AN INVESTIGATION INTO THE CROSS-LINGUISTIC ROBUSTNESS OF TEXTUAL EQUIVALENCE TECHNIQUES

A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy
in the Faculty of Science and Engineering

2018

By
Amal Mohammad Alshahrani
Computer Science

Contents

Word Count: 48,767

List of Tables

List of Figures

Abstract

This thesis explores a range of techniques that have been applied to the task of Textual Equivalence (TEQV), i.e., identifying whether one text snippet is equivalent to another. This task has been widely explored for English texts. In this study we investigate and analyse the extent to which these techniques generalise to other languages, in particular Arabic. Written Arabic is widely said to be more ambiguous than English. This ambiguity makes determining the relationships between text snippets particularly challenging. We have tried to use these techniques in settings which are as similar as possible so that any differences that appear in the experimental results can be reliably attributed to differences between the two languages, rather than to differences in the experimental set-up. In particular the dynamic time warping (DTW) algorithm has been used to measure the similarity between sentence pairs by calculating the minimum number of editing operations (Insert, Delete, Exchange) which are required to convert one sentence to another. Also WordNet similarity measures have been used as a cost function for the Exchange operation. This algorithm has been extended with an extra operation, Swap, which allows for local permutations to compensate for the comparatively free word order of Arabic.

The outcome is that when we extend the coverage of Arabic WordNet we obtain similar results to the use of English WordNet for TEQV for English; and that using the extended version of DTW provides more benefits for Arabic than for English.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

Acknowledgements

After God Almighty, who inspired and blessed this effort all the way to its completion, I would like to express my heartfelt sense of gratitude to a number of people who provided me with guidance and support throughout this journey.

First and foremost, I would like to express my sincere thanks to a wonderful supervisor Professor Allan Ramsay for his infinite patience, continuous guidance, and scholarly advice. His invaluable suggestions and guidance have shaped the main structure of this project even while helping me overcome the challenges that popped up along the way.

I would also like to thank the Umm Al-Qura University for their generous sponsorship and the staff of the University of Manchester for their help in providing for me an academically enriching experience and a friendly environment. I appreciate their academic support and the facilities provided to carry out the research work at the university.

I want to say Thank you to the members of my research group, who generously shared with me their knowledge and experience. I acknowledge and appreciate their personal and scholarly interactions, and their advice at various points of my research.

My special thanks and appreciations go to our volunteer annotators who were involved in annotating our dataset. To all of them, I would like to extend my sincere gratitude for their time spent helping me.

I will forever be thankful to Dr. Abdul-Mueed, and his wife Dr Aisha for supporting and helping me in my studying and my family.

I owe a lot to my Mother, who encouraged and helped me at every stage of my personal and academic life, and wished to see this achievement come true. I want to say Thank you for always being supportive and praying for me. I would like also to extend my gratitude to my brothers and sisters for their encouragement, support and prayers.

Last but not least, may I from the bottom of my heart thank my beloved husband Obaid. You have literally lived everything with me, the ups and downs, the latter for the most part, and you always knew how to make me feel better. Thank you for being supportive and understanding of my self-imposed isolation. I really appreciate all the sacrifices he made to follow me all the way through this PhD pursuit.

And to my three lovely children, Noura, Reem and Faris, thank you for always making me smile and for putting up with me when I had to work on my thesis instead of playing with you. Thank you. Only God knows how much I love all three of you. Thank you.

Buckwalter Transliteration Table

Arabic Letter	BW	Letter Name	Arabic Letter	BW	Letter Name
ا	A	ALEF	غ	g	GHAIN
أ	>	ALEF WITH HAMZA ABOVE	ف	f	FEH
إ	<	ALEF WITH HAMZA UNDER	ق	q	QAF
آ	|	ALEF WITH MADDA ABOVE	ك	k	KAF
ب	b	BEH	ل	l	LAM
ت	t	TEH	م	m	MEEM
ث	v	THEH	ن	n	NOON
ج	j	JEEM	ه	h	HEH
ح	H	HAH	ة	p	TEH MARBUTA
خ	x	KHAH	و	w	WAW
د	d	DAL	ي	y	YEH
ذ	*	THAL	ى	Y	ALEF MAQSURA
ر	r	REH	**Arabic Diacritics**		
ز	z	ZAIN	◌َ	a	FATHA
س	s	SEEN	◌ُ	u	DAMMA
ش	$	SHEEN	◌ِ	i	KASRA
ص	S	SAD	◌ً	F	TANWIN ALFATH
ض	D	DAD	◌ٌ	N	TANWIN ALDAM
ط	T	TAH	◌ٍ	K	TANWIN ALKASER
ظ	Z	ZAH	◌ّ	~	SHADDA
ع	E	AIN	◌ْ	o	Sukun

Table 1: Buckwalter translation table.

List of Abbreviations

Abbreviation	**Full Form**
ADJ	Adjective phrase
ADV	Adverb phrase
AWN	Arabic WordNet
BAMA	Buckwalter Arabic Morphological Analyser
CP	Complement phrase
DEL	Delete
DIRT	Discovery of Inference Rules from Text
DTW	Dynamic Time Warping
EWN	English WordNet
IC	Information Content
IDF	Inverse document frequency
IE	Information Extraction
if	inherited frequency
INS	Insert
IR	Information Retrieval
IRR	Inter-rater-reliability
LCS	Longest Common Subsequence
LHS	Left-Hand Side
LMF	Lexical Markup Framework
MADA	morphological analysis and disambiguation for Arabic
MT	Machine Translation
MSRP	Microsoft paraphrase corpus
MXL	Maximum Likelihood
NEs	Named Entities
NLP	Natural Language Processing

Abbreviation	Full Form
NP	Noun phrase
OVS	Object-verb-subject
P4P	Paraphrase of Plagiarism
PMI	Point-wise Mutual Information
POS	Part of Speech
PPDB	Paraphrase Database
QA	Question Answering
RHS	Right-Hand Side
RSS	Rich Site Summary
RTE	Recognising Textual Entailment
STS	Semantic Text Similarity
SVO	Subject-verb-object
TE	Textual Entailment
TEASE	Textual Entailment Anchor Set Extraction
TED	Tree Edit Distance
TEQV	Textual Equivalence
TF	Term frequency
TS	Text Summarization
VNN	Verb-noun-noun
VOS	Verb-object-subject
VSO	Verb-subject-object
WRPA	Wikipedia-based Rational Paraphrase Acquisition
XCH	Exchange
XDTW	eXtended Dynamic Time Warping

Chapter 1

Introduction

1.1 Research Overview

This thesis is concerned with the task of determining whether one text fragment is equivalent to another. This is an extremely important task in many Natural Language Processing (NLP) applications such as *Text Summarisation* (TS), *Information Retrieval* (IR), *Question Answering* (QA), or *Machine Translation* (MT). A number of approaches to this task have been developed, largely for English. The aim of the work reported here is to investigate the extent to which techniques that work well for English can be transferred unchanged to Arabic, and to look at ways of overcoming some of the problems that arise.

Equivalence can be defined as a relationship between two sentences S_1 and S_2, where the two sentences convey the same meaning, but are not identical (Jago 2007). In the examples below, there is an equivalence between S_1 and S_2 in (1.1) whereas S_1 is not equivalent to S_2 in (1.2).

Example 1.1. Equivalent sentences:

S_1. Google **bought** YouTube.

S_2. Google **purchased** YouTube.

Example 1.2. Non-equivalent sentences:

S_1. The student was **assassinated**.

S_2. The student is **dead**.

In example (1.1), S_1 and S_2 are equivalent, since *bought* has the same meaning as *purchased*, meaning that S_1 entails S_2 and S_2 entails S_1. Therefore, equivalence is a mutual entailment, and can also be defined as a pair of bidirectional entailment relations: A pair of expressions between which entailment relations of both directions hold. (Hashimoto et al. 2011). In contrast, example (1.2) is not an equivalence, since *assassinated* is not the same as *dead*. An entailment relationship can be found, since *assassinated* entails that the person in question is dead, but the reverse does not hold.

(Dagan et al. 2006a) have suggested a restricted notion called *Textual Entailment*. In relation to this, the authors state *"We say that T entails H if the meaning of H can be inferred from the meaning of T, as would typically be interpreted by people"*. This thesis is concerned with 'Textual Equivalence' (TEQV) that can be defined as a mutual textual entailment (TE).

A number of techniques for this task have been widely used for English texts. The aim of the current study is to investigate and analyse the extent to which these techniques generalise to other languages, in particular Arabic. In order to investigate this, it is necessary to apply these techniques in settings which are as similar as possible, so that any differences that appear in the experimental results can be reliably attributed to differences between the two languages, rather than to differences in the experimental set-up. To this end, an abstract work-flow has been designed for developing and evaluating a general TEQV system, which has then been instantiated as similarly as possible for the two languages.

There has been limited research on applying TEQV techniques to Arabic. The main problem that must be taken into consideration is that each language has different features; for example, Arabic is more ambiguous than English (Attia 2012). This ambiguity makes determining the relationships between text snippets particularly challenging, so much so that many of the existing approaches to TEQV are likely to be inapplicable. These challenges are explained in detail in Chapter 3.

1.2 Research Questions and Tasks

TEQV is an important task in NLP that requires language understanding. The techniques used for it have been developed significantly in the last few years, especially for English texts. The research proposed here attempts to address the

following research questions:

Research Question 1: Can Textual Equivalence (TEQV) techniques that work for English be equally well applied to Arabic?

Research Question 2: Is it possible to make the widely used string edit distance algorithm more robust and reliable without increasing the complexity, to help and cope with free word-order languages?

Research Question 3: How well does Arabic WordNet (AWN) support semantic similarity measures?

The following research tasks have been designed to answer these questions and help achieve the principal aims of the research:

Research Task 1: To collect and annotate comparable English and Arabic datasets.

Research Task 2: To develop an implementation of the standard string edit distance algorithm that can cope with free word-order languages by adding an extension allowing transposition of adjacent items.

Research Task 3: To integrate Arabic WordNet (AWN: (Black et al. 2006) and English WordNet (EWN: (Fellbaum 1998)) with Pyaramorph (PYA)x[1] to obtain potential synsets of an input Arabic word.

Research Task 4: To apply and evaluate different variants of the system to see how this affects performance in detecting paraphrases in the fragment texts.

In the following, the research tasks set out above are discussed in more detail.

Research Task 1

A number of tools for collecting and preprocessing English and Arabic text were designed and implemented. These texts were then annotated by human subjects to form Gold Standards for the experiments carried out in Research Task 4. In order to ensure that the English and Arabic datasets were as comparable as possible, an abstract workflow for the development TEQV system was designed and followed during this process. Research Task 1 supports Research Question 1 and Research Question 3 by providing the data for training and testing the various combinations of tools.

[1]Pyaramorph is a Python reimplementation of the Buckwalter Standard Morphological Analyser (BAMA) (Buckwalter 2004): see Section 5.3.2 for more details.

Research Task 2

The existing string edit distance algorithm is improved, updated and extended to make it more robust and more effective, by allowing the transposition of adjacent items as an edit operation 'SWAP', which partially tackles the problem of free word order languages. This enhanced version which is called 'eXtended Dynamic Time Warping' (XDTW) algorithm has been used to extend the Dynamic Time Warping (DTW) algorithm (Sakoe and Chiba 1978). XDTW operations are enhanced to produce cost-effective results compared to DTW operations by transferring one string to another, and calculating the minimum number of operations (insert, delete, exchange, and swap). This change to the standard algorithm has a modest but useful effect on the accuracy, as demonstrated in the results in Chapter 6. This research task is used to address Research Question 2.

Research Task 3

Part of the problem is that Arabic poses a number of problems that are not present in most other languages. For instance, it is a morphologically rich language. In order to overcome this problem in the current project, the Arabic morphological analyser (Pyaramorph) is integrated with AWN and EWN so that the information in Pyaramorph can be used to access the AWN. This research task used to address Research Question 3.

Research Task 4

A number of experiments were carried out to assess the effectiveness of different methods. The similarity scores of these methods were compared to our 'gold standard' by using precision rates. Then, we have carried out a statistical analysis of the results to find out how the precision varies as we vary the conditions. The aim of these analyses was to find out the best WordNet similarity measures for using with DTW and XDTW. In addition, because Arabic has free word order, and therefore, we expect the XDTW will produce better result for Arabic than English as discussed in Chapter 6. Research Task 4 is used to support RQ1.

1.3 Research Actions

The actions carried out to complete this study are comprised of five main stages as follows:

1. **Data collection**: In this stage relevant important data was extracted from different online newswire services (see Chapter 5). To achieve this goal, the TEQV system collected data automatically from different newswire services, utilizing their RSS[2] feeds. RSS feeds provide documents in a convenient form by splitting HTML pages into a header, summary, date and time. Hence, it is relatively straightforward to collect articles on events that happened on the same day and at the same time, which are likely to contain sentences that might be candidate paraphrases for the current study. A regular expression[3] classifier was used to clean all unwanted content, such as comments, HTML tags and other non human-readable elements.

2. **Data pre-processing**: This stage is used to normalise the text in which the selected articles were split first into sentences and then into words, and the whole sentences have been tagged. To achieve this goal, the main techniques used were: tokenisation and sentence splitter, part of speech (POS) tagging and morphological analysis.

3. **Data clustering**: In this stage, the dataset was clustered and classified into pairs of articles and then into pairs of sentences (S_1, S_2) that were likely to contain suitable candidate paraphrases. To achieve this goal, simple standard techniques were used (e.g., cosine similarity and tf-idf vector space), which are useful measures for clustering the data into pairs of articles and then into pairs of sentences. The output of this stage is a 'sentence pair' dataset that was used in the next stage. It is necessary to filter the data to obtain a balance corpus (sentence pairs): a cosine similarity threshold of 0.6 and above was selected for classification between positive and negative cases. This threshold was used to find a balanced set of pairs that were plausibly related for submission to the annotators to mark up and to the system to calculate the similarity scores, as illustrated in chapter 4.

[2]Rich Site Summary, which is a format for delivering regularly changing web content.

[3]Regular expression is a sequence of characters that define a search pattern, mainly for use in pattern matching with string matching.

4. **Similarity judgments**: This stage is used to check the human- and system-based similarity judgments between the sentence pairs.

    i. **Human-based judgment**: in this phase the datasets 'sentence pairs' are turned into a 'Gold Standard'. This is achieved by using human judgment to annotate these datasets by using an online annotation form. Following this, the reliability of the annotators was checked, using inter-rater reliability (IRR).

    ii. **System-based judgment**: the same datasets were used in this phase to align fragment sequences between pairs of sentences that have the same meaning. The DTW algorithm was used to align the words while maintaining their original order, and its extension, while the XDTW algorithm was used to allow transposition operations between adjacent words. These algorithms measure the minimum cost distance of operations (Insert, Delete, Exchange, and Swap) by converting one string to another. These algorithms were tested with a range of WordNet semantic similarity measures (see Chapter 4) to obtain similarity scores.

5. **Performance Evaluation**: The 'similarity scores' obtained from the system-based judgments and the 'Gold Standard' obtained from the human-based judgments were assessed by using a range of evaluation measures, namely: precision and recall. We have carried out a statistical analysis of the results to demonstrate the performance among baselines, DTW and XDTW systems. The main comparison was between the XDTW system applied to both English and Arabic and overall to study the differences in the results of these systems.

1.4 Research Contributions

The main contributions to the current research are as follows:

Contribution 1: Designing and implementing an abstract work-flow for a TEQV system to compare and evaluate the effectiveness of a set of common strategies across multiple languages.

Contribution 2: Constructing comparable English and Arabic datasets by following strictly comparable steps. This is an essential development, since while there are English and Arabic datasets available, these have not been uniformly collected and hence are not suitable for doing the comparison, and then to investigate the differences in the algorithm.

Contribution 3: Constructing a TEQV system for Arabic (some similar work has been carried out previously but it used different mechanisms, e.g., Alabbas (Alabbas 2013b), who built a system for Arabic in Textual entailment called (ArbTE) using Tree edit distance (TED) to compare the distance between two dependency trees, and taking the linguistic analysis into account).

The task of developing a TEQV for Arabic is an interesting one. We believe that the Recognising Textual Entailment (RTE) and the paraphrase research community are likely to benefit significantly from work about TEQV in languages other than English, particularly Arabic, since it has many characteristics (described in the next chapter) which make it challenging for Natural Language Processing (NLP) in general.

Contribution 4: Investigating an improvement to the standard Dynamic Time Warping (DTW) algorithm which adapted it to Arabic. This is achieved by extending the set of edit operations to cover transposition operations as well as the standard edit operations. Arabic is a free word order language, so the simple DTW alignment is not a suitable measure for some cases in Arabic. The extended version of the alignment method helps to address this problem.

Contribution 5: applying WordNet based-similarity measures for both English and Arabic datasets and comparing the results using statistical analysis. The initial test results we obtained from applying such measures is that the WordNet similarity measures work very well for English datasets but less effectively in Arabic datasets. We investigated the reasons behind such a disparate performance and concluded that the AWN is sparse and smaller than the lexical database of EWN which might affect the similarity scores. For this reason we planned for supporting AWN with additional resources.

Contribution 6: Integrating AWN and EWN with Pyaramorph in order to obtain more information that can be used to support the coverage of Arabic words synset and not depending solely on AWN for obtaining such information. The lexical database of AWN is smaller than the lexical database of EWN, and therefore, it is not possible to find the synset for a significant number of words, and this produces additional ambiguity in Arabic, making it difficult to find the right synset. Furthermore, it is relatively harder to obtain word forms in AWN comparing to EWN. Thus, the reason of such integration is to increase the richness of the lexical resources available to the Arabic version of our system.

1.5 Thesis Outline

The thesis is organised into seven chapters and the organisation of chapters is depicted in Figure 1.1 for reading guidance.

