based on the sentences generated by the semantic link network. In (Rush et al., 2015), a data-driven sentence abstraction approach has been proposed. This method, which is described as local attention-based, generates each word of the summary conditioned by the input sentence. The authors mention that the system could be trained end-to-end and can scale to large amounts of training data. In (Liu et al., 2015), a semantic representation, called abstractive meaning representation (AMR), was used to construct an abstractive framework. Original texts are represented and parsed as a set of AMR graphs. Then, these graphs are transformed into a graph of text summaries . A graph-to-graph transformation was used to generate the summary graph. The final step is then to generate the summary text from the graph. Mehdad et al. (2014) have proposed a query-based abstractive summarization system for conversations. Queries are defined so as to reflect the user's information needs. The authors have used a word graph-model to select sentences from the conversation. They generate the summary by selecting the best path in the constructed graph as a query-based

abstract sentence of each cluster. The summary consists of abstract sentences that represent the conversation content and the phrasal query information. In (Lloret and Palomar, 2011), a system is proposed which uses a word graph-based method to merge or compress information. The system combines extractive and abstractive information to generate the abstracts. To select relevant sentences from the source text, the COMPENDIUM system is used. The authors have reported that their experiments prove that the combination of extractive and abstractive information is a more suitable strategy to generate abstracts. In (Genest and Lapalme, 2011), an approach that uses an abstract representation of source documents is proposed. This representation makes use of the notion of Information Items (INIT), defined as the smallest element of coherent information in a text or a sentence. The approach selects the summary content from the representation. More recently, an interesting approach that uses a semantic representation to generate summaries was proposed (Lloret et al., 2015). This approach is a concept-level approach which aims to generate ultra-concise opinion summaries from the combination of several operations. These operations include sentence regeneration, integration of syntactic sentence simplification, and internal concept representation. Two versions of the system were proposed. The first version uses sentence generation while the second is without sentence generation. The authors have reported that both versions are reliable in generating summaries. Moreover, the system version that uses sentence regeneration was more robust in the presence of noisy data, while the one that does not use sentence regeneration yielded better results surpassing many state-of-the-art systems by 9% (global average between the readability of the summary, the content of the summary, and its overall responsiveness).

Recently some abstractive text summarization techniques have been proposed. Nema et al. (2017) proposed an approach for query based abstractive text summarization. This approach uses encode-attend-decode paradigm with two key additions (i) a query attention model which learns to focus on different portions of the query at different time steps and (ii) a new diversity based attention model which aims to alleviate the problem of repeating phrases in the summary. The authors reported that their approach outperforms vanilla encode-attend-decode models with a gain of 28% (absolute) in ROUGE-L scores. See et al. (2017) proposed an architecture that augments the standard sequence-to-sequence attentional model in two orthogonal ways. First, they proposed a hybrid pointer-generator network that can copy words from the source text via pointing, which allows the reproduction of accurate information, and by retaining the ability to produce novel words through the generator. Second, they used coverage to keep track of the summarized text to avoid repetition. They have tested this

approach using CNN / Daily Mail summarization task. The authors stated that their approach outperforms the current abstractive state-of-the-art by at least 2 ROUGE points. Zhou et al. (2017) have designed a selective encoding approach to extend the sequence-to-sequence framework for abstractive sentence summarization. They have evaluated the model on the DUC 2004, English Gigaword, and MSR abstractive sentence summarization datasets and reported that their approach outperforms state-of-the-art models. Tan et al. (2017) suggested a graph-based attentional neural model for abstractive text summarization. The authors reported that their approach has given comparable results to state-of-the-art extractive systems and achieved considerable improvement over previous neural abstractive models.

2.3 Analytical Study

In this section, we try to understand the task of text summarization and the different operations that have been proposed in the literature. At the end we give some conclusions.

2.3.1 Understanding Text Summarization

A number of studies have been done trying to understand the task of summarization. Jing (2002) has used Hidden Markov Models to decompose summaries produced by human experts. She has tried to infer whether a summary is constructed by reusing phrases from the original text, identifying these phrases and finding the positions in the original text these phrases come from. In another study, Hasler (2007) analyzed the operations performed by human abstractors and identified a set of operations that are involved in the summarization process.

2.3.1.1 The Work of Jing (2002)

In order to study how human abstractors generate summaries, Jing (2002) has analyzed the process of generating a set of articles which contains fifteen articles on telecommunication, ten articles about the legal domain and ten articles that deal with medical issues. The style and the structure of the documents vary even within the same specific domain.

From what they have observed about the studied corpus, they have found that human abstractors most of the time reuse text from the original document to generate a summary. They have also stated that this observation is consistent with the work of

Endres-Niggemeyer and Neugebauer (1998), who explained that professional abstractors most of the time rely on cutting and pasting from the original text in the summarization process.

After the analysis of the operations used in the summarization process, Jing (2002) has defined six operations that can be used to transform a source document into a summary document. These operations are sentence reduction, sentence combination, syntactic transformation, lexical paraphrasing, generalization or specification, and reordering. In the sequel, we give a description of each of these operations Jing (2002).

Sentence reduction Sentence reduction consists in removing non-essential phrases from a sentence. This operation has later been called sentence compression in the literature. But the term sentence compression in the literature is not only restricted to removing phrases but also individual words from any given document.

Sentence combination Sentence combination is the process of merging material from a set of sentences into a single sentence. Jing (2002) has stated that this operation is typically used together with sentence reduction.

Syntactic transformation Syntactic transformation is involved both in sentence reduction and sentence combination. It changes the syntactic structure of a sentence.

Lexical paraphrasing Lexical paraphrasing is the process of substitution of sentences with their paraphrases.

Generalization or specification Generalization (resp, specification) aims to replace phrases or clauses of a sentence with a more general (resp, specific) description.

Reordering In reordering, the order of extracted sentences is changed with respect to the original. For instance, the ending sentence of an article may be placed at the beginning of an abstract.

Jing (2002) stated that not all operations are listed here. In fact, some operations are infrequently used. Jing (2002) also stated that multiple operations are often involved to produce a single sentence summary. Another interesting conclusion is that in the human-written abstracts, some sentences are not based on cut and paste but are written from scratch. To distinguish between sentences that were cut and pasted and sentences written from scratch, the authors decided that sentences having more than

half of their words in the summary are borrowed from the source document (cut and pasted). Otherwise, they judge that these sentences have been written from scratch.

2.3.1.2 The Work of Hasler (2007)

Hasler (2007) has stated that to generate English summaries, humans tend to copy and paste snippets from the original document after some slight modifications. Hasler (2007) also classified the operations into five classes. After the analysis of a summarization corpus, these operations were classified as either atomic or complex operations. The atomic operations are insertion and deletion of words. The complex operations include replacement and reordering of words and merging of sentences. It was explained that atomic operations cannot be further divided into other operation, whereas complex operations can.

All the definitions provided below have been taken from Hasler (2007). Table 2.2 presents the operations that have been observed on the corpus by Hasler (2007).

Deletion Deletion is defined as the process of removing a unit from a certain place in the extract so that it does not appear in the same place in the abstract.

Insertion Insertion is defined as the process of adding a unit to the abstract which is not present in the extract.

Replacement Replacement is defined in terms of a deletion and an insertion: it is the deletion of one unit and the insertion of a different unit in the same place in the text.

Reordering Reordering occurs when a sentence or part of a sentence is moved from its original position in the extract into a new position in the abstract and is defined in this context as the deletion of a unit from one place in the extract and its insertion in a different place in the abstract.

Merging Merging is defined in this context as taking information from different units in the extract and presenting it as one unit in the abstract.

2.3.1.3 Other Text Summarization Operations

Here we present the operations that have been widely used in text summarization. We do not claim that this list of operations is exhaustive, but it covers definitions of

Type of operation	Operation	Sub-operation
Atomic	Deletion	Complete sentences
		Subordinate clauses
		Prepositional phrases
		Adverb phrases
		Reporting clauses and speech
		Noun phrases
		Determiners
		The verb be
		Specially formatted text
	Insertion	Connectives
		Formulaic units
		Modifiers
		Punctuation
Complex	Replacement	Pronominalization
		Lexical substitution
		Restructuring of noun phrases
		Nominalization
		Referred sentences
		Verb phrases
		Abbreviations
	Reordering	Emphasizing information
		Improving coherence and readability
	Merging	Restructuring of clauses and sentences
		Punctuation/connectives

Table 2.2 Text summarization operations

the main operations that have been used in the literature. These operations include natural language generation, text simplification, sentence compression, text revision and sentence fusion. We also consider extractive summarization (sentence selection) as an operation that aims to determine whether a sentence is worth including in the summary or not.

Natural language generation Various approaches have been used to tackle text summarization. Some of these are based on natural language generation. For instance, Radev and McKeown (1998) have developed a system, SUMMONS, which produces multi-document summaries of the same event by using the output of systems developed for the DARPA Message Understanding Conferences. In a similar study, (Kumar et al., 2009) have designed a learning-based system that generates a draft report as a mix of event data and the input text document. This learning system was trained on a corpus of reports prepared by experts in the target (conference replanning) domain.

Text simplification Various studies have tried to do text simplification. The focus here is on the use of rewriting operations that get applied to source sentences so as to decrease the syntactic or lexical level of complexity and at the same time to preserve their meaning (Siddharthan, 2002). In this context, Coster and Kauchak (2011) have used different simplification operations including (rewording, reordering, insertion and deletion) by introducing a data set that pairs Simple English Wikipedia with English Wikipedia. Similarly, Woodsend and Lapata (2011) have designed an approach to generate content selection rules from same-topic Wikipedia articles written in the main encyclopedia and its Simple English variant. Wubben et al. (2012) have proposed an approach for simplifying sentences using Phrase- Based Monolingual Machine Translation. Kauchak (2013) has used language modeling for text simplification.

Sentence compression A good effort has also been devoted to sentence compression. The goal here is to transform a given source sentence by means of reduction and/or paraphrasing operations while respecting the important information found in the source sentence (Clarke and Lapata, 2008). Recently, Huang et al. (2012) have used first-order logic Markov Logic Networks, a statistical relational learning framework that combines Markov Networks with First-Order Logic. This approach uses a set of weighted formulas to compress sentences. A tree-to-tree transduction method based on synchronous tree substitution grammars has also been proposed to tackle sentence compression (Feblowitz and Kauchak, 2013; Yamangil and Shieber, 2010). A semantic

constraint-based approach was proposed in (Yoshikawa et al., 2012). This approach relies on semantic roles constraints which aim to capture the relation between a given predicate and its arguments. An abstractive approach was proposed in (Cohn and Lapata, 2013). They used operations such as substitution, reordering, and insertion in a discriminative tree-to-tree transduction model to compress sentences. In another study, Zajic et al. (2007) have examined the use of sentence compression in single document text summarization using a parse-and-trim approach and a statistical noisy channel approach. They also introduced the Multi-Candidate Reduction (MCR) framework for multi-document summarization, in which many compressed candidates are generated for each source sentence. These candidates are then selected for inclusion in the final summary based on a combination of features.

Text revision Text revision has also been tackled in the context of text summarization. Nenkova (2008) studied the benefits and shortcomings of entity-driven noun phrase rewriting for multi-document summarization of news articles. Tanaka et al. (2009) studied text revision for broadcast news summarization using a Syntax-driven approach. They performed text revision by using operations like substitution, reordering, and insertion. Saggion (2009) studied the benefit of adding extra information to the abstract. The author proposed an approach whereby a predefined vocabulary (e.g., "to report", "to indicate", "to address", etc.) is used to combine different fragments of information. This system is machine learning-based; it tries to select the most suitable expression among the aforementioned expressions to combine two fragments of information.

Sentence fusion Another category of approaches have dealt with sentence fusion which was proposed in (Barzilay and McKeown, 2005). Sentence fusion is a text-to-text generation technique that aims to synthesize common information between documents. In order to evaluate sentence fusion techniques, McKeown et al. (2010) presented a semi-automatic approach to collecting fusions of similar sentences using Amazon's Mechanical Turk.

Sentence selection Probably due to the difficulty of abstractive text summarization, most studies have followed the extractive paradigm, selecting the important pieces of information from the source document verbatim (chunks of sentences, entire sentences, paragraphs), i.e., without adding any external text to the generated summary. Recently, Ferreira et al. (2013) have assessed the performance of 15 techniques for sentence scoring

(in extracted texts) which are the most common in the field. The extractive approaches have a lot of shortcomings in terms of the quality of the generated summary. One of these is the lack of coherence, especially due to the existence of "dangling anaphors" (Lloret and Palomar, 2012). Abstractive text summarization can theoretically solve this problem. In fact, it allows an internal representation of the source document so as to produce a faithful summary of the source, preserving thereby its readability and coherence. This approach allows the production of a summary by not only deleting (words, phrases, and/or sentences) from the source document, but by also allowing the addition to the summary of new material that was not necessarily present in the original document. An important feature of abstractive text summarization compared to the extractive one is that it generates a summary which is most of the time shorter and more informative (Jing and McKeown, 2000).

2.3.2 Text Summarization Taxonomies

Let us present the available taxonomies of text summarization that have been proposed in the literature.

2.3.2.1 The Taxonomy of Jones et al. (1999)

Jones et al. (1999) have proposed a taxonomy where three classes of context factors that have an impact on the summarization task have been considered. These are the input, purpose and output factors. The input factor deals with issues like genre, language and register. The output factor considers the style and the coverage of the summary. The purpose factor deals with issues related to the audience and the coverage of the summary.

Category	Description
Input	Genre, Language, Register
Output	Style and coverage
Purpose	Audience and use

Table 2.3 The taxonomy of Jones et al. (1999)

2.3.2.2 The Taxonomy of Hovy and Lin (1999)

Hovy and Lin (1999) have proposed a taxonomy similar to that of Jones et al. (1999). In fact, the authors tried to classify summarization work according to three relevant aspects: input, output and purpose. The difference between the two taxonomies is

that this taxonomy takes into account additional factors which are the subjectivity level and the coherence of the summary.

2.3.2.3 The Taxonomy of Mani and Maybury (1999)

Mani and Maybury (1999) have classified the summarization approaches according to the summarization level. In their work, they have distinguished three levels: surface-, entity- or discourse-level. The surface level approaches make use of shallow features (like the position of the sentence) to generate the summaries. Here, a function that takes into account these features is used to extract the most important content. The entity-level approaches use patterns of connectivity and relationships to represent the documents. These approaches use similarity between sentences, thesaural and semantic relationships. The discourse-level approaches represent the structure of the entire document.

Category	Description
Surface-level	Shallow features
Entity- level	Patterns of connectivity and relationships
Discourse-level	Document structure

Table 2.4 The taxonomy of Mani and Maybury (1999)

2.3.2.4 The Taxonomy of Lloret and Palomar (2012)

Lloret and Palomar (2012) have proposed a text summarization taxonomy that takes into account several summarization types according to several factors (see Table 2.5).

The first factor considered in their work is the media that characterize the summarization process. Although the main concern is text summarization, Lloret and Palomar (2012) have classified summarization according to the media used. They mentioned works where the summarization has been applied in other media. These include images (Fan et al., 2008), audio (Zechner and Waibel, 2000), video (He et al., 1999) and hypertexts (Sun et al., 2005).

Another factor that has been considered in the taxonomy of (Lloret and Palomar, 2012) is the input. They considered two types of input namely single or multi-document summarization. The output factor considers the generated summary which could be an extract, the summary in this case having been generated by selecting the most important sentences, chunks of sentences, paragraphs, words, etc., to generate the summary without adding any external material. The output could also be an abstract

where in this case the text generated as output could include operations that add external material to generate the summary. Another output that could be generated is the headline (the title of the document). They have also considered the purpose as a factor in the taxonomy. In this sense, the summary is either generic or query-focused. The difference between the two types of summaries is that the generic summary is generated regardless of any background information or any query expressed by a user, whereas the query-focused summary is driven by the query that the user expressed. Also this factor distinguished between indicative and informative summaries. An indicative summary is very short and indicates only the topic that is discussed in the document while an informative summary is used to give more detailed information by covering the topics of the source document. Another type which is the critical evaluative abstracts have also been considered. This type of summary considers the author feedback or point of view towards a specific topic. This includes opinion, reviews, feedback, recommendations, etc., and depends on cultural interpretations. This type of summary is very difficult to generate automatically due to its subjectivity. Other types of summaries such as update summaries where the background of the user towards specific content is considered (in this case the aim is to show the user only the new information that he probably does not know) have also been proposed. The sentiment-based summaries represent another type of summary where different degrees of subjectivity have been considered. Regarding the language of the summary, we distinguish three types: mono-lingual, multi-lingual, and cross-lingual summaries. If the language of the output (source document) is the same as the language of the output (summary), then we are dealing with mono-lingual summarization. In the case where the summarization system has the ability to take as input documents in several languages and generates a summary in the same language of the input then we are dealing with a multi-lingual summarization system. A cross-lingual summarization system supports many languages. Besides that, it may take an input in a language X and could output a summary in another language Y.

2.4 Arabic Text Summarization

For languages like English, and some other European languages like French, Swedish, and Spanish, various approaches have been developed to deal with the text summarization task (Lloret and Palomar (2012); Nenkova and McKeown (2011); Saggion and Poibeau (2013)). A lot of research effort is also underway for some Asian languages such as Japanese, Chinese and Indian. Unfortunately, research on Arabic text summa-

Category	Description
MEDIA	Text
	Images
	Video
	Speech
	Hypertext
INPUT	Single-document
	Multi-document
OUTPUT	Extract
	Abstract
	Headline
INPUT	Single-document
	Multi-document
PURPOSE	Generic
	Personalized
	Query-focused
	Update
	Sentiment-based
	Indicative
	Critical
LANGUAGE	Mono-lingual
	Multi-lingual
	Cross-lingual

Table 2.5 The taxonomy of Lloret and Palomar (2012)

rization is still very much underdeveloped compared to that on the aforementioned languages. This section reviews the main approaches that have tackled the problem of Arabic text summarization. As we will show later, all of the work that has been done on Arabic text summarization follows an extractive approach.

In (Azmi and Al-thanyyan, 2009) the authors present an automatic extractive Arabic text summarization system. Their system uses Rhetorical Structure Theory (RST) in conjunction with a sentence scoring scheme. Summaries of different lengths were compared to those made by a human expert. They explained that their system overcomes some of the other existing systems including those based on machine learning techniques.

In (El-Haj et al., 2011b) a multi-document summarizer is presented despite their complaint about the lack of Arabic multi-document corpora and gold standard for text summarization. The results produced by this system were compared to five systems in the DUC–2002 multi-document summarization task.

In (Keskes et al., 2012) a comparative study between three approaches for automatic summarization of Arabic documents is presented. The first method is based on a symbolic approach; the second uses a numerical approach while the third one is a hybrid between the two. They show that the numerical approach outperforms the symbolic one while the hybrid approach outperforms the other two.

In (Abdel Fattah and Ren, 2008) probabilistic neural networks (PNN) are used with several features. They studied the effect of each feature on the text summarization task. Then, they used a combination of all the features to train the probabilistic neural network (PNN) so as to generate a text summarizer.

Lakhas (Douzidia and Lapalme, 2004) is a system that generates 10-word summaries from news articles. The first step consists in summarizing the source articles; then it translates the documents into English. This approach was considered very successful for very short (headline) summaries.

El-Haj and Hammo (2008) proposed a query-driven summarization. This system tries to extract the most relevant paragraphs from a source document by using a vector space model and cosine similarity. Then, top ranked sentences are selected to generate the summary. The authors decided to set the size of the summary to half or less than half the source document. The system has been manually evaluated by 1500 evaluators having different educational levels and background. The corpus used is a collection of Wikipedia articles.

Haboush et al. (2012) have proposed a single document summarization system. To rank the sentences, they have relied on the weight of the word root (they have used

a stemming step to extract the root) instead of the weight of the whole word. The authors then used a tf formula to weight the sentences and generate the summary. The authors have created four summaries for each document and evaluated the approach using a manually created corpus that contains ten documents.

El-Haj et al. (2011a) have proposed a clustering- based approach to handle Arabic text summarization. TF-IDF is used to create the different instances which represent the sentences of the document. In the first approach, they have clustered sentences of the source document and the cluster having the largest amount of sentences is selected. The second approach takes the first sentence from each cluster. In this experiment, they have used a parallel English-Arabic of the DUC 2002 corpus developed by El-Haj et al. (2011a).

Schlesinger et al. (2008) have proposed CLASSY (Clustering, Linguistics, And Statistics for Summarization Yield) a generic and query driven approach. It is a multilingual (English and Arabic) extractive system. This system allows the generation of both single and multi-document summaries. CLASSY uses several defined trimming rules to shorten the sentences. Then, it selects and organizes sentences to produce a multi-document summary. The first step in CLASSY is to associate a type with each sentence. For example, type 0 is associated with headlines. Then, the trimming operation is performed to remove irrelevant phrases from sentences. Finally, sentences are scored using an approximate oracle score.

El-Haj and Rayson (2013) have proposed an extractive summarization system which is a language-independent, single and multi-document summarizer and it deals with both English and Arabic. In their approach, a log-likelihood score of each word is computed. Then, the weight of each sentence is the sum of the score of each word. The system selects sentences with the highest weight to generate the summary. The authors have participated in the MultiLing 2013 Workshop summarization task; in which, besides the provision of corpora for English and other languages, an Arabic language corpus was also made available.

Oufaida et al. (2014) have designed a single and multi-document summarization system which is based on the minimal-redundancy maximal-relevance (mRMR) (Peng et al., 2005) approach as well as a discriminant analysis method. First, the authors clustered sentences using the hierarchical clustering algorithm. The mRMR computes the score (or the rank) of each term, which represents its discrimination. This score is calculated in such as a way that its value will be high when its frequency varies across the different classes and the score value of the term will be low otherwise. The authors proposed three strategies to rank sentences. Then highly ranked sentences are selected

to generate the summary. The authors used EASC for single-document summarization and the MultiLing Pilot 2011 corpus for multi-document summarization. The authors stated that one of the configurations outperformed the lead baseline.

Boudabous et al. (2010) have proposed Support Vector Machines (SVM) to summarize documents. The authors have extracted features like sentence position and TF-IDF score, etc. The features were fed into the SVM to generate the learning model. The authors reported an F-measure of 0.991.

Sobh (2009) has proposed two different methods to generate an extractive summary. The first method uses a Naïve Bayesian classifier while the second approach uses genetic programming (GP). The final summary is then generated by either the union or the intersection of sentences of the two summaries generated by the two methods. For the evaluation, the authors have built a corpus of 123 documents. Each sentence in the summary has been annotated by a language specialist as either sentence summary or not. The authors stated that the summaries generated by the Naïve Bayesian classifier and those generated by genetic programming were relatively similar in terms of the F-measure. However, sentences generated by the union were much longer than those generated by the intersection. The summaries generated by the union have higher recall while summaries generated by the intersection have higher precision. They have also stated that their system provided summaries which are similar to those generated by humans.

El-Fishawy et al. (2014) have designed a machine learning-based system that summarizes Twitter posts made in the Egyptian dialect. First, the system applies some preprocessing steps. Very short posts are excluded. They have used features like the number of followers, the length of the post and the number of re-tweets. They have also used approaches to calculate the significance of the features such as term frequency. They have also calculated the similarity between the sentences. They have used a regression tree model (Frank et al., 1998) to give weights to the different posts. Then they have selected the posts with the highest score. To eliminate redundancy, they selected the post based on a similarity function between the current post and the previously chosen posts. The authors have used a corpus that contains 15 topics. Each topic contains between 300 and 1500 Twitter posts. According to the evaluation they reported, the system they proposed outperforms SumBasic (Vanderwende et al., 2007) and MEAD (Radev et al., 2002).

Imam et al. (2013) have proposed OSSAD, an Ontology-based Summarizer for the Arabic language. This summarizer is query-driven and relies on a machine learning approach to generate summaries. The system expands the user's query by using the

Arabic WordNet. It also adds concepts and relations. Then it uses the decision tree algorithm (C4.5) with four features (which are the Arabic WordNet expanded query, the number of words of the original query, the concept of the expanded query and the relationship of the expanded query) to produce the summary. The authors used their own corpus and also used EASC (El-Haj et al., 2011a) in the test stage. They reported promising results compared to the summaries generated by humans.

2.4.1 Discussion

As we mentioned above, the work that has been done on Arabic text summarization is very much underdeveloped compared to other languages. This is due to various factors among these we mention the following:

- The lack of validated gold standard corpora for the Arabic language. In fact, most studies use their own corpus. This makes the comparison between Arabic summarization systems difficult.
- The community working on Arabic text summarization is quite small.
- The difficulty of the Arabic language itself because of its complex spelling, vocabulary, morphology and syntax.
- The problem of diacritics (tashkyl) in the Arabic language makes it even more complex to handle.

However, it is worth pointing out that recently some efforts have started. For instance, El-Haj et al. (2011a) have designed the EASC corpus, which is a parallel corpus. Nevertheless, there is still a lot of effort to be done to produce Arabic summarization systems with a quality similar to the ones produced for other languages.

To improve the quality of the current Arabic summarization systems, the community needs to develop and improve NLP tools for Arabic. In fact, the summarization systems rely mainly on these NLP tools to generate summaries. This include part of speech tagging, named entity recognition, syntactic and semantic parsing, etc.

Let us emphasize once more that to the best of our knowledge the work done on Arabic text summarization has followed the extractive approach and that the abstractive approach has not been used for the Arabic language except our work.

2.5 Text Summarization with Other Applications

2.5.1 Text Categorization

Text categorisation aims to automatically associate a document with category from a predefined set (Sebastiani, 2002). Text summarization has been successfully combined with many NLP applications. For instance, in the work of Ker and Chen (2000) summarization features such as the position of a sentence and the word frequency were used to classify news documents. This approach has reached a precision of 82%. In another study, Shen et al. (2004) have suggested to use summaries instead of entire documents to do text categorization. This approach was used as a noise filter for web pages. This will generate a new document which will then be used to classify the document. The text categorization approach they proposed uses term frequency and Latent Semantic Analysis (LSA). They have used 150 000 web pages with 64 categories and have shown that this approach has improved the classification performance by 8.8%. In a similar study, Shen et al. (2007) have studied the impact of the compression rate of the summarization system in text categorization and concluded that the optimal compression rates are 20 and 30%.