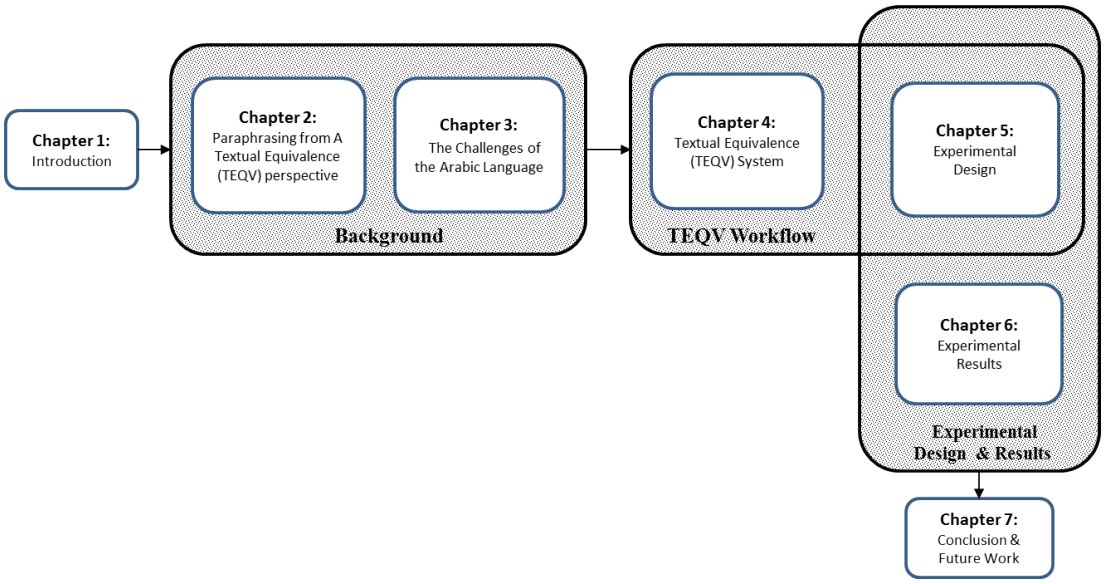


Figure 1.1: Thesis Reading Roadmap - the arrows indicate the reading path between parts.

A summary for each chapter is listed below:

Chapter 1 contains the introduction, which presents an overview of the research problem. Following this, research questions and tasks are highlighted, and the research actions that are used to build the TEQV system are discussed. Following this, the main contributions of the research to the literature are presented.

Chapter 2 discusses background issues such as entailment and equivalence. The problem of the entailment is introduced from a linguistics perspective, and then the TEQV task is presented in some detail in terms of the notion, inference rules, characteristics and the sources of textual equivalence. Moreover, the main applications of paraphrasing are explained. Finally, related work in the areas of paraphrasing is reviewed.

Chapter 3 focuses on the challenges of Arabic because it is more ambiguous than other languages such as English. Some examples are included elaborating these ambiguities.

Chapter 4 presents an abstract work-flow for building and evaluating the TEQV system, which contains four main components: data collection, pre-processing, data clustering, and finally similarity judgment was examined using human- and system-based judgments. Following this, the output of the human and system judgment were compared using evaluation measures precision (P) and recall (R).

Chapter 5 presents the series of experiments of design, such as Dataset preparation, Experimental implementation and results of the TEQV system when applied to two different languages (English and Arabic). These experiments were structured according to the general architecture presented in the previous chapter. The aim of this Chapter is to investigate the performance of systems that are used on English and Arabic datasets.

Chapter 6 presents the statistical analysis of the results of the experiments described in Chapter 5 to find how the different variations relate to the Gold Standard result.

Chapter 7 presents the final remarks of the thesis and concludes by suggesting some directions for future work and research.

Chapter 2

Paraphrasing from a TEQV perspective

2.1 Overview

In this chapter, the general notions of entailment and equivalence that are used in everyday language will be discussed from a linguistic perspective in Section 2.2. In Section 2.3 the notion of Textual Equivalence (TEQV) as paraphrasing with its inference rules, characteristics features, levelsand sources, and the main applications of paraphrasing will be explored in more detail. Finally, a brief summary of previous methods in the field will be presented in Section 2.4.

2.2 Entailment and equivalence in linguistics

Entailment in linguistics can be defined as '*A entails B*' if anyone who really understands *A and B* and thinks carefully about them, agrees that B will be true whenever A is. On that basis, in case S_1 is true, then S_2 should be true as well; likewise if S_2 is false, then S_1 should be false (Merrison et al. 2013). Besides, when S_1 entails S_2, then the information carried by S_2 is included in S_1. As a result, S_1 is considered to be more informative than S_2.

Example 2.1. Entailment.

(S_1) John and Mary went to the party.

(S_2) John went to the party.

Example 2.2. Non-entailment.

(S_1) No children came to the nursery for morning session.

(S_2) No children came to the nursery.

In (2.1) S_1 entails S_2, where John and Mary went to the party; nevertheless, the reverse relationship does not hold because Mary possibly did not go to the party. However, in (2.2) the first sentence does not entail the second, for the reason that children possibly came to the afternoon session. Therefore, it is not true to say that there were no children at the nursery; hence, this is non-entailment.

Entailment is related to other relationships (e.g. equivalence, contradiction, presupposition and implicature), which play a significant role in determining the connections between sentences. In particular, when S_1 entails S_2 and vice versa, then both S_1 and S_2 are equivalent, synonymous, or are paraphrases of each other. In other words, in a paraphrase two, sentences have the same or almost the same meaning, which can be seen as mutually entailing. Example (2.3) demonstrates equivalence in relation between two sentences.

Example 2.3. Equivalent sentences.

(S_1) Sarah **bought** a book.

(S_2) Sarah **purchased** a book.

Example 2.4. Non-equivalent sentences.

(S_1) The student was **assassinated**.

(S_2) The student is **dead**.

In (2.3), there is an equivalent semantic content given that *bought* is the synonym used instead of *purchased*, whereas in (2.4), there is no equivalent semantic content, since *assassinated* is not the same as *dead*. Thus, to a certain extent there is an entailment relationship as *assassinated* entails that the person in question is dead. In this research, we shall focus on the '*equivalence*' relation.

A further relationship that could be tested via entailment is that of *contradiction*, that can be defined in terms of entailment as *"S_1 and S_2 are contraries if and only if S_1 entails not-S_2, but not-S_2 does not entail S_1 (and vice versa)"*

(Cruse 2011). This means that a pair of sentences is considered as contradictory if and only if any of them entails the negation of the another, as illustrated in example (2.5):

Example 2.5. Contradiction.

(S_1) Mary came to the party and had a good time.

(S_2) Mary did not come to the party.

In S_1 entails not-S_2 because not-S_2 states '*Mary did not come to the party*', however not-S_2 does not entail S_1 because $Mary$ might not have had a good time at the party. Therefore S_1 and S_2 are contraries.

Entailment could be used to examine a *presupposition*; assume there are two propositions P and Q as follows: P presupposes Q if both P and not-P entail Q (Bublitz and Norrick 2011), as in (2.6).

Example 2.6. Presupposition.

(S_1) Mary's car is blue.

(S_2) Mary's car is not blue.

(S_3) Mary has a car.

Here, 'Mary's car is blue' and 'Mary's car is not blue' both entail 'Mary has a car', hence S_1 presupposes S_3.

In addition, entailment could be used to test an *implicature*, where the truth of S_1 suggests the truth of S_2, but does not require it, as seen in (2.7).

Example 2.7. Implicature.

(S_1) Most people enjoyed the party.

(S_2) Some people did not enjoy the party.

Entailments absolutely must follow from the basic sentence, which is a necessary implication. However, implicature is the action of implying a meaning beyond the literal sense of what is explicitly stated, which can be a cancellable implication.

Trying to write programs that deal in fine detail with all these issues has proved very difficult. Dagan et al. (2006b) have proposed investigating a possibly simpler notion which they have called *Textual Entailment* (TE): *"We say that T entails H if the meaning of H can be inferred from the meaning of T, as would typically be interpreted by people."*. This thesis is mainly concerned with *Textual Equivalence*, namely with pairs of sentences where TE holds in both directions.

2.3 Textual Equivalence as Paraphrasing

Paraphrasing has a close relation with entailment in a semantic field. Therefore, paraphrasing is referred to as a *semantic equivalence*. There are many definitions of paraphrasing in the literature provided by different authors. Some of them are illustrated in Table 2.1.

Authors	Definitions
Shinyama and Sekine (2003)	*"Equivalent patterns (expressions) that capture the same information"*
Glickman and Dagan (2004)	*"Equivalencies between different expressions that correspond to the same meaning"*
Bar and Dershowitz (2012)	*"paraphrases are sometimes referred to as dynamic translations or semantic equivalents"*
Ganitkevitch et al. (2013)	*"Differing textual realizations of the same meaning"*
(Zhao et al. 2009b)	*"Alternative ways that convey the same meaning"*
Rus et al. (2009)	*"text A is a paraphrase of text B if and only if A entails B and B entails A"*
Madnani and Dorr (2010)	*"an alternative surface form in the same language expressing the same semantic content as the original form"*
Hashimoto et al. (2011)	*"A pair of expressions between which entailment relations of both directions hold"*
Ho et al. (2014)	*"different words, phrases or sentences that express the same or almost the same meaning"*

Table 2.1: Some definitions of paraphrases.

2.3.1 Inference Rules for Paraphrasing

Inference rules for paraphrasing can be described as the bidirectional relationships between two text fragments (templates) with variables or lexical terms, which signify that the left-hand side (LHS) entails its right-hand side (RHS) and vice versa. This is indicated by LHS → RHS and RHS → LHS, as in Example 2.8. Several automatically acquired inference rule and paraphrase collections are obtainable, for instance Szpektor et al. (2004) point out that DIRT[1] and TEASE[2] are the most extensively employed inference rules for paraphrasing.

- DIRT was created by Lin and Pantel (2001), which is based on the Extended Distributional Hypothesis. The original Distributional Hypothesis (Harris 1954) was based on the observation that that words that occur in similar contexts have a meaning that is related; however, the extended version applies this observation to phrases rather than just isolated words. The purpose of DIRT is to choose expressions for linking two nouns and for inferring the relationship that could be between them. The input of this algorithm is considered as an expression representing a relation, such as templates 'X buy Y' and 'X acquire Y'. Two templates with similar co-occurrence distributions are suggested as paraphrases (termed inference rules)from which sets of equivalent expressions can be inferred.

- TEASE is a bootstrapping-based method suggested by Szpektor et al. (2004). Unlike the DIRT method, TEASE uses the web to collect its rules rather than parsed corpora. It starts with templates which share the same anchors. Anchors are lexical elements describing the context of a sentence. Then, it uses the input template to extract verb-based expressions for the other candidate templates for entailment relations.

These inference rules should be applied in specific contexts, defined as '*relevant contexts*' by Szpektor et al. (2007). For example, the rule (2.8b) can be used in the context of *'buying'* events such as "Yahoo *acquired* Overture" and "Yahoo *bought* Overture", but we should not apply it for "the baby acquired a new habit", because a habit is just not the kind of thing you can buy, in other words we are not in a relevant context. It is often hard to tell whether you are in fact in such

[1]DIRT stands for Discovery of Inference Rules from Text.
[2]TEASE stands for Textual Entailment Anchor Set Extraction

context. Consider "The nun *acquired* a new habit" and "The nun *purchased* a new habit". The clothes that nuns wear are known as "habits", so it is possible for a nun to buy a new habit; in order to realise this we need to do lexical disambiguation which is itself a very hard task.

Example 2.8. Relevant contexts and Entailment rules.

(a) $(X \text{ wrote } Y) \rightarrow (X \text{ is the author of } Y)$ (Template with variable)

(b) $(X \text{ buy } Y) \rightarrow (X \text{ acquire } Y)$ (Template with variable)

(c) office $\rightarrow$ bureau (Lexical term)

2.3.2 Characteristic Features of Paraphrases

Numerous linguistic phenomena produce paraphrases. Bhagat and Hovy (2013) indicated that quasi-paraphrases involve two specific kinds of change, namely lexical and structural, which can occur in a sentence or a phrase despite the fact that the approximate meaning is retained (semantics). These changes can be summarised as:

1. **Substitution:** a substitution occurs when a word or phrase in the original sentence or phrase is replaced by a different word or phrase in its paraphrase.

2. **Addition/Deletion:** the addition of an extra word in the paraphrase, which does not have a corresponding word or phrase in the original sentence or phrase, is known as *addition*. Deletion – removing a word from the original sentence without introducing any new words – is the opposite of addition. Consider the pair

   - I know she loves me.
   - I know that she loves me.

   The second of these could be obtained from the first by addition of the word 'that'; conversely, the first could be obtained from the second by deleting this word.

3. **Changing the order or format:** in this case, the orders of words or phrases in a paraphrase are changed; thus, the corresponding words or phrases in the original sentence or phrase have a different relative order. This is known as a permutation.

It is worth noticing that (1) is a lexical change, whereas (2) and (3) are structural changes. Additionally, the lexical changes go hand in hand with modifications in the structure of the original sentence or phrase. This can be seen in Table 2.2.

Features	Description	Example
Synonym substitution	This denotes a replacement of a word or a phrase with its synonym or phrase in the proper context.	• John is **slim**. *(TEQV)* John is **skinny**. • Sarah **bought** a book for you. *(TEQV)* Sarah **acquired** a book for you.
Active/passive substitution	This denotes a modification of the verb from active form to passive form and vice versa. This alteration may possibly involve the removal or addition of a number of function words. It may also mean that the sentence is restructured.	• Mum cooked the food. • The food was cooked by Mum.
Function word variation	This describes a modification of function words in a sentence without altering the meaning.	• Results **of** the competition were announced. • Results **for** the competition were announced.
External knowledge	This denotes a replacement of a word or a phrase by means of another word or phrase based on extra linguistic knowledge, in the appropriate context.	• **Obama** was named the 2009 Nobel Peace Prize laureate. • **The President of the United States** was named the 2009 Nobel Peace Prize laureate.
Change of format	This is where numbers and/or symbols are written in a different way (e.g., %, $, £, etc.).	• Home sales fell **2%**. • Home sales fell **two percent**.

Table 2.2: Features of Paraphrases.

2.3.3 Levels of Paraphrases

Paraphrases can occur at different levels of linguistic structure. Different words, phrases or sentences could have the same meaning and hence be described as

paraphrases. According to Ho et al. (2014), we can distinguish three levels or types: the lexical, the phrase and the sentential level, as can be noted in Figure 2.1.

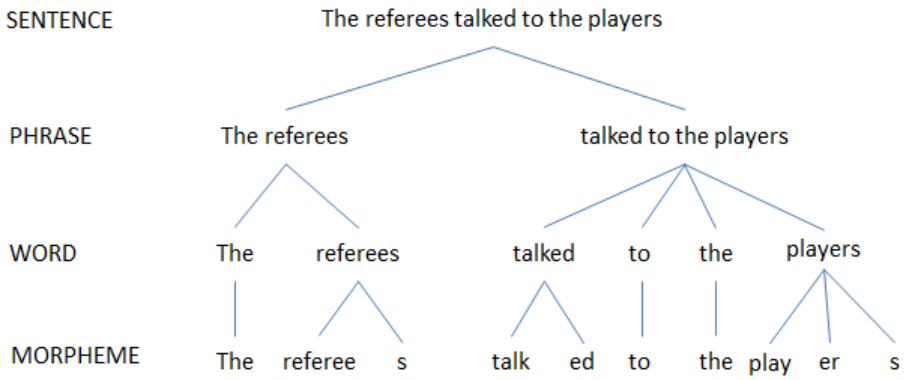


Figure 2.1: The hierarchy of linguistic units. The lexical level is formed from the bottom two levels: morphemes and words.

(i) **Lexical level** refers to individual lexical entities that convey the same meaning and are known as lexical phrases or synonyms, as is the case with happy/glad, for example. However, in addition to synonyms, lexical phrases can take the form of hypernyms, where one of the words is either more specific or more general than the other within the 'paraphrastic' relationship, as in 'reply/say' and 'landlady/hostess'.

(ii) **Phrase level** refers to phrasal fragments sharing the same semantic content (Keshtkar and Inkpen 2010). These fragments are referred to as phrasal paraphrases. These phrases appear frequently in the form of syntactic phrases, as in 'very happy/so glad'.

(iii) **Sentential level** refers to the entire sentence. A number of researchers have examined sentential paraphrases or large paraphrasing templates (Ravichandran and Hovy 2002; Barzilay and Lee 2003; Pang et al. 2003; Brockett and Dolan 2005; Regneri et al. 2014; Bach et al. 2014). Two sorts of sentential paraphrases exist at this level. Firstly, there are surface paraphrases, where the syntactic structure is similar, although the surface words are differen, as in (2.9). Secondly, there are structural paraphrases, where the syntactic structure is different with the surface words being either similar or different, as in (2.10).

Example 2.9. Surface paraphrases.

(a) The student is smart.

(b) The pupil is clever.

Example 2.10. Structural paraphrases.

(a) He is brave.

(b) Is he brave?

2.3.4 Sources of Paraphrases

Before examining previous methods of recognising paraphrases, it is important to discuss the sources of paraphrases, given that occasionally the choice of a certain method depends on the type of these sources. Ho et al. (2014) indicate that it is possible to divide the source of paraphrases into two sorts, specifically, the *lexical* source and the *corpora* source. Examples of the lexical source include thesauri, dictionaries and lexical databases, such as WordNet. As far as the corpora are concerned, they can be divided into the following types based on the sorts of data that have been used for paraphrasing:

1. A **Free Corpus** is a collection of texts in which the correspondence between texts is unknown where this uncertainty is specified by the term free. Word Wide Web (WWW) is an example of free corpora (Ringlstetter et al. 2006; Zhao et al. 2007; Keshtkar and Inkpen 2010).

2. A **Comparable Corpus** is a collection of texts describing the same event or topic in such corpora. An instance of Comparable corpora are newspapers, various reports discussing similar event, and dictionaries that contain words with multiple version of definitions. Examples of studies using such corpora to extract paraphrases are Shinyama and Sekine (2003); Barzilay and Lee (2003); Brockett and Dolan (2005); Regneri et al. (2014); Bar and Dershowitz (2012); Bar (2013).

3. A **Monolingual Parallel Corpus** is a collection of texts in the same language that serves as a mutual translation of the source text written in a different language. Examples of studies using such corpora to extract paraphrases are Barzilay and McKeown (2001); Ibrahim et al. (2003); Pang et al. (2003).

4. A **Bilingual Parallel Corpus** is a collection of texts written in a certain language, which appear in conjunction with their translation in another language. Examples of studies using such corpora to extract paraphrases are Bannard and Callison-Burch (2005); Madnani and Dorr (2010); Madnani et al. (2007); Cohn et al. (2008); Zhao et al. (2009b).

Comparable corpora are different from free corpora. The texts of newspapers are regularly written by professional writers, including publishers and editors. This implies that such corpora are cleaner than the free corpora. Additionally, it is often possible to detect paraphrases in these corpora as different articles reporting the same fact or event on the same day and time can commonly be discovered in newspapers. This makes such corpora particularly attractive when searching for paraphrases – different articles reporting the same event are extremely likely to contain sentences whose meaning are almost identical.

Monolingual parallel corpora have the advantage that pairs of sentences represent different patterns of the same meaning are put forward by different translators regarding the source text, and hence are paraphrases. To put it differently, there are in such corpora pairs of sentences which have either semantic equivalence or considerable semantic overlap. Consequently, paraphrases can be extracted by means of extracting correspondences from sentences that have either the same or a similar meaning. They are thus even more likely to contain paraphrases than comparable corpora, but they are less readily available. Zhao et al. (2009a, b) further indicate that in the inter-language process, there is no guarantee that the target language will include a translation of all the source language words or phrases, making the problem of word alignment worse as a result. This affects the quality of the extracted paraphrases, as they will have deteriorated. Moreover, the interpretations and translation may possibly vary since different authors have different writing styles. An additional significant point to mention is that these corpora suffer from the problem of scarce availability, for the reason that it is not logical to translate a foreign language text multiple times when there is a translation already.

Regarding bilingual parallel corpora, they are similar to monolingual parallel corpora, except that the former include the source texts. In addition, these corpora are more available than monolingual parallel corpora given that it is not necessary to have multiple translations of the source text, in order to build bilingual parallel corpora. Even one translation would be sufficient for that purpose.

Moreover, Ho et al. (2011) indicate that the use of such corpora is beneficial seeing that there is an exact semantic equivalence between sentences in the other language and those in the intended paraphrasing language. Consequently, generating paraphrases with such corpora is achieved by the use of the alignment of phrases across the two languages and by considering all of the co-aligned phrases in the intended paraphrasing languages to be paraphrases.

As mentioned above, the lexical source is the one which is used the least to extract paraphrases. However, according to Ho et al. (2011) there are five reasons why one should collect paraphrase candidates that use several instances from online dictionaries rather than corpora to extract paraphrases:

- To take advantage of the similarity characteristic of synonyms with regards to meaning.
- To capitalise on the candidates' correctness, on the basis of the belief that synonyms provided by them which are usually man-made products are less likely to contain erroneous paraphrases.
- To exploit the available lexical resources.
- To take advantage of the feature of lexical resources that are up-to-date.
- To capitalise on the quantity without sacrificing quality.

The current research focuses on comparable corpora for the reason that correspondence exists between comparable texts at a high level (i.e. different articles reporting the same topic), which are liable to comprise paraphrase fragments.

In summary, this section highlighted TEQV taxonomy which consists of the following: the level of paraphrasing that leads to the same meaning might be as words, phrases and sentences. Afterwards, we discussed the type of corpus with regards to TEQV. In the next section, we will discuss the research undertaken on previous methods in this field.

2.3.5 Applications of Paraphrasing

Studies related to paraphrases play an increasingly important role in NLP. Ho et al. (2014) indicate that such studies are important in the following areas:

1. **Document Summarisation** (Jusoh et al. 2011; Tatar et al. 2009; Lloret et al. 2008; Barzilay et al. 1999). This is due to the fact that sentential paraphrases are extracted by first getting rid of the words and phrases that are unnecessary and then by replacing the original phrases by either equivalent shorter phrases or by single words.

2. **Information Retrieval** (IR) (Clinchant et al. 2006; Parapar et al. 2005; Riezler et al. 2007; Zukerman et al. 2002). There is a generation of query paraphrases from the source query that the user enters via the replacement of some of its content words. It is possible to use all the queries in order to retrieve documents whose quality is better in terms of coverage and relevance.

3. **Information Extraction** (IE) (Kouylekov 2006; Romano et al. 2006). When a query about a specific event or person is submitted by a user, the relevant documents are retrieved, and from these documents a set of patterns is extracted. Afterwards, semantically equivalent patterns identified through the use of a paraphrase table become employed in the extraction of the relevant information.

4. **Question Answering** (Q&A) (Duboue and Chu-Carroll 2006; Negri and Kouylekov 2009; Celikyilmaz et al. 2009; Heilman and Smith 2010; Ou and Zhu 2011). When there is a submission of a question by a user to a Q&A system, and when such a question is actually absent from its database, the question paraphrases will be identified by the system which will then attempt to return to the user the answer to the question paraphrases if any can indeed be found

5. **Text-to-Speech generation** (Kaji and Kurohashi 2005). Speech generated directly from a written text can sound odd due to the fact that written language is different from spoken language. On the basis of a paraphrase table built through the use of both written and spoken languages, it is possible to generate more natural sounding speech through paraphrasing written text into spoken language.

6. **Plagiarism Detection** (Uzuner et al. 2005; Burrows et al. 2013). To identify plagiarisms through multiple documents.

2.4 Previous methods

Previous approaches fall into three groups: lexical-based approaches, syntactic-based approaches and hybrid-based approaches. In what follows we briefly describe all of them and provide supporting references.

2.4.1 Lexical-based approaches

A lexical-based approach uses two features to represent the context of a word or phrase: the surrounding words and named entities.

Surrounding words: As defined by Ho et al. (2014), surrounding word refer to any number of contiguous words that appear at both the left and right sides of a given word or phrase. Surrounding words are also known as N-grams where N is the number of the neighbouring words.

A general approach is to collect distinct words that are surrounded by common words in different sentences as the candidates, and then to validate those candidates against their frequencies.

Several methods that rely on the use of surrounding words to extract paraphrases from different corpora (e.g. comparable and monolingual parallel corpora) are developed in (Murata et al. 2005; Shimohata and Sumita 2002; Wang et al. 2009).

A similar approach was developed by Grigonytė et al. (2010) for free corpora and Bannard and Callison-Burch (2005) for bilingual parallel corpora in which sentences are first aligned by the so-called Sumo metric (a metric measures lexical strength between sentences based on the number of overlapping characters), and then compared as in (Murata et al. 2005; Shimohata and Sumita 2002; Wang et al. 2009). However the drawback here is that the Sumo metric relies too much on the presence of overlapping characters and therefore a large number of paraphrases get discarded. Between the two approaches, the one of Bannard and Callison-Burch is able to extract a greater number of paraphrases from the bilingual parallel corpora, however many of those paraphrases are not linguistically meaningful.

Overall, the methods employing surrounding words techniques require a high degree of presence of common words between sentences in order to extract paraphrases, which is a very limiting factor. It was found by Herrera et al. (2007) that methods similar to the ones mentioned above have unsatisfying results when applied to comparable corpora, therefore the surrounding words approach may

work only with monolingual parallel corpora.

The primary drawback of using N-grams is that they neglect the syntactic and semantic relations. For instance, the two sentences *'John knows Mary'* and *'John knows Mary is very nice'* both contain the bigrams *'John knows'* and *'knows Mary'*. Using the first of these bigrams to suggest that these two sentences are similar makes sense, but using the second bigram would be misleading because it does not capture the fact that *'Mary'* in the first sentence directly related to *'knows'* as its object, whereas in the second sentence the relationship between *'Mary'* and *'knows'* is much more indirect.

Named entities (NEs): As defined in Ho et al. (2014), NEs are proper noun expressions which represent the name of a person, an organization, location, date or number (Shinyama et al. 2002).

So far, four NE-based methods have been proposed: (Bhagat and Ravichandran 2008; Hasegawa et al. 2005; Sekine 2005) and (Bhagat et al. 2009). The first three methods use NEs in pairs as a representation of context, whereas the one proposed by Bhagat et al. (2009) makes use of NEs that occur at one side only.

These methods use one of the two techniques to assign weight to NEs, namely Pointwise Mutual Information/Association (PMI) (Bhagat and Ravichandran 2008) and Term Frequencey and Inverse Document Frequenct (TF-IDF) (Hasegawa et al. 2005; Sekine 2005), and then treat the weighted NEs as context vectors. The similarity between these vectors is measured by cosine similarity and those with high similarity scores are then grouped as paraphrases.

A limitation of this method is that larger sentences are more likely to introduce noise, and therefore only shorter sentences are appropriate to these methods. Furthermore, Azmi-Murad and Martin (2004) argue that TF-IDF is not a good indicator of semantic similarity since it only counts the number of co-occurring words without taking into account any synonymous relations or syntactic information.

2.4.2 Syntactic-based approaches

Syntactic-based approaches analyse texts at the syntactic level by converting pairs of sentences into syntactic trees by using grammatical parsing (dependency is the most widely used framework for this) to extract paraphrases.

The most common dependency relations that are used are the SUBJ-OBJ (subject-object) relations. The SUBJ-OBJ relations are dependent on verbs,

hence approaches employing them are limited to extracting verbal paraphrases only. For example, in the sentence *'John likes Mary'* , where *'likes'* is the root word, *'John'* is the subject of *'likes'* and *'Mary'* is the object of *'likes'*.

Approaches that extract paraphrases by using SUBJ-OBJ relations have been developed by (Connor and Roth 2007; Glickman and Dagan 2003; Lin and Pantel 2001; Shinyama and Sekine 2003; Shinyama et al. 2002). The first three of these use free corpora as their resources. In these approaches first all the sentences are parsed and then all verbs that are connected to a pair of subject and object are collected as candidates. Then, the approach of Lin and Patel uses association measures to extract paraphrases by determining the strength of associations of sentences with groups of similar subjects and objects, while the approach of Glickman and Dagan uses a strict matching technique in which two candidates are matched as paraphrases if and only if they are connected to the same pairs of subject and object, and the corresponding sentences share a number of common words.

The remaining two approaches by Shinyama and Sekine (2003); Shinyama et al. (2002) use comparable corpora as their resources. In this approach a topic detection technique is used which aligns all the articles and sentences, and then the sentences are tagged by using an extended list of NEs. The candidates whose SUBJ-OBJ relations are connected to identical pairs of NEs are then extracted as paraphrases. It is worth to note that Wu and Zhou (2003) have developed a method for extracting paraphrases that is not dependent only on the SUBJ-OBJ relations.

Two approaches that extract paraphrases from bilingual parallel corpora without relying on the use of dependency relations have been developed by Callison-Burch (2008) and Zhao et al. (2008). Zhao et al. (2009b) subsequently refined their method. These methods make use of syntactic labels instead.

Barzilay and McKeown (2001) have developed an approach that extracts paraphrases from monolingual parallel corpora by extracting patterns of identical words in parallel sentences. Ibrahim et al. (2003) have combined this approach with the one of (Lin and Pantel 2001); in their approach, instead of all identical words in parallel sentences, only identical nouns and pronouns are considered. This way more phrasal paraphrases can be obtained, of which verbal paraphrases will be a majority.

Similar methods where, instead of a small number of features, a large number

of features is used to extract paraphrases have been developed by (Zhao et al. 2010; Keshtkar and Inkpen 2010; Hashimoto et al. 2011). Zhao et al. make use of overlapping characters or words, lengths of phrases and sentences, N-grams and NEs among some other features, with 8 features in total. Keshtkar and Inkpen make use of the lengths of sequences, numbers of surrounding verbs, nouns, adjectives and adverbs and so on, with 18 features in total. Hashimoto et al. use N-grams, dependency relations, the number of morphemes and so on, with 17 features in total.

An approach that extracts paraphrases by computing edit distances between syntactic trees was developed by Zhang and Shasha who generalised Wagner and Fischers edit distance from sentences to trees ((Zhang and Shasha 1989); (Dulucq and Tichit 2003)): given two trees $T1$ and $T2$, their distance is computed as the least cost sequence of insert, delete and exchane operations needed to transform $T1$ into $T2$.

Wu (2010) has developed a general framework for considering alignment, including tree alignment. Further work on tree edit distances was done by (Vila and Dras 2012) who used dependency trees and the tree edit distance between them, i.e. the number of insert, delete and substitute operations needed to convert one tree into the other, as a paraphrase representation baseline.

Only a few studies in the entailment literature are concerned with Arabic. Alabbas (2011) developed the ArbTE system for assessing existing TED techniques and proposed to extend the tree edit distance with subtrees so to obtain a more flexible matching algorithm for identifying TE in Arabic (Alabbas and Ramsay 2013), which measures the distance between two trees by applying operations to subtrees rather than to single nodes.

2.4.3 Hybrid-based approaches

Hybrid-based approaches combine two different approaches, very often syntactic and lexical. Approaches such as: Hearst (1992) extracted paraphrases from free corpora, and Pasca and Dienes (2005) extracted paraphrases from monolingual corpora. Both used lexical-syntactic approaches.

Hearst uses three initial lexical-syntactic patterns that are created manually by an observation of the texts which are then used to extract paraphrases. These paraphrases, together with their contexts, are then used to extract more patterns. The drawback of this method, however, is that it is very expensive as every step

has to be performed manually, and therefore consumes more time to execute. In contrast, Pasca and Dienes did not apply any initial patterns but instead apply some heuristic rules to remove noise from sentences that are collected from the top search results of a search engine. Then, they use several combinations of features to extract paraphrases such as: (i) N-grams, (ii) N-grams and NEs, where an anchor contains the preceding and following NE that are close to the N-gram, (iii) N-gram-Relative, same as the second combination, but the anchor contains an adverbial clause that links the N-grams to their NEs. These combinations are then used to create the lexical-syntactic patterns.

Deléger and Zweigenbaum (2008) tried to extract 'lay-technical French paraphrases' from comparable corpora rather than extracting 'lay-lay English paraphrases', where 'lay' indicates common vocabulary, whereas 'technical' indicates jargon vocabulary. To detect and match comparable texts they applied topic segmentation and cosine similarity. Then, under the assumption that technical articles contain more nouns and lay articles contain more verbs, 'lay-technical' pairings were obtained by pairing deverbal nouns with verbs according to a set of predefined matches. Such pairs, together with their contexts, were then used to create more lexical-syntactic patterns with which more paraphrases can be extracted. In Deléger and Zweigenbaum (2009) an improved method was introduced in which paraphrases can be extracted from specific sorts of technical terms known as 'neo-classical' compounds (words arising from Greek and Latin) by first decomposing these compounds into words and their corresponding definitions with the 'DriF' parser, and then using any content words contained in the definitions to match their equivalents in the lay language. This method was developed to extract English paraphrases in (Deléger and Zweigenbaum 2010). All these methods developed by Deleger and Zweigenbaum have a common disadvantage in that only a fixed number of patterns is used to extract paraphrases and therefore only limited type of paraphrases can be extracted.

On the other hand, hybrid approaches can include different combination of lexical, syntactic, and semantic relation. The semantic relations that have been used for this task are WordNet Hierarchical Semantic Structure.

Liu et al. (2007) presented a method to measure the semantic similarity between sentences by using the Dynamic Time Warping (DTW) technique (we will discuss this technique in more details in Chapter 4). They took into account the semantic information, word order and the Parts of Speech (POS) in a sentence. In

addition, Islam and Inkpen (2008) developed a method to combine three similarity functions (e.g. string, semantic and common word order) with normalization so to calculate the semantic similarity between two texts. They modified the Longest Common Subsequence (LCS) measure by taking into account the length of the string. This method is known as the Semantic Text Similarity (STS) method.

Table 2.3 shows the range of approaches in paraphrasing that have been used in the literature classified in terms of corpus type, level of paraphrases, and type of similarity.

Corpus type	Level of paraphrase	Type of similiarity	Method	Author
Comparable corpora	Lexical	Lexical + Syntactic	Dependency relations, named entities	Shinyama et al. (2002)
	Lexical	Hybrid	N-gram, named entities, syntactic	Pasca and Dienes (2005)
	Lexical-level	Lexical + Syntactic	Dependency relations, N-gram	Dagan et al. (2006b)
	Phrase	Syntactic	Dependency relations, named entities	Shinyama and Sekine (2003)
	Sentences	Lexical + Syntactic	Multiple-Sequence Alignment	Barzilay and Lee (2003)
	Sentences	Lexical + Semantic	Support vector machine-based classifier	Brockett and Dolan (2005)
	Phrases	Hybrid	Multiple-Sequence Alignment	Deléger and Zweigenbaum (2008)
	Lexical	Lexical + Semantic	Synonym, semantic relations	Ho et al. (2011)
	Phrases	Lexical	N-gram	Wang and Callison-Burch (2011)
	Lexical & phrases	Hybrid	Syntactic parser, N-gram, named entities	Bar (2013)
	Sentences	Syntactic + Semantic	Syntactic parse, Dependency trees	Alabbas (2013b)
	Sentences	Hybrid	Semantic rules	Regneri et al. (2014)
Monolingual parallel corpora	Phrases	Lexical + Syntactic	Dependency relations, N-grams	Ibrahim et al. (2003)
	Phrases & sentences	Lexical + Syntactic	Syntactic parser, N-gram	Barzilay and McKeown (2001)
	Sentences	Syntactic + Semantic	Syntactic parser, synonyms	Pang et al. (2003)
	Sentences	Lexical + Syntactic	Dependency relations, N-grams	Glickman and Dagan (2003)
	Phrases	Syntactic	Dependency relations, N-grams	Lin and Pantel (2001)
	Sentences	Hybrid	DTW technique	Liu et al. (2007)
	Phrases & Sentences	Lexical	Named entities	Bhagat and Ravichandran (2008)
Biligual parallel corpora	Sentences	Lexical + Syntactic	Syntactic parser, alignment	Callison-Burch (2008)
	Sentences	Lexical	SMT techniques	Riezler et al. (2007)
	Sentences	Lexical + Syntactic	Syntactic parser, alignment	Zhao et al. (2010)
	Sentences	Lexical + Semantic	statistical MT technique	Madnani et al. (2007)
	Phrases & lexical	Lexical	N-gram, alignment	Bannard and Callison-Burch (2005)
	Phrases	Lexical + syntactic	Dependency relations, Alignment	Hwang et al. (2008)

Table 2.3: Classification of systems by corpus type, paraphrase levels and similarity measure

In the work described in the remainder of this thesis, we investigate the possibility of using techniques that have been shown to work well for discovering paraphrases in English texts for doing the same with Arabic texts. This means that the relevant infrastructure has to be available, and in particular it makes it difficult to investigate the applicability of approaches based on dependency parsing: the state-of-the-art in dependency parsing for Arabic is some way behind that for English, and corpora for training Arabic parsers are less extensive and less easy to obtain than for English (the Penn Arabic Treebank is the best known such corpus, but it consists of less than two hundred thousand words, and is in any case made up of phrase structure trees which have to be converted to dependency format and appropriately labelled before they can be used for training a dependency parser). Alabbas and Ramsay (2012a) report around 80% accuracy on unlabelled data using a combination of MST and MALT, which is not accurate enough to be used for the current task. We use comparable corpora (group 1) in Table 2.3, extracted by collecting news articles from Arabic and English news feeds, linking them by using TF-IDF scores and cosine similarity and then linking pairs of sentences from within linked articles by the same method but with a tighter threshold to look for sentences that are likely to contain matching phrases. We then look for lexical and phrasal matches, using an extended alignment algorithm that pays attention to word similarity on the basis of WordNet similarity relations and that also allows a limited degree of permutation. In terms of Table 2.3 the most closely related work is highlighted in blue: we are attempting to fill the spaces occupied by Bar (2013), Bannard and Callison-Burch (2005) and Hwang et al. (2008), using dictionary-like information as suggested by Ho et al. (2011) to look for sequences of similar words, where we make the very rough working assumption that if two sequences of words have very similar meanings then they may well constitute substitutable phrases.