2.5.2 Opinion Mining

The aim of opinion mining is to detect whether a given text is subjective or objective. In the case of a subjective text, sentiment analysis tries to determine the category of the text: neutral, positive or negative. Text summarization has improved the performance for this task where a user gives an opinion about a certain topic, product, service, etc. In this situation a user could express this by a short text which could be classified as positive, neutral or negative (Wilson et al., 2005). In (Saggion and Funk, 2009), the summarization task has improved the quality of the system and achieved (74%) and (32%) when using full texts.

2.5.3 Information Retrieval

Text summarization has been combined with information retrieval in several ways. The most common way is that information retrieval systems output a set of documents then the summarization system takes the output and generate a summary to the user. For instance, Radev and Fan (2000) designed a multi-document summarizer that summarizes web search results. The SWEeT system, developed by Steinberger et al. (2008) uses an information retrieval system that retrieves documents from the

web and uses Latent Semantic Analysis to select the most important sentences from these documents. To remove redundancy, they used cosine similarity to generate the final summary. Dunlavy et al. (2007) have designed the QCS system which uses an information retrieval system to retrieve documents not from the web but from a static document collection. Then, the documents are clustered according to the main topic. In the next step, a single-summary is produced for each cluster. Then, the authors used a Hidden Markov Model as a machine learning model to determine whether sentences are worth to be including in the summary. In another scenario, some work has investigated if the summarization systems where benefial to the information retrieval systems. For instance, Sakai and Sparck-Jones (2001) have proven that the indexing stage has improved the precision of the information retrieval system when using summaries with compression rates between 10 and 30%.

2.6 Conclusion

In this chapter, we have covered the main approaches that have been proposed for text summarization. We presented the main two approaches in the field of text summarization, namely extractive and the abstractive summarization. We have shown that the most dominating approach in the field is the extractive approach. Most studies avoid using the abstractive approach probably due to the difficulty of this one. In fact, this approach requires a deep understanding of the text to generate the summary. However, we mention that we have recently seen an increasing interest in this research area since the performance of the extractive approach seems to be reaching its limits.

We have tried to conduct an analytical study within which we have extracted the main operations that are applied in the text summarization process. This analysis has been done by looking at two interesting works (Hasler, 2007) and (Jing, 2002) which have tried to analyze the operations that human abstractors perform when generating summaries.

We have also investigated the use of text summarization with other applications. We have discussed the use of text summarization with text categorization, opinion mining, and information retrieval.

We have also presented the main works that has been done on Arabic text summarization and stressed on the fact that this task is not sufficiently addressed for the Arabic language. One of the issues that one has to raise is that for the Arabic language, and to the best of our knowledge, no abstractive text summarization system has been developed.

Chapter 3

A Supervised Approach to Arabic Text Summarization Using AdaBoost

3.1 Introduction

The work presented in this chapter focuses on selecting important sentences using AdaBoost. This work could be seen as an operation in the proposed text summarization framework (See Figure 3.1). One way to address the problem of text summarization is by means of machine learning techniques. These techniques have shown their usefulness and effectiveness in various natural language processing applications such as information retrieval, text classification, machine translation and speech recognition. The aim of machine learning techniques is to learn a hypothesis(function) instead of algorithmically programming it. In other words, instead of knowing exactly how to solve a problem, in whichever area, one of the machine learning techniques can be used to learn the way to solve the problem from examples. As such, by feeding input-output pairs , i.e. characteristics of the problem such as an original text in our case (input) and a representation of the solution such as the summary of the original text (output), the technique used learns a hypothesis, i.e. how to automatically produce a summary given a text. In this chapter, we propose a new approach to text summarization based on AdaBoost which we will explain in Section 3.2.

The main idea here is to build a hypothesis that could select the most relevant sentences in the source document so as to keep the most important information as well as preserve the readability of the summary. In our case, the inputs are sentences and

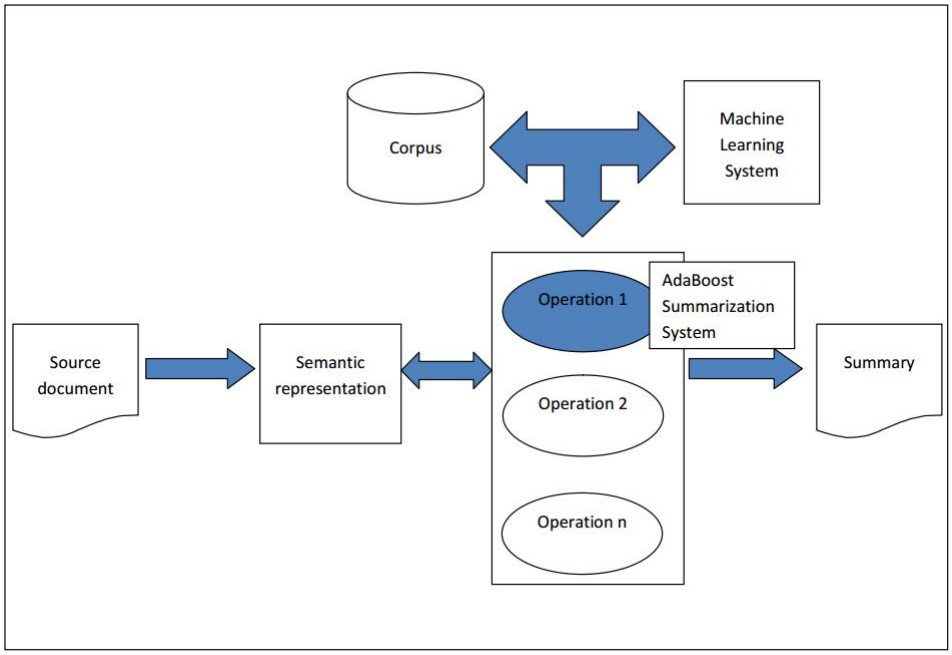

Fig. 3.1 AdaBoost summarization system in the global summarization framework

the outputs are the corresponding labels. Two types of labels are considered. Label "one" (1) is used to indicate that the sentence should be included in the summary and "zero" (0) to indicate that the sentence should be discarded from the summary. These decisions are taken by the AdaBoost learning model.

The remainder of this chapter is as follows. Section 3.2 describes the design of the approach adopted in this work. Section 3.3 presents the results of the implementation. A conclusion is then given in Section 3.4.

3.2 The AdaBoost-based summarizer

3.2.1 The AdaBoost algorithm

The boosting mechanism was introduced by Schapire (1990) for boosting the performance of a weak learning algorithm. After several improvements AdaBoost (Freund et al., 1996) was presented by (Bauer and Kohavi, 1999). The general AdaBoost algorithm is as follows.

Input: training set S of size m, Inducer $\mathcal{I}$, integer T (number of trials).

1. $S' = S$ with instance weights assigned to be 1.
2. For $i = 1$ to T {
3. $\quad C_i = \mathcal{I}(S')$
4. $\quad \epsilon_i = \frac{1}{m} \sum_{x_j \in S' : C_i(x_j) \neq y_j} \text{weight}(x)$ (weighted error on training set).
5. $\quad$ If $\epsilon_i > 1/2$, set S' to a bootstrap sample from S with weight 1 for every instance and goto step 3 (this step is limited to 25 times after which we exit the loop).
6. $\quad \beta_i = \epsilon_i / (1 - \epsilon_i)$
7. $\quad$ For-each $x_j \in S'$, if $C_i(x_j) = y_j$ then $\text{weight}(x_j) = \text{weight}(x_j) \cdot \beta_i$.
8. $\quad$ Normalize the weights of instances so the total weight of S' is m.
9. }
10. $C^*(x) = \arg\max_{y \in Y} \sum_{i : C_i(x) = y} \log \frac{1}{\beta_i}$

Output: classifier C^*.

The AdaBoost algorithm combines several induction algorithms, so that in the presence of a new instance, it tries to determine the class/label of this instance as a function of the decisions (votes) made by all these algorithms.

This chapter presents a method to extract the most relevant sentences from an original document using AdaBoost which is also called AdaBoost.M1. This algorithm produces a set of classifiers and assigns a weight to each one of them. AdaBoost also modifies the training instances weights delivered as input to each classifier on the basis of the models formerly constructed.

The aim of the process adopted by AdaBoost is to minimize the error over the different inputs distribution. The different parameters of the algorithm are:

T: number of iterations
$S_1, S_2, ..., S_t$: t weighted training sets
$C_1, C_2, ..., C_t$: t classifiers
x_i is the input vector of the training instance number i

y_i is the label of the training instance number i

For more details about the algorithm, we refer the reader to the paper by Bauer and Kohavi (1999).

3.2.2 System design

Our architecture is based on two phases: the first one aims to build the AdaBoost learning model. In the second phase, this model which was produced in the previous phase is tested by producing a summary using the model and assessing its quality.

3.2.2.1 Stage one: Building the learning model

The general idea of this process is to construct the training data from a parallel corpus of pairs of <source, summary>documents. This process is presented in more detail in Algorithm 1.

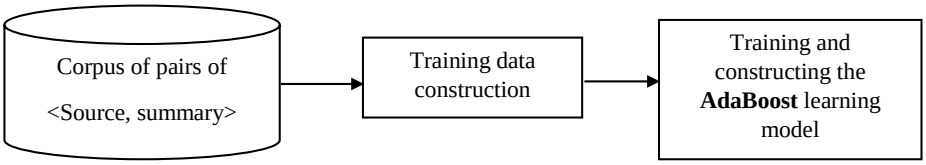

Fig. 3.2 Stage one: Building the learning model

Building the data that will be used to construct the classifier is the cornerstone of our proposal. Given a set of training documents, i.e. a corpus of pairs <source, summary>, the process starts by segmenting each pair of documents <d_i,d_j>from the training corpus which gives us a document d_i= {s_{i1}, s_{i2},..,s_{im}} and a document d_j= {s_{j1}, s_{j2},..,s_{jn}}, each of which consists in a set of sentences. The second step is to find for each sentence in the source document whether it has been removed or kept in the corresponding summary. Depending on the result, we assign a value of one (1) as a label to the sentence to indicate that the sentence has been kept and zero (0) to indicate that the sentence has been deleted in the summary document. The function extract_features gets the different features from a given sentence. The different features considered in this work are: the number of words in a sentence that are common with those in the title; a discrete function which assigns the value zero (0) in case the sentence is not the first in the text and one (1) otherwise; the position of the sentence in the document (the first sentences of the article are most of the time

Algorithm 1 Building the training data

```
1: Input C: corpus of pairs <source, summary >of documents
2: Output T: Training data
3: T=Null {The training set is initialy empty}
4: for each pair <D_i,D_j >in C do
5:    D_i:=Segments (D_i)
6:    D_j:=Segments (D_j)
7:    for each sentence S_k in D_i do
8:       if S_k exists in D_j then
9:          S_k.label:=1
10:      else
11:         S_k.label:=0
12:      end if
13:      F_i:=Extract_features (S_k)
14:      T_i:=CreateInstance (F_i, S_k.Label)
15:      T.add(T_i)
16:   end for
17: end for
18: Return T
```

informative); the number of keywords in the sentence; the length of the sentence; a function that returns zero (0) if the sentence is the last one in the text and one (1) otherwise. Then, given the label as well as the different features, these parameters are passed as arguments to the function CreateInstance. Thus, a new instance is created and added to the training data. The process is repeated for each pair of documents until we get the entire training data.

3.2.2.2 Stage two: Construction of a Summary

The process depicted in figure 3.3 is responsible for the generation of a summary from a source document. The different procedures used in this process are described in more detail in Algorithm 2.

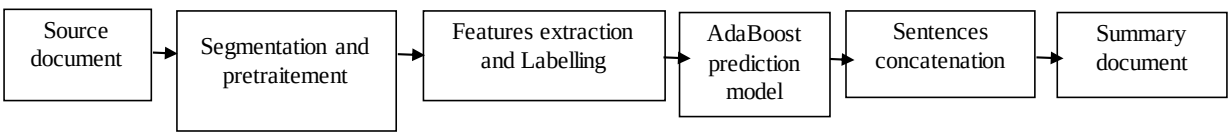

Fig. 3.3 Stage two: summary construction

Algorithm 2 Summary generation

```
1: Input D_i (Source document)
2: Output D_j (Summary made by the system)
3: D_i:=Segments (D_i)
4: D_j:=Null {The summary is initialy empty}
5: for each sentence S_i in D_i do
6:    F_i:=Extract_features(S_i)
7:    Decision:=AdaBoost_predict(F_i) { Get the decision about S_i from the AdaBoost
      model}
8:    if decision=1 then
9:       D_j:=D_j+S_i { Concatenation of two sentences }
10:   end if
11: end for
12: Return D_j
```

The algorithm takes as input a source document D_i then applies a set of operations to generate a summary D_j. The system starts with the initialization of the summary document D_j by associating an empty value to it. Then it segments the source document into sentences and performs the pretreatment specific to the Arabic text that we will explain latter. Once the documents get segmented, for each sentence S_i in the source document D_i, the different features are extracted from the sentence S_i. Next, the extracted features are passed to the AdaBoost classifier which decides whether the sentence should be included in the summary. If so, this sentence is concatenated to the summary under construction. This process is repeated until the complete summary is generated.

3.3 Implementation and test

3.3.1 Corpus

Machine learning allows to learn rules from the observation of a set of instances and their correspending labels. In our case, a corpus of source documents and their corresponding summaries was used. The instances are the feature vectors of the sentences in a document, and the labels are the decisions about the importance of the sentence: label zero(0) indicates that the sentence is not important and label one (1) indicates that the sentence is important and should be included in the summary.

We have built a corpus of documents about technology news. These articles have been collected from the websites cnnarabic.com and bbcarabic.com . The total number

of documents is 20. This dataset is a parallel corpus of <source, summary>pairs. The summaries were manually produced.

3.3.2 Pretreatment and Normalization of the Arabic Text

An initial step in our system is the normalization and pretreatment of the text. Most of the operations that we will present in the sequel are specific to the Arabic language. We have performed the following:

- Words encoding: We have used UTF-8 to encode the different characters in the documents.
- Removal of vowels: the various diacritics have been removed.
  Example: The word الْعَرَبِيّةُ becomes العربية
- Removal of Elongation: it is purely aesthetic and it has nothing to add to the meaning of the word.
  Example: The word العـــــربية becomes العربية
- Normalization of "HAMZA": the following letters: ALEF_HAMZA_ ABOVE, ALEF_MADDA, HAMZA_ABOVE, BELOW ALEF_HAMZA, and BELOW_HAMZA are transformed to ALEF by removing the Hamza.
  Example: إستقلال becomes استقلال

3.3.3 Comparison

Arabic text summarization is not studied enough in the literature. Moreover, we can state that it is very difficult to compare our technique against other existing techniques, due to the lack of gold standard corpora on the one hand and the different measures used to assess text summarization on the other. This is why we have decided to compare AdaBoost against the results obtained using multilayer perceptrons (MLP) and j48 decision trees. These techniques have been very successful for many AI applications. Table 3.1 below shows the results of comparison of the AdaBoost algorithm to a

multilayer perceptrons and the j48 decision tree in terms of F1 score. In this setting, AdaBoost was used to boost the support vector machine classifier, which is a robust machine learning classifier introduced by (Vapnik, 2000). We have used 10-fold cross validation for all the machine learning algorithms.

Precision reflects the ratio of sentences extracted by the system and which were judged to be relevant.

Recall reflects the ratio of relevant sentences that the system extracted (i.e. did not miss).

We use TP to denote true positives and FP to denote false negatives. The recall(R), precision(P) and F1 score(F1) are calculated as follows.

$$P = \frac{TP}{TP + FP} \tag{3.1}$$

$$R = \frac{TP}{TP + FN} \tag{3.2}$$

F_β score makes a balance between recall and precision; its general formula is given as follows:

$$F_\beta = (1 + \beta^2) * \frac{P * R}{\beta^2 * P + R} \tag{3.3}$$

F1 score: By setting the value of β to 1 we get

$$F_1 = 2 * \frac{P * R}{P + R} \tag{3.4}$$

Approach	**MLP**	**AdaBoost**	**j48**
F1-score	66,4%	66,60%	63.20%

Table 3.1 Comparaison between AdaBoost, J48 and MLP in terms of F1 score

3.3.4 Discussion

From the above results we conclude that AdaBoost has given better results than MLP and j48 decision trees in terms of F1 score. This is due to the voting mechanism adopted by AdaBoost. If we look at the task of text summarization itself and its

subjective definition, we conclude that 66.60% F1 score is considered as an acceptable result. We can also state that it is very difficult to compare the proposed approach (AdaBoost) to other existing systems for various raisons. Among these, we can mention the following:

1. It is difficult to compare the performances of the approaches proposed for Arabic text summarization, because each work uses a different dataset.

2. Various researchers use different evaluation measures, which is due to the difficulty of the task itself. In fact, researchers have devoted a lot of effort developing new metrics to assess the quality of a summarization system (Lin, 2004; Louis and Nenkova, 2013). In fact, the evaluation of text summarization systems is still an open issue that has to be tackled.

3. The community working on Arabic natural language processing, and specifically Arabic text summarization, is still quite small.

4. The complexity of the Arabic language in terms of its spelling, vocabulary as well as its morphology makes lexical, syntactic, and semantic ambiguity even higher.

5. The problem of diacritics (tashkyl) in the Arabic language makes it even more complex to handle.

3.4 Conclusion

We have presented in this chapter a machine learning-based approach to Arabic text summarization that uses the AdaBoost algorithm. We have shown the results of the evaluation of the implementation using the F1-score measure. We have shown that this approach outperforms other existing machine learning systems in terms of the F1-score.

We envision a further development of this work in several directions. One direction would be to test our approach on an extended corpus that contains more documents. We also intend to introduce more features like part-of-speech tagging and semantic relations between sentences. Another interesting area of research is to propose a new approach that treats the problem of text summarization but using an abstractive paradigm. In the next chapter we present concept generalization and fusion which is an abstractive sentence generation operation.

Chapter 4

Concept Generalization and Fusion for Abstractive Sentence Generation

4.1 Introduction

In this chapter, we address the problem of abstractive text summarization with a focus on the task of concept fusion and generalization. The latter can be seen as one operation among several ones that can contribute to text summarization (See Figure 4.1) . It is considered a difficult one as it requires a cognitive effort to achieve it. We are particularly interested in generalizing sentences, i.e. such that the system be able to generate from a sentence like *"Selma ate bananas, apples and potatoes"* an output like *"Selma ate fruits and vegetables"* or *"Selma ate some food"*. This task requires the use of world knowledge. In our case, we use WordNet[1] (Miller, 1995) as a source of external knowledge to generalize concepts, hence to abstract sentences.

We automatically generate the generalization and fusion of the concepts of a given sentence through a sequence of steps. The first step is to decide whether a given sentence is generalizable or not. If it is, we generate the set of possible generalizations (versions) of the sentence. The next step is to reduce the space of generalization versions. And, in order to further reduce this space and get a set of generalization versions that are acceptable in natural language, a heuristic-based and a Machine Learning-based model are proposed. Once the best generalization version is found, we generate the compressed sentence. The methodology proposed can generalize even

[1] http://wordnet.princeton.edu/

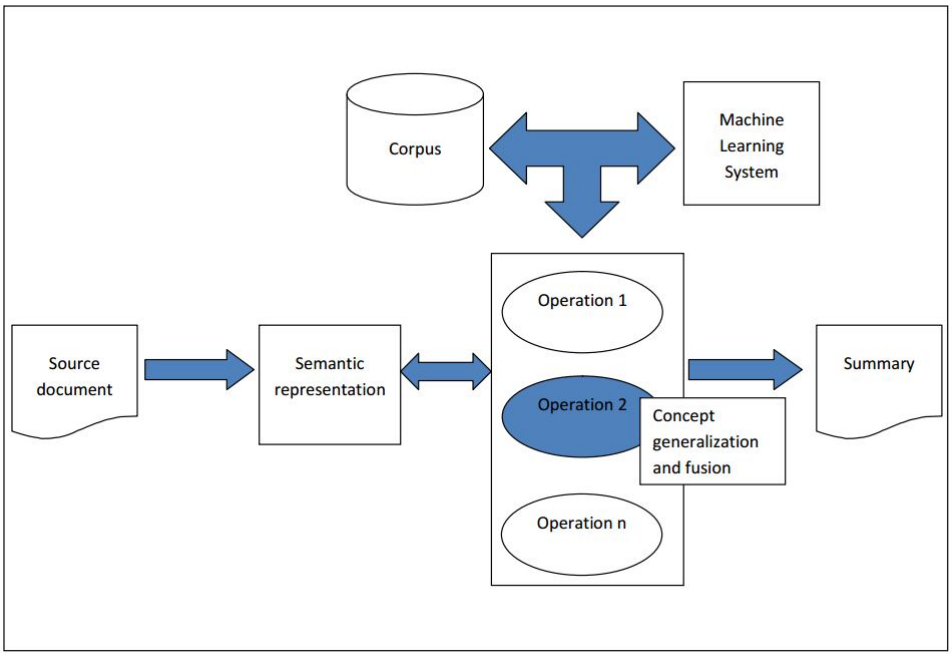

Fig. 4.1 Concept generalization and fusion in the global summarization framework

complex sentences thanks to the dependency parsing module which is used and is described below.

The remainder of this chapter is organized as follows. Section 4.2 introduces the problem statement and definitions. Section 4.3 explains the system design. First, we tackle the problem of extraction of generalizable sentences. We then show how the space of generalization versions can be generated and then reduced. Next, we describe the heuristics we use to select acceptable versions from the space of generalization versions. The evaluation methodology and experimentation work are presented in Section 4.4. A running example is used in Section 4.5 to illustrate the whole approach. Section 4.6 discusses the results we have obtained and Section 4.7 gives a conclusion as well as a listing of some possible directions for the development of text summarization based on this work.

4.2 Problem Statement

From our review of the related work, we have identified a number of research gaps. Firstly, there has been limited previous work on abstractive text summarization, most studies having focused on extractive text summarization. Secondly, there has been

almost no summarization system that considers concept fusion and generalization techniques for abstractive text summarization.

In the sequel, we develop an approach to sentence abtraction (i.e. generalization) which can be used in the context of text summarization. This approach is based on the generalization and fusion of concepts as explained above with the *"eating food"* example. We define the notion of sentence generalization and what it means for a sentence to be generalizable or not. Once we have detected that a given sentence is generalizable, we show how its generalizations can be generated. Among all the generalization versions that get generated, those that are acceptable are selected, which also allows to keep the best one. The system performance is then evaluated.

We consider indeed that abstraction is an important part of summarization in that it allows the reformulation of sequences of text fragments (concepts) in terms of simple concepts that are more general in a taxonomy of concepts and hence capture more concisely the meaning conveyed in a sentence. The notion being quite complex, we have focused our attention in this chapter on the generalization and fusion of conjunctions or disjunctions of concepts within a sentence. We leave it for future work the investigation of this abstraction for more varied types of linkages of concepts as well as for the generalization and fusion across contiguous sentences in a paragraph.

In our approach to concept fusion and generalization within sentences, we will consider that sentences may contain concepts that appear in a taxonomy such as WordNet. As such, we will be able to generalize and/or fuse concepts using the is-a relation, hence hyperonymy paths that may exist between concepts in the taxonomy.

4.2.1 Definitions

Definition 1. *Hyperonymy path* of a concept C
We define a *hyperonymy path* of a concept C as a sequence of concepts [C_1,...,C_n] of the taxonomy such that C_1 = C, C_n is the root concept, and for every pair of concepts (C_i,C_{i+1}), the relation is-a(C_i,C_{i+1}) holds, for $i = 1$ to $i = n - 1$.

Example 1. In WordNet,
Hyperonymy path(*teacher*)= [teacher, educator, professional, adult, person, organism, living_thing, whole, object, physical_entity, entity]

Example 1 shows a hyperonymy path where *"educator"* is the first parent of the concept *"teacher"* and *"entity"* is the root concept. We note that a given concept could have more than one hyperonymy path. In this case, all the hyperonymy paths of the

concept will be merged to produce a single path. This merged path is the *set* of all concepts of the hyperonymy paths of the concept (i.e. without repetition).

Example 2. Paths and merged paths example
Path$_1$(milk)= [milk, beverage, liquid, fluid, substance, matter, physical_entity, entity]
Path$_2$(milk)= [milk, beverage, liquid, fluid, substance, part, relation, abstraction, entity]
Path$_3$(milk)= [milk, beverage, food, substance, matter, physical_entity, entity]
Path$_4$(milk)= [milk, dairy_product, foodstuff, food, substance, matter, physical_entity, entity]
MP(milk)= Path$_1$(milk) ∪ Path$_2$(milk) ∪ Path$_3$(milk) ∪ Path$_4$(milk)
MP(milk)= [milk, beverage, liquid, fluid, substance, matter, physical_entity, entity, part, relation, abstraction, food, dairy_product, foodstuff]

In example 2, the first as well as the second paths of the concept *"milk"* do not include the concept *"food"*. If we consider only one of these paths, we fail at the end to generalize and fuse the concepts. As a solution, we propose to produce a single path (merged paths) which is the union of the concepts of all the paths for the specific concept of interest.

Definition 2. *Merged path*
Let C be a concept and *hyperonymy_paths*(C)=$\{Path_1, Path_2, ..., Path_n\}$, where path$_i$ is one of the hyperonymy paths of the concept C. We define *merged_path(C)* as $\{Path_1 \cup Path_2 \cup ... \cup Path_n\}$.

Definition 3. *Generalizable concept*
A concept C$_i$ ∈ Taxonomy is generalizable (or, equivalently, can be generalized) if ∃ C$_j$ ∈ Taxonomy such that is-a(C$_i$,C$_j$) holds.

The following example will motivate the definition of generalizable and non-generalizable sentences that comes after it.

Example 3. Generalizable and Non-Generalizable sentences
(1) Selma ate (**non-generalizable sentence**)
(2) Selma ate bananas, carrots and apples (**generalizable sentence**)
(3) Selma ate fruits and vegetables (**generalizable sentence**)
(4) Selma ate bananas (**non-generalizable sentence**)
(5) Selma ate food (**non-generalizable sentence**)

In Example 3, the sentence *"Selma ate bananas, carrots and apples"* is generalizable because it has three generalizable concepts and the conjunctive relation holds between them. The sentence *"Selma ate fruits and vegetables"* is generalizable as well since it contains two concepts that are generalizable and the conjunctive relation holds between them. The sentence *"Selma ate"* is not generalizable since it does not contain generalizable concepts connected by a conjunctive or disjunctive relation. Finally, the sentence *"Selma ate bananas"* is not generalizable according to our definition, since it has only one generalizable concept, *"bananas"*, while at least two are required.

Definition 4. *Generalizable sentence*
A sentence S is said to be generalizable if it contains at least two *generalizable concepts* which are connected by a conjunctive or a disjunctive relation. We denote a given generalizable sentence by the set S= $\{C_1,...,C_n\}$ where the set S contains only the concepts of the sentence S which are connected by a conjunctive/disjunctive relation.

Definition 5. *Generalization Version* of a *Generalizable Sentence*
Let S= $\{C_1,...,C_n\}$ be a *generalizable sentence* and C_i a concept of S. We say that V=$\{V_1,...,V_n\}$ is a *generalization version* of S if and only if $\forall i \in (1..n)$ $V_i \in$ *merged path* of C_i.

In example 3, for instance, one generalization version of the generalizable sentence *"Selma ate bananas, carrots and tomatoes"*, which we denote by S={bananas, carrots, tomatoes}, is *"Selma ate fruits, vegetables, and vegetables"* denoted by V={fruits, vegetables, vegetables}. Another one could be *"Selma ate food, food, and food"* denoted as W={food, food, food}. Note that in the sequel, the fusion operation will take care of removing the redundancies from the generalization versions.

Definition 6. *Space of generalization versions* of a *generalizable sentence*
The *space of generalization versions* of a *generalizable sentence* S is the set SGV of all possible *generalization versions* that could be derived from S by applications of generalization operations. SGV={V, where V is a *generalization version* of S}.

Definition 7. *Fusible generalization version*
Let V= $\{V_1,...,V_n\}$ be a *generalization version* of some generalizable sentence S. We say that V is a *fusible generalization version* of S if there exist at least two concepts in V that are the same, in which case the *fused generalization version* is obtained by fusing these concepts, i.e. removing the redundant concepts. Otherwise, by abuse of language, we will say that the fused generalization version V is S itself.

The generalization version V1={fruits, vegetables, vegetables} is a fusible version because the second and third concepts are equal. The fused version of this generalization version is FV1={fruits, vegetables}. The generalization version V2={food, food, food} is fusible as well since all of its concepts are equal. Its fused version is FV2={food}. However, the generalization version V3={fruits, vegetables} is not fusible since all of its concepts are different. In this case, FV3=V3, i.e., remains unchanged.

Definition 8. *Compression Ratio* (CR)
Let V={V_1,...,V_n} be a *generalization version* of a given generalizable sentence S, where n is the number of its concepts, and FV={FV_1,...,FV_m} be its *fused version*. We define the compression ratio CR of V between V and its fused version FV as

$$CR(V) = \frac{m}{n} \tag{4.1}$$

We define below a list of highly abstractive concepts. This list is proposed mainly because when fusing concepts we may end up with fused generalization versions which are not acceptable in the human language, such as *"Selma ate entity and food"* denoted by FV={entity, food}. We have judged that *"entity"* belongs to the list of highly abstractive concepts. We discard any generalization version that contains any of the highly abstractive concepts. This list has been constructed by observing concepts used on different generalization paths in WordNet. We point out that this list could be not exhaustive.

Definition 9. *List of highly abstractive concepts in WordNet*
We have selected the concepts listed in Table 4.1 as the list of *highly abstractive concepts* of WordNet.

abstraction, entity, attribute, whole, physical_entity, matter, object, relation

Table 4.1 List of highly abstractive concepts in WordNet

It is worth pointing out that even if a concept is polysemous, the process of merging paths does not create any confusion in terms of the expected generalization and fusion. Consider, for instance, the concept *"bat"* which has at least two meanings, *"bat"* the animal and *"bat"* the piece of wood used in Cricket and Baseball. If we have a sentence like *"My son fears bats and owls"*, the attempt to merge the hyperonymy paths of the two meanings of *"bat"*, will not find (for fusion purposes) any common ancestor with *"bat"* the wooden piece, except highly abstractive concepts such as *"entity"* and

"physical_entity" which will be filtered out as just explained. Thus the only possible fusions will include parents of *"bat"* the animal and *"owl"*.

4.3 System Design

In the sequel, we give a description of the main steps of our approach. The system works as a pipeline as shown in the global system architecture (Figure 4.2). In the first step, the input raw text is segmented into sentences. Next, the sentences are parsed using a dependency parser. The generalizable sentences are then detected and, for each one, the different paths are generated and the highly abstractive concepts are dropped. Then, the space of generalization versions (SGV) is generated. Due to the large number of generalization versions produced in this step, and because most versions are not acceptable (non-fusible), the space of generalization versions is reduced. In the following step, the reduced SGV is passed on to one of two different approaches (appearance in the same context or the Ml-based approach) in an attempt to further reduce the SGV and select only the versions whose human language equivalents are acceptable . At this stage, the system takes the best version and generates the compressed sentence. The final step is to generate the compressed document.

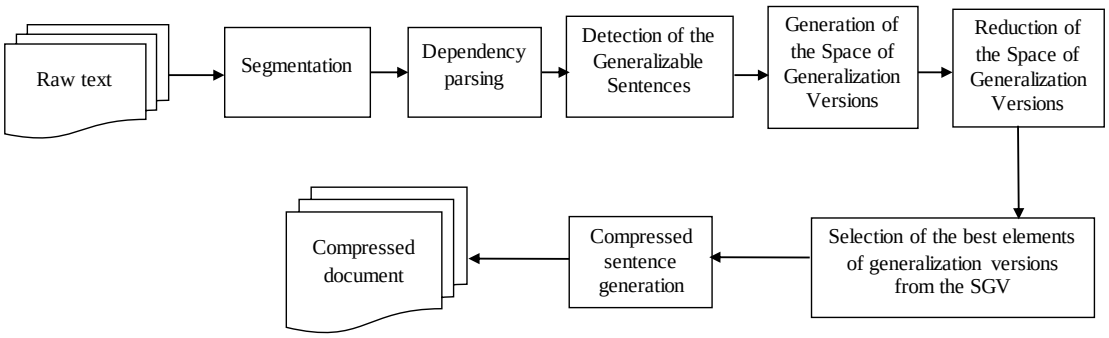

Fig. 4.2 Architecture of the Concept Generalization and Fusion System

4.3.1 Dependency Parsing

Since we need to identify those concepts that are related to the sentence predicate in a similar way and we want to generalize and fuse these concepts, we start by parsing the sentence we want to generalize. Dependency parsing gives us the kinds of relations we look for. Thus, after segmenting the raw text into sentences, syntactic parsing is

performed using the Stanford dependency parser[2] (De Marneffe and Manning, 2008). Example 4 presents how the English sentence *"Selma ate apples, bananas, and potatoes"* is parsed with the Stanford dependency parser.

Example 4. Dependency parsing of the sentence *"Selma ate apples, bananas, and potatoes":*
nsubj(ate-2, Selma-1)
root(ROOT-0, ate-2)
dobj(ate-2, apples-3)
dobj(ate-2, bananas-5)
conj_ and(apples-3, bananas-5)
dobj(ate-2, potatoes-8)
conj_ and(apples-3, potatoes-8)

This states that the sentence predicate, the root, is the verb *"ate"* whose subject is *"Selma"*. The words *"apples"*, *"bananas"* and *"potatoes"* are the direct objects of the predicate *"ate"* and are connected through a conjunction.

4.3.2 Detection of Generalizable Sentences

For every sentence, we select all the concepts that are connected by a conjunctive or a disjunctive relation. The second step is to check whether at least two of these concepts are generalizable. If true, we decide that this sentence is generalizable.

In example 4, we say that *"Selma ate apples, bananas, and potatoes"* is a generalizable sentence since the concepts *"apples"*, *"bananas"* and *"potatoes"* of the sentence could be generalized to give less specific information about the eating event. One possible generalization of this sentence is *"Selma ate food"*. While another more specific could be *"Selma ate fruits and vegetables"*.

4.3.3 Generation and Reduction of the Space of Generalization Versions

4.3.3.1 Generation of the Space of Generalization Versions

The first step in generating the space of generalization versions is to generate the hyperonymy paths of each concept of the generalizable sentence. Then, if a generalizable concept has more than one path, we produce the merged path. Next, an exhaustive

[2]http://nlp.stanford.edu/software/stanford-dependencies.shtml

search algorithm (Breadth-First Search) takes the different merged paths as input and produces as output all the possible combinations between the merged paths, which we call the space of the generalization versions (SGV). One of the shortcomings of this approach is the exponential complexity of the algorithm $O(N^M)$ where N is the number of concepts in the generalizable sentence and M is the size of the merged path.

Figure 4.3 shows how the space of generalization versions is generated, where C_{ij} is the concept j of the merged path i. As explained above, a Breadth-First search takes all the merged paths as input and outputs the space of generalization versions. It explores the neighbourhood nodes, where each node is a concept of a merged path. The algorithm exhaustively searches the entire space until it generates the whole space of generalization versions.

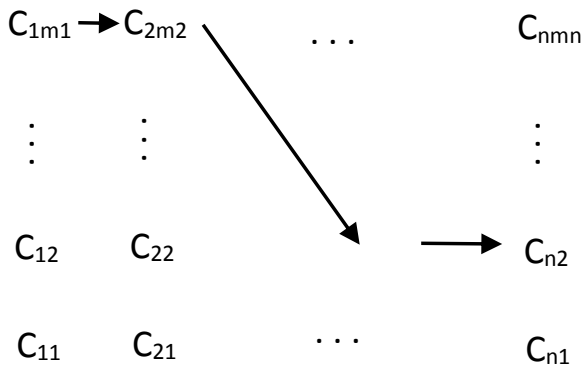

Fig. 4.3 Space of generalization versions

4.3.3.2 Reducing the Space of Generalization Versions

The step of SGV generation produces a large number of generalization versions. This is due to the exponential complexity of the generation of their space. In fact, the versions that are not shorter than their corresponding generalizable sentences are not acceptable, since in the context of abstractive sentence generation we aim at producing shorter sentences, thus shorter documents. Also, versions that contain concepts like *"entity"* or *"physical_entity"* are generally not acceptable in the human language because, if we don't rule out these kinds of versions, we might end up with generalization versions like *'Selma ate an entity'* or *'Selma ate a physical_entity'*. For these reasons, we need to reduce the number of versions of generalizable sentences.

This section discusses the process of reducing the space SGV of generalization versions. As explained in Section 4.3.3.1, the size of SGV grows exponentially. We will

thus adopt a filtering process to reduce it, keeping only admissible versions. To this end, we discard the generalization versions V of SGV whose compression ratio is equal to one.

Algorithm 3 details the process of filtering the SGV of a given generalizable sentence. It takes as input the SGV and, after filtering out some of its elements, returns the resulting SGV. For all the generalization versions V of SGV, the program checks whether the condition CR(V)=1 is true. If so, the generalization version V is removed from SGV. This condition in algorithm 3 amounts to discarding generalizable versions that are not fusible since CR(V)=1 means that the number of concepts of V is equal to the number of concepts of the fused generalization version and hence it would not lead to a compression of the sentence.

Algorithm 3 Reducing the SGV

```
1: Input Space SGV of a generalizable sentence
2: Output (Possibly reduced) SGV
3: for all V in SGV  do
4:    if CR(V)=1 then
5:       SGV.remove(V)
6:    end if
7: end for
8: Return SGV
```

We point out that an initial step that contributes to the reduction of SGV, which would come before the SGV generation step, is to remove from the set of possible paths all the highly abstractive concepts (listed in Table 4.1). By removing highly abstractive concepts, we avoid producing generalization versions that are not natural. For example, the sentence *'Selma ate a physical entity'* does not sound natural to a human judge. This sentence contains the word *'Physical entity'* from the list of highly abstractive concepts. In natural language, one would say *'Selma ate food'* or *'Selma ate fruits and vegetables'* but not *'Selma ate a physical_entity'*.

4.3.4 Heuristics to Select Acceptable Versions from a Given SGV

In order to further reduce the SGV so that it contains only versions that are considered acceptable in human language, we propose three different approaches. The first approach is used as a baseline, it randomly selects the generalization versions, regardless of their meanings. The second approach estimates the acceptability of the sentence

according to the number of appearances of the concepts of the generalization version within the same context in four different NLTK corpora. The third one is a machine learning-based approach which takes as input a set of annotated generalization versions to train the machine learning-based model and, based on the learned model, decides whether the new generalization versions are acceptable or not.

4.3.4.1 Random Selection of Acceptable Versions

Algorithm 4 Algorithm for the RANDOM Selection of Acceptable Generalization Versions

```
1: Input Space SGV of a generalizable sentence
2: Output list_acceptable_V          % The list of acceptable generalization versions

3: list_acceptable_V=Null
4: for all concept V in SGV  do
5:    Generate a random value RandVal (0 or 1)
6:    if RandVal=1 then
7:       list_acceptable_V.add(V)
8:    end if
9: end for
10: Return list_acceptable_V
```

In algorithm 4, the system takes as input the space of generalization versions SGV of a given generalizable sentence and randomly selects the set of acceptable generalization versions which it produces as output (list_acceptable_V). This heuristic is a naive method since it does not rely on any semantic information and does not take into account any feature for the selection of the set of acceptable versions. We use this method as a baseline to see whether the features that we will adopt later will improve the performance of the system or not.

4.3.4.2 Selection of Acceptable Versions Based on the Appearance of Concepts within the Same Context

This heuristic is based on the idea that if the concepts of a given version as well as the predicate of these concepts appear together many times within the same context, then it more likely refers to an acceptable generalization version. We have thus set a threshold so as to select only generalization versions that appear in the same context in the four NLTK corpora more than this predefined threshold number of times. (The selection of the threshold will be discussed in Section 4.4.2.)

Algorithm 5 Algorithm for the Selection of Acceptable Versions Based on the Appearance of Concepts within the same Context

```
1: Input Space SGV of a generalizable sentence,
2:        threshold
3: Output list_acceptable_V        % The list of acceptable generalization versions

4: list_acceptable_V=Null
5: for all concept V in SGV do
6:   if appearTogether(V,predicate) >threshold then
7:     list_acceptable_V.add(V)
8:   end if
9: end for
10: Return list_acceptable_V
```

4.3.4.3 Selection of Acceptable Versions Based on a ML-Based Model

Support Vector Machines Support Vector Machines have been introduced as a machine learning model in (Cortes and Vapnik, 1995). The goal of the technique is to generate a model from the observation of a number of pairs of input-output. This technique aims to find a decision boundary that maximally separates the space into two regions, through the hyper-plane that correctly classifies the data. SVMs have been adopted in this work because they have proven to be effective and robust in numerous Artificial Intelligence applications. We have also compared their performance to other machine learning techniques (See Table 4.4).

Suppose we have a generalization version j, which has to be classified as acceptable or not acceptable according to the SVM score. For linear SVMs the score is a linear combination of relevant features x_j= (x_{j1}, x_{j2},...,x_{jd}), where x_j is a vector with d features and x_{jk} is the value of the feature number k of the generalization version j, k=*1..d*. So z_j, the score of the generalization version j can be defined as:

$$z_j = w_1 x_{j1} + w_2 x_{j2} + ... + w_d x_{jd} + b \tag{4.2}$$

In a compact form:

$$z_j = w x_j^T + b \tag{4.3}$$

where w is considered as a vector that contains the weights of the d features, x_j^T the transpose of vector x_j and b is a constant.

With the aim of using this model for the classification of the generalization version j, the SVM has to learn the values of the parameters w and b on a training corpus.

Suppose the corpus consists of a set of n generalization versions. To calculate the value of the parameters w and b, the SVM finds the hyper-plane that best separates acceptable from non-acceptable versions, maximizing the margin between these two data classes. This margin is the distance between the hyper-planes bounding each class. For more details about the technique, we refer the reader to (Platt, 1999).

Machine learning features Let us present the two features that we have selected in the ML-Based system.

1. Generalization version features Here, we have built a feature vector whose elements are the concepts of a given generalization version. We have gained some insight from the bag of words representation which is widely used in text classification. There, the documents are the training instances and the words of the documents are the features. In this work however, the instances are the generalization versions and the features are the concepts of the generalization versions.

2. SCF By Same Context Frequency (SCF) we refer to the number of appearances of concepts within the same context in the sentences of four different NLTK corpora. This feature tries to capture whether a given generalization version is actually used in natural language. Let V= $\{C_1,...,C_n\}$ be a version where C_i is a concept. The *SCF* of a given generalization version V is calculated as follows.

$$SCF(V) = \sum_{i=1}^{m} appearTogether(V, S_i) \tag{4.4}$$

where m is the number of sentences in the four training NLTK corpora, and

the function *appearTogether* returns 1 if all the concepts of the generalization version appear in sentence S_i, and 0 otherwise.

Annotation Algorithm 6 shows how the process of annotation of an SGV is performed. It takes as input the SGV and outputs the training data T. Initially the training set is empty. For every version in SGV, the human annotator makes a decision about the version, i.e. whether it is accepted or not. The next step is to extract the different features from this version. Then, given the label and the different features, the function *CreateInstance* is invoked; it takes as input the features and the label and creates a new instance which is then added to the training data. The process is repeated for all the versions of the given SGV.

Algorithm 6 Annotation

Input Space SGV of a generalizable sentence
Output T: Annotated training data
T=Null %The training set is initially empty
for all V_i in SGV **do**
  decision=Annotate (V_i)
  if decision=1 **then**
    V_i.label:=1
  else
    V_i.label:=0
  end if
  F_i:=Extract_features (V_i)
  T_i:=CreateInstance (F_i, V_i.label)
  T.add(T_i)
end for
Return T

Algorithm 7 ML-Based model

Input Space SGV of a generalizable sentence
Output list_acceptable_V
list_acceptable_V=Null
for all V in SGV **do**
  if ML_Classifier(V)=1 **then**
    list_acceptable_V.add(V)
  end if
end for
Return list_acceptable_V

ML-based model selection Algorithm 7 is responsible for selecting the set of acceptable versions. As for the previous approach, it takes as input the space of generalization versions of a generalizable sentence and outputs the set of acceptable versions. In this case, however, the decision about the selection of a given version is based on a trained SVM classifier which predicts whether the generalization version should be selected or not.

4.3.5 Generation of the Compressed Sentence

At this stage, we take the best element from the set of acceptable generalization versions of the SGV and generate the compressed sentence. For this we have defined a set of rules. These rules work on the syntactic dependency parsing of the original generalizable sentence and try to give a correct compressed sentence as output. Algorithm 8 is used to generate the compressed sentence from a given generalization version V and a dependency parse tree of the generalizable sentence. If two concepts or more in the generalization version V could be fused, we apply the *fusion rule*. Then, we apply the *"Some" rule*, the *"Adjective Deletion" rule* and the *"Number Deletion" rule*. At the end, the compressed sentence is output.

In this work we have tried to cover the basic rules to generate a correct sentence. We are aware that there might exist other rules that could enhance the quality of the generated compressed sentence, but our focus here is just on the basic requirements that a sentence should have when fusing the concepts.

Algorithm 8 Rules to generate the compressed sentence

1: **Input** a generalization version V=$\{V_1,V_2,..,V_n\}$ and a dependency parsing of a generalizable sentence
2: **Output** a compressed sentence C
3: **if** V is a fusible version **then**
4: ***Fusion rule***
5: ***Some rule***
6: **if** there exists any concept in V that is connected to an adjective **then**
7: ***Adjective Deletion rule***
8: **end if**
9: **if** there exists any concept in V that is connected to a number **then**
10: ***Number Deletion rule***
11: **end if**
12: **end if**
13: **Return** C

Fusion rule Having a generalization version V=$\{V_1\, V_2 \ldots V_n\}$ and the parse tree of a given generalizable sentence, we should first verify the category of every concept V_i of V (countable or mass noun) because if we have to fuse concepts which are countable nouns, we have to add the plural suffix *'s'*. However, if the concept is a mass noun, we should not add this suffix. For instance, in the sentence *"Selma ate apples, bananas, and potatoes"*, the concepts *"apples"*, *"bananas"* and *"potatoes"* could be generalized to yield the concept *"food"*. The concept *"food"* is a mass noun so we do not add the suffix *'s'* to it. But if we have, for example, the sentence *"I visited my mother and father"*, which contains the concepts *"mother"* and *"father"*, these two concepts could be generalized into the concept *"parent"*. Since the latter is a count noun, when we generate the compressed sentence by fusing the concepts, we should add the plural suffix *'s'* to the concept *"parent"*. The generated sentence will be *"I visited my parents"*.

The Some rule Suppose that Selma likes bananas, apples but not mangos. When we generalize this sentence we might generate the sentence *"Selma likes fruits"* or *"Selma likes food"*. If we take as input the sentence *"Selma likes bananas and apples."* and the system outputs the sentence *"Selma likes food"*, then we have generalized more than needed. The problem with this generalization is that we have reached a level of generalization where we have included other concepts that Selma does not necessarily like (e.g. mangos). To overcome this problem, we have added the *some rule*. In this case, we will output the sentence *"Selma likes some fruits"* or *"Selma ate some food"*.

Adjective Deletion rule In order to keep only the important content of a sentence, we remove the adjectives that are related to the direct objects which we want to generalize. This will yield sentences that are much shorter, conveying only the most important information. For instance, if we have the sentence *"I ate an apple and a delicious banana"*, we drop the adjective *"delicious"*. This adjective is connected to the direct object *"banana"*, as shown in example 5.

Example 5. Dependency parsing of the sentence *"I ate an apple and a delicious banana."*
nsubj(ate-2, I-1)
root(ROOT-0, ate-2)
det(apple-4, an-3)
dobj(ate-2, apple-4)
cc(apple-4, and-5)
det(banana-8, a-6)

amod(banana-8, delicious-7)
conj(apple-4, banana-8)

Number Deletion rule We delete numbers that are connected to the direct objects. So, if we have as input the sentence *"I have two apples and three bananas"*, then we could output the sentence *"I have some fruits"* or *"I have some food"*. The generated sentence has been produced using the fusion rule, the Some rule and the Number Deletion rule.

Example 6. Dependency parsing of the sentence *"I have two apples and three bananas."*
nsubj(have-2, I-1)
root(ROOT-0, have-2)
num(apples-4, two-3)
dobj(have-2, apples-4)
cc(apples-4, and-5)
num(bananas-7, three-6)
conj(apples-4, bananas-7)

4.4 Experimentation and Evaluation

4.4.1 Evaluation Methodology

We have followed a methodology to asses the performance of the system which uses the recall, precision and F1-score defined as follows:

		Actual value		
		Positive	Negative	Total
System's value	Positive	TP	FP	$TP+FP$
	Negative	FN	TN	$FN+TN$
	Total	$TP+FN$	$FP+TN$	N

Table 4.2 Confusion Matrix

Precision Of all generalization versions which the system predicts as acceptable, precision measures the fraction which is actually acceptable:

$$P = \frac{TP}{TP+FP} \tag{4.5}$$

Recall Of all generalization versions that are actually acceptable, recall measures the fraction the system correctly detects as acceptable:

$$R = \frac{TP}{TP + FN} \tag{4.6}$$

F_β-score This measure makes a balance between recall and precision; the general formula is as follows:

$$F_\beta = (1 + \beta^2) * \frac{P * R}{\beta^2 * P + R} \tag{4.7}$$

F1-score By setting the value of β to one we get:

$$F_1 = 2 * \frac{P * R}{P + R} \tag{4.8}$$

In order to study this problem, we have used four different NLTK corpora, namely the Brown Corpus, the Gutenberg Corpus, the Reuters Corpus, and the Web and Chat Text. We notice that for the evaluation, the initial number of generalizable sentences (GS)that we have extracted from the four NLTK corpora is 980 (that satisfy definition 4). Among the 980 GSs that we have extracted, and after a manual check, we have selected only 108 GSs. The data set that we will use in the evaluation is composed of the generalization versions of all the 108 SGVs.

The total number of instances (generalization versions) that we have used to train the system and evaluate its performance is 366 out of which 245 are negative (annotated as non-acceptable) and 121 positive examples (annotated as acceptable). This number is the sum of the generalization versions of 108 SGVs. This number is obtained after reducing the size of all the SGVs by eliminating *non-fusible versions*.

4.4.2 Evaluation of the Selection of Acceptable Versions Based on the Algorithm of Appearance of Concepts within the Same Context

The context-based approach has a threshold parameter upon which it takes the decision whether it selects a generalization version or not. Figure 4.4 presents the performance of the system (F1 score) in terms of the variation of the threshold. We observe that the system performance starts decreasing as the value of the threshold gets bigger. This is so because the chance of a given generalization version to be accepted will be very low (due to the high value of the threshold) and, at the end, none of the generalization

versions will be selected. The best threshold value that we get is 0 i.e., we select a generalization version if its concepts along with its predicate appeared in the training corpus more than zero times (at least once) in the same context. The best F1-score value that we have reached for this approach is 0.53.

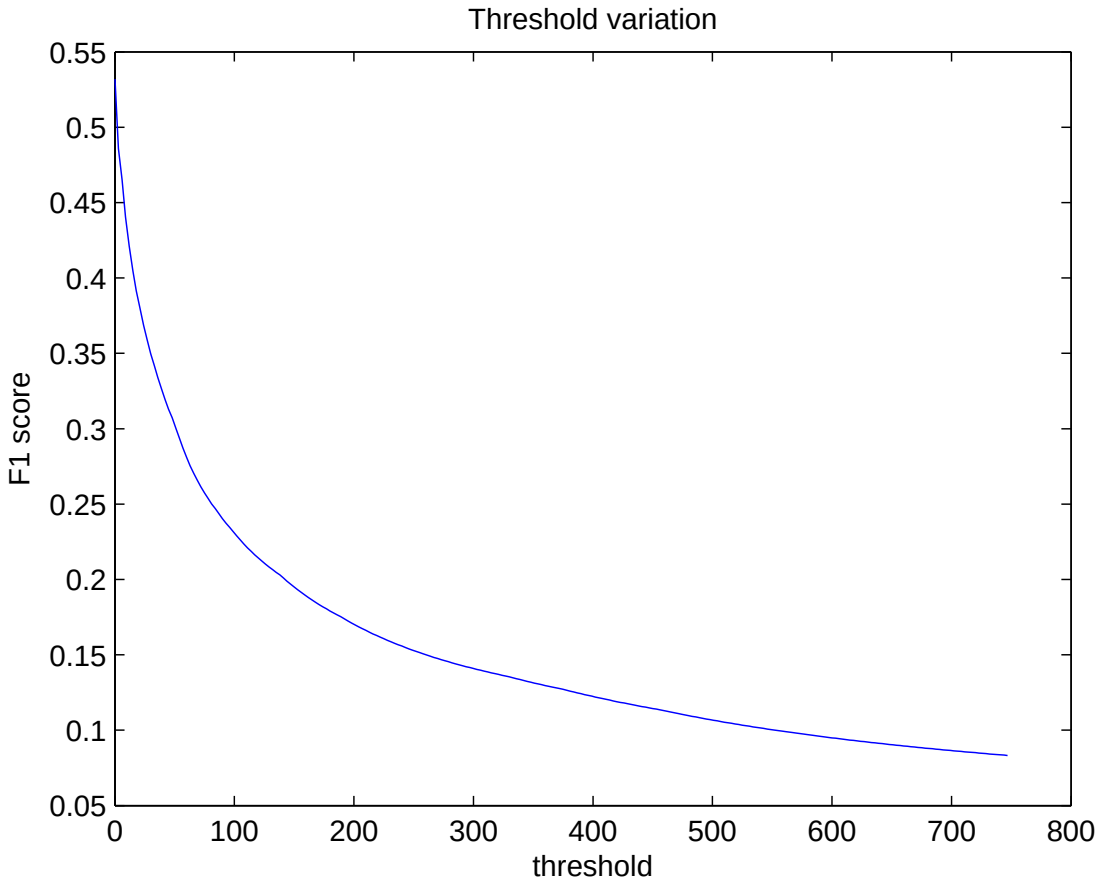

Fig. 4.4 F1-Score of the system as a function of the threshold of the selection of Acceptable Versions using the Algorithm of Appearance of Concepts within the same Context

4.4.3 Evaluation of the Ml-Based Approach

We have used the Libsvm package (Chang and Lin, 2011) to train and test the SVM-based model; all the parameters have been set to their default values. In order to

train and test the other Machine Learning-based approaches, we have used the Weka[3] framework (Bouckaert et al., 2013).