Chapter 3

The Challenges of the Arabic Language

3.1 Introduction

As mentioned in the introductory chapter, Arabic has a complex structure, which creates a lot of ambiguities in the language. Since we have conducted some of the experiments in this study on Arabic, and because Arabic differs in a number of ways from other languages such as English, we dedicate this chapter to describing some of the structural complexities of Arabic, which might affect the performance of various TE algorithms. (Chalabi 2004) and (Daimi 2001) all argue that there are a lot of complexities in Arabic that are not present in other languages, while Holes (2004) states that Arabic has a complex syntactic structure. In this chapter, we will briefly highlight some of the complexities of Arabic which may make processing Arabic text difficult, and then we present the structure of Arabic words.

3.2 Arabic ambiguity sources

There are various sources of ambiguity in Arabic related to its properties, particularly the written form of this language, which are discussed in some detail below.

3.2.1 Lack of diacritic marks

It is optional to write Arabic with diacritics[1], which consist of short vowels and some other phonological effects. These diacritics are usually absent, hence causing various ambiguities (Nelken and Shieber 2005). These diacritics are the only source of differentiating between different words and their inflected forms. This results in making the analysis of the morphological structure of the language very challenging. This is due to the fact that it is possible to interpret a single lexeme in Arabic in various ways. Accordingly, it is possible to have a single word with various meaning that could be determined on the basis of the context of the word. Besides, it is possible to interpret a noun in Arabic in three different ways with respect to the nominative, accusative and genitive cases, causing extra ambiguities at the structural or grammatical level.

Examples of the effect of diacriticisations on the meaning of the word are given below. In the Example (3.1), there are different diacriticisations that distinguish between a noun and a verb, in (3.2) they distinguish between active and passive, in (3.3) they distinguish between the declarative and the imperative, in (3.4) they distinguish between various gender and person differences, and in (3.5) there is a duplication of the middle letter of the verb to make it transitive.

Example 3.1. A surface form that would be recognisable as either a verb or a noun if it were diacriticised.

درس	drs		
Verb:	دَرَسَ	darasa	‘study’
Noun:	دَرّس	dars	‘lesson’

Example 3.2. A surface form that would be recognisable as either an active or a passive if it were diacriticised.

رسم	rsm		
Active:	رَسَمَ	rasama	‘drew’
Passive:	رُسِمَ	rusima	‘was drawn’

Example 3.3. A surface form that would be recognisable as either an imperative or a declarative if it were diacriticised.

[1]In Arabic special symbols called حركات HrkAt “diacritical marks” can be added to help reading language. Some of these symbols are put above Arabic characters (e.g., Damma ضمة, Fatha فتحة, Sukun سكون, Dammatan تنوين ضم, Fathatan تنوين فتح), and some of them are put under Arabic characters (e.g., Kasra كسرة, Kasratan تنوين كسر).

انتبه	Antbh		
Imperative:	اِنَتبِه	Antabih	pay attention
Declarative:	انتَبَهَ	Antabaha	paid attention

Example 3.4. A surface form that would be recognisable as either a various gender or person if it were diacriticised.

استمعت	AstmEt		
اَستَمَعتُ	AstamaEtu	(1st.sg.)	'I listened'
اَستَمَعتَ	AstamaEta	(2nd.masc.sg.)	'You listened'
اَستَمَعتِ	AstamaEti	(2nd.fem.sg.)	'You listened'
اَستَمَعت	AstamaEt	(3rd.fem.sg.)	'She listened'

Example 3.5. A surface form that would be recognisable as either an intransitive verb or transitive (causative) through duplication of the middle letter of the verb if it were diacriticised.

وصل	wSl	
وَصَلَ	waSala	'arrived'
وَصَّلَ	waS~ala	'connect'

Figure 3.1 presents the Arabic word علم Elm, without diacritics, and how the addition of diacritic marks gives seven different readings. Hence there is a great deal of ambiguity as to which of these readings is intended in the absence of diacritics.

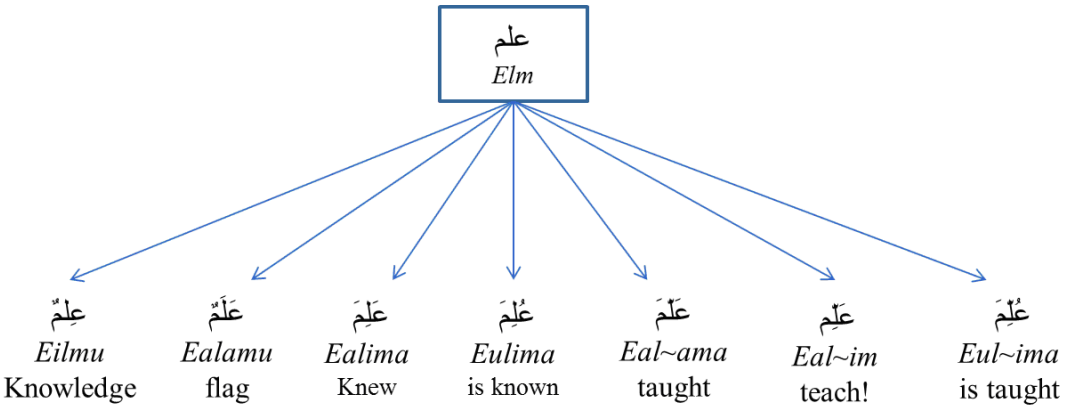


Figure 3.1: Ambiguity caused by the lack of diacritics.

3.2.2 Free word order

Arabic has a high degree of syntactic flexibility (Daimi 2001). It has to a certain extent a free word order since the components of a sentence could be exchanged without any effect on the core meaning of this sentence. This is one of the sources of ambiguities in Arabic (Attia 2012). As a result, in addition to the regular sentences that have the verb-subject-object (VSO) constructions, it is possible to have in Arabic the verb-object-subject (VOS), subject-verb-object (SVO) and object-verb-subject (OVS) constructions. Sometimes different constructions lead to a large amount of ambiguity. Examples of such constructions are given in example (3.6).

Example 3.6. Word order (distinguishing between subject and object).

a)	Arabic:	قتل الجندي المجرم	(VNN)
	BW[2]:	Almujrem Aljundy ktl	
	English gloss:	the-criminal the-soldier killed	
b)	Arabic:	قَتَلَ الجنديُ المجرمَ	(VSO)
	BW:	Almujrema Aljundyu ktl	
	English gloss:	the-criminal(OBJ) the-soldier(SUBJ) killed	
	Translation:	The soldier killed the criminal	
c)	Arabic:	قَتَلَ الجنديَ المجرمُ	(VOS)
	BW:	Almujremu Aljundya ktl	
	English gloss:	the-criminal(SUBJ) the-soldier(OBJ) killed	
	Translation:	The criminal killed the soldier	

In example (3.6) (a) where there is no visible case marking (because there are no diacritics) we have a V followed by two Ns (VNN), but it is difficult to identify which order it is (VSO or VOS) i.e., it is difficult to decide the subject and the object in the sentence. The default order is VSO, and hence in the absence of any other information this would be the most likely interpretation. However, if the diacritics are supplied then they will include case markers which can be used to distinguish between the two readings. In Example (b), *'the soldier'* is in the nominative case, and *'the criminal'* is in the accusative, so the allocation of roles is indeed the default VSO order. In (c) *'the soldier'* is in the accusative case while

[2]Buckwalter transliteration, `http://www.qamus.org/transliteration.htm`

'the criminal' is in the nominative, so the sentence must have the non-standard VOS order.

3.2.3 Zero copula

A copula is defined as a verb connecting the predicate of the sentence with the subject. Arab grammarians point out that there are two kinds of Arabic sentences, namely, a nominal sentence starting with a noun, and a verbal sentence starting with a verb. Besides, there is a special kind of a nominal sentence called an *equational sentence*. This kind of sentence has no verb, and it is made up of two main parts. The first part is called المبتدإ Al+mbtdÂ "the subject", made up of a noun phrase (NP), and the second part is called الخبر Al+xbr "the predicate", which could be made up of a NP, an adjective phrase (ADJ), a complement phrase (CP), or an adverb phrase (ADV). Such a sentence usually begins with a definite noun, which is the subject, followed by the predicate. Examples of equational sentences are given in (3.7).

Example 3.7. Arabic equational sentences.

a)	Arabic:	الرجل مهندس	(NP predicate)
	BW:	muhandes Alrjl	
	English gloss:	engineer The-man	
	Translation:	The man (is) (an) engineer	
b)	Arabic:	الطبيب مخلص	(ADJ predicate)
	BW:	mxlS Altabyb	
	English gloss:	honest The-doctor	
	Translation:	The doctor (is) honest	
c)	Arabic:	المعلم في الصف	(PP predicate)
	BW:	AlSf fi AlmElm	
	English gloss:	the-classroom in the- teacher	
	Translation:	The teacher (is) in the classroom	

Under certain constraints, the standard subject-predicate order of equational sentences is reversed, as is the case when the subject is indefinite, as shown in example (3.8).

Example 3.8. An indefinite subject following a predicate phrase.

Arabic:	في الصف كتاب	(PP predicate)
BW:	kitab AlSf fy	
English gloss:	book the-classroom In	
Translation:	In the classroom (is) book	

The above appears to be an example where the copula is omitted. This only happens in the case of present tense affirmative sentences. In the case of past, future or present tense negative sentences, the verbs كان kan 'to be' and ليس lys 'be not' are used, making the first noun, i.e., the subject, in the nominative case and the second noun, i.e., the predicate, in the accusative case.

3.2.4 Arabic clitics

Clitics are defined as morphemes that have the syntactic properties of a word but are at the same time bound to other words (Attia 2012). Arabic is a clitic language. It has a large number of clitic items, be they prepositions, pronouns, or conjunctions. This leads to a difficulty in determining which items are actually present. For instance, in (3.9) it is possible to analyse the word 'والى wAly' into five different forms, with each form having a different decomposition into lexemes (Salloum and Habash 2011). As a result, combining three Arabic words results in more than two hundred different meanings.

Example 3.9. Numerous clitic items.

والى	wAly'	
والي	wAly	'ruler'
و + الى + ي	w+Aly'+y	'and to me'
و + اَلى + ي	w+ Āl+y	'and my clan'
و + اَلى	w+Āly	'and automatic'
و + أى	w+Âly	'and I follow'

3.2.5 Arabic pro-drop

Arabic is also characterized as being a pro-drop language. The pro-drop theory states that "a null class (pro) is permitted in a finite clause subject place if the agreement features on the verb are rich enough to enable its content to be recovered" (**?**)). The pro-drop is referred to as Al+Dmyr Al+msttr "tacit pronoun". A great amount of structural ambiguity is caused by the pro-drop since

the syntactic parser needs to determine whether there is in the subject position a dropped pronoun or not. This situation becomes worse by the fact that it is possible for many Arabic verbs to have both transitive and intransitive forms, or ditransitive and transitive forms, or even all three of these three forms. What further complicates the situation is the fact that it is impossible in general to distinguish the active form from the passive one through the inspection of the surface form. If one of these verbs is followed by only one NP, ambiguity emerges. For example, there are three different interpretations for the Arabic sentence in (3.10).

Example 3.10. Arabic pro-drop.

Arabic:	أَكَلت التُفَاحة
BW:	AltufaHa ¿kalat
English gloss:	the-apple ate(fem.)
Translation:	The apple ate, (she) ate the apple or The apple was eaten

The ambiguities in the example (3.10) arise from three different types of structural analysis. First, the verb أَكَلَ Âakala 'to eat' could be either transitive or intransitive, hence the meaning is أَكَلت التُفَاحة ¿kalat AltufaHa. Second, a pro-drop subject is potentially present هي hy 'she' that is inferred from the feminine marker of the verb, meaning accordingly أكلت (هي) التفاحة Akalat (hy) Al+tufaHa '(She) ate the apple'. Third, it could mean 'The apple was eaten', where the verb 'eaten' أُكِلت is a passive transitive verb.

3.2.6 Construct phrases

It is possible to use nouns as adjectival modifiers (i.e, in noun-noun compounds), or as possessive determiners, forming what is called "construct phrases", "genitive constructs", or "annexation structures", typically marked with little inflectional morphology (Alabbas and Ramsay 2011). Ryding (2005) pointed out that *"in Arabic, two nouns may be linked together in a relationship where the second noun determines the first by identifying, limiting, or defining it, and thus the two nouns function as one phrase or syntactic unit"*. Furthermore, it is possible to link possessive uses of Arabic nouns with no clear markers, in contrast to English in which such uses are joined together by means of different markers, such as the -s suffix on the possession noun or the possessive phrase 'of'. It is essential to note

that this is a problem only in the case of written Arabic because in reading Arabic, case markers tend to be pronounced in such cases, distinguishing hence the role of each noun. The construct phrase is referred to by Arab grammarians as Idafa, إضافة 'annexation'. In this phrase, the first noun, referred to as مضاف mDaf 'the added', should be indefinite. It could be in any case, i.e., nominative, genitive or accusative, and does not take the nunation (Schulz et al. 2000). Besides, this noun will have the case marker for the definite form, despite not having a definite article. As for the second noun, referred to as مضاف اليه mDAf Alyh 'annexing noun or amplifying noun', which is in what is called 'the construct state', it could be either definite or indefinite, and is always in the genitive case. In (3.11), some examples of the construct phrase are given; while in (3.12) examples of a noun used as an adjective are given.

Example 3.11. Construct phrases (Idafa).

a) Arabic: وزير التعليمِ
BW: AltElim Wazir
English gloss: the-education minister
Translation: the minister for education

b) Arabic: أم سامرٍ
BW: samer ¿m
English gloss: samer um
Translation: Samer's mother

Example 3.12. Noun as an adjective.

Arabic: ساعة يد
BW: sAEt yd
English gloss: watch hand
Translation: wristwatch

It should be noted that an NP could form the second part of a different construct phrase. It is possible to extend this recursively, leading to the creation of an Idafa chain, in which all the words must be genitive, except for the first word, and all the words must be in the construct state, except for the last one. An example of an Idafa chain is given in (3.13).

Example 3.13. Idafa chain.

Arabic:	ابن عمة صديق رئيس مجلس إدارة الشركة
BW:	Alshrkeh Adart magles r}is Sadiq Emt Abn
English gloss:	the-company management committee chief friend uncle son
Translation:	The cousin of the CEO's friend

3.2.7 Noun multi-functionality (Coordination)

In Arabic, coordination is either syndetic, in that an explicit conjunction links terms, or asyndetic, in that terms are linked with no explicit conjunction. Syndetic conjunction is preferable, and very common. When this kind of coordination takes place, some linguistic units are omitted in one conjunctive or more, which are و+ w+ 'and', ف+ f+ 'and' and ثم θuma 'then', and are used in connecting words, phrases, clauses and simple sentences in order to generate complex or compound sentences. In (3.14), there is an illustration of an Arabic syndetic example.

Example 3.14. Syndetic coordination.
و+w+ 'and'

Arabic:	تزوج ماهر و هدى أمس
BW:	>mes huda wa maher tzwj
English gloss:	yesterday Huda and Maher Married
Translation:	Maher and Huda married yesterday

'Maher got married to Huda' or 'Maher got married and so did Huda'

In the case of the second potential meaning, i.e., 'Maher got married and so did Huda', there is verbal ellipsis, referring to the syntactically zero realization of the verb of the subsequent clause, which has a structurally parallel construction to the preceding clause and its meaning could be recovered from the preceding clause as well.

These phenomena cause significant problems when trying to parse Arabic. These phenomena occur in other languages as well.

Every single one of these problems occurs in English text, where English has pro-drop, free word order, zero copula, construct phrases and clitics, as shown in Example (3.15).

Example 3.15. Examples of pro-drop, free word order, zero copula, construct phrases and clitics in English.

- English pro-drop

  (∅)[3] Keep (∅) away from children.
  (**you**) Keep (**it**) away from children.

- English free word order

  On the bus sat an old man. (PP + V + S)
  An old man sat on the bus. (S + V + PP)

- English zero copula

  I think he is a fool.
  I think him (∅) a fool.

- English Clitics
  I want *some fruit* to eat.
  I want *something* to eat.

- Construct NPs

  I saw the man's friend.
  (In English has a possessive marker on the first NP, In Arabic the first NP would have a genitive case marker , but case markers are not written)

Arabic is difficult to handle not because of these problems on their own, but because they occur in combinations in Arabic text in a way where it is not easy to distinguish which phenomena are actually present. Particularly with undiacriticised text we cannot detect which one of the phenomena in that text is causing the problem. For example, if in Arabic text we have three consecutive nouns in combination, we may not know which phenomena is showing; it could be that two nouns are a zero copula sentence, a complex noun or a construct phrase and in any case it is hard to tell whether something is an indefinite noun or a verb.

3.3 The Structure of Arabic Words

Arabic is a highly inflected language. Arabic words are made up of a root, a template or a pattern, and a number of prefixes and suffixes. A single root may

[3]The symbol ∅ will show the position of the omitted pronoun.

consist of 3 or 4 consonants and can be used to form words of different meaning by varying the template. The template is a sequence of consonants and variables (long vowels) for root letters.

Words are marked for grammatical categories to represent the inflectional process on the words. Thus, Arabic grammar is classified into eight categories: tense, person, mood, number, voice, case, definiteness and gender (Ryding 2005); proclitics and enclitics are added to indicate definiteness, conjunction, various prepositions, and possessive forms.