Table 4.3 gives the result of comparing Support Vector Machines to other Machine Learning approaches for our problem. It shows the superiority of SVMs in terms of F1-score over the other Machine Learning models .

Technique	Precision	Recall	F1-score
Naive Bayes	0.797	0.488	0.605
J48	0.699	0.826	0.758
Random Forest	0.717	0.752	0.734
SVM	0.798	0.818	0.808

Table 4.3 Comparison of the Ml-based techniques

4.4.4 Comparison

Table 4.4 compares the different approaches we have used to filter the set of acceptable generalization versions from the different SGVs. The comparison is performed using the F1-score. The table shows that the performance of the system is poor when using a very naive method like "random selection". The performance increases when relying on a corpus-based approach like the context-based approach which decides whether to select the version based on a threshold (number of appearances of the concepts of the generalization version and the predicate within the same context in the four NLTK corpora). Yet this approach suffers from some problems which are due to the fact that it is very difficult to find the concepts of the generalization version within the same context in the 04 NLTK corpora. So, in this situation, many versions will have a score of zero and the system will fail to detect that these versions are acceptable in human language. This is due to the small size of the 04 NLTK corpora. To improve the performance of this approach, we have proposed a Machine Learning-based approach in which we have combined two types of features. In the first, the features are specific to the generalization version; they include the concepts and the predicate of the generalization version. The second is the number of appearances of concepts of the version in the four NLTK corpora. We have trained all the machine learning approaches using 10-fold cross validation and SVM has achieved a significant improvement over the other approaches reaching an F1-score of 0.808.

[3]http://www.cs.waikato.ac.nz/ml/weka/

Approach	Precision	Recall	F1-score
RANDOM selection	0.300	0.460	0.370
Context-based approach	0.750	0.410	0.530
SVM-based approach	0.798	0.818	0.808

Table 4.4 Comparison between the approaches that filter the set of acceptable generalization versions

4.5 A Running Example

In this section, we present how a given sentence could be generalized using the proposed system. Table 4.5 depicts a passage from the book *"The Ball and The Cross" by G.K. Chesterton (1909)*, which has been extracted from the file *chesterton-ball.txt* of the *gutenberg corpus*.

... I have the biscuits and the tinned meat, and the milk. You have the chocolate, I think? ... " "Yes," said MacIan, like a soldier taking orders. "Very well, then, come on. March. We turn under that third bush and so down into the valley." And he set off ahead at a swinging walk. ...

Table 4.5 Passage from the file *"chesterton-ball.txt"* of the *"gutenberg"* corpus

Dependency Parsing and Generalizable Sentence (GS) Detection After segmenting the text into sentences and feeding the sentence *"I have the biscuits and the tinned meat and the milk."* to the Stanford dependency parser it will detect that such a sentence has the predicate *"have"* as its root which has three direct objects *"biscuits"*,*"meat"* and *"milk"*. All the objects are related to each other by means of the conjunctive relation. The sentence concepts could be generalized and fused by means of is-a relations. Below is the result of the parsing of the sentence by the Stanford dependency parser. The result of this step is to extract the generalizable sentence that we denote as GS={biscuits, meat, milk} and the predicate *"have"* as its root.

Example 7. Dependency parsing of the sentence *"I have the biscuits and the tinned meat and the milk"*

nsubj(have-2, I-1)
root(ROOT-0, have-2)
det(biscuits-4, the-3)
dobj(have-2, biscuits-4)
det(meat-8, the-6)
amod(meat-8, tinned-7)
dobj(have-2, meat-8)
conj_and(biscuits-4, meat-8)
det(milk-12, the-11)
dobj(have-2, milk-12)
conj_and(biscuits-4, milk-12)

Paths Generation Once we have identified a generalizable sentence S, we feed the different concepts of S to WordNet which will generate the different hyperonymy paths. Table 4.6 shows the two paths of the concept *"biscuits"*, table 4.7 the path of the concept *"meat"* and table 4.8 the three different paths of the concept *"milk"*.

Path$_1$(biscuits)= [biscuit, quick_bread, bread, baked_goods, food, solid, matter, physical_entity, entity]
Path$_2$(biscuits)= [biscuit, quick_bread, bread, starches, foodstuff, food, substance, matter, physical_entity, entity]

Table 4.6 Paths of the concept *"biscuits"*

Path$_1$(meat)= [meat, food, solid, matter, physical_entity, entity]

Table 4.7 Paths of the concept *"meat"*

Path$_1$(milk)= [milk, beverage, liquid, fluid, substance, matter, physical_entity, entity]
Path$_2$(milk)= [milk, beverage, liquid, fluid, substance, part, relation, abstraction, entity]
Path$_3$(milk)= [milk, beverage, food, substance, matter, physical_entity, entity]
Path$_4$(milk)= [milk, dairy_product, foodstuff, food, substance, matter, physical_entity, entity]

Table 4.8 Paths of the concept *"milk"*

Removal of Highly Abstractive Concepts This step consists in removing the highly abstractive concepts from the paths which helps in reducing the complexity of the problem.

Path Merging Path merging (PM) is the operation defined in section 4.2.1. This will help in having a higher coverage for the generalization versions of the generated SGVs. In fact, if we do not merge the paths, the system will fail at the end to produce the sentence *"I have the food"*. Suppose that for the concept *"milk"* we have taken into consideration only the first path. Because this path does not contain the concept *"food"*, the system will fail to produce the sentence *"I have the food"*. The different merged paths (MPs) are defined as follows:

1. MP(biscuits)= Path_1(biscuits) $\cup$ Path_2(biscuits)
2. MP(meat)= Path_1(meat)
3. MP(milk)= Path_1(milk) $\cup$ Path_2(milk) $\cup$ Path_3(milk) $\cup$ Path_4(milk)

Tables 4.9, 4.10, and 4.11 present the merged paths for the concepts *"biscuits"*, *"meat"* and *"meat"*, respectively.

MP(biscuits)= [biscuit, quick_bread, food, foodstuff, bread, solid, baked_goods, substance, food, starches]

Table 4.9 Merged paths for the concept *"biscuits"*

MP(meat)= [meat, food, solid]

Table 4.10 Merged paths for the concept *"meat"*

MP(milk)= [milk, beverage, liquid, fluid, substance, food, dairy_product, foodstuff]

Table 4.11 Merged paths for the concept *"milk"*

Generation and Reduction of the Space of Generalization Versions Table 4.12 presents a sample of the first 20 versions of the space of generalization versions of the generalizable sentence *"I have the biscuits and the tinned meat, and the milk"* denoted as GS={biscuits, meat, milk}. The first column represents the initial generalization

Initial generalization version	Fused generalization version	Compression Ratio (CR)
beverage food food	beverage food	0.66
food food food	food	0.33
foodstuff food food	foodstuff food	0.66
fluid food food	fluid food	0.66
dairy_product food food	dairy_product food	0.66
liquid food food	liquid food	0.66
substance food food	substance food	0.66
milk food food	milk food	0.66
beverage solid food	beverage solid food	1
food solid food	food solid	0.66
foodstuff solid food	foodstuff solid food	1
fluid solid food	fluid solid food	1
dairy_product solid food	dairy_product solid food	1
liquid solid food	liquid solid food	1
substance solid food	substance solid food	1
milk solid food	milk solid food	1
beverage meat food	beverage meat food	1
food meat food	food meat	0.66
foodstuff meat food	foodstuff meat food	1
fluid meat food	fluid meat food	1

Table 4.12 Sample from the space of generalization versions of the generalizable sentence *"I have the biscuits and the tinned meat, and the milk"*

versions. The second column represents the fused versions. The third column gives the compression ratio (CR), which is calculated using Formula 4.1. For instance, the second initial generalization version has three concepts. By applying the fusion operator we obtain only one concept. In this case, the compression ratio is one over three, i.e. 0.33. The tenth initial generalization version has three concepts and its fused generalization version has three versions as well. As a consequence, the compression ratio is three over three, i.e. 1. The versions that have a compression ratio equal to one will be eliminated as explained above.

Selection of Acceptable Generalization Versions The following step is to feed the selected generalization versions to one of the two proposed methods so as to select generalization versions that are natural in human language. After eliminating the versions that are not natural, only the generalization version V={food, food, food} will remain.

Generation of the Compressed Sentence The final step is to generate the compressed sentence *"I have some food"* which is composed of the predicate *"have"*, the subject *"I"*, the direct object *"food"* and the quantifier *"some"*. The initial sentence was *"I have the biscuits and the tinned meat, and the milk"*. The compressed sentence has been generated by applying the *fusion rule* and the *some rule*. For the *fusion rule*, the concept *"food"* is a mass noun; in this case, when we fuse the concepts we do not add the plural suffix 's'. We have also used the *some rule* so as not to generalize more than needed. As such we generalize and fuse concepts in such a way that we preserve the meaning of the original sentence. On the other hand, we have lost in specificity as the resulting sentence is less specific than the original one, but this loss of detail was to be expected since this is the side effect of summarization.

4.6 Discussion

In this chapter, we have tackled the problem of sentence generalization. In so doing, we have faced several problems and challenges, some of which are as follows:

1. In some cases, the parser makes some errors and this hinders the decision whether a sentence is generalizable or not. In order to concentrate on the problem at stake, we have decided to select only syntactically correct sentences.

2. The high complexity of the space of generalization versions $O(N^M)$ yields a huge number of versions, which increases the difficulty of finding the best way to reduce this space. Our approach uses various filters that attenuate this problem.

3. The use of WordNnet as a concepts resource has indeed shown an interesting dimension of the suggested solution. A richer lexical-semantic sentence abstraction is made possible. Nevertheless, it would be interesting to experiment with a WordNet Domains resource so as to extract domain-specific knowledge within the approach we have suggested.

4. Regarding the experiments, we have tried to investigate the ability of the system to select acceptable generalization versions. We have proposed three approaches to find the one that will perform better in selecting the best generalization version.

    - The first one is a naive method that does not rely on any feature to select the best generalization version. This method was the worst in terms of F1-score.

    - The second approach is a heuristic-based one that counts the number of times the concepts of the generalization versions appear together in different corpora. If this number is greater than a given threshold (the best threshold that we have found experimentally is zero as shown in Figure 4.4), this version will be selected. The intuition behind this idea is that generalization versions which have already been used in real corpora are acceptable. The problem with the heuristic method is that many versions will have a score of zero according to formula 4.4 thus they get rejected even though they are acceptable according to the human annotators. This method is unable to deal with generalized sentences that have not been seen before in the training corpora.

    - The third approach was proposed to deal with the weaknesses of the heuristic approach. To tackle this weakness, a machine learning method which takes into consideration different features has been proposed. Experiments have shown that this approach has performed better than the heuristic approach; it has indeed shown its ability to select generalization versions which are acceptable in human language even though these generalization versions have not been seen before in the training corpora.

In terms of the results that we have obtained, the approach that tries to estimate the acceptability of a given version on the basis of the number of appearances of the concepts along with the predicate of a given version in the four NLTK corpora has not given good enough results. It seems that the four NLTK corpora are not large enough to handle such a problem. As a consequence, a lot of versions will have a score of zero and will not be selected. To solve this problem, we should think about estimating the acceptability of the generalization versions using larger corpora to enhance the performance of the context-based approach. The ML-based approach has benefited from the strength of two features (contextual and generalization version features) and has achieved a significant improvement over the context-based approach.

Overall, our approach of concept generalization and fusion for abstractive sentence generation has some obvious strengths as well as some shortcomings that are now discussed:

- Strengths:
  - Our approach to sentence abstraction by means of concept generalization and fusion is able to go beyond word deletion. As a matter of fact, it produces as output abstractive sentences that contain words not necessarily seen in the source sentence and which are the result of the generalization and fusion of the concepts of the source sentence.
  - From a theoretical point of view the approach addresses some reasoning aspects. The fact that the system we propose tries to abstract sentences by generalizing and fusing concepts is a cognitive effort which opens a door for other types of abstractions (than what we have explored) within the same sentence as well as between contiguous sentences. The results we have achieved are interesting and encourage us indeed to do further research on such richer abstractive aspects.
  - Regarding the mechanism underlying the approach, we are convinced that having selected to use a dependency parsing module will allow us to cope with more difficult cases thanks to its ability to detect different syntactic constituents and binary relations between them. This will in turn facilitate the tackling of the richer forms of abstractions mentioned in the previous (Strengths) item, provided appropriate operations are defined on parts of the dependency trees.

- Limitations:
  - The algorithm adopted to generate the SGVs has an exponential complexity. We need to implement a more intelligent algorithm that could explore the SGVs with a better complexity.
  - We have decided to limit our study to only some abstractive cases (sentences that satisfy definition 4, i.e. with concepts linked by conjunctions or disjunctions); this is to avoid the risk of generating inappropriate or ill-formed sentences as output. As a further step, we intend to cover more complicated cases, which we have started investigating.
  - The approach is restricted to abstracting at a sentence level. The approach should scale up to deal with document level abstraction. i.e., the system should be able to generalize and fuse concepts which do not necessarily belong to the same sentence. We leave this for future work.

The concept fusion and generalization operation should be integrated within a full summarization framework. Because the concepts fusion and generalization approach is only one among other operations that a summarization system should perform, ideally such a system should incorporate operations like sentence fusion, sentence compression, concepts fusion and generalization as well as a system that allows the selection of the important sentences, etc. We should also think about a methodology on how to combine these operations and to define the priority between them. This will be tackled in chapter 6.

To the best of our knowledge, and with the exception of Lin (1995), where a methodology for the generalization of concepts was presented, and then Hovy and Lin (1998), where the authors included this operation in their SUMMARIST[4] system using Wavefront to select the most relevant concepts which they take as input to be generalized, the work presented here is the first to investigate the operation of concept generalization and fusion for sentence abstraction with its potential use as an operation among several ones in abstractive text summarization, and to propose an efficient computational method for its implementation. This idea of concept fusion and generalization operation is one form of abstraction where we accept to lose on specificity keeping the general meaning and coherence of the text. What we do in our work is new in that we tackle a difficult operation that is not addressed in the literature.

[4]http://www.isi.edu/natural-language/projects/SUMMARIST.html

One novelty in this work is a method that takes into consideration the "syntactic level" of concepts given by a dependency parser (the Stanford dependency parser in our case); this is used in our approach to detect the sentences that can be generalized. We have also handled the semantic aspect by using WordNet to generalize the concepts of a given sentence. The main contribution of this work consists in (1) the methodology we present to generate what we define as the space of generalization versions, and (2) the different approaches we use to select the best sentence generalizations from this space so we can generate the “best” one as output.

4.7 Conclusion

In this chapter, we have addressed the problem of concepts fusion and generalization for abstractive sentence generation. We have shown that this problem is not well addressed due to the difficulty of this task which, at the core, is about the difficulty of reasoning and language generation. The methodology we have adopted has allowed us to answer various research questions which include the detection of generalizable sentences and the generation and reduction of the space of generalization versions. We have also proposed two approaches to select the set of acceptable generalization versions. The first approach estimates the acceptability of a given generalization version by counting the number of times the concepts and the predicate of a given generalization version that appear within the same context in four different NLTK corpora. The second one is a machine learning-based approach which relies on a SVM classifier to distinguish between acceptable and non-acceptable generalization versions. The results have shown the superiority of the latter approach over the other approaches since it benefits from the strengths of two types of features.

This research can be useful to researchers interested in natural language processing. For example, it could be used in conjunction with other tasks like text summarization or categorization. This could improve the quality of the results of these tasks. It could be used, for instance, to display shorter texts in applications for mobile devices. It should also improve the quality of the generated text summaries by mentioning key (general) concepts. One can think of using the approach in reasoning systems where different concepts appearing in the same context are related to one another with the aim of finding a more general representation of the concepts. This could be in the context of Goal Formulation, expert systems, scenario recognition, and cognitive reasoning more generally. Sentence abstraction could also be of interest to a broader community interested in Artificial Intelligence and reasoning in general. Indeed, the

notion of abstraction is an issue that has already been addressed from reasoning and philosophical perspectives. As stated in Ganascia (2015), the notion of abstraction is one of the foundations of Floridi's Philosophy of Information (Floridi, 2011). We point out that Floridi has various contributions on abstraction from a philosophical viewpoint Floridi (2008), Floridi (2009), and Floridi (2013). It was also stated in Ganascia (2015) that the notion of abstraction is not new; since it has been given different meanings in various fields, more specifically in scientific disciplines and, particularly, in computer science.

We have shown through this work that the abstraction aspect could be addressed in practice thanks to the natural language processing tools available today. Although the approach presented in this chapter has shown interesting results regarding sentence abstraction, it does not deal with all abstraction cases and is limited to working at a sentence level. Nevertheless, it opens a door on interesting problems in front of the NLP and AI communities. These problems, if solved, should generalize our work towards reaching fully abstractive abstraction systems as first step, and, as an ultimate goal, summarization systems which produce summaries similar to those produced by humans.

For our part, we envision a further development of this work in several directions. One direction would be to introduce more features and thus try to give more meaning to the generalization versions. Another way to improve the system performance is to use a larger corpus to estimate the acceptability of the generalization versions, especially for the context-based approach. Another problem that has to be solved is the use of a richer external knowledge source with more domain-specific knowledge since WordNet seems to provide almost none. An interesting other future direction is to extend the idea of concept fusion and generalization to not only intra-sentence generalization but to inter-sentence generalization. We leave these for future work. The ultimate goal of this work is to integrate the concept fusion and generalization operation that we have studied in this work into a global summarization framework and test the effect of this operation within the context of abstractive text summarization. This is discussed in chapter 6. In the next chapter we present a corpus for Arabic sentence compression.

Chapter 5

TALAA-ASC: A Corpus for Arabic Sentence Compression

5.1 Introduction

Sentence compression is still considered as one of the most challenging tasks in natural language processing Huang et al. (2012). It could be seen as a subtask of text summarization Clarke and Lapata (2008). Instead of considering a whole document, the focus here is to drop from any single original sentence a set of tokens, while keeping as much information as possible from the original sentence, and ensuring that the compressed sentence remains grammatically well-formed.

Sentence compression has been useful for many artificial intelligence applications. For example, in text summarization, the quality of the generated summaries has improved when using sentence compression (Chali and Hasan, 2012; Jing, 2000; Li et al., 2013; Lin, 2003; Liu and Liu, 2009; Martins and Smith, 2009; Qian and Liu, 2013; Zajic et al., 2007). It has also been beneficial for tasks like text simplification (Angrosh et al., 2014) and mobile devices (Corston-Oliver, 2001; Sarkar, 2008).

Compared to other languages than Arabic, a lot of effort has been done, especially for the English language, both in developing sentence compression systems, and building resources that help in the evaluation and building of these systems. Unfortunately, if we look at the work on Arabic sentence compression, we find very few contributions in this area. This could be explained by the fact that it is very difficult to build sentence compression systems without having a corpus dedicated to this task. This is what has aroused our interest in developing an Arabic corpus for sentence compression.

In this chapter, we present TALAA-ASC, the first Arabic sentence compression corpus (See Figure 5.1). This corpus will (1) help evaluate the Arabic sentence com-

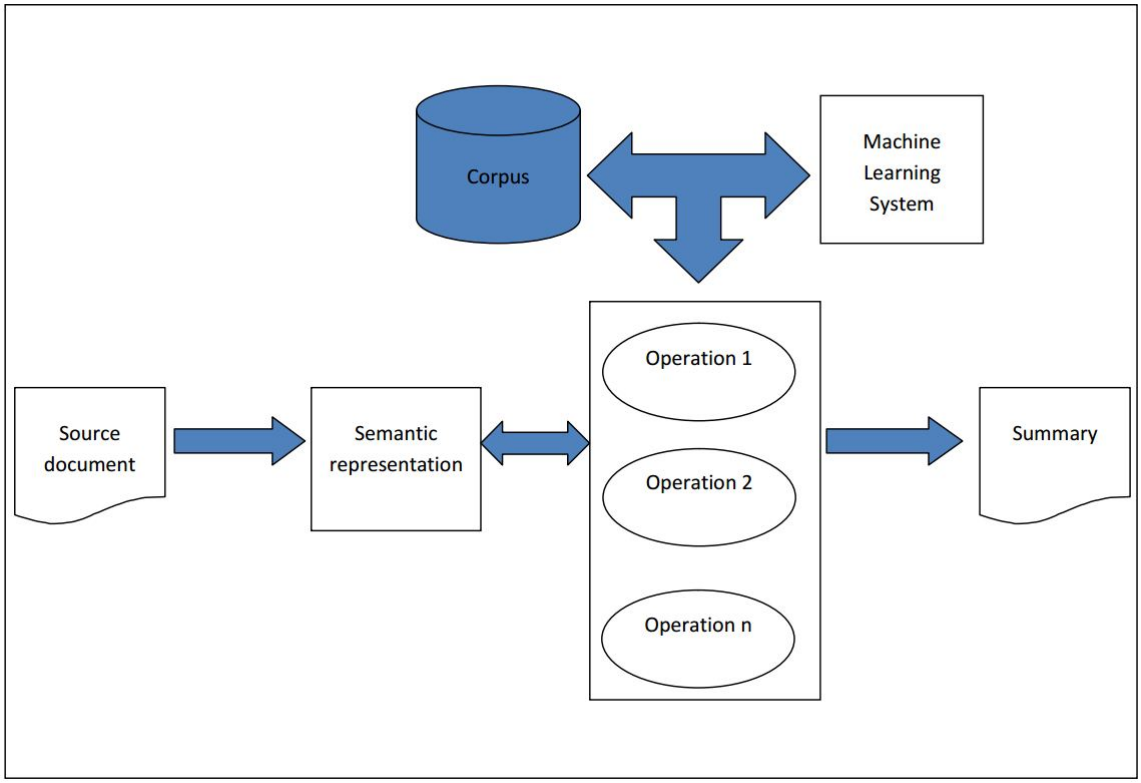

Fig. 5.1 The summarization corpus in the global summarization framework

pression systems and (2) be a much-needed resource for building sentence compression systems through the generation of learning models from this corpus.

The following is an example from the Arabic sentence compression corpus that shows an original sentence and its compression.

Original sentence:

بعد ما يقرب من 100 عام قد يكون الباحثون قد اقتربوا الآن من صنع لقاح يقضي على وباء السل الذي يفتك بنحو مليون ونصف المليون انسان سنويا.

Compressed sentence:

بعد ما يقرب من 100 عام قد يكون الباحثون قد اقتربوا الآن من صنع لقاح يقضي على وباء السل.

Here, the compressed sentence has been constructed by deleting the word sequence

ألذي يفتك بنحو مليون ونصف المليون انسان سنوياً.

The remainder of this chapter is organised as follows. Section 5.2 presents the methodology we have followed in order to build the Arabic sentence compression corpus (TALAA-ASC). The different statistics performed on the corpus are presented in Section 5.3. Section 5.4 concludes this work.

5.2 The TALAA-ASC corpus

5.2.1 Annotation Guidelines

When constructing the Arabic sentence compression corpora, we have followed the same guidelines provided in (Clarke and Lapata, 2008) . This corpus has been annotated (compressed) by one annotator and validated by two human experts in the Arabic language. The annotator has been given the following instructions:

- Feel free to remove from sentences any words considered unessential;
- Preserve the most important information in the source sentence;
- Ensure that the compressed sentence remains grammatically-well formed;
- Keep a sentence uncompressed, if you consider it as inappropriate for compression;
- Do not remove entire sentences even if you believe they contain no informational content with regard to the topic of the document.

Following these guidelines, we have compressed 70 documents, articles published in newspaper websites, belonging to five different categories (see Tables 5.1 and 5.3). These articles have been collected from the web. The total number of sentences of the corpus is 609 sentences (see the details in Table 5.2).

Table 5.1 Number of documents in each category in TALAA-ASC

Category	management and work	medicine	news and politics	sports	technology
# of Documents	10	11	14	10	25

Table 5.2 Number of sentences in each category in TALAA-ASC

Category	management and work	medicine	news and politics	sports	technology
# of Sentences	104	84	104	88	229

Table 5.3 Number of words in each category in TALAA-ASC

Category	management and work	medicine	news and politics	sports	technology
# of Words	2467	2657	2876	1856	5991

5.2.2 Corpus structuring using XML

We have used XML (eXtensible Markup Language) to represent and structure the corpus Cunningham (2006). Below is an example of a document from the corpus structured in XML format. Every file of the corpus is represented in XML format which contains pairs of sentences of the form <source, compression>. The top node is the document id.

5.3 Statistics

5.3.1 Size of the TALAA-ASC corpus

As mentioned above, the TALAA-ASC corpus consists of a collection of 70 articles published in newspaper websites. The articles were classified into five different categories as shown in Table 5.1. The table also presents the number of articles in each category. Table 5.2 shows the number of sentences in each category and table 5.3 the number of words per category.

5.3.2 Distribution of sentences according to their length

Figure 5.3 presents the distribution of sentences according to their length in the TALAA-ASC corpus. It shows that the majority of sentences have a length between 20 and 30 words. We observe also that the sentences that have more than 60 words

```
<?"UTF-8"-encoding "1.0"-version xml?>
<"sport.12"-id doc>
<"technology.12.0.S"-id original>
تفوقت شركة سامسونغ على شركة آبل في مبيعات الهواتف الذكية
، خلال الأشهر الثلاثة الأولى من العام، لتنهي بذلك شركة الإلكترونيات الكورية الجنوبية احتكار "نوكيا"،
أكبر مصنع للهواتف المحمولة في العالم من حيث حجم التصنيع.
<original/>
<"technology.12.0.C"-id compressed>
تفوقت شركة سامسونغ على "آبل" في مبيعات الهواتف الذكية، خلال الأشهر الثلاثة الأولى من العام،
لتنهي بذلك احتكار نوكيا.
< compressed />
<"technology.12.1.S"-id original>
وقال أليكس سيكتور من شركة "ستراتيجي أناليتكس" إن شحنات سامسونغ العالمية ارتفعت بواقع 253 في المائة
إلى $ 44,5 مليون جهاز، مع نمو الطلب على هواتفها الذكية من طرز "غالاكسي نوت" و$"S2"$ و$"Y"$
<original/>
<"technology.12.1.C"-id compressed>
وقال أليكس سيكتور من شركة "ستراتيجي أناليتكس" إن شحنات سامسونغ العالمية ارتفعت،
مع نمو الطلب على هواتفها الذكية من طرز "غالاكسي نوت" و$"S2"$ و$"Y"$
< compressed />
<"technology.12.2.S"-id original>
وأرجع سيكتور ارتفاع مبيعات هواتف سامسونغ النقالة، إلى اعتمادها على مزيج من التصميمات الأنيقة لمكونات
الأجهزة مع خدمات الأندرويد، فضلاً التوزيع الكثيف عالمياً.
<original/>
<"technology.12.2.C"-id compressed>
وأرجع سيكتور ارتفاع مبيعات هواتف سامسونغ، إلى اعتمادها على مزيج من التصميمات الأنيقة لمكونات
الأجهزة مع خدمات الأندرويد، فضلاً التوزيع الكثيف عالمياً.
< compressed />
<"technology.12.3.S"-id original>
والجمعة أعلنت "سامسونغ" عن تحقيق أرباح قياسية بلغت$ 4,44 مليار دولار خلال الفصل الأول من العام
مدفوعة بالطلب القوي على مبيعات الهواتف الذكية.
<original/>
<"technology.12.3.C"-id compressed>
والجمعة أعلنت "سامسونغ" عن تحقيق أرباح قياسية خلال الفصل الأول من العام.
< compressed />
<"technology.12.4.S"-id original>
وساهمت مبيعات "ابل" من هواتف "اي فون" الذكية، والتي بلغت$ 35,1 مليون جهاز
خلال الربع الأول، في مضاعفة الأرباح الفصلية للشركة الأمريكية.
<original/>
<"technology.12.4.C"-id compressed>
وساهمت مبيعات "ابل" من هواتف "اي فون" الذكية في مضاعفة الأرباح الفصلية للشركة الأمريكية.
< compressed />
<"technology.12.5.S"-id original>
وتراجعت مبيعات "نوكيا" الفنلندية بأكثر من النصف إلى $11,9 هاتف خلال الفترة من يناير/ كانون
الثاني إلى مارس/ آذار، لتهوى حصتها في سوق الهواتف الذكية إلى$ 8,2 في المائة
من$ 23,5 في المائة بالفصل الأول العام الماضي.
<original/>
<"technology.12.5.C "-id compressed>
وتراجعت مبيعات "نوكيا" بأكثر من النصف خلال الفترة من يناير/ كانون الثاني إلى مارس/ آذار.
< compressed />
...
<doc/>
```

Fig. 5.2 Example of a document from the TALAA-ASC corpus

Table 5.4 Percentage of part-of-speech (POS) tags dropped for the TALAA-ASC corpus. #drop refers to the frequency the POS tag was dropped from the source data. % drop is the percentage of times the POS tag was dropped in forming the compression.

Tag	% drop	# drop
NN	32.60%	1364
DTNN	11.69%	489
NNP	11.54%	483
DTJJ	6.57%	275
IN	6.21%	260
VBD	5.57%	233
JJ	5.49%	230
VBP	4.20%	176
CD	2.72%	114
NNS	2.22%	93
WP	2.15%	90
DTNNS	1.98%	83
PUNC	1.48%	62
DT	1.12%	47
RP	0.83%	35
VBN	0.69%	29
DTNNP	0.47%	20
CC	0.47%	20
VN	0.43%	18
NOUN	0.31%	13
WRB	0.31%	13
ADJ	0.23%	10
RB	0.23%	10
PRP	0.14%	6
JJR	0.11%	5
VBG	0.07%	3
DTJJR	0.02%	1
VB	0.02%	1

are very rare. This plot gives an indication about the complexity that Arabic sentence compression systems will encounter when trying to compress sentences. Indeed when sentences are very long, the complexity of the problem will increase and sentence compression systems will struggle to output high quality compressions.

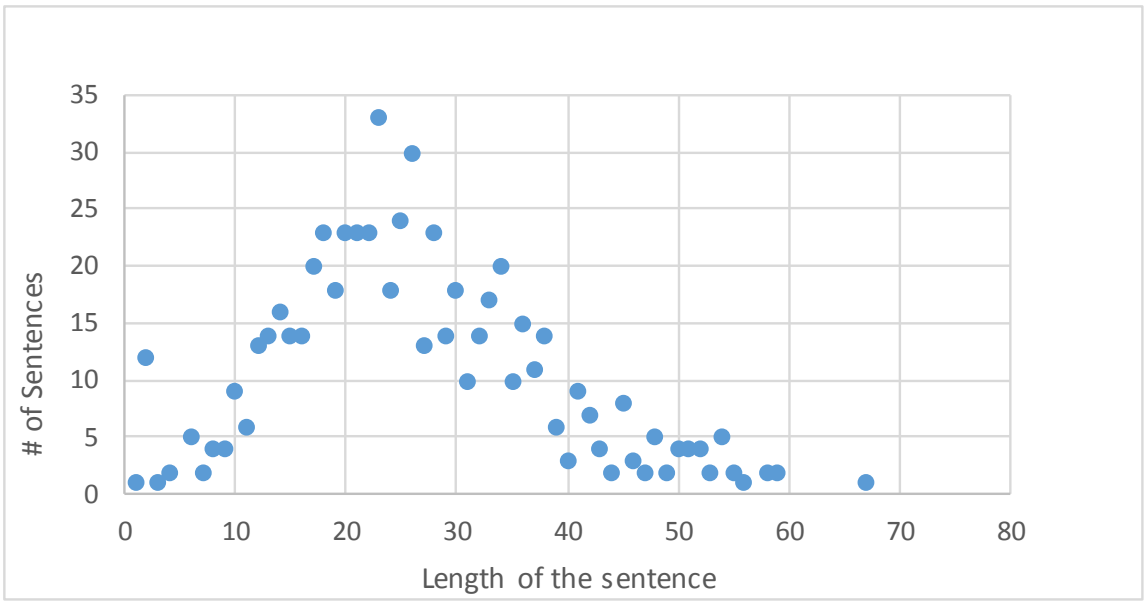

Fig. 5.3 Distribution of sentences according to their length in the TALAA-ASC corpus.

5.3.3 Percentage of Drops

Table 5.4 gives details of the grammatical categories being dropped by the annotator. In order to tag the corpus sentences, we have used the Stanford POS Tagger[1] . The various part-of-speech (POS) tags, their frequencies and the percentage of times each POS tag is dropped from the corpus is also given in Table 5.4. By examining Table 5.4, our conclusions support those of Clarke and Lapata (2008) and Jing (2000) that a naive baseline that merely removes parts of speech like all adjectives is not sufficient to create high quality compressions. In fact, if we look at the percentage of drops of all the adjectives (DTJJ, JJ, ADJ, JJR, and DTJJR) in the corpus we find that it represents only 12.42% from the percentage of all the drops.

5.3.4 Compression rate

The compression rate for a given sentence is calculated by dividing the number of tokens of the compressed sentence by the length of the source sentence. The average compression of the corpus is calculated by summing up the compressions of all the sentences of the corpus and dividing this sum by the total number of sentences of the

[1]http://nlp.stanford.edu/software/tagger.shtml

Table 5.5 Compression in each category in TALAA-ASC

Category	management and work	medicine	news and politics	sports	technology
# Compression	0.798	0.761	0.800	0.766	0.810

Table 5.6 Compression range for the five categories of the TALAA-ASC corpus

Category	Compression range (lowest-highest)
management and work	11.42% - 100%
Medicine	28.77% - 100%
news and politics	38.96% - 100%
sports	31.25% - 100%
technology	25% - 100%

corpus. Table 5.5 shows the compression rate for the five different categories of the TALAA-ASC corpus.

We also confirm the results of Clarke and Lapata (2008) that the compression depends mainly on the sentence structure and the information content not on the length of the sentence. Our results also confirm that the compression rate cannot be fixed for a given corpus or a given category of documents or even for sentences of the same length (see table 5.6).

5.3.5 Scatter plots of the lengths of source sentences against compression rate for the TALAA-ASC corpus

As in Clarke and Lapata (2008) we will examine the relationship between the compression rate and the sentence length (but for Arabic, in our case). This analysis may give us some hints as to how to select the best compression rate for a given sentence. Figure 5.4 shows plots of the source sentence length against the compression rate for the five categories of the TALAA-ASC corpus, while Figure 5.5 shows the plot for the entire corpus. All plots are very scattered and do not prove any correlation between compression rate and source sentence length.

5.3.6 Graph bars for the distribution of compression rates for the TALAA-ASC corpus

In order to study the distribution of compressions we plot the followings graph bars (Figures (5.6,5.7,5.8,5.9,5.10,5.11)). They allow to investigate the effect of categoriza-

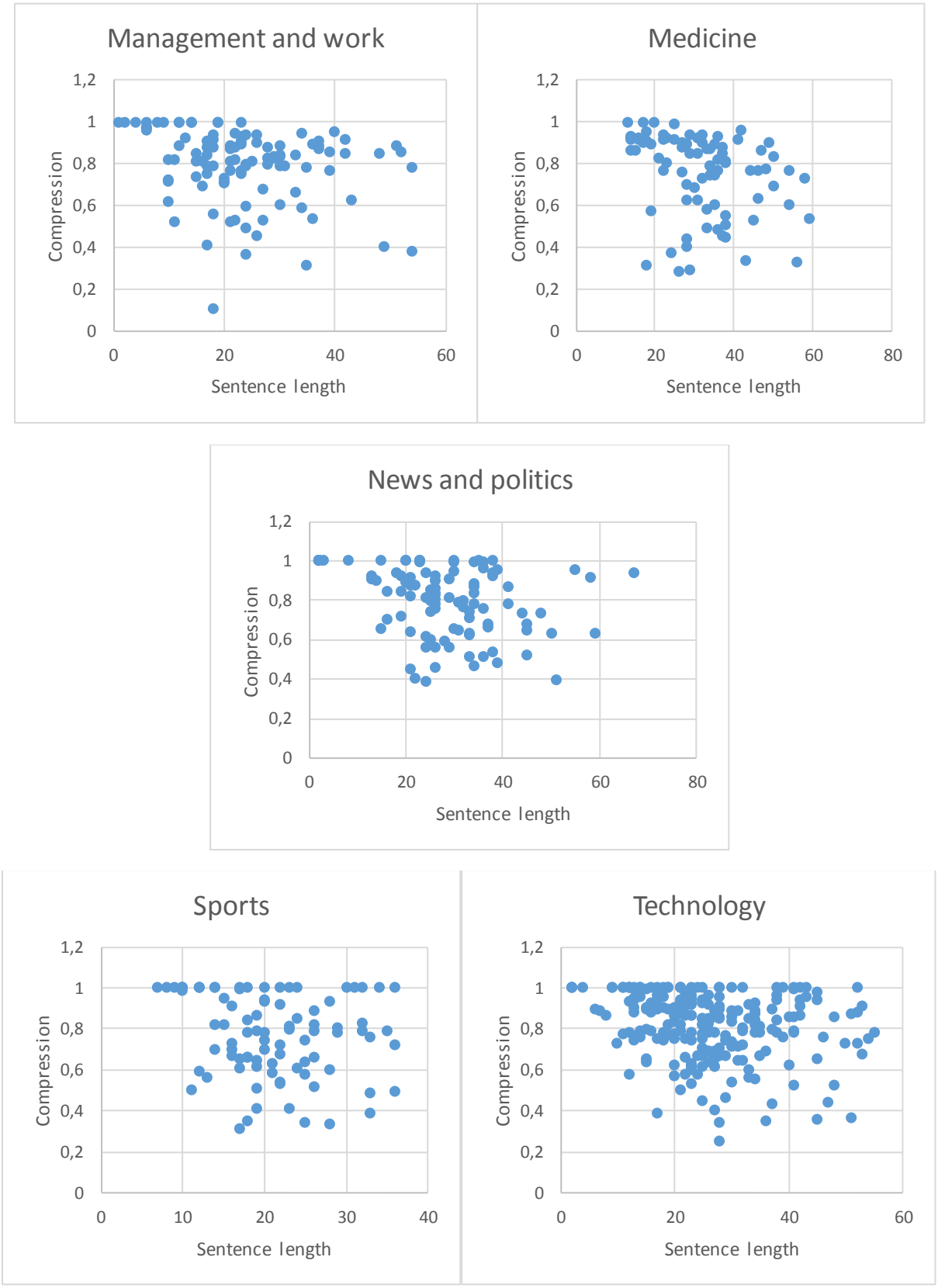

Fig. 5.4 Relation between the length of a sentence and its compression for the five categories of the TALAA-ASC corpus

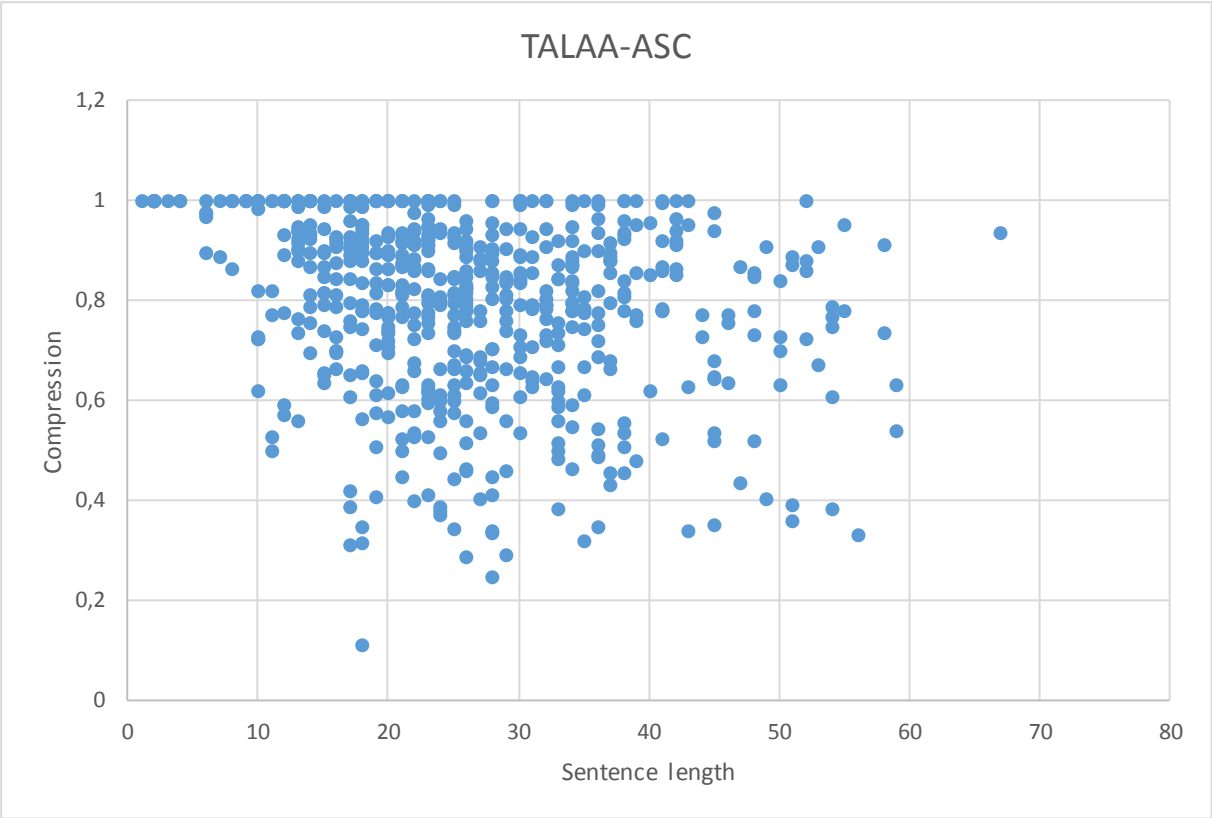

Fig. 5.5 Relation between the length of a sentence and its compression for the TALAA-ASC corpus

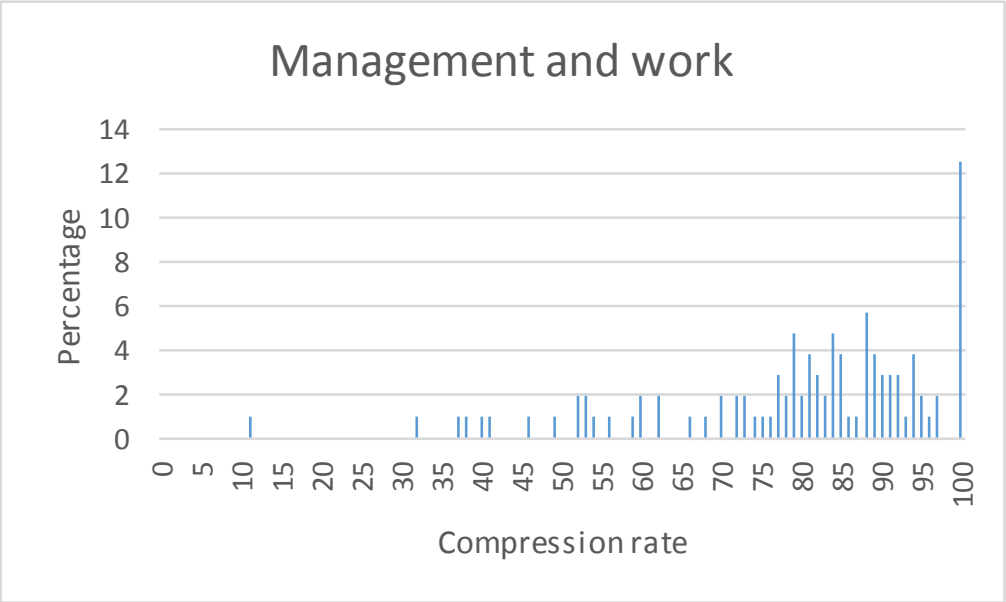

Fig. 5.6 Compression rate distribution for the "management and work" category in the TALAA-ASC corpus

tion on sentence compression. The percentage of sentences that remain uncompressed is very different between the categories. This gives an indication that the category of a document has an impact on the sentence compression system.

5.3.7 Why do some sentences remain uncompressed?

Here we intend to investigate why some sentences remain uncompressed. In fact, the high percentage of compression rate is met when sentences remain uncompressed, i.e. the compression rate is equal to one. We observe also that sentences that have a compression rate below 0.3 are very rare. In the sequel we try to analyze the percentage of sentences that remain uncompressed in each category of the corpus.

From the TALAA-ASC corpus we have observed that sentences that remain uncompressed fall into one of the following categories:

- **Direct speech:** We have found out that a lot of sentences identified as direct speech remain uncompressed.
- **Elementary sentences:** Removing some words from elementary sentences results is ungrammatical sentences.

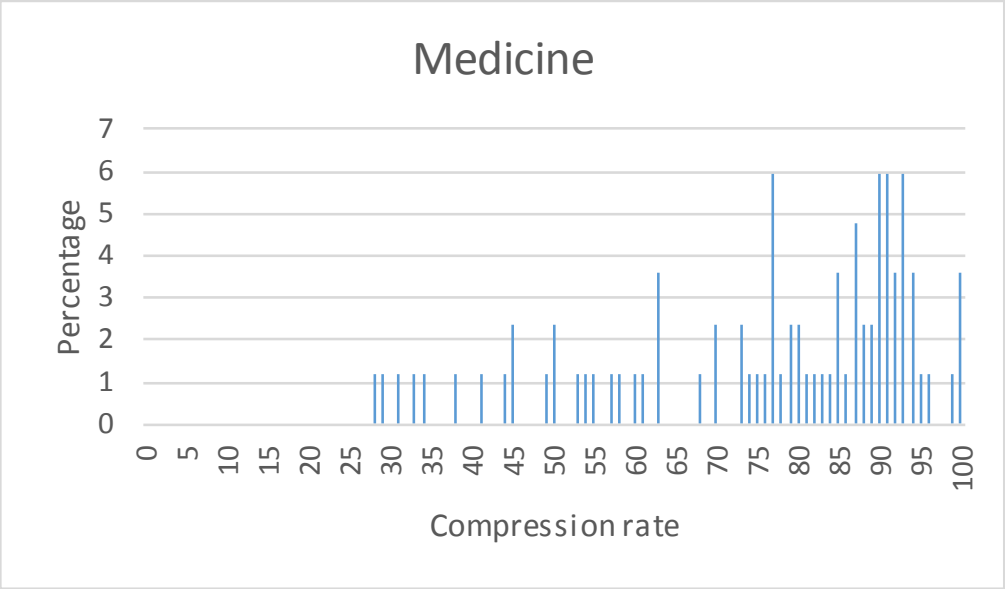

Fig. 5.7 Compression rate distribution for the "medicine" category in the TALAA-ASC corpus

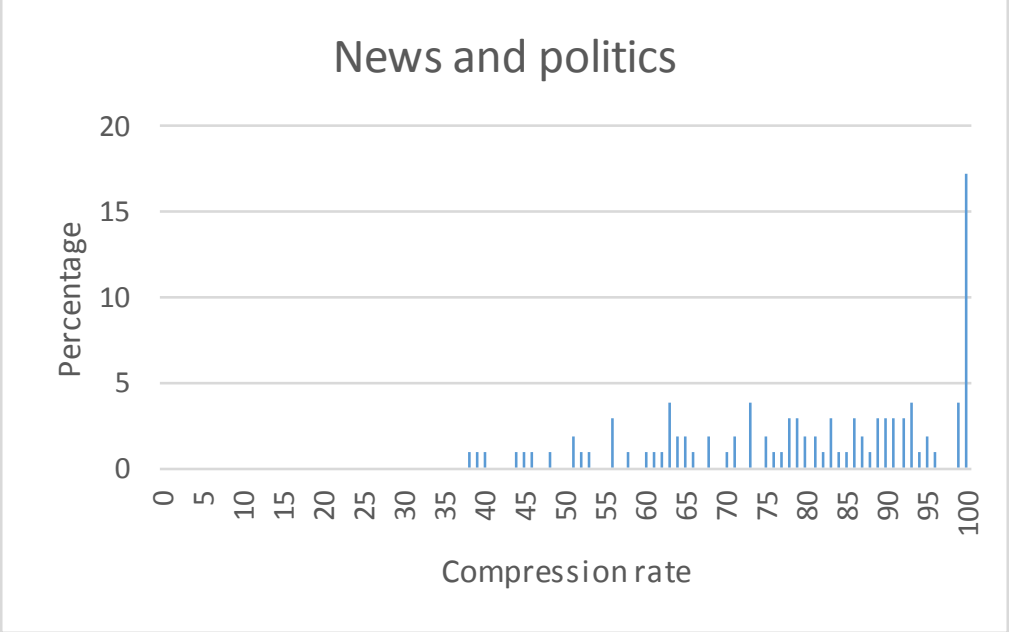

Fig. 5.8 Compression rate distribution for the "news and politics" category in the TALAA-ASC corpus

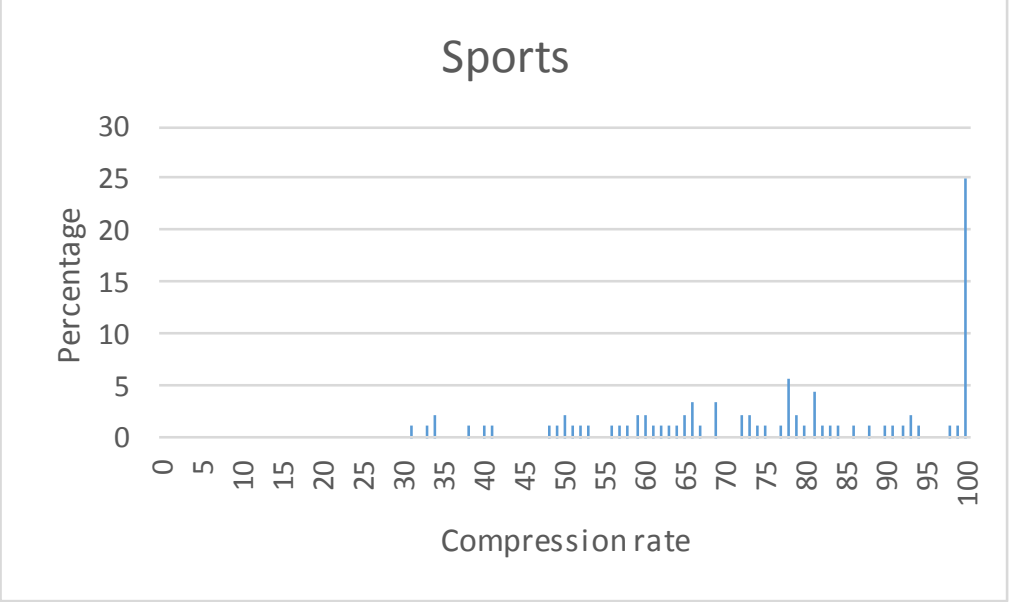

Fig. 5.9 Compression rate distribution for the "sports" category in the TALAA-ASC corpus