An instructive example is given by the root 'k.t.b' and the template XaYaZa, where X, Y, and Z are variables (long vowels). The result is the verb `katab`, meaning 'wrote'. We can impose different grammatical roles for this word by inflecting it; for example '`kataba`'/ 'he wrote' is the inflected form of '`katab`' representing the 3rd person masculine singular of the perfect verb form. 'katabotu'/ '`I wrote`', is the inflected form of `katab` representing the 1st person of the perfect verb form.

By slightly altering the template into XaAYaZ (a variable (long vowel) A has been added before Y) we obtain the verb كاتب (`kaAtib`), 'he corresponded with'. The Figure 3.2 shows the important word structure as follows:

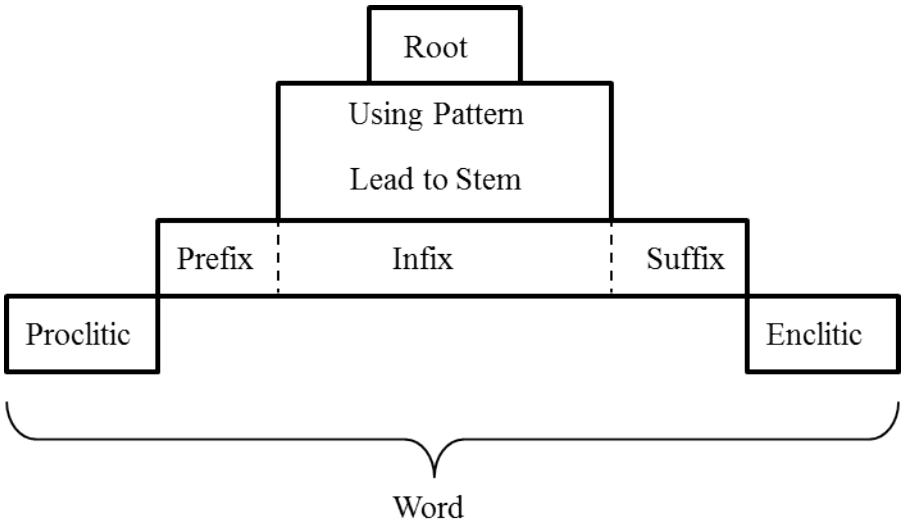


Figure 3.2: Word structure for Arabic.

The elements in Figure 3.2 are described below:

Root: the root of a word is the set of consonants that appear in most of its realisations. Most nouns and verbs have the same consonants in all their

inflected forms, but for so-called weak verbs and for some broken nouns this does not hold.

Pattern: sequence of vowels for filling in the gaps between the elements of the root.

Stem: of a written form of a word is the consonants of the root with the diacritics that occur in the gaps between these consonants. The stem does *not* include any diacritics that are part of any inflectional affixes

Affixes: each is a set of morphemes attached to the stem. It could be before the stem, prefix; within the stem, infix; or after the stem, suffix.

Clitics: morphemes that attach to the stem after affixes. They are categorized by whether they are placed in the beginning or the end of the word to be consecutively proclitic or enclitic. Proclitics include conjunctions and prepositions and enclitics are generally pronouns (Althobaiti et al. 2014).

Standard form: of a word is the diacriticised singular form of the word for nouns (masculine singular if it is a noun that can have both masculine and feminine forms) and the third singular masculine past active form of a verb.

The distinction between the stem and the standard form is fairly tricky. The critical issues are that a single word will have a number of alternative stems, since it can be written with different diacritics in different forms; and that the standard form for a verb will generally include a final short vowel, whereas a stem will *never* end with a vowel. In almost all cases, the consonants in the stem and standard form will be the same and will be the root, but this does not always hold for weak verbs and broken nouns. Below we apply the Arabic word structure on the word بقراراتهم, as shown in Figure 3.3.

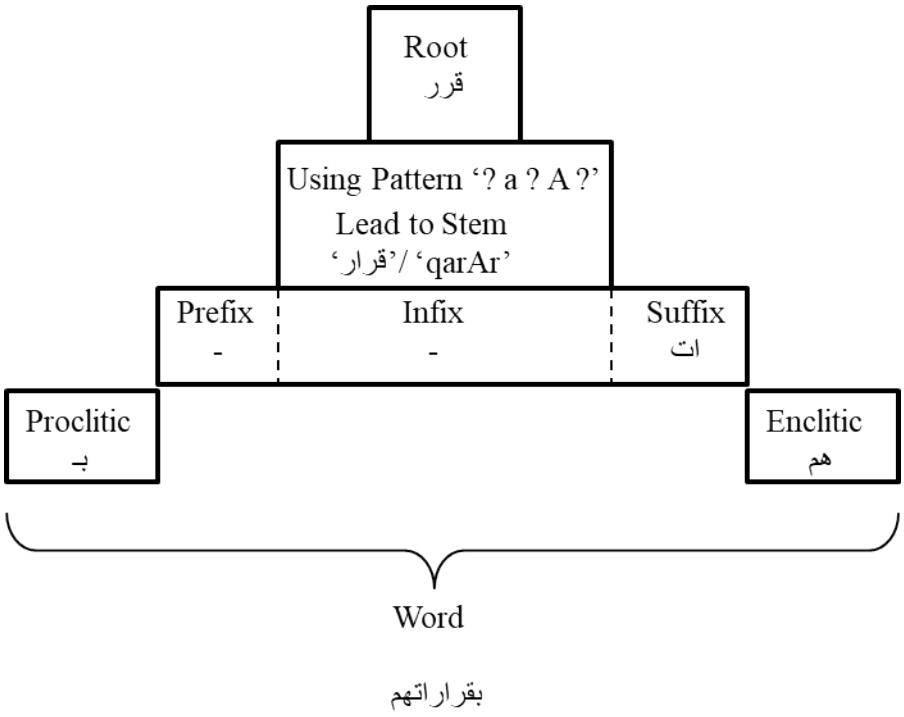


Figure 3.3: Word structure in Arabic language applied to بقراراتهم.

3.4 Summary

This chapter focused on the most prominent sources of ambiguities in Arabic that might affect the performance of Arabic systems. These ambiguities also exist in other languages such as English. However, the problem is not due to these ambiguities by themselves, but because in Arabic text these phenomena occur in combination in one sentence, and because the lack of diacritics in normal written text hides a numbers of markers that would help distinguish between different cases. Therefore, often we are not able to distinguish which one of these phenomena causes the problem.

Chapter 4

TEQV Architecture

4.1 Introduction

Textual equivalence TEQV or paraphrases are pairs of text fragments that convey the same meaning, while using different words, phrases or structures for expressing and describing the same concept. Textual equivalence plays an important role in many applications in NLP, such as QA, MT, IR, TS), and Information Extraction (IE), where these applications need to detect paraphrases in some of their operations.

This chapter focuses on describing an overall architecture for building a general TEQV system, which will be populated differently with two languages, English and Arabic. In order to compare the effectiveness of a set of common strategies across both of them, we need to apply them in settings which are as similar as possible. Figure 4.1 shows an abstract workflow; while a detailed discussion of applying this framework to both Arabic and English language will be the subject of Chapter 5.

The general components of this system consist of: data collection, pre-processing pipeline, data clustering, similarity judgment or checking, and finally performance evaluation.

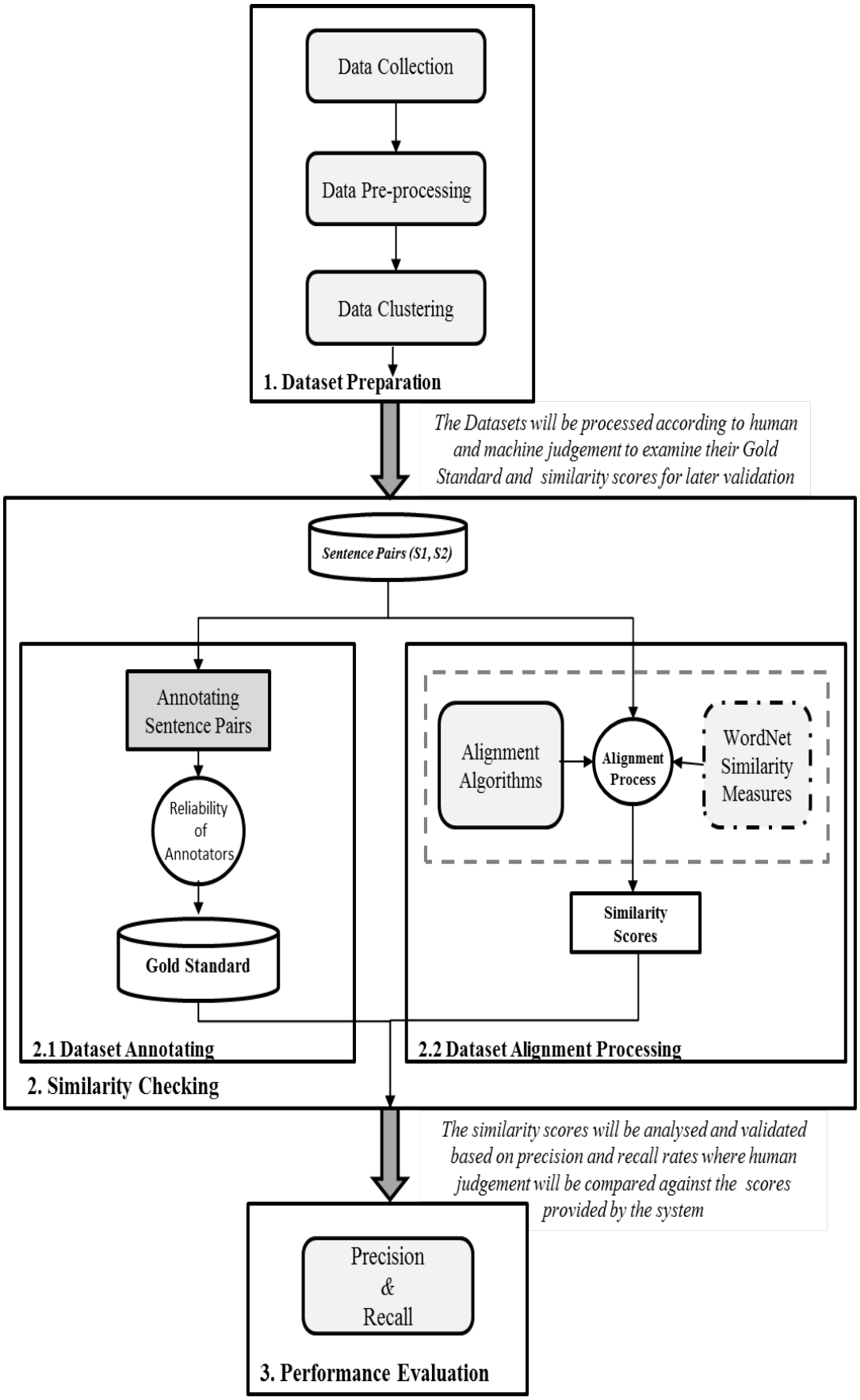


Figure 4.1: General Workflow for TEQV.

The data collection component is responsible for collecting the raw data (articles) from different newswire sources using online RSS feeds. Then, this data that has been collected will be normalised through a pre-processing pipeline, which involves three subtasks: Part of Speech (POS) tagging, sentence splitting, and morphological analysis. Following this, to produce pairs of sentences that might contain a candidate paraphrase, a clustering process is applied to cluster pairs of articles and then sentence pairs using simple standard similarity measures, namely cosine similarity and TF-IDF vector space. The next step will show how the dataset of sentence pairs is turned into a 'Gold Standard' by assessing the sentence pairs by human judgment using an online annotation form to annotate the dataset. To check the reliability of the annotators, an inter-annotator-reliability (Barrón-Cedeño et al. 2013) assessment will be used. Matching fragment sequences that occur in the same order between pairs of sentences are then found having the same meaning, by applying one of the string alignment methods (e.g., Dynamic Time Warping (DTW) introduced by Sakoe and Chiba (1978)) to transfer one sentence to another by calculating the minimum number of editing operations (Insert, Delete, and Exchange). Next, an enhanced string alignment method referred to as eXtended Dynamic Time Warping (XDTW) will be used to identify paraphrases in the fragment texts, which deals flexibly with adjacent items with free word order by adding a new edit operation (Swap). The DTW and XDTW approaches will be used to measure the degree of similarity between words using information available from WordNet, for instance, WordNet semantic similarity functions. Finally, an overall evaluation will be conducted to assess the effectiveness of these methods by comparing their outputs (similarity score) to the 'Gold Standard' using precision and recall rates.

4.2 TEQV Dataset Creation and Collection

In order to develop and evaluate a TEQV system for any language, an appropriate dataset is needed. A number of paraphrase datasets have been produced for different languages, such as English, Spanish, Japanese and Turkish. Table 4.1 shows a number of widely used English corpora[1]:

[1] `http://clic.ub.edu/corpus/en/paraphrases-en`

Authors	Corpus	Size	Number of labels and annotators
Dolan et al. (2005)	MSRP[2] (Microsoft paraphrase corpus)	5801	Annotated by 2 expert Annotators, 2 labels (equivalent, not equivalent)
Vila et al. (2010)	WRPA[3] (Wikipedia-based Rational Paraphrase Acquisition)	1000	Annotated by 2 annotators: native English speaker and native Spanish speaker
Barrón-Cedeño et al. (2013)	P4P (Paraphrase of Plagiarism)	847	Annotated by 3 experienced postgraduate linguists
Ganitkevitch et al. (2013)	PPDB[4] (Paraphrase Database)	220 million, 73m phrasal, 8m lexical, 140m paraphrase pattern	Annotated by Amazon Mechanical Turk[5], labelled from 15 where 54 is equivalent, 3 roughly equivalent, and 21 not equivalent)

Table 4.1: The existing corpus in English.

As noted, Table 4.1 contains the various corpora which differ in their types, sizes, labels and number of annotators. The key observation for Table 4.1 is that the MSRP, WRPA and P4P corpora are all significantly smaller in size than the PPDB. Therefore we split these corpora into two groups: small and large. The MSRP, WRPA and P4P corpora all belong to the small group, while the PPDP belongs to the large group. The MSRP and the P4P corpora were created manually from different news sources on the web, while the WRPA corpus was created automatically from Wikipedia. The PPDB corpus was created automatically as well.

There are notable differences between the two groups of corpora in terms of some features, namely: the cost, the size of datasets and the number of annotators; these features are listed in Table 4.2 along with the mention of the differences

[2] https://www.microsoft.com/en-us/download/details.aspx?id=52398

[3] http://www.talp.upc.edu/index.php/technology/resources/multilingual-lexicons-and-machine-translation-resources/multilingual-lexicons/178-wrpa

[4] http://www.cis.upenn.edu/~ccb/ppdb/

[5] https://www.mturk.com/mturk/

between them.

Feature	Small corpora	Large corpora
Cost	It is available for free.	It is available with extremely high cost.
Dataset Size	Modest: less than 10k words.	Huge: greater than 100m words.
Annotators	Small number of annotators to annotate a small set of carefully selected data, where it was possible to give carful guidance and to check on their reliability using inter-annotator-reliability.	A huge number of annotators recruited via Amazon Mechanical Turk, to annotate large number of self-selected data not necessary careful annotators. It is difficult to check on their reliability.

Table 4.2: Comparison between small and large corpora.

From our point of view, because the annotators for small corpora are well-trained and are easily monitored, it is likely that they will be more accurate and consistent.

In contrast, the Arabic paraphrasing corpora are limited by comparison, as listed in the Table 4.3:

Authors	Size	Number of labels and annotators
Denkowski et al. (2010)	728	Annotated via Amazon Mechanical Turk. 2 labels ('yes' or 'no')
Bar and Dershowitz (2014)	690	Annotated by two native Arabic speakers. 2 labels ('yes' or 'no')

Table 4.3: The existing corpora in Arabic.

It is worth noticing in Table 4.3 that the two corpora are small. Denkowski et al. (2010) corpus was extracted automatically from NIST Open MT 2002, and annotated using Amazon's Mechanical Turk. The corpus developed by Bar and Dershowitz (2014) was extracted automatically from Arabic Gigaword 4.0, and it was annotated manually by two native Arabic speakers. The main purpose of the two listed Arabic-based corpora is to improve the evaluation of an English-to-Arabic machine translation system. These corpora are not available for public access.

Because the Arabic corpora were collected in different conditions and by different mechanisms from the English one, we would not be able to use them even if they were publically available since we cannot be sure they are comparable. Therefore, we collected our own dataset.

The reason for using the same machinery for collecting the data in English and Arabic is to minimise the differences between the English and Arabic corpora, as explained in Chapter 6.

As discussed in section 2.5.2, there is a range of types of corpora for paraphrasing, for example: monolingual corpus, monolingual parallel corpus, monolingual comparable corpus of documents, and bilingual parallel corpus. Based on that discussion, a comparable corpus was selected, which is a collection of articles that are derived from different newswire services to describe the same event or topic. This type of corpora was selected due to these advantages:

1. Comparable corpora are most likely to be written by professional article writers such as publishers, editors and columnists. Therefore, comparable corpora are much cleaner than other corpora.

2. It is easy to find paraphrases in comparable corpora. The reason is that correspondence exists between comparable texts. For example,

   (i) Newspapers report different articles about the same event or topic on the same day.

   (ii) A question with only one answer can be asked in many different ways.

   (iii) The editors who write the articles are not (usually) specialists in a specific field and they do not have background knowledge about every field (e.g., medical, political etc.). Hence, articles about politics can be written in specialised English and general English for people with and without political background knowledge, respectively.

Therefore, by building a comparable corpus of articles that have been automatically acquired from newswire using online RSS[6] feeds, these articles cover a number of topics such as business, politics, sports, and general news. The reason for using RSS feeds is that they provide a simple way to collect articles from multiple sources, which also have a set structure and therefore are easy to manipulate

[6]Rich Site Summary, which is a format for delivering regularly changing web content.

and extract specific data from, including, title, date, time, and summary. Additionally, a regular expression[7] matcher is used to clean out all unwanted content, such as comments, HTML tags and other non-human readable elements. This technique provides a large number of potential articles. As there are multiple reports of the same events or topics on the same day, it is likely that sentences with the same meaning will appear.

This section has described an automatic technique for creating and collecting a dataset for the TEQV system. The next section discusses the pre-processing process, which is applied to the dataset that has been collected in this section.