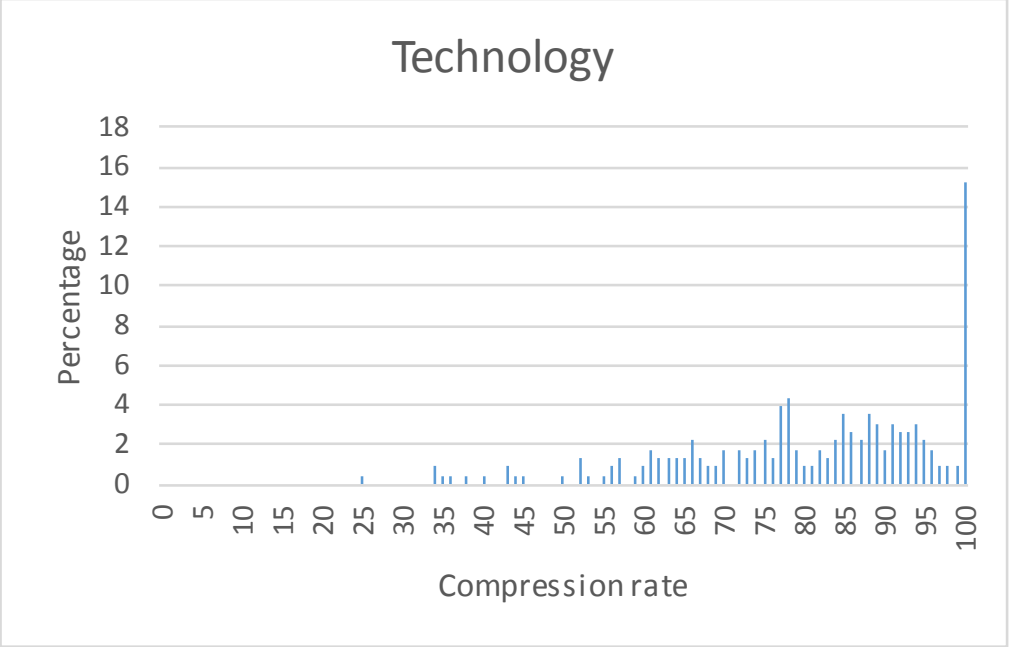

Fig. 5.10 Compression rate distribution for the "technology" category in the TALAA-ASC corpus

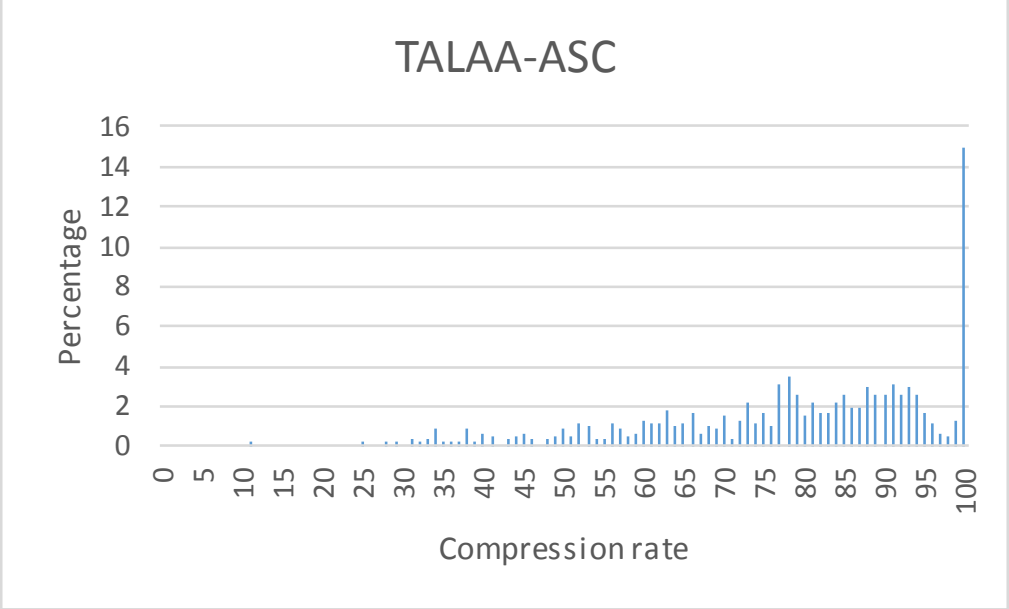

Fig. 5.11 Compression rate distribution for the TALAA-ASC corpus

Table 5.7 Percentage of uncompressed sentences in the different categories of the TALAA-ASC corpus

Category	Percentage of uncompressed sentences
Management and work	12.5%
Medicine	3.57%
News and politics	17.30%
Sports	25%
Technology	15.28%

- **Section title:** Since some documents contain sections with titles, the annotator has kept the title uncompressed.
- **Loss of meaning:** In some situations, when we remove some words from a sentence we lose the meaning of the original sentence even though the compressed sentence is grammatically well-formed.

5.4 Conclusion

In this chapter, we have presented TALAA-ASC, the first Arabic sentence compression corpus that we will make available to the NLP community. We have performed different analyses and statistics on this corpus.

One of the aims of this study was to investigate the effect of categorization on sentence compression. Even though the global compression differences between categories was not significant, we have observed from Table 5.7 that the percentage of sentences that remain uncompressed is higher for some categories like the "sports" category (25%), average for "news and politics" (17.3%), "technology" (15.28%) and "management and work" (12.5%), while very low for the "medicine" category (3.57%). This indicates that future sentence compression systems should take into consideration the categories of sentences before compression.

The characteristics of the Arabic Language also had an effect on the statistics. In fact, Arabic language sentences tend to be very long. This makes it very difficult for compression systems to output high quality compressions (the complexity of the problem being very high). In fact, for the case of sentence compression, we have to find the best compression out of the $2^n - 1$ compressions. When sentences are very long, the number n increases, which in turn increases the problem size.

We have observed also that there is no relation between sentence length and compression rate. In fact, we can find different compression rates for sentences of the same length. We conclude that the only criteria that direct the selection of compressions are the sentence structure and the information content.

Some kinds of sentences remain uncompressed. Generally, sentences that remain uncompressed are either direct speech, elementary sentences, section titles, or sometimes when we compress the sentence we observe some loss of meaning, even though the sentence remains grammatically well-formed.

This work could be improved and extended in several directions. One possible improvement will be to enrich the current TALAA-ASC corpus by adding more documents. We also intend to get help from the ANLP community in order to extend the corpus. As a future direction, it is worth investigating the effect of inter-annotation in sentence compression. For this work, the corpus has been compressed by only one annotator and validated by two human experts to ensure that the compressed corpus is grammatically and semantically correct. In the next chapter we present the global text summarization framework.

Chapter 6

TALAA-ATSF: A Global Operation-Based Arabic Text Summarization Framework

6.1 Introduction

Text summarization is a cognitive effort that requires a deep understanding of the source document. According to (Hasler (2007); Jing (2002)), humans tend to use a number of operations to generate summaries. From our investigation of the aforementioned and other text summarization studies, we have come to the conclusion that these operations fall into one of two categories: *single-sentence operations* or *multi-sentence operations*. A *single-sentence operation* is applied to a single sentence. These are for instance *sentence compression*, *concept generalization and fusion*, *sentence selection*, etc. *Sentence compression* is a *single-sentence operation* because it is applied to a single sentence, while *sentence fusion* is a *multi-sentence operation* because it needs more than one sentence to be applied. On the other hand, a *multi-sentence operation* is applied in a setting where two sentences or more are merged.

Most of the work that we have found in the literature followed the extractive approach because of the difficulty of the abstractive one. We also found a lack of summarization frameworks that can integrate the various operations proposed in the literature within the same system.

In this chapter, we give a whollistic vision of text summarization by proposing a framework that tries to cover all the possible basic operations of text summarization that have been developed in the literature. Our approach allows the integration of

these operations within the same framework. It sets a strategy to select the most suitable operation for a given portion of text. It is also adaptable to the different operations. For instance, we find in the literature operations that have mainly relied on templates, others that select the best sentences, etc.

We have used different mathematical and computer science theories to develop our framework. First, we use *number theory* and specifically *Bell numbers theory* (Bell, 1934) to generate what we will define later in this chapter as *partitions of the document*. We also use *graph theory* for the *multigraph* that we define later. A machine learning approach has also been used in this work to assign the suitable operations to the different portions of the source document.

The remainder of this chapter is organized as follows. Section 6.2 presents the problem statement and introduces various definitions. Section 6.3 explains the system design. The experimentation work is presented in Section 6.4. Section 6.5 discusses the extension of the framework. Section 6.6 discusses the results and Section 6.7 gives a conclusion and some possible directions for the development of Arabic text summarization based on this work.

6.2 Problem Statement

Having analyzed the available literature, we have come to the conclusion that there is a need for a full summarization framework. In fact, many summarization operations[1] have been developed, but we do not have a system that can integrate all of them in a compositional manner. In this work, we try to respond to the following research questions:

- **Q1:** What representation model should we consider to represent the document to be summarized?
- **Q2:** How should we assign summarization operations to the different portions of the source document?
- **Q3:** After applying a sequence of such operations, what is the stopping condition in the summarization process?

  When humans summarize documents, they tend to do it in an iterative process. After applying the operations to the source document, one ends up with the

[1] In the remainder of this chapter, whenever we use the term "operation(s)" it will obviously stand for "summarization operation(s)"

first version of a summary. At this moment, one (or the system) should decide whether it should summarize the summary again or stop the process. The most common way that has widely been used in text summarization was to set a retention rate upon which a decision is made whether to stop summarizing or not. In our case, we should come up with a more convincing approach to stop summarizing because the retention rate is not fixed for all the texts; it rather varies according to the type and the content of the text.

In the sequel, we give definitions and examples of the different concepts that are used in the proposed framework.

6.2.1 Definitions

We start from the intuitive idea here that before summarizing a document, we should find the different combinations (partitions) of sentences that we could generate from the source document. This will later allow the association of operations to the different partitions.

Definition 10. *Partition of a document*
Given a document D, a Partition P of D is a set of non-empty subsets of sentences $SP_i \subseteq D$, $1 \leqslant i \leqslant k$ (the (subsets) of the partition), such that $\mathrm{U}_{\mathrm{i}=1}^{\mathrm{k}}\ SP_i = D$ and for every $i <> j$, $SP_i \cap SP_j = \emptyset$. In other words, a partition is a set of non-empty subsets of sentences of D whose union is equal to D.

The notion of *subset of sentences of a partition of a document* is introduced because the summarization operations will be associated with these subsets of sentences.

Definition 11. *Subset of sentences of a partition of a document*
Given a partition $P = \{sp_1, sp_2, ..., sp_n\}$ of a document D. An element sp_i of P is a subset of sentences of the partition P of the document D.

A document could have many possible partitions. Below we define the notion of *set of partitions of a document*. Understanding the possible partitions is an important step as it allows us to understand the various possible summaries and hence the operations that need to be applied to the *subsets of sentences of the partition* to produce the summaries. As we will explain later, we select the best partition from the set of partitions of a document and also the best operations to be applied to this partition.

Definition 12. *Set of partitions of a document* (SPD)
Given a document D. The set of partitions of a document D is the set SPD of all

possible partitions that could be derived from D. $SPD = \{P$, where P is a partition of $D\}$.

It is known in combinatorics that the number of all partitions of a set of n elements is given by the *Bell number* B_n (Bell, 1934). As such, the number of partitions of a document D with n sentences is *Bell number* B_n, where $B_0 = 1$, $B_1 = 1$, $B_2 = 2$ and $B_3 = 5$. The Bell numbers satisfy the following relation:

$$B_{n+1} = \sum_{k=0}^{n} \binom{n}{k} B_k \tag{6.1}$$

Example 8. Let us consider a document D which has three sentences: a, b, c. there are five possible partitions of D:

1. $\{\{a\},\{b\},\{c\}\}$
2. $\{\{a,b\},\{c\}\}$
3. $\{\{a,c\},\{b\}\}$
4. $\{\{b,c\},\{a\}\}$
5. $\{\{a,b,c\}\}$

For example, the first partition contains three subsets of sentences each one having one single sentence. The second, third, and fourth partitions have two subsets of sentences, where the first subset has two sentences and the second has one sentence. The fifth partition has one subset of sentences which contains all three sentences.

Several taxonomies have been proposed in the literature (Jones et al., 1999; Mani and Maybury, 1999). These works classified text summarization systems according to a number of factors: input, output, language, etc. But none of these taxonomies mentioned the nature of the operations that a summarization system performs. In this work, we propose a classification of text summarization operations with respect to the nature of the operation that is performed in the summarization process. We distinguish two classes of operations: *single-sentence operations* and *multi-sentence operations*.

We define below *single-sentence operation, multi-sentence operation, set of single-sentence operations* and the *set of multi-sentence operations*. These concepts are related to the nature of the operations that are used in the summarization process. For example, *sentence compression* is considered as a *single-sentence operation* since it acts on only one sentence at a time, while *sentence fusion* is a *multi-sentence operation* since it merges sentences and needs at least two sentences for its application.

Definition 13. *Single-sentence operation* ($\odot$)
Given a subset of sentences SP of a partition P of a document D, a *single-sentence operation*, denoted by $\odot(SP)$, is an operation that is applied to a singleton subset of sentences of a partition of D, i.e. one that has **only one sentence**.

Definition 14. *Set of single-sentence operations* (SM)
The set of single-sentence operations of a partition P of a document D is denoted by $SG = \{\odot_1, \odot_2, ..., \odot_n\}$.

Definition 15. *Multi-sentence operation* ($\oplus$)
Given a subset of sentences SP of a partition P of a document D. A multi-sentence operation denoted by $\oplus(SP)$ is an operation that is applied to a subset of sentences of a partition that has **two or more sentences** .

Definition 16. *Set of multi-sentence operations* (SG)
The set of multi-sentence operations of a partition P of a document D is denoted by $SM = \{\oplus_1, \oplus_2, ..., \oplus_n\}$.

The process of mapping operations to the different subsets of sentences which make the partition of the document is used to get a summary document at a given iteration.

Definition 17. *Mapping (Θ) of operations to a partition of a document*
Let $P = \{sp_1, sp_2, ..., sp_n\}$ be a partition of a document D and let $\odot_i \in SG$ and $\oplus_j \in SM$. The process of mapping operations to P is the association of operations to all the subsets of sentences of the partition P. This is denoted by $\Theta(P) = \{\Phi(sp_1), \Phi(sp_2), ..., \Phi(sp_n)\}$. Where $\Phi(sp)$ is defined as follows:

$$\Phi(sp) = \begin{cases} \odot_i(sp), & \text{if } |sp| = 1 \\ \oplus_j(sp), & \text{otherwise} \end{cases} \tag{6.2}$$

If the number of sentences of the chosen subset of sentences of the partition is equal to one, then we should select an operation that belongs to the set of *single-sentence operations*. Otherwise, we select an operation that belongs to the set of *multi-sentence operations*.

We note that $\Theta(P)$ represents at the end a summary document since it performs two things: (1) it associates the operations to the subsets of sentences of the partition P and (2) generates a summary document which is the result of the application of theses operations to the subsets of sentences.

Below we define the *space of mappings of operations to a partition of a document*. For a given partition, we could generate many possible mappings. If we consider a

partition $P = \{sp_1, sp_2, ..., sp_n\}$ of a document D, a *set of single-sentence operations* $SG = \{\odot_1, \odot_2, ..., \odot_{n_1}\}$, a *set of multi-sentence operations* $SM = \{\oplus_1, \oplus_2, ..., \oplus_{n_2}\}$, $m = n_1 + n_2$ the total number of operations, and n the total number of *subsets of sentences of the partition* P, the worst-case complexity of generating this space is $O(m^n)$.

Definition 18. *Space of mappings of operations to a partition of a document*
Let P be a partition of a document D. The space of mappings of operations to the partition P of D, denoted by $SMOP(P) = \{\Theta_1(P), \Theta_2(P), ..., \Theta_{nb}(P)\}$, is the set of all the possible combinations of the operations that could be assigned to P.

Definition 19. *Best mapping ($\bar{\Theta}$) of operations to a partition of a document*
Let $SMOP(P) = \{\Theta_1(P), \Theta_2(P), ..., \Theta_n(P)\}$ be the *space of mappings of operations to a partition* P of a document D. The best mapping of operations to a partition of a document D, denoted by $(\bar{\Theta})$, is defined as the mapping of operations that yields a summary document which maximizes a fitness function that represents the quality of the document.

$$\bar{\Theta}(P) = \arg\max_{\Theta} f(SMOP(P)) \tag{6.3}$$

We have chosen to represent a document using a *multigraph*, i.e. consisting of multiple layers each of which represents one relation between the sentences. The representation of a document as a *multigraph* will set the ground for the application of the different operations of text summarization. A more detailed discussion about this *multigraph* is provided in the System Design section 6.3.

Definition 20. *Multigraph of a document*
The *multigraph* $G = (V, E)$ over a document D represents the different relations between the sentences of this document, where:

- V is a set of vertices or nodes (each node corresponding to a sentence in the document D)
- E is a multiset of edges (an edge representing a relation between two sentences)

This *multigraph* has several layers, each one representing a relation between sentences.

The summarization task is an iterative process such that, after mapping the operations to the partition, we generate a summary for the current iteration which represents a new summary.

Definition 21. *Summarization function*
Given a document D, the summarization function $S(D)$ represents the sequence of transformations of the document D. A function Γ_i of the document D is the result of applying the best mapping of operations of all the partitions of document D at iteration i. $P_1, P_2, ..., P_n$ are the partitions of the document D.

$$\Gamma_i(D) = \arg\max_{\Theta}(f(U_{\mathrm{k=1}}^{\mathrm{n}} SMOP(P_k))) \tag{6.4}$$

$\Gamma_i(D)$ takes the best mapping of operations of all the partitions of the document D and generates the current summary of iteration i. The summarization function $S(D)$ is the composition of the results of all the iterations, which is denoted by:

$$S(D) = \Gamma_n \circ \Gamma_{n-1} \circ ... \Gamma_2 \circ \Gamma_1(D) \tag{6.5}$$

i.e.:

$$S(D) = \Gamma_n(\Gamma_{n-1}(...(\Gamma_2(\Gamma_1(D))))) \tag{6.6}$$

We note that initially the best summary is the first generated summary. After the generation of a summary at every iteration $i > 1$, its quality is assessed and, if it outperforms the quality of the best generated summary of the previous iterations, then the current summary will be the best summary. At each iteration $i > 1$ the document that is given as an argument to the function Γ_i is the best summary.

6.3 System Design

6.3.1 General idea

In our approach, we consider text summarization in a graph-theoretic setting. A document is decomposed into a set of subsets of sentences where the number of subsets of sentences is given by a *Bell number* (Bell, 1934) as explained above. We also use a *multigraph* representation to represent the source document.

One novel and important aspect of this work is the use of a *multigraph*. The graph nodes represent the different sentences, and the edges between sentences represent the different relations that hold between them. Each layer of the graph represents a semantic relation between the sentences of the document. These relations could include, for instance, the semantic relations between sentences as well as the rhetorical

relations that might exist between them. A more detailed discussion about this issue will be given in the sequel. This representation will also prepare the ground for the operations that will be applied to generate the summary.

The mapping of operations to a partition uses a machine learning approach. To select the operation that will be applied to each subset of sentences of a partition, a probability is associated with each operation and the operation that has the highest probability is selected. This probability is estimated by a machine learning algorithm that has already been trained on an annotated corpus. The annotated corpus would contain the subset of sentences and the operation that should be applied on this subset of sentences. The machine learning system is trained to learn the operations that the system should associate with a partition.

6.3.2 Why a multi-layer graph?

In the text summarization literature, several representation models have been adopted to represent the text document (graphs, trees, term vectors, lexical representation, etc.). To justify the choice that we will consider in this approach, we start by providing the representation models that have been used in the literature. Then, we present the representation model that has been chosen in this work.

6.3.2.1 Rhetorical structure theory

The work presented in (Mann and Thompson, 1988) has gained a lot of interest for text summarization. The approach therein analyzes the input text and provides a tree representation that represents the text rhetorical relations. This representation has been used in several studies (Gonçalves et al., 2008; Marcu, 1997, 1998, 1999, 2000).

6.3.2.2 Lexical chains

Another approach is to use lexical or coreference chains. This approach has first been used in (Baldwin and Morton, 1998). The idea of this approach is to select the longest coreference chain. The assumption here is that the longest coreference chain contains the main topic of the document while the shorter chains indicate subtopics. To generate coherent summaries, this approach selects the longest coreference chain. Lexical-chains-based approaches aim to determine sequences of words which are semantically related to each other (for example, synonyms). It is assumed that the main topic could be detected just by selecting the longest chain. This technique has been used in a number

of studies (Barzilay and Elhadad, 1999; Barzilay et al., 1999; Ercan and Cicekli, 2008; Medelyan, 2007).

6.3.2.3 Graph-based representation

Graph-based summarizers have been proposed in the literature. In this approach, a node represents a sentence and an edge represents the relation between two sentences. These relations include semantic relations, semantic distances between sentences, etc. The assumption here is that the topology of the graph could provide interesting insights into the most important content of the text. Among the most relevant approaches in this category, we mention (Erkan and Radev (2004), Mihalcea (2004), Giannakopoulos et al. (2008), Qazvinian and Radev (2008), Mihalcea and Tarau (2004)).

The use of a *multigraph* aims to build a semantic representation of the document. This *multigraph* will prepare the ground for the operations that the mapping process will select. After an iterative process, we generate a summary at each iteration. If the quality of the newly generated summary at the current iteration is better than the best summary thus far, then the newly generated summary will be the best summary.

The choice of a multi-layer graph instead of a simple graph is motivated by the fact that many operations in the literature have relied on several relations (a simple graph cannot represent all the relations). For instance, we find in the literature operations that have mainly relied on the centrality of a sentence (Erkan and Radev, 2004; Patil and Brazdil, 2007) which is expressed by the number of connections between a sentence and the other sentences of the document. To represent this aspect, we need to define a relation which expresses the distance between sentences (aggregate similarity is an example of these relations). Another relation that might be considered as a representation layer in the *multigraph* is the RST relation (Mann and Thompson, 1988) which shows the rhetorical structure of a document. This kind of relations has been useful in various summarization systems. Moreover, RST relations could be used by a number of operations. It could help both the *single-sentence* and *multi-sentence operations*. Other relations that could be represented by the *multigraph* are lexical chains (Baldwin and Morton, 1998), latent semantic analysis (Yeh et al., 2005), and coreference information (Azzam et al., 1999). We also mention that several studies have reported that using a graph representation is very useful in text summarization (Erkan and Radev, 2004; Giannakopoulos et al., 2008; Mihalcea, 2004; Wan and Xiao, 2007) since it allows to generate coherent documents.

6.3.3 The Process of Summary Generation

Algorithm 9 gives the steps of the general algorithm for the generation of a summary. It takes as input a source document and the maximum number of iterations and outputs the summary of this document. The algorithm follows an iterative process. First, it generates the partitions of the document and the *multigraph*. For each partition of the document, it builds the space of mappings of operations of the partition denoted by $SMOP(P)$ and selects the best mapping of operations of the partition $\bar{\Theta}(P)$. If the summary of the mapping $\bar{\Theta}(P)$ is better than the best summary that far according to the fitness function being used, then the best summary will be the summary of $\bar{\Theta}(P)$. At the end, Algorithm 9 outputs the best summary. The whole process implements the summarization function $S(D)$ and each iteration i of the process is the function $\Gamma_i(D)$.

Algorithm 9 General algorithm for summary generation

```
1: Input Source document D, maxIter
2: Output bestSummary
3: i =1;
4: bestSummary = D;
5: while i <= maxIter do
6:    SPD = Generate the partitions of the bestSummary;
7:    multiGraph = Construct the multigraph;
8:    for all P in SPD  do
9:       SMOP(P) = Generate the space of mappings of operations of P
10:      Θ̄(P) = Select the best mapping of operations from SMOP(P)
11:      summary = Generate the summary from the best mapping Θ̄(P) of operations
         of P;
12:      if fitness(summary) > fitness(bestSummary) then
13:         bestSummary = summary;
14:      end if
15:   end for
16:   i = i + 1
17: end while
18: Return bestSummary;
```

6.3.3.1 Selection of the Best Operation by Machine Learning

To select the best operation to be applied to generate the summary, Algorithm 10 takes as input a subset of sentences. The algorithm uses features like the position of the subset of sentences, the number of keywords of the subset of sentences, the number

of words shared with the title, whether the subset of sentences is the first or the last subset of sentences and the number of words in the subset of sentences. The algorithm returns the operation that has the highest probability.

Algorithm 10 Algorithm for the selection of best operation using machine learning

```
1: Input A subset of sentences sp
2: Output best_operation
3: F= extract_features (sp);
4: best_operation = Machine_learning_prediction(F);
5: Return best_operation;
```

Once we generate all the partitions of a document, we calculate the quality of each partition which is expressed by a fitness value. The fitness measures the quality of the summary document which would be the result of applying the mapping of operations to obtain the considered partition.

Algorithm 11 Algorithm for the construction of the *multigraph*

```
1: Input listNodes is a list of n nodes which represent the sentences of the source
   document
2: Output The multigraph represented by nb matrices
3: for k := 1 to nb do
4:    for  i:= 1 to n  do
5:       for  j := i+1 to n do
6:          M_k[i,j] = Relation_k(listNodes[i],listNodes[j]);
7:       end for
8:    end for
9: end for
10: Return M_1,M_2,...,M_nb matrices ;
```

To generate the *multigraph*, Algorithm 11 captures the relations that exist between sentences. Let us consider an instance of the general problem and let us suppose that we are interested in tackling the problem taking into account only two relations, the similarity between sentences and the semantic relations that could exist between sentences represented by the RST relations. In this case, we have only two matrices A and B. Matrix A stores the similarity distances between sentences, while Matrix B stores the RST relations that might exist between the different sentences. These two matrices will be needed for the operations that will be applied to generate the summary.

6.3.3.2 Defining the fitness function

In order to evaluate the quality of the summary (S), we define a fitness function (f) which combines the grammaticality and the concentration of information to evaluate the importance of a summary. The concentration of information is a semantic measure of the quality of the summary as will be explained below.

$$fitness(S) = information\_concentration(S) + grammaticality(S) \tag{6.7}$$

Information concentration The information concentration is measured by the amount of important content present in the summary. We define two parameters: the aggregate similarity and the *tf*. The aggregate similarity gives high value to summaries with central sentences in the text. While the *tf* allows to give high ranks to documents that have important words.

$$information\_concentration(S) = \alpha * (Aggregate\_similarity(S)) + \beta * (TF(S)) \tag{6.8}$$

Aggregate similarity Let $S = \{s1, s2, s3, s4, s5, s6, s7\}$ be a summary document. The aggregate similarity technique allows to assign weights to the different sentences of the document to assess its quality. The weight of any given sentence is calculated as the sum of the weights of its connections. For example, the weight of Sentence s_6 is equal to $0.65+0.84+0.12$, while the average weight of the sentence s_6 is $(0.65+0.84+0.12)/3$

In our case, the aggregate similarity of a summary of a document is calculated by summing up the average weights of all the sentences of the summary and dividing this sum by the number of sentences of the document.