4.3 Dataset Pre-processing

Pre-processing is also called text normalisation and is applied to datasets for normalising the text. Some NLP tools are available for pre-processing, e.g., GATE[8], Ling Pipe[9], Mallet[10], Stanford toolkit[11], and Natural Language Tool Kit (NLTK)[12]. These tools contain the essential techniques for the text pre-processing stage: sentence splitter, tokeniser, POS tagger, parser, and morphological analyser. These essential operations are briefly described below.

- **POS tagging** is the process of assigning a POS tag (e.g., noun, verb, adjective, or adverb) to each word of a sentence. It is an important step in the pre-processing component in the architecture. POS tagging will be used to analyse the type of each word in the sentences, and is clarified in the next chapter.

- **Sentence Splitting** is the process of finding boundaries of sentences in text, e.g., by using regular expressions or by looking at full stops which denote the end of sentence. The sentence splitter chosen will depend on the language that is being used, as discussed further in Chapter 5.

[7]Regular expression is a sequence of characters that define a search pattern, mainly for use in pattern matching with string matching, such as particular characters, words, or patterns of characters.

[8]`https://gate.ac.uk/`

[9]`http://alias-i.com/lingpipe/`

[10]`http://mallet.cs.umass.edu/`

[11]`http://nlp.stanford.edu/software/`

[12]`http://www.nltk.org/download`

- **Morphological Analysis** is the process of defining the root of the word. A morpheme is the smallest piece of a word that contributes to its meaning, for instance, the word *derivations* has three morphemes: *derive-tion-s*. There are three cases of word form modification (inflectional morphology, derivational morphology, and cliticisation). For more details see Section 5.4.

4.4 Dataset Clustering

According to Aggarwal and Reddy (2013), clustering is the task of grouping related objects in such a manner that the objects in the same cluster are more similar to one another than to the objects in other clusters.

In this section we will discuss two types of clustering, namely the clustering of article pairs and the clustering of sentence pairs.

4.4.1 Clustering Article Pairs

The dataset that has been pre-processed is then clustered into pairs of articles. The purpose of this clustering process is to be able to detect similarities between articles, since similar articles probably contain paraphrases. To do so, the dataset is treated as a monolingual comparable corpus containing articles about the same events or topics, which are written in one language by different authors (Barzilay and Lee 2003). Articles published during the same time period have a higher chance of being similar than those that are not chronologically close (Wang and McCallum 2006). However, it cannot be simply assumed that any two articles are similar only based on their timestamp. Although the comparable documents are not parallel, at the sentence or sub-sentence level, the paraphrased fragments may still exist. To quantify the comparability between two documents (articles) the standard similarity measure (cosine similarity with TF-IDF vector space) was applied.

Term Frequency and Inverse Document Frequency Vectors (TFIDF), developed by Salton and McGill (1986), is a weight often used in information retrieval and text mining. This weight is a statistical measure used to evaluate how important a word is to a document in a collection or a corpus.

- *tf* (term frequency): is the number of times the term occurs in the original

document.

- *df* (document frequency): is the count of how many articles in a corpus contain the word.

- *idf* (Inverse document frequency): is a measure of how important a word is, that is, whether the term is common or rare across all documents. Whilst computing TF, all words are considered equally important. However it is known that common words, such as "is", "of", and "that", may appear a lot of times but have little importance. Thus we need to weigh down the frequent terms while scaling up the rare ones, by computing the following equation:

$$tfidf = tf \cdot \log\left(\frac{1}{df+1}\right) \tag{4.1}$$

The *log* has been used to smooth out the effect of large changes in document frequency, and 1 is added because if a term occurs in one document and has not occurred in any other documents then *df* will be zero and *log* of zero is infinity.

The TF-IDF weighting scheme is often used in the vector space model with cosine similarity to measure the similarity between two vectors X and Y by finding the cosine of the angle between them. The cosine similarity of any pair of vectors can be calculated by taking their dot product and dividing that by the product of their norms. That yields the cosine of the angle between the vectors.

Given two vectors of attributes X and Y, then

$$similarity(X,Y) = \cos(\theta) = \frac{X.Y}{|X||Y|}. \tag{4.2}$$

where θ represents the angle between the vectors. As θ ranges from 0 to 90 degrees, $\cos(\theta)$ ranges from 1 to 0, where 1 means exactly the same, 0 indicates dissimilar, and intermediate values indicate somewhere between similarity or dissimilarity. The angle θ can only range from 0 to 90 degrees, because TF-IDF vectors are non-negative.

In this section, the articles have been clustered into pairs of articles that might be similar using similarity metrics. Then in the next section, we will use the same metric similarity to cluster the sentences into sentence pairs.

4.4.2 Clustering Sentence Pairs

This section describes the process of selecting pairs of related sentences from within the previously paired articles. The aim of this clustering process is to find similarities between sentences, since similar sentences are likely to contain candidates of paraphrases. To find sentence pairs that are similar, articles from two different sources were used that were published about the same story on the same day. Such a selection is more likely to produce candidate pairs of sentences that describe the same set of basic facts, but are not excessively similar ((Shinyama and Sekine 2003; Barzilay and Lee 2003; Brockett and Dolan 2005; Wang and Callison-Burch 2011; Bar 2013)). For instance, Table 4.4 shows how the editors express the same fact differently on different newswires.

Newswire	Sentence
BBC	Women 'nearly twice as likely to have anxiety' as men
The Guardian	Women twice as likely as men to experience anxiety, research finds
The Independent	Women are biggest worriers and are TWICE as likely to suffer anxiety as men, study says
Express	Women are almost TWICE as likely to suffer from anxiety as men

Table 4.4: The same fact is expressed differently in different news sources.

A sentence of one newswire can therefore be matched with related sentences from another. The motivation behind selecting pairs of related sentences from within paired articles is that the alignment algorithms (see Section 4.6) can be easily applied to the paired sentences as well as articles.

To improve the quality of the sentence pairs we use some conditions to filter the results and reject:

- Sentence pairs where the sentences are identical, or differ only in punctuation.
- Sentence pairs where the cosine similarity threshold is less than 60%, because anything greater than or equal this threshold is sensible for classification between positive and negative cases of sentence pairs.
- Duplicate sentence pairs.

This section discusses the clustering process from articles into article pairs, and then into sentence pairs. Afterwards, the pairs of sentences that have been

collected will be assessed with human judgment by annotating the data and turning it into 'Gold Standard' as described in Section 4.5. An inter-annotator-reliability measure (Barrón-Cedeño et al. 2013) is used to check the reliability of annotators.

4.5 Dataset Annotation (Gold Standard)

This section discusses how to evaluate the human-based judgment of the dataset to turn it into a 'Gold Standard'. The steps are described for annotating the dataset. First, an online annotation form is designed that contains collections of sentence pairs that are distributed to the annotators to mark up as detailed in Section 4.5.1. In Section 4.5.2, the inter-annotator-agreement is calculated using the kappa test (Cohen 1960).

4.5.1 Annotating Pairs of Sentences

The pairs that are collected in the second stage still have to be marked up by human annotators, but at least the process of collecting them is nearly bias-free. The annotation is performed by volunteers, and the study has to rely on their goodwill both in terms of how many examples they do, and how carefully they do them. The task therefore has to be made as easy as possible, to encourage them to do it, and the problems have to be managed that arise from having as annotators a mixture of people with different backgrounds. Having non-experts can bring benefits: TEQV is about the judgments that a typical speaker would make, and not about the judgments that a logician, or a carefully briefed linguist would make. From this point of view, having a mixture of volunteers carrying out the task is preferable since they represent the majority of typical speakers.

Since the annotators are distributed across the globe, an online annotation system has been developed. The system presents the annotator with pairs of sentences that they have not yet seen and asks them the question: Do these two sentences have the same meaning or not? The annotator marks up this pair as follows:

- If they consider that the sentence pair (S_1, S_2) has the same meaning, they will tag the pair as positive 'YES'. That means that they are equivalent.

- If they consider that the sentence pair (S_1, S_2) differs in meaning, they will tag the pair as negative 'NO'. That means that they are not equivalent.

This chapter addresses the general architecture, while the next chapter discusses the online annotation forms that display 300 sentence pairs for different languages (e.g., English, and Arabic). These pairs cover a number of topics such as politics, business, sport and general news. Five volunteer annotators[13] were chosen for each language to annotate the different pairs as 'YES' or 'NO' in the Textual Equivalence Dataset (TEQVDS). The judgment of semantic equivalence of the two sentences refers to the idea that both sentences (S_1, S_2) are paraphrases of each other. All information about sentence pairs, annotators, annotations and other is stored in a MySQL database for later processing.

4.5.2 Reliability of Annotators

There are problems associated with this strategy, however. The volunteers may misunderstand what they have to do, or they may know what is required but are not accurate in carrying out the annotation process. Annotators must be identifiable if, for whatever reason, they have not done the job properly. It is therefore necessary to be able to measure the reliability of each annotator to avoid unreliable ones. Reliability of annotators was addressed by measuring the agreement between them by using inter-annotator-agreement.

The challenge is that it is hard to know in advance how reliable those annotators are. There are several operational definitions of 'inter-annotator-reliability' (IAA, also called inter-annotator-agreement) in use by many researchers, reflecting different viewpoints about what is meant by reliable agreement between annotators (Banerjee et al. 1999). Here, a statistical measure for assessing the reliability of agreement among the annotators is applied when assigning category annotations to annotated sentence pairs. This measure is called ***kappa***, which takes chance agreement into consideration. Fleiss's kappa (Fleiss 1971) is used, which is a generalisation of Cohen's kappa statistic that provides a measurement of agreement among a constant number of annotators n, where each of the k pair of sentences is annotated by $n > 2$ annotators.

Let k_{ij} be the number of annotators who assign the i$^{\text{th}}$ pair of sentences to

[13] In our case, all the annotators are native speakers of English or Arabic. Some of them are PhD students in linguistics, whereas the others are working in fields related to NLP.

the jth category ($i = 1, \ldots, k$ and $j = 1, \ldots, c$). The kappa can be defined as:

$$kappa = \frac{p_0 - p_e}{1 - p_e}, \tag{4.3}$$

where

$$p_0 = \frac{\sum_{i=1}^{k} \sum_{j=1}^{c} k_{ij}^2 - nk}{kn(n-1)} \tag{4.4}$$

and

$$p_e = \sum_{j=1}^{c} p_j^2, \tag{4.5}$$

where

$$p_j = \frac{1}{nk} \sum_{i=1}^{k} k_{ij}. \tag{4.6}$$

The numerator ($p_0 - p_e$) of Equation 4.3 gives the degree of agreement actually achieved above chance, whereas the denominator ($1 - p_e$) gives the degree of agreement that is attainable by chance. Kappa score is a number between 0 and 1, with higher kappa for better agreement: kappa in the range of 0.01–0.20 stand for slight agreement; 0.21–0.40 for fair agreement; 0.41–0.60 for moderate agreement; 0.61–0.80 for substantial agreement; 0.81–0.99 for almost perfect agreement.

To detect the unreliable annotators, the kappa must be calculated using the above equations for each annotator with their co-annotators and another kappa for their co-annotators only, for the five annotators for each language.

This is the way that will be used for annotating the dataset to turn it into 'gold standard' by assessing human judgment, and the way that has been used for checking the reliability of annotators using inter-annotator-agreement. Afterwards, the 'gold standard' data can be used to compare different similarity measurement algorithms to see which one most closely matches human judgment.

4.6 Using Alignment Methods to Measure the Sentences Similarity

In the alignment process, a text is represented as a sequence, which could be a word or an entire sentence. To measure the similarity between two sentences (sequences), it is proposed to compare them by aligning the two sequences and count the number of editing operations (e.g., Insertion, Deletion, and Exchanging)

which are required to convert one sequence into another.

The process of detecting the paraphrases can be considered as a sequence alignment problem, which aims to identify matching fragments of text between pairs of sentences. To detect paraphrases, a set of algorithms is proposed for aligning the text fragments based on the concept of aligning biological sequences (Mount 2004).

Defining the distance between two strings by measuring the minimum cost of edit operations needed to transform one string into the other is called the string edit distance problem. The edit distance between two strings refers to the Levenshtein distance presented by Levenshtein (1966). The Dynamic Time Warping (DTW) algorithm is a technique, which uses efficient dynamic programming to calculate the edit distance between two different strings that might vary in length (Sakoe and Chiba 1978).

DTW is a sequence alignment technique mainly utilised to find an optimal alignment between two sequences under certain restrictions. Initially, DTW became popular in the context of speech recognition, and then in time series data mining, particularly in similarity measurement and pattern recognition. DTW is widely used in science, medicine, industry and finance (Ratanamahatana and Keogh 2005). According to Kovacs-Vajna (2000), the DTW also has been successfully used to align biometric data, such as gait, signatures and even fingerprints.

The advantage of DTW is that it can handle different lengths of sequences. However, one of the drawbacks of DTW is that the time and the computational cost will increase as the square of the length of the sequences. Additionally, another main drawback of DTW is that the transformation operations (i.e., Insert, Delete, and Exchange) are matched monotonically when applied to the items that occur in the same order. When applied to the items that occur in a free word order language, we cannot assume that corresponding items occur in the same order. This problem is discussed and to a certain extent is resolved in this thesis. As an enhancement to the monotonic matching issues by the DTW algorithm, an extended version of the algorithm is proposed to allow transposition transformation operations between adjacent items with a free word order language. This makes the extended version of the algorithm more flexible than the standard one, as shown in Chapter 5.

First we will explain how the DTW algorithm will be used for aligning between pairs of sentences (S_1, S_2) in the proposed system to detect matching fragments of

sequences in the text. Afterwards, the extended version of DTW will be discussed by adding a new operation 'SWAP', which works fairly well with free word order languages.

4.6.1 Dynamic Time Warping (DTW) Algorithm

DTW is based on a dynamic programming technique (Müller 2007), used for measuring the similarity between any two sequences of strings with arbitrary lengths, by calculating the minimum number of editing operations (Insert, Delete, Exchange) which is required to convert one sequence into the other. DTW operates by aligning the items while maintaining their original order. The basic operations provided by the DTW algorithm are essential in our system and the algorithm was enhanced by adding a new operation as explained in the next section. However, this section will explain the classical DTW.

Assume the following two sequences of strings S and Q of length n and m, respectively, where

$$S = \langle s_1, s_2, \ldots, s_n \rangle, \tag{4.7}$$

$$Q = \langle q_1, q_2, \ldots, q_m \rangle. \tag{4.8}$$

To align (match) two sequences using **DTW**, an *n-by-m* matrix is constructed where the (i, j)-th element of the matrix contains the distance $d(s_i, q_j)$ between the two points s_i and q_j . Each matrix element (i, j) corresponds to the alignment between the points s_i and q_j.

As a dynamic programming technique, the problem is divided into several sub-problems ((Dasgupta et al. 2006; Cormen 2009)), each of which contributes to calculating the distance cumulatively as follows.

1. **Initialization:**

   $d(s_i, 0) = i$;

   $d(0, q_j) = j$;

2. **Recurrence Relation:**

   For each $i = 1, \ldots, n$

For each $j = 1, \ldots, m$

$$\text{dtw}(s_i, q_j) = \min \begin{cases} d(s_i - 1, q_j) + INS, & \\ d(s_i, q_j - 1) + DEL, & \\ d(s_i - 1, q_j - 1) + XCH, & \text{if } s_i \neq q_j, \\ 0, & \text{if } s_i = q_j. \end{cases} \tag{4.9}$$

Note that *INS*, *DEL* and *XCH* are constants here; different values can be used to produce different behaviour. It is important that the cost(XCH) should be less than cost(INS) + cost(DEL) in order to obtain the optimal path.

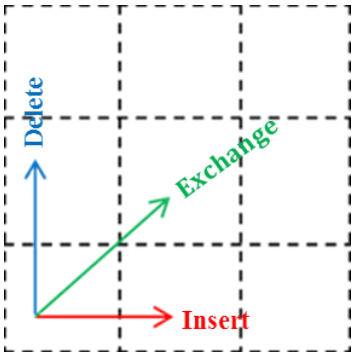


Figure 4.2: Directions for the search grid in DTW.

There are two ways to implement this algorithm, backward and forward. In this case, the forwards algorithm is used, and the possible warping can be illustrated in Figure 4.2, which means only three directions →Insert, ↗Exchange, ↑Delete will be searched from the previous step to construct the current step.

Here is a concrete example: we will compute the minimum edit operation required to transfer one string $W1$ to another string $W2$, where $W1 = ABCE$ and $W2 = ACBE$. In this example, a matrix is taken of dimension $n \times m = 4 \times 4$, where m and n represent the length of two strings $W1$, and $W2$. The string $W1$ is placed at the bottom row of matrix, and $W2$ is placed as the leftmost column. On execution of the DTW algorithm, each cell of the matrix is filled with the difference of edit operations performed, assuming $INS = 2$, $DEL = 2$, $XCH = 3$ on the basis of Equation 4.9. After filling the warping matrix, the final step for the DTW is to report the optimal warping path and the DTW distance. The warping path is the set of adjacent matrix elements that identifies the mapping between S and Q. It represents the path that minimizes the overall distance between S and Q, which is 4 as seen in matrix 6 of Figure 4.3.