We note that the values of the TF, aggregate similarity and the grammaticality are between 0 and 1.

where α, β and γ are set empirically.

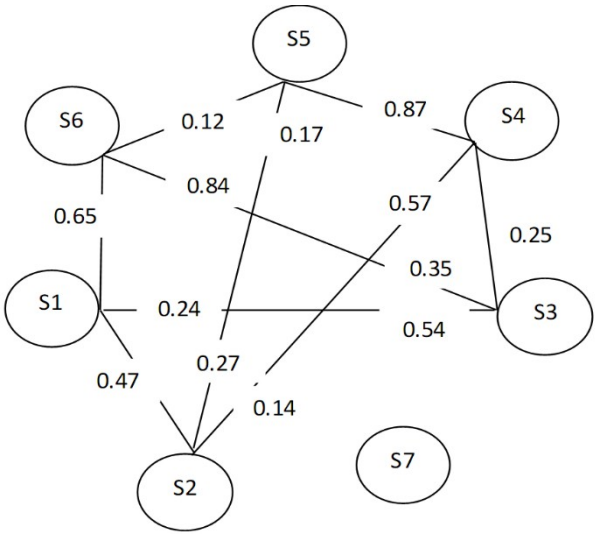

Fig. 6.1 Similarity graph of a document

TF

$$TF(term, D) = \frac{freq(term, D)}{freq(term, D) + 0.5 + 1.5 * \frac{lengthdoc}{avrlengthdoc}} \tag{6.9}$$

where:
D: A summary document
term: A term in the summary document D
TF: term frequency
IDF: Inverse Document Frequency
freq: the frequency of a term in the document
lengthdoc: the document length
avrlengthdoc: the average length of training documents
The *tf* value is calculated for each term of the document; the global value of the *tf* of the document is the sum of all the *tf* values of the terms divided by the total number of the terms of the document.

Grammaticality The grammaticality of the summary should also be maximized. For this reason, we propose to use a trigram language model. Collins (2013) stated that the trigram assumption is arguably quite strong, and linguistically naive. However, it leads to models that are very useful in practice. The grammaticality of the document is represented as the sum of all trigrams of the document over the total number of trigrams of this document. The value of the grammaticality ranges from 0 to 1.

$$Grammaticality(S) = \gamma * Trigrams(S) \tag{6.10}$$

6.4 Evaluation of the Global Summarization System

6.4.1 Evaluation of the Algorithm for Affectation of Operations to Document Partitions

6.4.1.1 Operations Used in the Evaluation

We have used three operations in the experimentation. All the operations used belong to the *single-sentence operation* category.

- **Sentence extraction using AdaBoost (Belkebir and Guessoum, 2015a)**

  This operation is based on the AdaBoost Machine Learning-based technique to generate Arabic summaries. This technique is used to predict whether a new sentence is likely to be included in the summary or not.

- **Concept generalization and fusion for abstractive sentence generation**

  This operation addresses the problem of abstractive text summarization by fusing and generalizing concepts. It tackles the problem of generalization of sentences, i.e. such that the system will be able to generate from a sentence like *"Selma ate bananas, apples and potatoes"* an output like *"Selma ate fruits and vegetables"* or *"Selma ate some food"*. This task requiring the use of world knowledge, the operation uses WordNet to generalize the concepts. This operation has been basically proposed for the English language in (Belkebir and Guessoum, 2016) and we have ported it to Arabic for the purposes of this work.

- **Sentence compression (Belkebir and Guessoum, 2015b)**

  This operation tries to generate a single sentence that preserves the most salient information from the original sentence while ensuring that it is grammatically well-formed. We have modeled sentence compression as an optimization problem using differential evolution. This method allows to search the space of compressions in a polynomial time.

6.4.1.2 Discussion of the learning process and the corpus annotation

The approach adopted in this work requires the use of a mapping process which is responsible for assigning operations to the different portions of the text (subsets of sentences of a partition). After segmenting the source document into sentences and generating all the possible partitions, we need to select the best partition, hence the best mapping of operations for this partition of the source document. At this stage, a prediction function that assigns a certain probability that represents the suitability of the operation to each subset of sentences of the partition is needed.

To assign a probability to any operation, a machine learning system is trained to learn the mapping between the subsets of sentences of the partition and the operations. This mapping function takes as input the different features of the problem and outputs an estimated value that gives hints about the suitability of this operation to this portion of text.

To train the system we have developed a parallel corpus of pairs <subsets of sentences, operation>. Figure 6.2 shows an example from the TALAA-ASC corpus (Belkebir and Guessoum, 2015b). The document is represented using the XML format. It contains packets each of which represents a subset of sentences of the document. Inside each packet tag, we have defined the "id" and the "operation" attribute. For the operation attribute, we have used the following codes:

- S: Select a sentence
- R: Remove a sentence
- C: Compress a sentence
- G: Generalize a sentence
- F: fuse sentences

6.4.1.3 Comparison

In order to evaluate the performance of the system in associating the operations, we have used the annotated TALAA-ASC corpus. We have used 10-fold cross validation. Table 6.1 presents the results of the recall, precision and F1-score.

We have used WEKA (Hall et al., 2009) to asses the performance of the different machine learning algorithms. We have tested the system with the following machine learning techniques: J48, SVM, Logistic regression and Naive Bayes; and have used

```
<?xml version="1.0" encoding="UTF-8"?>
<file>
<packet  id="1" operation="S">
<source>حث وزير الخزانة الامريكي جاك لو دول منطقة اليورو على "تعزيز الطلب" على منتوجاتها من
أجل تخفيف نسبة البطالة وتجنب الدخول في مرحلة الانكماش الاقتصادي. </source>
</packet>
<packet  id="2" operation="S">
<source>وقال لو خلال اجتماع الدول العشرين التي تحتفن العديد من كبار الدول الاقتصادية
إن " أوروبا بحاجة الى حل مشاكلها والقضاء على اختلافاتها داخل منطقة اليورو"، مضيفاً
" ما يبدو واضحاً من التجربة الامريكية هو ضرورة الجمع بين إتخاذ خطوات لدفع عجلة الاقتصاد على المدى القصير
وإجراء تغييرات هيكلية على المدى الطويل". </source>
</packet>
<packet  id="3" operation="S">
<source>وأشار "على الاوروبيين العمل على تطبيق هاتين الخطوتين". </source>
</packet>
<packet  id="4" operation="R">
<source>وعبر وزير الخزانة الامريكي عن "قلقه من التوتر القائم بين الدول الاوروبية،
منبهاً إلى مضار الدفع بسياسات عاجلة". </source>
</packet>
<packet  id="5" operation="R">
<source>وأشار إلى أن ما يقلقه هو "تأجيل الجهود لدفع عجلة الاقتصاد مما سينعكس سلباً
على حاجات بلدان منطقة اليورو". </source>
</packet>
<packet  id="6" operation="F6_7">
<source>وكان البنك المركزي الاوروبي قد فرض تدابير جديدة لإنعاش الاقتصاد في منطقة اليورو.</source>
<fusion>و بسبب الضغوط لدفع عجلة الاقتصاد في منطقة اليورو فرض البنك المركزي الاوروبي تدابير
جديدة لإنعاش الاقتصاد في منطقة اليورو</fusion>
</packet>
<packet  id="7" operation="F6_7">
<source>ويواجه البنك الكثر من الضغوط لدفع عجلة الاقتصاد في منطقة اليورو جراء التباطؤ الذي
يشهده قطاع الانتاج والتضخم الذي هبط فقط 3 في المئة. </source>
<fusion>و بسبب الضغوط لدفع عجلة الاقتصاد في منطقة اليورو فرض البنك المركزي الاوروبي تدابير
جديدة لإنعاش الاقتصاد في منطقة اليورو</fusion>
</packet>
</file>
```

Fig. 6.2 Example of a document from the annotated TALAA-ASC corpus

the default values for their parameters. These techniques have been trained on the TALAA-ASC (Belkebir and Guessoum, 2015b) corpus which we have later enriched and annotated with the following operations: *sentence selection*, *sentence compression*, *concept generalization and fusion* and *sentence fusion*. We note that the sentence fusion operation which merges sentences is not used when training the machine learning system that selects the operations to be applied, this is one of our future work directions.

Table 6.1 Evaluation of the Algorithm for Affectation of Operations

Evaluation	j48	SVM	Logistic	MLP	Nb	Random forest
Weighted Precision	0.639	0.623	0.668	0.638	0.715	0.597
Weighted Recall	0.645	0.618	0.654	0.633	0.630	0.610
Weighted F-Measure	0.623	0.587	0.623	0.607	0.585	0.598

Table 6.1 shows that the machine learning techniques that have given the best results are the J48 and Logistic regression with F-Measure equal to 0.623. SVM comes next with an F-Measure of 0.587 followed by Naive Bayes with an F-Measure of 0.585.

6.4.2 Evaluation of the Summarization Framework

ROUGE-N is an n-gram recall measure that compares between a summary and a set of reference summaries. It is computed as follows:

$$ROUGE-N = \frac{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count(gram_n)} \tag{6.11}$$

where n is the length of the n-gram. $Count_{match}(gram_n)$ is the maximum number of n-grams co-occurring in a set of reference summaries and a summary.

In order to evaluate the framework, we have used a number of machine learning algorithms and the Rouge package (Lin, 2004) to evaluate the system.

Table 6.2 Evaluation of the Summarization Framework

Evaluation		AdaBoost	j48	SVM	Logistic	ML Op	Nb
Rouge 1	R	0.43308	0.13690	0.29116	0.14934	**0.74566**	0.15340
	P	**0.53312**	0.32110	0.41174	0.36549	0.48694	0.32575
	F	0.44071	0.17442	0.31470	0.19166	**0.55995**	0.18813
Rouge 2	R	0.15690	0.00935	0.10429	0.01641	**0.37068**	0.02339
	P	0.18658	0.04216	0.12997	0.05356	**0.28130**	0.07482
	F	0.15771	0.01503	0.10458	0.02471	**0.31244**	0.03400
Rouge SU4	R	0.17887	0.00631	0.12812	0.01172	**0.44232**	0.02927
	P	0.22046	0.04911	0.17082	0.05981	**0.28683**	0.08999
	F	0.16966	0.01092	0.12331	0.01912	**0.33117**	0.03511

We have compared the performance of the proposed system against the performance of a number of machine learning systems (See Table 6.2) . All of these existing systems are extractive and try to select the most relevant sentences from a document. These systems have been trained on an external corpus of Belkebir and Guessoum (2015a) which contains 30 documents and then tested on the enriched TALAA-ASC corpus (Belkebir and Guessoum, 2015b).

We have compared the performance of our Summarization framework against the AdaBoost system proposed in (Belkebir and Guessoum, 2015a), J48 (Quinlan, 1993), SVM (Vapnik, 1999), logistic (Le Cessie and Van Houwelingen, 1992) and Naive Bayes (John and Langley, 1995). The proposed system has a better performance compared to the other existing systems since it has the ability to select the suitable operation for a given portion of text. In our case, for the purposes of testing the proposed framework we have included only three operations which are *sentence selection/deletion*, *sentence compression* and *concept generalization and fusion*. We note that the performance of the system could be improved if we include other operations from the *multi-sentence operation* category. We are currently working on the implementation of some *multi-sentence operations* so we can include them into the framework and test it.

6.5 Extension of the framework: TALAA-ATSF with BSO-CHI2-SVM

As we mentioned in related work, text summarization has been successfully combined with different applications. An interesting future extension of the framework is to

categorize documents before summarization. In fact, if we know the category of the document it will be beneficial for some text summarization operations such as sentence compression. This is what we have discussed and concluded in chapter 5. We have concluded that the compression ratio varies between the different categories of the corpus and discussed that it will be interesting to design compression systems which takes into consideration the category of the document. The categorization operation could improve as well the performance of the sentence selection operation. In fact, if we know the category of the document, it will help choose the best strategy to select sentences and it will help to select the best keywords that the summarization system should rely on since the keywords are directly related to the category of the document. In the sequel, we present the general architecture of BSO-CHI2-SVM (Belkebir and Guessoum, 2013) .

The aim of BSO-CHI2-SVM is to assign a category to a document. The problem of selecting the set of attributes is NP-hard. On the other hand, meta-heuristics have given good results for several optimization problems. Among these meta-heuristics, we can mention Genetic Algorithms (GA), Particle Swarm Optimisation (PSO), Tabu Search (TS), etc. In the area of automatic classification, although the methods based on meta-heuristics are very powerful, little research has addressed the problem of feature selection using these approaches. Researchers avoid using them for reasons related to computation time which is extremely high when compared with methods based on filters (Chi-square, information gain, mutual information, etc.). Indeed, when dealing with this problem of feature selection using meta-heuristics, it is necessary to repeat the learning process after the generation of any solution and hence the learning time becomes very high. As a remedy, a meta-heuristic that exploits the parallelism on a large scale can be adopted, hence the choice of BSO. This is coupled with Chi-square (X^2) in a way to guide the search so as to avoid bad solutions. The set of features is fed into the SVM, which produces a learned model for the categorization problem using this subset of features.

We start by generating the vocabulary, i.e. all the corpus words without redundancy. A feature selection process using Chi2-BSO takes this vocabulary and produces a subset of features. The latter is fed into the SVM classifier which produces a model of the categorization system. This in turn gets evaluated and the result of this evaluation is affected to this model. The process is repeated iteratively a predefined number of times. Note that the call to the SVM module is done from within the general BSO master process.

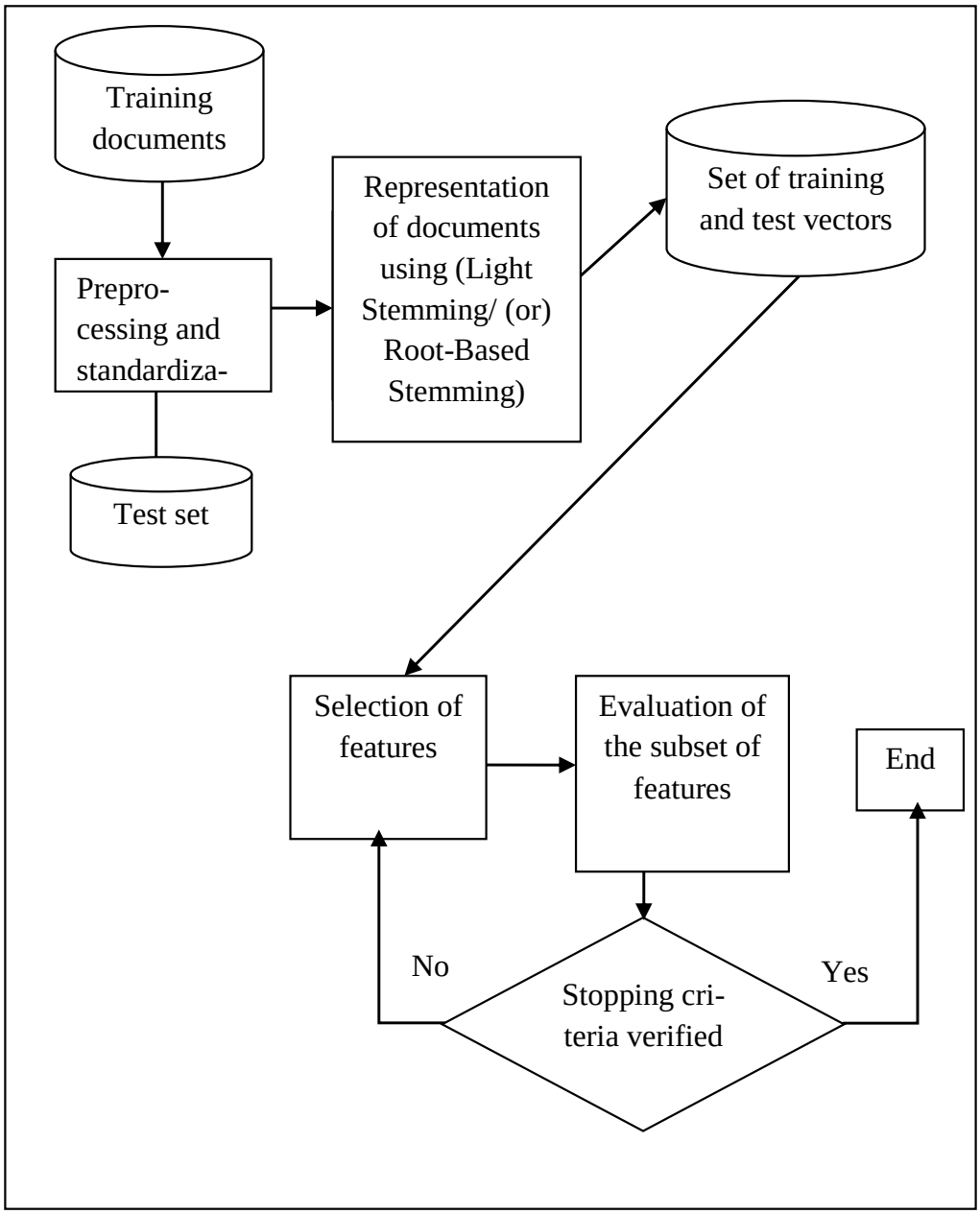

Fig. 6.3 Architecture of BSO-CHI2-SVM

As we discussed above, using BSO-CHI2-SVM as an initial step before categorization could improve the performance of the framework since it allows to choose the appropriate strategy of summarization according to the category of the document. An interesting future direction is to investigate the effect of using BSO-CHI2-SVM within the framework.

6.6 Discussion

We now discuss some of the most relevant points about the proposed approach.

The approach could automatically select the portions of the document (set of subsets of sentences) and the most suitable operations that we should apply to them. In our context, a document portion represents a set of subsets of sentences. We select an operation that belongs to the set of *single-sentence operations* if the set of subsets contains one sentence or, in case we are dealing with multiple sentences (the set of subsets contains more than one sentence), we should use an operation from the *set of multi-sentence operations*. In this case, we have to summarize several sentences using one operation.

The *multigraph* has been constructed to facilitate the task for the other operations that will be applied to generate the summary. For instance, the *single-sentence operation* of deleting sentences could benefit from the use of the *multigraph* and some relations that the *multigraph* conveys, like the semantic similarity between sentences. This information could be used as a feature for this operation (delete sentence). Among the other operations that could benefit from the *multigraph* are the fusion and generalization of concepts, especially, if concepts that we intend to generalize do not belong to the same sentence. This is being investigated in a separate research work. In this situation, the *multigraph* provides information about the RST relation, and here we use some rules that will give us the ability to decide whether two adjacent sentences could be fused or not.

An important issue that has to be emphasized is that the efficiency of the approach will mainly depend on the quality of the operations that make up the entire framework. To have a high-quality summarizer, all of the operations should be of good quality. The framework is only responsible for selecting the best operation(s) to apply to the best portion of text, and it is not responsible for the quality of the operations which are external to the framework.

We note that this framework is the first that handles the Arabic language using both abstractive and extractive operations. Combining the two operations has improved the

results reached by other existing machine learning systems which have been used in the literature to tackle Arabic text summarization.

6.7 Conclusion

In this work, we have presented an approach that can integrate several operations into one single framework which has benefitted from several aspects. It uses concepts inspired from number theory. In this sense, we have generated the partitions of a document where the number of partitions is the *Bell numbers* (Bell, 1934) . We also use probabilities to select the most likely partition and the set of operations that will be applied to the source text to produce the summary. Another interesting issue in this approach is the use of a *multigraph* which embodies several relations represented by different matrices. The *multigraph* could also be useful for the operations selected by the machine learning system.

This work could be extended in several directions. One possible improvement is to propose an active learning approach that will have the ability to adapt new operations in real time. Currently, if a new operation has to be integrated into our summarization framework, we must retrain the machine learning system to learn which operation to use among the available ones.

It will also be interesting to integrate in an enrichment of the implementation an operation that belongs to the set of *multi-sentence operations*. For the moment, though the theoretical framework includes two categories of operations, we have included in the implementation only operations that belong to the *set of single sentence operations*; we are currently working on the suggested expansion of the system.

We state that the approach presented in this chapter is very novel and gives different dimensions to research in text summarization. The main strength of this framework is its ability to seemlessly include any text summarization operation. In the next chapter we present a general conclusion of the thesis and discuss the future research directions.

Chapter 7

Conclusions

This chapter summarizes the main findings of this work and discusses further research directions.

7.1 Main Contributions

This thesis has addressed the task of text summarization. We have investigated the work done on text summarization. We have proposed a framework for text summarization. We have designed approaches for sentence compression, concept generalization and fusion as well as sentence selection. The following is a summary of the main findings and contributions of this thesis:

- We have conducted an analytical study of the work that has been already done in the literature for text summarization and proposed a taxonomy that classify text summarization work according to the operations used to generate the summary. Two classes of operations have been identified: *single-sentence* operation and *multi-sentence* operation. If the operation is a *single-sentence* operation, then the operation could be applied to a single sentence (example: sentence compression). For the *multi-sentence* operation, the operation is used to handle several sentences (example: sentence fusion).

- We have shown how text summarization can be viewed as a "factory" and proposed a global summarization framework that could integrate the different operations of text summarization within the same system. This framework allows to select the most suitable operation for each portion of the text to be summarized.

- We have designed an abstractive summarization operation. This operation use concept fusion and generalization techniques to generate abstractive sentences. It uses Wordnet as a semantic resource to generalize the concept.
- We have proposed an approach that uses AdaBoost as a machine learning technique to select the important sentences that are worthy to be included in the summary. It has achieved good results compared to other existing techniques.
- We have proposed TALAA-ASC which is the first sentence compression corpus for the Arabic language. This could be used both to generate new sentence compression systems that are trained on this corpus. It also allows to compare the performance of the sentence compression systems.

7.2 Further Research

In this thesis, we have investigated the work done on text summarization and proposed a global framework that deals with text summarization. We have proposed three summarization operations and discussed the limitations and the improvements that could be done for each operation. These could be summarized as follow.

Regarding the work that has been done on sentence seletion operation where we proposed an AdaBoost approach, one direction would be to extend the corpus by adding more documents. It will be also interesting to introduce more features like part-of-speech tagging and semantic relations between sentences.

A lot of improvement could be done for the concept generalization and fusion approach. One direction would be to introduce more features and thus try to give more meaning to the generalization versions. Another way to improve the system performance is to use a larger corpus to estimate the acceptability of the generalization versions, especially for the context-based approach. Another problem that has to be solved is the use of a richer external knowledge source with more domain-specific knowledge since WordNet seems to provide almost none. An interesting other future direction is to extend the idea of concept fusion and generalization to not only intra-sentence generalization but to inter-sentence generalization.

Regarding the text summarization framework. One possible improvement is to propose an active learning approach that will have the ability to adapt new operations in real time. Currently, if a new operation has to be integrated into our summarization framework, we must retrain the machine learning system to learn which operation to use among the available ones. It will also be interesting to integrate in an enrichment

of the implementation an operation that belongs to the set of *multi-sentence operation*. For the moment, though the theoretical framework includes the categories of operations, we have included in the implementation only operations that belong to the *set of single sentence operations*; we are currently working on the suggested expansion of the system. An interesting future direction is to use deep learning for the machine learning systems that have been developed in this thesis since this technique has achieved state-of-the-art results in many artificial intelligence problems. Furthermore, the generation of partitions has a high computational complexity and yields a big number of partitions when the number of sentences of a document is high. To alleviate this problem, a future research direction would be to use meta-heuristics as a mean to explore the search space intelligently. The use of more semantic relations to represent the textual information could be one of the research directions to improve the global summarization framework proposed in this thesis. Using ontologies could represent an interesting future solution since it allows to represent the documents semantically by providing a number of semantic relations.

It is interesting to investigate the task of summarization by taking into account the user background information. In this case, the summary should contain the important content with respect to the user.

Another research direction is to study the impact of using the Tashkyl in Arabic text summarization. It is also interesting to investigate the use of some advanced morphological tools to tackle some complex Arabic words.

One possible improvement will be to enrich the current TALAA-ASC corpus by adding more documents. We also intend to get help from the ANLP community in order to extend the corpus.

References

Abdel Fattah, M. and Ren, F. (2008). Probabilistic neural network based text summarization. In *International Conference of Natural Language Processing and Knowledge Engineering, 2008. NLP-KE'08*, pages 1–6. IEEE.

Angrosh, M., Nomoto, T., and Siddharthan, A. (2014). Lexico-syntactic text simplification and compression with typed dependencies. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING 2014)*.

Aone, C., Okurowski, M. E., and Gorlinsky, J. (1998). Trainable, scalable summarization using robust nlp and machine learning. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 62–66. Association for Computational Linguistics.

Azmi, A. and Al-thanyyan, S. (2009). Ikhtasir—a user selected compression ratio arabic text summarization system. In *International Conference on Natural Language Processing and Knowledge Engineering, 2009. NLP-KE 2009*, pages 1–7. IEEE.

Azzam, S., Humphreys, K., and Gaizauskas, R. (1999). Using coreference chains for text summarization. In *Proceedings of the Workshop on Coreference and its Applications*, pages 77–84. Association for Computational Linguistics.

Baldwin, B. and Morton, T. S. (1998). Dynamic coreference-based summarization. In *EMNLP*, pages 1–6.

Barzilay, R. and Elhadad, M. (1999). Using lexical chains for text summarization. *Advances in automatic text summarization*, pages 111–121.

Barzilay, R., Elhadad, M., and McKeown, K. (1999). Text summarizations with lexical chains. *Advances in automatic text summarization*, pages 111–121.

Barzilay, R. and McKeown, K. R. (2005). Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328.

Bauer, E. and Kohavi, R. (1999). An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine learning*, 36(1-2):105–139.

Belkebir, R. and Guessoum, A. (2013). A hybrid bso-chi2-svm approach to arabic text categorization. In *International Conference on Computer Systems and Applications (AICCSA), 2013 IEEE/ACS*, pages 1–7. IEEE.