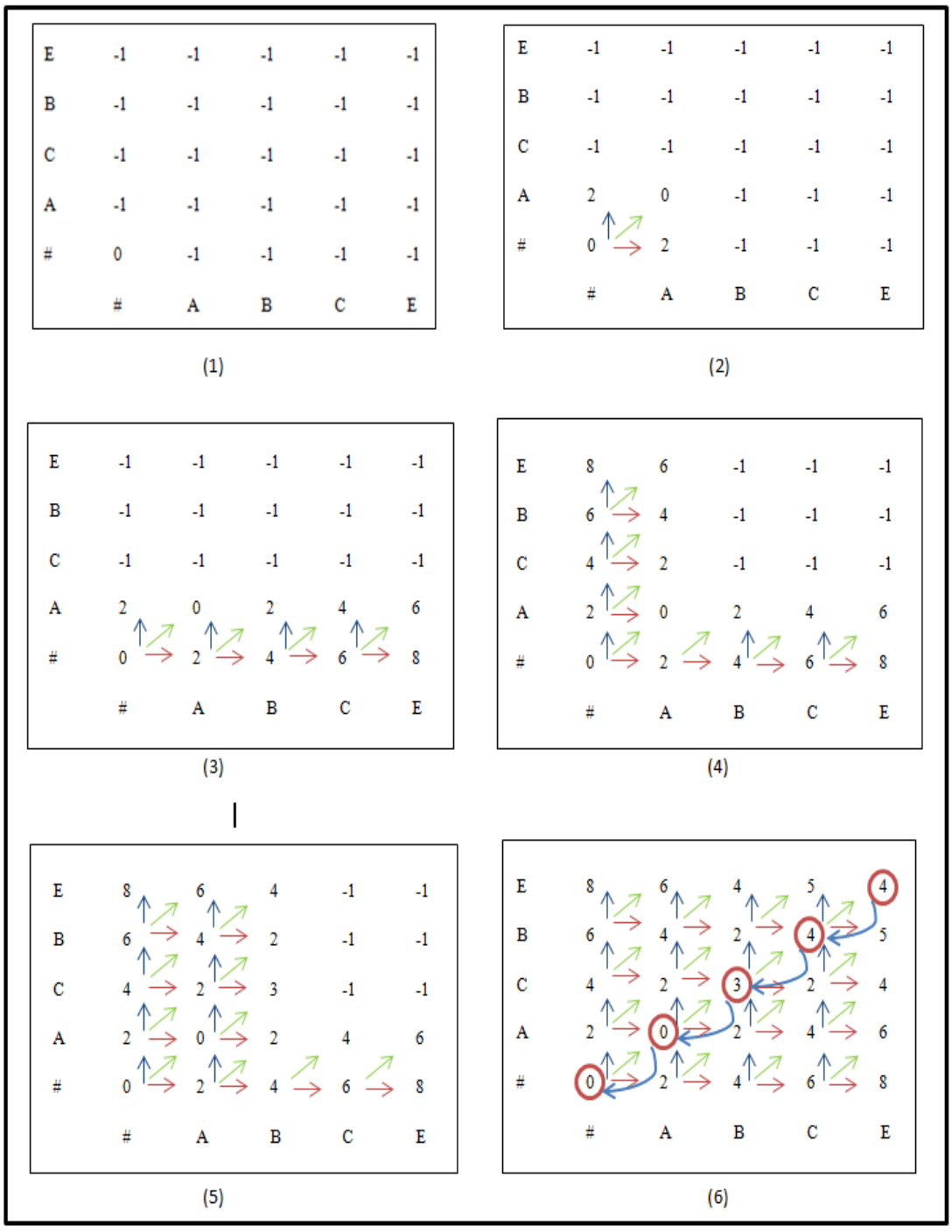


Figure 4.3: The minimum cost of edit operations to transfer *W1*='ABCE' into *W2*='ACBE' is 4.

Figure 4.3 shows that 4 is the minimum cost of edit operations to transform one string to another, and that represents the cost of two exchange operations

fewer than the cost of DEL and INS.

An Example on Spelling Correction with the use of the 'Exchange' operation

So far, the cost for an 'exchange' operation has been fixed. In some applications, it is a good idea to have a variable cost for 'exchange', for instance, in spelling correction. Many spelling correction algorithms (e.g., edit distance, n-gram-based techniques, rule-based techniques, and similarity keys) use a measure of distance between the misspelled word and the target by calculating the minimum number of operations (Insert, Delete, Exchange) to convert one string into another (Damerau 1964). The key difference between the proposed

In this case, suppose the spelling correction has a cost function for the Exchange operation, then the calculation of the distance on the keyboard between the characters depend upon how likely a certain character is to have been mistaken by the user (see Figure 4.4).

There are two different misspelling cases when the user types on the keyboard that involves:

1. A character that is physically close on the keyboard to the correct character (adjacent Keys).

2. A character that is physically far on the keyboard from the correct character.

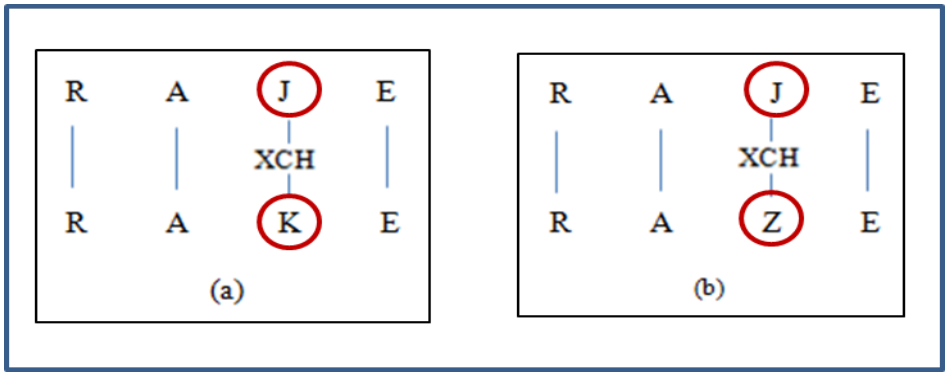


Figure 4.4: Example of typing two different words on the keyboard.

The maximum distance on the keyboard as shown in Figure 4.5 probably is for instance between the character '1' and '/', which is 13. Thus, to measure the cost of Exchange operation we will use the equation $(D/13 * 3)$ (e.g., the distance divided by the maximum distance on the keyboard between two keys times 3, which is the standard cost of Exchange operation).

Figure 4.4 (a) and (b) show the example of typing two different words. Suppose the user had typed RAJE. This is not an English word, so we need to find out what the user did intend to type. There are a number of possibilities: we will consider RAKE and RAZE, both of which are legal English words. Obviously, the strings are the same length but differ in one position on the keyboard, as shown in Figure 4.5. The typist intended to type 'K' or 'Z', but they made a mistake and typed 'J' on the keyboard. The 'J' is close to the 'K' on the keyboard and it is quite easy to hit the adjacent key 'J' instead of 'K', while it is difficult to type 'J' when the intended character is 'Z', because the letter 'Z' is physically far from the letter 'J' on the keyboard. The distance between the adjacent keys is 1, so to measure the cost of Exchange to convert one string to another by using the previous equation, we obtained $(1/13 * 3) = 0.23$, while the distance between 'J' and 'Z' which are far from each other is 7, so to calculate the cost of Exchange we obtained $(7/13 * 3) = 1.62$. Therefore, the cost of exchanging for adjacent keys should be less than the cost of exchanging for those which are physically far from each other.

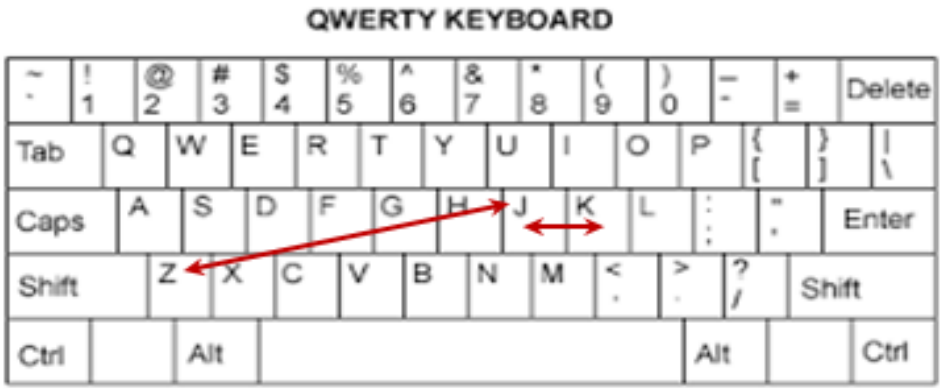


Figure 4.5: Standard QWERTY keyboard.

This section discussed the DTW algorithm with an example to see how it works with fixed cost of operations (e.g., Insert, Delete, and Exchange) for converting one string to another, and then a spelling correction example was given,

which used a variable cost with operations instead of fixed cost. The next section will discuss an enhancement of the algorithm so that it works well with adjacent items, which is the same DTW but the transposition of adjacent symbols is allowed as an edit operation.

4.6.2 Extended Dynamic Time Warping (XDTW) Algorithm

One of the main weaknesses of DTW is that transformation operations (Insert, Delete, and Exchange) presume that items to be matched occur in the same order. The output of DTW on single items has the lowest cost sequence of operations. This algorithm will be extended to find the lowest cost sequence of operations on adjacent items (characters) with free word order. The enhanced DTW algorithm proposed here is called the 'eXtended Dynamic Time Warping' (XDTW) algorithm, which will be more flexible than the existing one.

XDTW operations are enhanced to produce cost-effective results compared to DTW operations, and such a method has been presented by Damerau-Levenshtein distance model. Damerau (1964) suggested adding a new edit step, swap, to the standard three. The key difference between Damerau's algorithm and the proposal here is that, as with the standard Exchange operator, we take into account the "similarity" between the words being exchanged when calculating the cost. XDTW is used to transfer one string to another by calculating the minimum number of operations (Insert, Delete, Exchange, Swap), with different cost fixed (Insert = 2, Delete = 2, Exchange = 3 or less, and Swap = 0.5) depending on the weighted minimum distance. The XDTW algorithm tries to set the cost of (Exchange/Swap) to be lower with similar words than dissimilar ones.

To explain the computational process of XDTW, we assume the following two sequences of strings S and Q of length n and m, respectively, where

$$S = \langle s_1, s_2, \ldots, s_n \rangle, \tag{4.10}$$

$$Q = \langle q_1, q_2, \ldots, q_m \rangle. \tag{4.11}$$

To align (match) two sequences using X**DTW**, an *n-by-m* matrix is constructed where the (i, j)-th element of the matrix contains the distance $d(s_i, q_j)$ between the two points s_i and q_j. Each matrix element (i, j) corresponds to the alignment

between the points s_i and q_j.

1. **Initialization:**
   $d(s_i, 0) = i$;
   $d(0, q_j) = j$;
2. **Recurrence Relation:**

   For each $i = 1, \ldots, n$

   For each $j = 1, \ldots, m$

$$\text{xdtw}(s_i, q_j) = \min \begin{cases} d(s_i - 1, q_j) + INS, \\ d(s_i, q_j - 1) + DEL, \\ d(s_i - 1, q_j - 1) + XCH, \text{ if } s_i \neq q_j, \\ \qquad\qquad 0, \text{ if } s_i = q_j, \\ d(s_i - 2, q_j - 2) + XCH(s_i, q_j - 1) \\ +XCH(s_i - 1, q_j) + SWAP \end{cases} \tag{4.12}$$

The above equation is similar to the DTW equations (4.9) with the new operation 'SWAP', which means that two exchanges are added to the fixed value 'SWAP'.

To explain the XDTW algorithm, the example discussed in section 4.6.1 will be re-addressed with the enhanced operations. We will compute the minimum edit operation required to transfer one string $W1$ to another string $W2$ respectivly, where $W1 = `ABCE'$; $W2 = `ACBE'$. Here, a matrix is taken of dimension $n \times m = 4$, where m and n represent the length of two strings $W1$ and $W2$. the string $W1$ is placed at the bottom row of the matrix, and $W2$ is placed at the leftmost column. On execution of the XDTW algorithm, each cell of the matrix is filled with the difference of edit operations performed, assuming INS = 2, DEL = 2, XCH = 3 and SWAP = 0.5 on the basis of (4.12). After filling the warping matrix, the final step for the XDTW is to report the optimal warping path and the XDTW distance. The idea behind this is to obtain a cheaper route.

Figure 4.6 shows that 0.5 is the minimum cost of edit operations for transforming one string to another, and that represents the cost of 'SWAP', which is less than the cost of other operations.

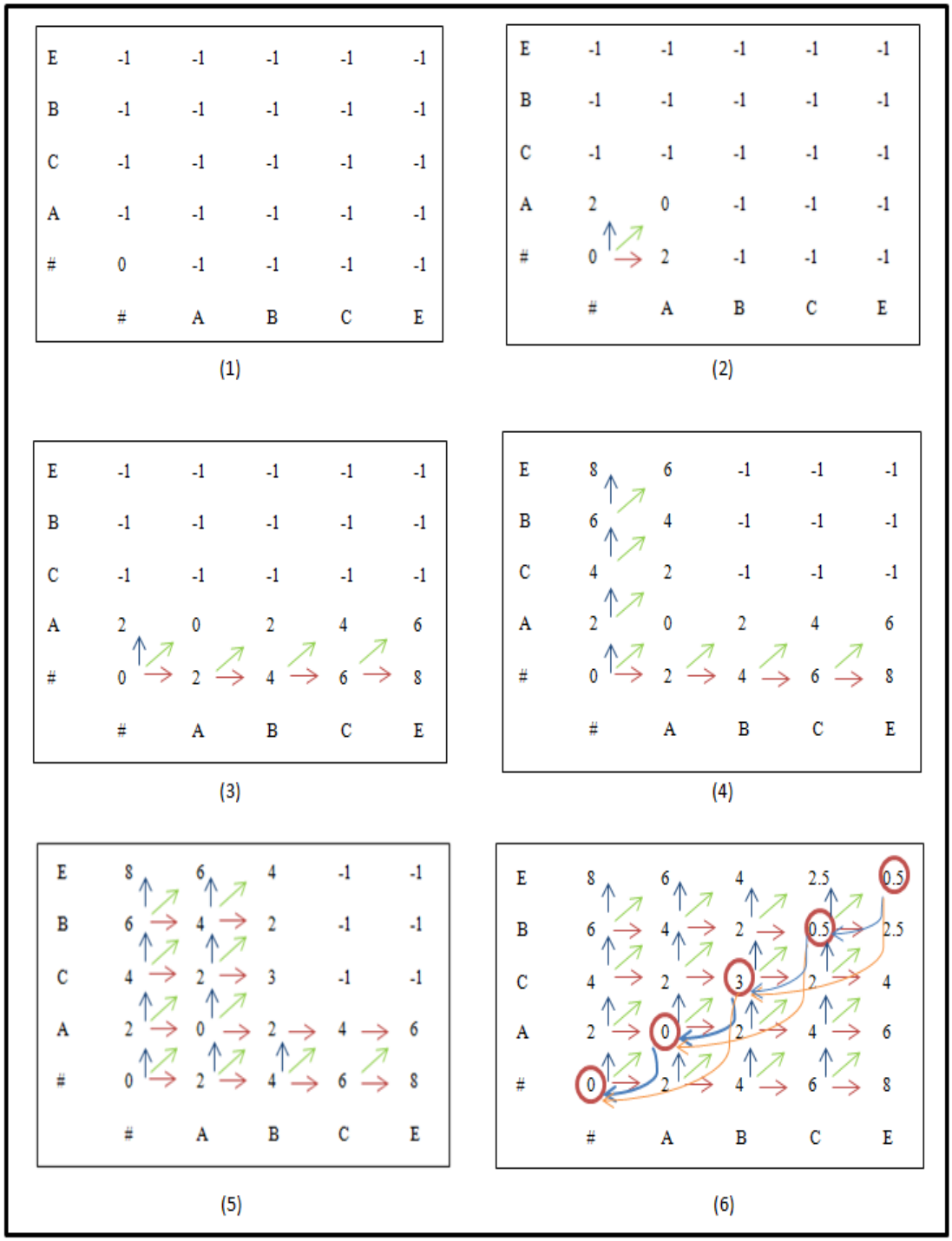


Figure 4.6: The minimum cost of edit operations to transfer *W1*='ABCE' into *W2*='ACBE' is 0.5.

An Example of Spelling Correction with the use of the 'Swap' operation

In the earlier example of spelling correction, we noted that people are likely to make mistakes in typing by pressing a key that is adjacent to the one they intended to press, so therefore that the cost of exchanging letters that are close together should be low.

Another common problem is that people get the sequence of key presses with the fingers of one hand muddled up with the sequence of key presses with the other hand, leading to a transposition of characters that are far apart on the keyboard, since these would be pressed with fingers on different hands. Figure 4.7 shows an example of typing the word in the wrong order. Suppose the user has typed 'RAKE'. The sequence of typing the word 'RAKE' by using two hands is *'left-left-right-left'*, and if the user has made a mistake with one hand and typed 'RKAE', then the sequence is *'left-right-left-left'*. Obviously, that can happen because the key 'A' is on the opposite side of the keyboard to the key 'K', and we would make this mistake if the sequence of using our hands is wrong. Therefore, the cost of swapping between two hands should cost less than the cost of the exchange for the keys that are physically far from each other, which is 0.5. The distance (D) between the keys 'A' and 'K' is 7, and then to measure the cost of Swap using the previous equation, we obtained $(7/13 \cdot 0.5) \approx 0.25$ (note that 13 is the maximum distance between two keys on the keyboard as we have mentioned before in Section 4.6.1).

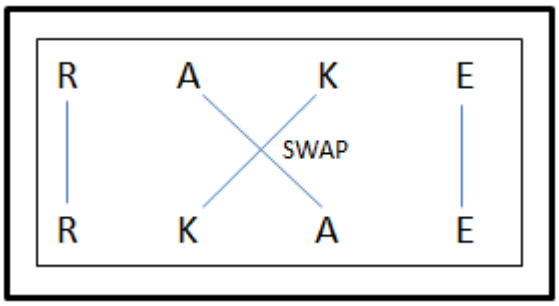


Figure 4.7: Typing the word with wrong order when using two hands.

In much the same way that the cost of exchanging two letters or of reserving the order of two letters when doing spelling correction might depend on where they appear on a keyboard, the cost of the various edit operations for calculating the similarity between pairs of sentences may depend on properties of the words

being inserted, deleted, exchanged, or swapped.

Consider the fact that a sentence is composed of words with different POS tags, such as nouns, verbs, adjectives, and adverbs. Nouns and verbs form the essential part of a sentence, while adjectives and adverbs play less important roles in its meaning. The measure of similarity between two sentences is achieved by calculating the number of edit operations (Insert, Delete, Exchange, and Swap) to transform one sentence to another. Different costs for edit operations can be used for different POS tags for Insert and Delete operations as seen in (4.1) and (4.2), and on the degree of similarity between the words involved for Exchange and Swap operations as will be shown in (4.5) and (4.6) in Section 4.6.4.

The following two examples illustrate why the cost of adding adverbs or adjectives should be less than the cost of adding a noun or a verb.

Example 4.1.
(S_1) The man is running.
(S_2) The old man is running.

Example 4.2.
(S_1) The man is running.
(S_2) The man is running races.