Belkebir, R. and Guessoum, A. (2015a). A supervised approach to arabic text summarization using adaboost. In *New Contributions in Information Systems and Technologies*, pages 227–236. Springer.

Belkebir, R. and Guessoum, A. (2015b). Talaa-asc: A sentence compression corpus for arabic. In *12th International Conference of Computer Systems and Applications (AICCSA), 2015 IEEE/ACS*, pages 1–8. IEEE.

Belkebir, R. and Guessoum, A. (2016). Concept generalization and fusion for abstractive sentence generation. *Expert Systems with Applications*, 53:43–56.

Belkebir, R. and Guessoum, A. (2017). Talaa-atsf: A global operation-based arabic text summarization framework. In *Intelligent Systems Reference Library series*. Springer.

Bell, E. T. (1934). Exponential numbers. *The American Mathematical Monthly*, 41(7):411–419.

Bing, L., Li, P., Liao, Y., Lam, W., Guo, W., and Passonneau, R. (2015). Abstractive multi-document summarization via phrase selection and merging. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1587–1597, Beijing, China. Association for Computational Linguistics.

Boguraev, B. and Kennedy, C. (1999). Salience-based content characterisation of text documents. *Advances in automatic text summarization*, pages 99–110.

Bouckaert, R. R., Frank, E., Hall, M., Kirkby, R., Reutemann, P., Seewald, A., and Scuse, D. (2013). Weka manual for version 3-7-8.

Boudabous, M. M., Maaloul, M. H., and Belguith, L. H. (2010). Digital learning for summarizing arabic documents. In *International Conference on Natural Language Processing*, pages 79–84. Springer.

Boudin, F., Mougard, H., and Favre, B. (2015). Concept-based summarization using integer linear programming: From concept pruning to multiple optimal solutions. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1914–1918, Lisbon, Portugal. Association for Computational Linguistics.

Chali, Y. and Hasan, S. A. (2012). On the effectiveness of using sentence compression models for query-focused multi-document summarization. In *COLING'12*, pages 457–474.

Chali, Y. and Joty, S. R. (2008). Improving the performance of the random walk model for answering complex questions. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 9–12. Association for Computational Linguistics.

Chang, C.-C. and Lin, C.-J. (2011). Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27.

Christensen, H., Kolluru, B., Gotoh, Y., and Renals, S. (2004). From text summarisation to style-specific summarisation for broadcast news. In *European Conference on Information Retrieval*, pages 223–237. Springer.

Clarke, J. and Lapata, M. (2008). Global inference for sentence compression: An integer linear programming approach. *J. Artif. Intell. Res.(JAIR)*, 31:399–429.

Cohn, T. and Lapata, M. (2013). An abstractive approach to sentence compression. *ACM Trans. Intell. Syst. Technol.*, 4(3):41:1–41:35.

Collins, M. (2013). Course notes for nlp by michael collins, columbia university.

Conroy, J. M. and O'leary, D. P. (2001). Text summarization via hidden markov models. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 406–407. ACM.

Corston-Oliver, S. (2001). Text compaction for display on very small screens. In *Proceedings of the NAACL Workshop on Automatic Summarization*, pages 89–98. Association for Computational Linguistics.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297.

Coster, W. and Kauchak, D. (2011). Simple english wikipedia: A new text simplification task. In *ACL (Short Papers)*, pages 665–669.

Cunningham, L. A. (2006). Language, deals, and standards: The future of xml contracts. *Wash. UL Rev.*, 84:313.

Da Cunha, I., Fernández, S., Morales, P. V., Vivaldi, J., SanJuan, E., and Torres-Moreno, J. M. (2007). A new hybrid summarizer based on vector space model, statistical physics and linguistics. In *Mexican International Conference on Artificial Intelligence*, pages 872–882. Springer.

De Marneffe, M.-C. and Manning, C. D. (2008). Stanford typed dependencies manual. *URL http://nlp. stanford. edu/software/dependencies manual. pdf.*

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391.

Douzidia, F. S. and Lapalme, G. (2004). Lakhas, an arabic summarization system. *Proceedings of DUC2004.*

Duboue, P. A., McKeown, K. R., and Hatzivassiloglou, V. (2003). Progenie: Biographical descriptions for intelligence analysis. In *International Conference on Intelligence and Security Informatics*, pages 343–345. Springer.

Dunlavy, D. M., O'Leary, D. P., Conroy, J. M., and Schlesinger, J. D. (2007). Qcs: A system for querying, clustering and summarizing documents. *Information processing & management*, 43(6):1588–1605.

Edmundson, H. P. (1969). New methods in automatic extracting. *J. ACM*, 16(2):264–285.

El-Fishawy, N., Hamouda, A., Attiya, G. M., and Atef, M. (2014). Arabic summarization in twitter social network. *Ain Shams Engineering Journal*, 5(2):411–420.

El-Haj, M., Kruschwitz, U., and Fox, C. (2011a). Exploring clustering for multi-document arabic summarisation. In *Asia Information Retrieval Symposium*, pages 550–561. Springer.

El-Haj, M., Kruschwitz, U., and Fox, C. (2011b). Multi-document arabic text summarisation. In *Computer Science and Electronic Engineering Conference (CEEC), 2011 3rd*, pages 40–44. IEEE.

El-Haj, M. and Rayson, P. (2013). Using a keyness metric for single and multi document summarisation. Association for Computational Linguistics.

El-Haj, M. O. and Hammo, B. H. (2008). Evaluation of query-based arabic text summarization system. In *Natural Language Processing and Knowledge Engineering, 2008. NLP-KE'08. International Conference on*, pages 1–7. IEEE.

Elsner, M. and Charniak, E. (2008). Coreference-inspired coherence modeling. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 41–44. Association for Computational Linguistics.

Endres-Niggemeyer, B. and Neugebauer, E. (1998). Professional summarizing: no cognitive simulation without observation. *Journal of the American Society for Information Science*, 49(6):486–506.

Ercan, G. and Cicekli, I. (2008). Lexical cohesion based topic modeling for summarization. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 582–592. Springer.

Erkan, G. and Radev, D. R. (2004). Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479.

Fan, J., Gao, Y., Luo, H., Keim, D. A., and Li, Z. (2008). A novel approach to enable semantic and visual image summarization for exploratory image search. In *Proceedings of the 1st ACM international conference on Multimedia information retrieval*, pages 358–365. ACM.

Feblowitz, D. and Kauchak, D. (2013). Sentence simplification as tree transduction. In *Proceedings of the Second Workshop on Predicting and Improving Text Readability for Target Reader Populations*, pages 1–10, Sofia, Bulgaria. Association for Computational Linguistics.

Ferreira, R., Lins, R. D., Freitas, F., Cavalcanti, G. D., Lima, R., Simske, S. J., Favaro, L., et al. (2013). Assessing sentence scoring techniques for extractive text summarization. *Expert Systems with Applications*, 40(14):5755–5764.

Filatova, E. and Hatzivassiloglou, V. (2004). Event-based extractive summarization. In Marie-Francine Moens, S. S., editor, *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 104–111, Barcelona, Spain. Association for Computational Linguistics.

Floridi, L. (2008). The method of levels of abstraction. *Minds and machines*, 18(3):303–329.

Floridi, L. (2009). *Philosophical conceptions of information*. Springer.

Floridi, L. (2011). *The philosophy of information*. Oxford University Press.

Floridi, L. (2013). *The ethics of information*. Oxford University Press.

Frank, E., Wang, Y., Inglis, S., Holmes, G., and Witten, I. H. (1998). Using model trees for classification. *Machine Learning*, 32(1):63–76.

Freund, Y., Schapire, R. E., et al. (1996). Experiments with a new boosting algorithm. In *ICML*, volume 96, pages 148–156.

Fuentes, M., Alfonseca, E., and Rodríguez, H. (2007). Support vector machines for query-focused summarization trained and evaluated on pyramid data. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 57–60. Association for Computational Linguistics.

Ganascia, J.-G. (2015). Abstraction of levels of abstraction. *Journal of Experimental & Theoretical Artificial Intelligence*, 27(1):23–35.

Genest, P.-E. and Lapalme, G. (2011). Framework for abstractive summarization using text-to-text generation. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 64–73. Association for Computational Linguistics.

Giannakopoulos, G., Karkaletsis, V., and Vouros, G. (2008). Testing the use of n-gram graphs in summarization sub-tasks. In *Proceedings of the text analysis conference (TAC)*.

Gonçalves, P. N., Rino, L., and Vieira, R. (2008). Summarizing and referring: towards cohesive extracts. In *Proceedings of the eighth ACM symposium on Document engineering*, pages 253–256. ACM.

Gong, Y. and Liu, X. (2001). Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 19–25. ACM.

Gotti, F., Lapalme, G., Nerima, L., Wehrli, E., and du Langage, T. (2007). Gofaisum: a symbolic summarizer for duc. In *Proc. of DUC*, volume 7.

Haboush, A., Al-Zoubi, M., Momani, A., and Tarazi, M. (2012). Arabic text summarization model using clustering techniques. *World of Computer Science and Information Technology Journal (WCSIT) ISSN*, pages 2221–0741.

Hachey, B., Murray, G., and Reitter, D. (2006). Dimensionality reduction aids term co-occurrence based multi-document summarization. In *Proceedings of the workshop on task-focused summarization and question answering*, pages 1–7. Association for Computational Linguistics.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18.

Hasler, L. (2007). *From extracts to abstracts: human summary production operations for computer-aided summarisation*. PhD thesis, University of Wolverhampton.

He, L., Sanocki, E., Gupta, A., and Grudin, J. (1999). Auto-summarization of audio-video presentations. In *Proceedings of the seventh ACM international conference on Multimedia (Part 1)*, pages 489–498. ACM.

Hovy, E. and Lin, C. (1999). Automated multilingual text summarization and its evaluation. Technical report, Technical Report, Information Sciences Institute, University of Southern California, Los Angeles, SC, USA.

Hovy, E. and Lin, C.-Y. (1998). Automated text summarization and the summarist system. In *Proceedings of a workshop on held at Baltimore, Maryland: October 13-15, 1998*, pages 197–214. Association for Computational Linguistics.

Hovy, E. and Marcu, D. (2005). Automated text summarization. *The Oxford Handbook of computational linguistics*, pages 583–598.

Huang, M., Shi, X., Jin, F., and Zhu, X. (2012). Using first-order logic to compress sentences. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Imam, I., Nounou, N., Hamouda, A., and Khalek, H. A. A. (2013). An ontology-based summarization system for arabic documents (ossad). *International Journal of Computer Applications*, 74(17).

Jing, H. (2000). Sentence reduction for automatic text summarization. In *Proceedings of the sixth conference on Applied natural language processing*, pages 310–315. Association for Computational Linguistics.

Jing, H. (2002). Using hidden markov modeling to decompose human-written summaries. *Computational linguistics*, 28(4):527–543.

Jing, H. and McKeown, K. R. (2000). Cut and paste based text summarization. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 178–185. Association for Computational Linguistics.

John, G. H. and Langley, P. (1995). Estimating continuous distributions in bayesian classifiers. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pages 338–345. Morgan Kaufmann Publishers Inc.

Jones, K. S. et al. (1999). Automatic summarizing: factors and directions. *Advances in automatic text summarization*, pages 1–12.

Kauchak, D. (2013). Improving text simplification language modeling using unsimplified text data. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1537–1546, Sofia, Bulgaria. Association for Computational Linguistics.

Ker, S. J. and Chen, J.-N. (2000). A text categorization based on summarization technique. In *Proceedings of the ACL-2000 workshop on Recent advances in natural language processing and information retrieval: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics-Volume 11*, pages 79–83. Association for Computational Linguistics.

Keskes, I., Boudabous, M. M., Maaloul, M. H., and Hadrich Belguith, L. (2012). Étude comparative entre trois approches de résumé automatique de documents arabes. In *Proceedings of the Joint Conference JEP-TALN-RECITAL 2012, volume 2: TALN*, pages 225–238, Grenoble, France. ATALA/AFCP.

Khan, A., Salim, N., and Kumar, Y. J. (2015). A framework for multi-document abstractive summarization based on semantic role labelling. *Applied Soft Computing*, 30:737–747.

Khan, A. U., Khan, S., and Mahmood, W. (2005). Mrst: A new technique for information summarization. In *WEC (2)*, pages 249–252.

Kumar, M., Das, D., Agarwal, S., and Rudnicky, A. I. (2009). Non-textual event summarization by applying machine learning to template-based language generation. In *Proceedings of the 2009 Workshop on Language Generation and Summarisation*, pages 67–71. Association for Computational Linguistics.

Kupiec, J., Pedersen, J., and Chen, F. (1995). A trainable document summarizer. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 68–73. ACM.

Le Cessie, S. and Van Houwelingen, J. C. (1992). Ridge estimators in logistic regression. *Applied statistics*, pages 191–201.

Leskovec, J., Milic-Frayling, N., and Grobelnik, M. (2005). Impact of linguistic analysis on the semantic graph coverage and learning of document extracts. In *AAAI*, pages 1069–1074.

Li, C., Liu, F., Weng, F., and Liu, Y. (2013). Document summarization via guided sentence compression. In *EMNLP*, pages 490–500.

Li, S., Ouyang, Y., Wang, W., and Sun, B. (2007). Multi-document summarization using support vector regression. In *Proceedings of DUC*. Citeseer.

Lin, C.-Y. (1995). Knowledge-based automatic topic identification. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 308–310. Association for Computational Linguistics.

Lin, C.-Y. (2003). Improving summarization performance by sentence compression: a pilot study. In *Proceedings of the sixth international workshop on Information retrieval with Asian languages-Volume 11*, pages 1–8. Association for Computational Linguistics.

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81.

Liu, F., Flanigan, J., Thomson, S., Sadeh, N., and Smith, N. A. (2015). Toward abstractive summarization using semantic representations. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1077–1086, Denver, Colorado. Association for Computational Linguistics.

Liu, F. and Liu, Y. (2009). From extractive to abstractive meeting summaries: Can it be done by sentence compression? In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 261–264. Association for Computational Linguistics.

Lloret, E., Boldrini, E., Vodolazova, T., Martínez-Barco, P., Muñoz, R., and Palomar, M. (2015). A novel concept-level approach for ultra-concise opinion summarization. *Expert Syst. Appl.*, 42(20):7148–7156.

Lloret, E. and Palomar, M. (2009). A gradual combination of features for building automatic summarisation systems. In *International Conference on Text, Speech and Dialogue*, pages 16–23. Springer.

Lloret, E. and Palomar, M. (2011). Analyzing the use of word graphs for abstractive text summarization. In *Proceedings of the First International Conference on Advances in Information Mining and Management, Barcelona, Spain*, pages 61–6.

Lloret, E. and Palomar, M. (2012). Text summarisation in progress: a literature review. *Artificial Intelligence Review*, 37(1):1–41.

Louis, A. and Nenkova, A. (2013). Automatically assessing machine summary content without a gold standard. *Computational Linguistics*, 39(2):267–300.

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of research and development*, 2(2):159–165.

Mani, I. and Maybury, M. T. (1999). *Advances in automatic text summarization*, volume 293. MIT Press.

Mann, W. C. and Thompson, S. A. (1988). Rhetorical structure theory: Toward a functional theory of text organization. *Text-Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281.

Marcu, D. (1997). From discourse structures to text summaries. In *Proceedings of the ACL*, volume 97, pages 82–88. Citeseer.

Marcu, D. (1998). To build text summaries of high quality, nuclearity is not sufficient. In *Working Notes of the AAAI-98 Spring Symposium on Intelligent Text Summarization*, pages 1–8.

Marcu, D. (1999). Discourse trees are good indicators of importance in text. *Advances in automatic text summarization*, pages 123–136.

Marcu, D. (2000). *The theory and practice of discourse parsing and summarization*. MIT press.

Martins, A. F. and Smith, N. A. (2009). Summarization with a joint model for sentence extraction and compression. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*, pages 1–9. Association for Computational Linguistics.

McKeown, K., Rosenthal, S., Thadani, K., and Moore, C. (2010). Time-efficient creation of an accurate sentence fusion corpus. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 317–320. Association for Computational Linguistics.

Medelyan, O. (2007). Computing lexical chains with graph clustering. In *Proceedings of the 45th Annual Meeting of the ACL: Student Research Workshop*, pages 85–90. Association for Computational Linguistics.

Mehdad, Y., Carenini, G., and Ng, R. T. (2014). Abstractive summarization of spoken and written conversations based on phrasal queries. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1220–1230, Baltimore, Maryland. Association for Computational Linguistics.

Mihalcea, R. (2004). Graph-based ranking algorithms for sentence extraction, applied to text summarization. In *The Companion Volume to the Proceedings of 42st Annual Meeting of the Association for Computational Linguistics*, pages 170–173, Barcelona, Spain. Association for Computational Linguistics.

Mihalcea, R. and Tarau, P. (2004). Textrank: Bringing order into texts. In Lin, D. and Wu, D., editors, *Proceedings of EMNLP 2004*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Miller, G. and Fellbaum, C. (1998). Wordnet: An electronic lexical database.

Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. J. (1990). Introduction to wordnet: An on-line lexical database. *International journal of lexicography*, 3(4):235–244.

Mitkov, R., Evans, R., Orăsan, C., Pekar, V., et al. (2007). Anaphora resolution: To what extent does it help nlp applications? In *Discourse Anaphora and Anaphor Resolution Colloquium*, pages 179–190. Springer.

Mori, T. (2002). Information gain ratio as term weight: the case of summarization of ir results. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics.

Nema, P., Khapra, M. M., Laha, A., and Ravindran, B. (2017). Diversity driven attention model for query-based abstractive summarization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1063–1072. Association for Computational Linguistics.

Nenkova, A. (2006). *Understanding the process of multi-document summarization: content selection, rewriting and evaluation.* PhD thesis, Columbia University.

Nenkova, A. (2008). Entity-driven rewrite for multidocument summarization. In *In Proceedings of IJCNLP'08*.

Nenkova, A. and McKeown, K. (2011). Automatic summarization. *Foundations and Trends® in Information Retrieval*, 5(2–3):103–233.

Orasan, C. (2004). The influence of personal pronouns for automatic summarisation of scientific articles. In *Proceedings of the Discourse Anaphora and Anaphor Resolution Colloquium*, pages 127–132.

Orasan, C. (2007). Pronominal anaphora resolution for text summarisation. *Proceedings of the Recent Advances in Natural Language Processing*, pages 430–436.

Orăsan, C. (2009). Comparative evaluation of term-weighting methods for automatic summarization*. *Journal of Quantitative Linguistics*, 16(1):67–95.

Oufaida, H., Nouali, O., and Blache, P. (2014). Minimum redundancy and maximum relevance for single and multi-document arabic text summarization. *Journal of King Saud University-Computer and Information Sciences*, 26(4):450–461.

Patil, K. and Brazdil, P. (2007). Text summarization: Using centrality in the pathfinder network. *Int. J. Comput. Sci. Inform. Syst*, 2:18–32.

Peng, H., Long, F., and Ding, C. (2005). Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on pattern analysis and machine intelligence*, 27(8):1226–1238.

Pighin, D., Cornolti, M., Alfonseca, E., and Filippova, K. (2014). Modelling events through memory-based, open-ie patterns for abstractive summarization. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 892–901.

Platt, J. C. (1999). Advances in kernel methods. chapter Fast Training of Support Vector Machines Using Sequential Minimal Optimization, pages 185–208. MIT Press, Cambridge, MA, USA.

Qazvinian, V. and Radev, D. R. (2008). Scientific paper summarization using citation summary networks. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 689–696. Association for Computational Linguistics.

Qian, X. and Liu, Y. (2013). Fast joint compression and summarization via graph cuts. In *EMNLP*, pages 1492–1502.

Quinlan, J. R. (1993). C4.5: Programming for machine learning. *Morgan Kauffmann*, 38.

Radev, D. R. and Fan, W. (2000). Automatic summarization of search engine hit lists. In *Proceedings of the ACL-2000 workshop on Recent advances in natural language processing and information retrieval: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics-Volume 11*, pages 99–109. Association for Computational Linguistics.

Radev, D. R., Hovy, E., and McKeown, K. (2002). Introduction to the special issue on summarization. *Computational linguistics*, 28(4):399–408.

Radev, D. R. and McKeown, K. R. (1998). Generating natural language summaries from multiple on-line sources. *Computational Linguistics*, 24(3):470–500.

Rush, A. M., Chopra, S., and Weston, J. (2015). A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal. Association for Computational Linguistics.

Saggion, H. (2009). A classification algorithm for predicting the structure of summaries. In *Proceedings of the 2009 Workshop on Language Generation and Summarisation*, pages 31–38. Association for Computational Linguistics.

Saggion, H. and Funk, A. (2009). Extracting opinions and facts for business intelligence. *RNTI Journal, E (17)*, 119:146.

Saggion, H. and Poibeau, T. (2013). Automatic text summarization: Past, present and future. In *Multi-source, Multilingual Information Extraction and Summarization*, pages 3–21. Springer.

Sakai, T. and Sparck-Jones, K. (2001). Generic summaries for indexing in information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 190–198. ACM.

Salton, G., Singhal, A., Mitra, M., and Buckley, C. (1997). Automatic text structuring and summarization. *Information Processing & Management*, 33(2):193–207.

Sarkar, K. (2008). Syntactic sentence compression: Facilitating web browsing on mobile devices. In *International Conference on Information Technology, 2008. ICIT'08*, pages 283–286. IEEE.

Schapire, R. E. (1990). The strength of weak learnability. *Machine learning*, 5(2):197–227.

Schilder, F. and Kondadadi, R. (2008). Fastsum: fast and accurate query-based multi-document summarization. In *Proceedings of the 46th annual meeting of the association for computational linguistics on human language technologies: Short papers*, pages 205–208. Association for Computational Linguistics.

Schlesinger, J. D., Okurowski, M. E., Conroy, J. M., O'Leary, D. P., Taylor, A., Hobbs, J., and Wilson, H. T. (2002). Understanding machine performance in the context of human performance for multi-document summarization.

Schlesinger, J. D., O'leary, D. P., and Conroy, J. M. (2008). Arabic/english multi-document summarization with classy—the past and the future. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 568–581. Springer.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1):1–47.

See, A., Liu, P. J., and Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083. Association for Computational Linguistics.

Shen, D., Chen, Z., Yang, Q., Zeng, H.-J., Zhang, B., Lu, Y., and Ma, W.-Y. (2004). Web-page classification through summarization. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 242–249. ACM.

Shen, D., Yang, Q., and Chen, Z. (2007). Noise reduction through summarization for web-page classification. *Information processing & management*, 43(6):1735–1747.

Siddharthan, A. (2002). An architecture for a text simplification system. In *Language Engineering Conference, 2002. Proceedings*, pages 64–71. IEEE.

Sobh, I. M. A. H. (2009). *An optimized dual classification system for Arabic extractive generic text summarization*. PhD thesis, Citeseer.

Spärck Jones, K. (2007). Automatic summarising: The state of the art. *Information Processing & Management*, 43(6):1449–1481.

Steinberger, J., Jezek, K., and Sloup, M. (2008). Web topic summarization. In *ELPUB2008*.

Steinberger, J., Poesio, M., Kabadjov, M. A., and Ježek, K. (2007). Two uses of anaphora resolution in summarization. *Information Processing & Management*, 43(6):1663–1680.

Sun, J.-T., Shen, D., Zeng, H.-J., Yang, Q., Lu, Y., and Chen, Z. (2005). Web-page summarization using clickthrough data. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 194–201. ACM.

Svore, K. M., Vanderwende, L., and Burges, C. J. (2007). Enhancing single-document summarization by combining ranknet and third-party sources. In *EMNLP-CoNLL*, pages 448–457.

Tan, J., Wan, X., and Xiao, J. (2017). Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1171–1181. Association for Computational Linguistics.

Tanaka, H., Kinoshita, A., Kobayakawa, T., Kumano, T., and Kato, N. (2009). Syntax-driven sentence revision for broadcast news summarization. In *Proceedings of the 2009 Workshop on Language Generation and Summarisation*, pages 39–47. Association for Computational Linguistics.

Vanderwende, L., Suzuki, H., Brockett, C., and Nenkova, A. (2007). Beyond sumbasic: Task-focused summarization with sentence simplification and lexical expansion. *Information Processing & Management*, 43(6):1606–1618.

Vapnik, V. N. (1999). An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999.

Vapnik, V. N. (2000). The nature of statistical learning theory. statistics for engineering and information science. *Springer-Verlag, New York*.

Victoria, M. (2004). Statistical approaches to automatic text summarization. *Bulletin of the American Society for Information Science and Technology*, 30(4).

Wan, X. and Xiao, J. (2007). Towards a unified approach based on affinity graph to various multi-document summarizations. In *International Conference on Theory and Practice of Digital Libraries*, pages 297–308. Springer.

Wilson, T., Wiebe, J., and Hoffmann, P. (2005). Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pages 347–354. Association for Computational Linguistics.

Wong, K.-F., Wu, M., and Li, W. (2008). Extractive summarization using supervised and semi-supervised learning. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 985–992. Association for Computational Linguistics.

Woodsend, K. and Lapata, M. (2011). Wikisimple: automatic simplification of wikipedia articles. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, pages 927–932. AAAI Press.

Wubben, S., van den Bosch, A., and Krahmer, E. (2012). Sentence simplification by monolingual machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 1015–1024. Association for Computational Linguistics.

Yamangil, E. and Shieber, S. M. (2010). Bayesian synchronous tree-substitution grammar induction and its application to sentence compression. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 937–947. Association for Computational Linguistics.

Yeh, J.-Y., Ke, H.-R., Yang, W.-P., and Meng, I.-H. (2005). Text summarization using a trainable summarizer and latent semantic analysis. *Information processing & management*, 41(1):75–95.

Yoshikawa, K., Hirao, T., Iida, R., and Okumura, M. (2012). Sentence compression with semantic role constraints. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 349–353. Association for Computational Linguistics.

Zajic, D., Dorr, B. J., Lin, J., and Schwartz, R. (2007). Multi-candidate reduction: Sentence compression as a tool for document summarization tasks. *Information Processing & Management*, 43(6):1549–1570.

Zechner, K. and Waibel, A. (2000). Diasumm: Flexible summarization of spontaneous dialogues in unrestricted domains. In *Proceedings of the 18th conference on Computational linguistics-Volume 2*, pages 968–974. Association for Computational Linguistics.

Zhou, Q., Yang, N., Wei, F., and Zhou, M. (2017). Selective encoding for abstractive sentence summarization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1095–1104. Association for Computational Linguistics.