The sentence pair in (4.1) seems more similar than the pair in (4.2) because the former pair has a modifier (adjective) and the latter contains an entire new notion (noun), which adds new content to the sentence. In other words, adding an adjective such as 'old' has less effect on the meaning of a sentence than adding a noun such as 'races', which increases the difference in meaning. Based on these examples the cost of adding modifiers (e.g., adverbs or adjectives) should be less than the cost of adding a noun or a verb. Therefore, depending upon the POS tags, different costs are set (see Section 5.3.2 for more explanation).

4.6.3 Using alignment cost as a similarity measure

There is, however, a problem with using string edit distance as a similarity measure. This distance measures the difference between strings and ignores the characters that are in common. Consider the following two pseudo-examples of pairs of strings:

Example 4.3.
(S_1) A X
(S_2) A Y

Example 4.4.
(S_1) A B C X P Q R
(S_2) A B C Y P Q R

The string edit distance between the pairs of sentences in (4.3) and (4.4) are the same, because there is only one exchange operation. However, note that the string edit distance does not account for the number of characters that are the same: the sentence pair in (4.4) can be considered more similar than the sentence pair in (4.3) in the sense that there are more characters in common, but the string edit distance fails to measure that.

To rectify this, we will also account for the characters that are in common in a pair of strings. We will denote by *sim* this new similarity score.

Let us assume that we have a pair of strings (S_1, S_2) such that S_1 has length N_1 and S_2. has length N_2. Furthermore, assume that the cost of Insert and Delete is 2 and that the cost of Exchange is 3 as described in Section 4.6.1 and 4.6.2.

In order to measure the similarity between the strings, we measure the actual cost C and divide by the worse possible cost W, where

$$W = (3 * \min\{N_1, N_2\} + 2 * |N_1 - N_2|),$$

since we can exchange only as many characters as there are in the shorter string, and since we can either insert into the shorter string or delete from the longer string exactly $|N_1 - N_2|$ characters.

The similarity score *sim* is then defined as:

$$sim = 1 - C/W. \tag{4.13}$$

We should therefore expect that the pairs of strings that have a large number of characters in common will have the similarity score *sim* close to 1. Going back to (4.3) we find that the similarity score is $(1-(3/(3*2+2*0))) = (1-(1/2)) = 0.5$, while the similarity score in (4.4) is $(1 - (3/(3 * 7 + 2 * 0))) = (1 - (1/7)) = 0.86$.

In this section we have shown how to adapt costs produced by alignment algorithms so that they can be used more sensitively as similarity scores using

the formula (4.13).Then, in the next section we will use the WordNet similarity measures as a cost function for Exchange operation with alignment methods. Thus, the WordNet similarity measures will be discussed in more detail.

4.6.4 Using WordNet similarity measures with alignment methods

To detect sentence similarity, relying only on the string edit distance between sentences will not provide the accurate results since similarity also depends on the meaning of the words.

The concept of sentence similarity is extended to convey the meaning of words involved in each sentence and the semantic similarity or relatedness between these words. Therefore, a range of different word semantic similarity measures is needed based on the information available from WordNet to measure the degree of semantic similarity between the words. Such an observation will be elaborated more in (4.5) and (4.6):

Example 4.5.
(S_1) Minimum wage to rise to £6.50 an hour.
(S_2) Minimum wage to increase to £6.50 an hour.

Example 4.6.
(S_1) Google decided to increase production of self-driving cars.
(S_2) Google decided to decrease production of self-driving cars.

The sentence pair in (4.5) seems more similar than the sentence pair in (4.6) because the meanings of 'rise' and 'increase' are very close, and therefore the cost of exchange between 'rise' and 'increase' needs to be set much lower than the cost of exchange between 'increase' and 'decrease', which should be high (see Section 5.3.2 for more explanation).

To measure the degrees of similarity of words as in (4.5) and (4.6), WordNet semantic similarity functions can be plugged into the DTW and XDTW algorithms as cost functions for the Exchange and Swap operations. WordNet::similarity[14] is a freely available software package that makes it possible to measure the degree of semantic similarity between a pair of concepts (or word senses) (Pedersen et al. 2004). WordNet is a large lexical database of English and provides six different

[14] `http://wn-similarity.sourceforge.net/`

metrics of similarity (Fellbaum 1998). Three of these measures are based on the information content such as: *res* (Resnik 1995), *lin* (Lin 1998), and *jcn* (Jiang and Conrath 1997), see Table 4.5. The three other measures are based on the length of the path such as: *lch* (Leacock and Chodorow 1998), *wup* (Wu and Palmer 1994), and *Shortest path* (Rada et al. 1989), see Table 4.6. The idea behind this is to obtain different degrees of similarity between words by obtaining the synonym sets (synsets).

Before discussing the WordNet similarity metrics further, as shown in Tables 4.5 and 4.6, some basic definitions of the related concepts will first be explained.

(1) $len(c_1, c_2)$: is the length of the shortest path from synset c_1 to synset c_2 in WordNet.

(2) $lcs(c_1, c_2)$: is the lowest common subsumer of c_1 and c_2.

(3) $depth(c_i)$: is the length of the path from the global root entity to synset c_i

(4) $deep_{max}$: is the max depth of the taxonomy.

(5) *IC* stands for information content, The information content of a concept (term or word) is the negative logarithm of the probability of finding the concept *p(c)* in a given corpus.

$$IC(c) = -logp(c) \tag{4.14}$$

(6) $sim(c_1, c_2)$: is the semantic similarity between concept c_1 and concept c_2.

Similarity measure	Description	Mathematical formula
Resnik's measure	Resnik proposed an information content-based similarity measure. He assumes that for two given concepts, similarity depends on the information content that subsumes them in the taxonomy.	$sim_{res}(c_1, c_2) = IC(lcs(c_1, c_2))$ (4.15)
Lin's measure	Lin proposed a similarity measure that uses both the amount of information needed to state the commonality between the two concepts and the information needed to fully describe these terms.	$sim_{lin}(c_1, c_2) = \frac{2 \cdot IC(lcs(c_1, c_2))}{IC(c_1) + IC(c_2)}$ (4.16)
Jiang's measure	Jiang calculated semantic distance to obtain semantic similarity. Semantic similarity is the opposite of the distance.	$sim_{jcn}(c_1, c_2) = (IC(c_1) + IC(c_2)) - 2 * IC(lcs(c_1, c_2))$ (4.17)

Table 4.5: WordNet Similarity measures based on the Information Content (IC).

Similarity measure	Description	Mathematical formula
Wu and Palmer	This measure takes the position of concepts c_1 and c_2 in the WordNet taxonomy relatively to the position of the most specific common concept $\text{lcs}(c_1, c_2)$ into account.	$sim_{wup}(c_1, c_2) = \frac{2 \cdot depth(lcs(c_1, c_2))}{len(c_1, c_2) + 2 \cdot depth(lcs(c_1, c_2))}$ (4.18)
Shortest Path	Shortest path only takes $len(c_1, c_2)$ into account. This measure assumes that the $sim(c_1, c_2)$ depends on how close the two concepts are in the taxonomy.	$sim_{path}(c_1, c_2) = 2 \cdot deep_{max} - len(c_1, c_2)$ (4.19)
Leacock and Chodorow	This measure takes the maximum depth of the taxonomy into account.	$sim_{lcs}(c_1, c_2) = -\frac{\log(len(c_1, c_2))}{2 \cdot deep_{max}}$ (4.20)

Table 4.6: WordNet Similarity measures based on the length of the path.

For more clarification see Figure 4.8 that shows the relations between words in WordNet. Looking at this diagram we can see that $len(Car, Fork)$, the shortest path between 'Car' and '$Fork$', is 8; that $lcs(Car, Cycle)$, the lowest common subsumer, of 'Car' and '$Cycle$' is '$Vehicle$'; and that the depth of 'Car' is 5, which is the same as the maximum depth of the whole taxonomy.

In addition, to obtain the Information content (IC) we need to calculate the following:

- term frequency (tf), obtained from a corpus, which could be WordNet definitions, or it could be some other corpus (e.g., Brown) and inherited frequency (if), which is the cumulative total of all the term frequencies of nodes lower in the hierarchy.
- the probability of a concept $p(c) = (tf+if)/N$, where N is the total number

of the words in the hierarchy.

Figure 4.8 shows the value of term and inherited frequency, and Figure 4.9 shows the value of IC by using the equation (4.14), and (f), which is the result of calculating (tf+if).

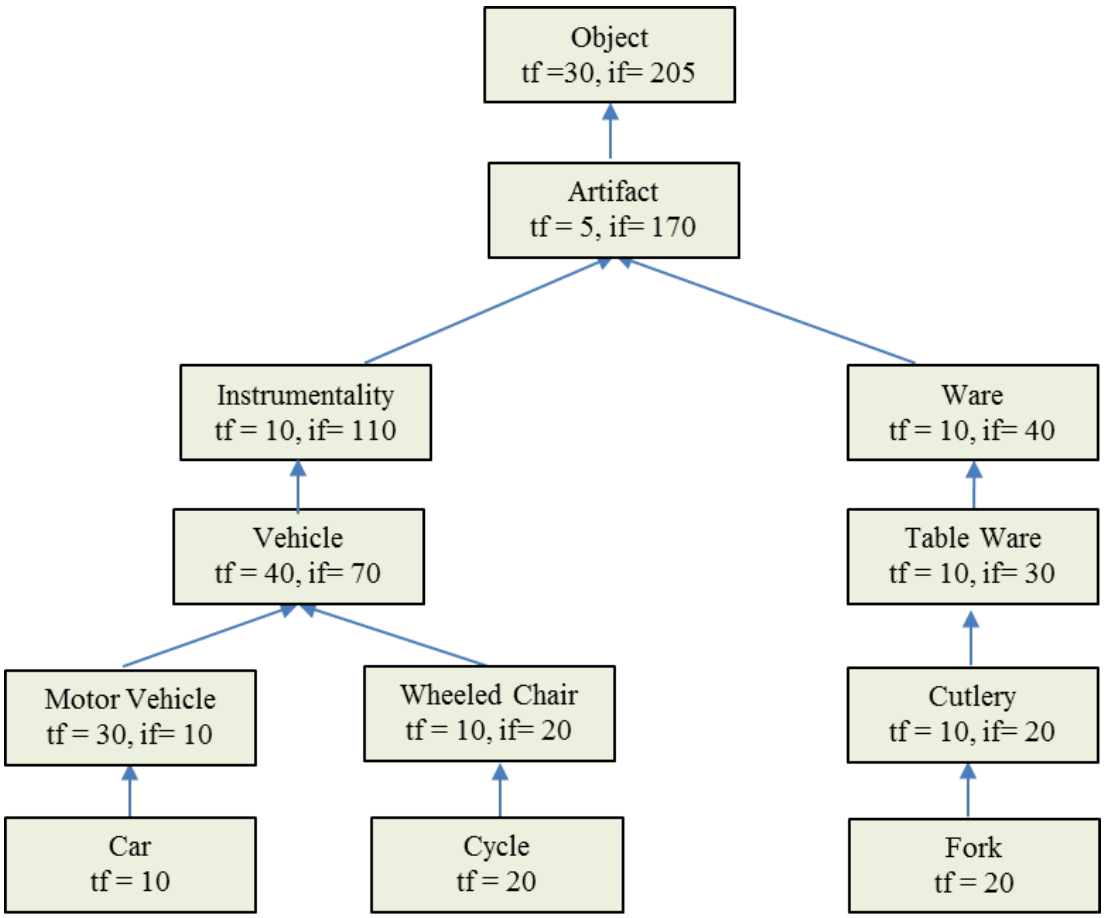


Figure 4.8: The Term Frequency and Inherited Frequency for the words in WordNet hierarchy.

The advantages of the Shortest Path, Wu and Palmer, and Leacock and Chodorow measures is that they are simple to implement. However, a disadvantage for all of them is that depth in the hierarchy is no guarantee of specificity, so two terms may be at the same depth but the similarities between their daughters are not the same.

Similarly, the advantage of *res*, *lin*, and *jcn* measures is that they are fairly robust because they depend on the natural distribution of words, whereas the

disadvantage is that they depend on the size of the corpus and whether it is appropriate for your domain, and moreover the sparse data problem is not avoided.

WordNet semantic similarity functions have to be normalised within a fixed range; some of them already are normalised and give values between 0 and 1, but some of them are not actually guaranteed to be in that range and can produce arbitrarily high values. Therefore, a way has to be found to normalize them, which is done by dividing the similarity score provided by a given measure with the maximum observed score for that measure.

The reason for normalising the value of semantic similarity functions to (0–1) is because there are some constraints:

- The cost of Exchange should be less than Insert and Delete ($i.e., XCH < INS + DEL$),
- The cost of Swap is $(2 \cdot XCH)$,

which means that the cost of Swap must be less than or equal to the costs of two Insert and two Delete operations, ($i.e., SWAP \leq 2(INS + DEL)$).

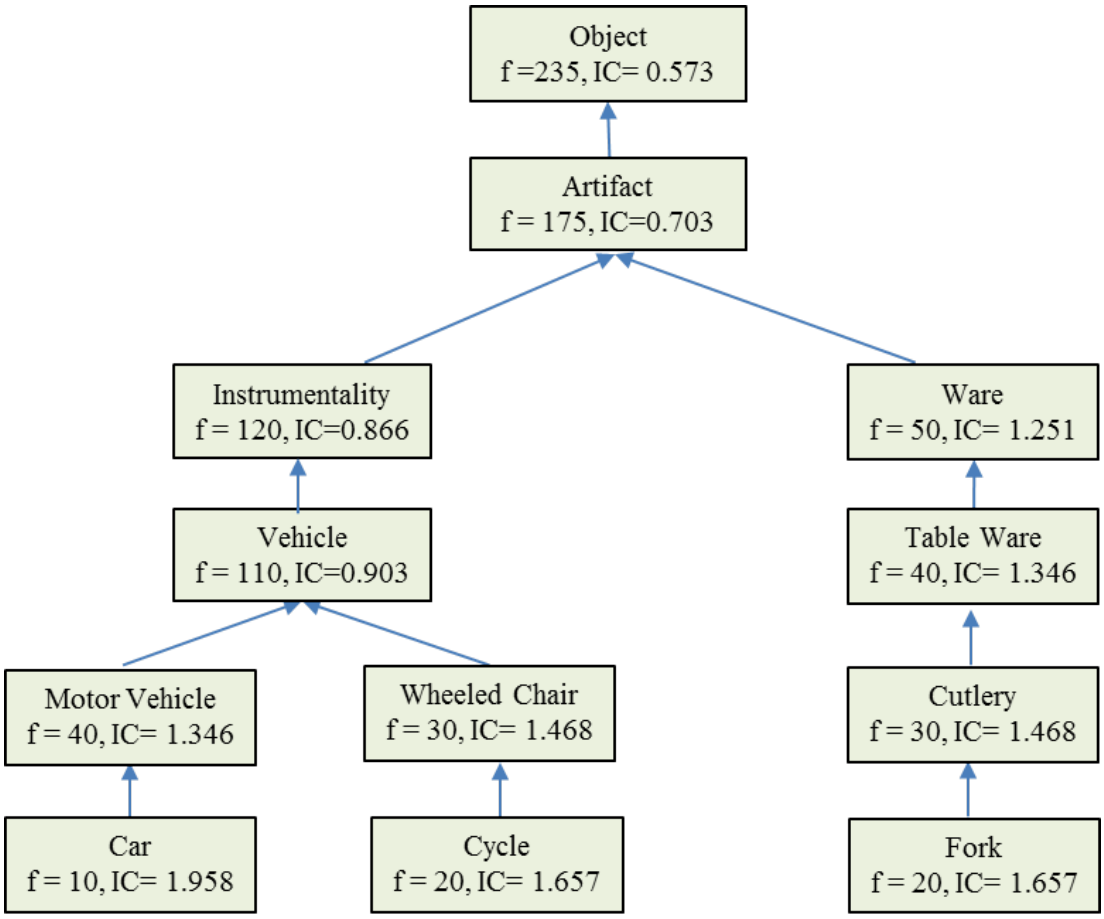


Figure 4.9: Calculating IC in WordNet hierarchy.

In Section 5.3.2, we will explain how to align and calculate the similarity scores between pairs of sentences in examples 4.2, 4.4, 4.5, and 4.6.

In this section, we discussed sentence similarity measures on different cost functions for edit operations (e.g., Insert, Delete, Exchange, and Swap) based on the degree of similarity between the words. A comparison of the effectiveness of the results of these methods by using different evaluation measures such as *precision* and *recall* will be presented in the next section.

4.7 Performance Metrics

The goal of this section is to assess and analyse the comparative effectiveness of different methods on results of similarity scores to the gold standard when applied to English and Arabic. There is a number of evaluation measures namely: precision (P), recall (R) and F-measure (F), which we now discuss.

- **Precision (P)**, defined as

$$P = \frac{\text{True Positive (TP)}}{\text{True Positive (TP)} + \text{False Positive (FP)}}, \tag{4.21}$$

  is a measure of how reliable the system is when it assigns an item to a class (i.e. the number of items correctly labelled as belonging to that class).

- **Recall (R)**, defined as

$$R = \frac{\text{True Positive (TP)}}{\text{True Positive (TP)} + \text{False Negative (FN)}} \tag{4.22}$$

  is a measure of how good the system is at finding items that belong to a target class (how many of the items that belong to that class the system selects).

- **F-measure**, defined as

$$F\text{-measure} = 2 \cdot \frac{P \cdot R}{P + R} \tag{4.23}$$

  is a measure of the harmonic mean of precision and recall, which provides a useful compromise measure: it is easy to obtain very high precision, by being very selective about which items are chosen; and it is easy to obtain very high recall, e.g. by simply selecting every single item. F-measure assigns a very low score to these extreme cases, and hence provides a better overall measure of the performance of the system.

In this thesis the precision and recall measures will be used. However, there are a huge number of sentences that occur in all the articles published in a given period, and those sentences may have no relation to each other. In order to obtain the recall we will have to compare every single pair, and then we have to ask the annotators to annotate an enormous number of data, which is impossible because they are volunteers. So we applied some filtering on the sentence pairs to find the plausible pairs, and may have lost some actual pairs. Therefore, we cannot calculate the recall but we can only estimate it, see Chapter 6 for more details.

Chapter 5

Experimental Design

5.1 Introduction

In Chapter 4 we introduced the general workflow of TEQV, to be used for detecting semantic equivalence (paraphrases) between two text snippets. This chapter presents a series of experiments to investigate to what extent the standard techniques perform differently when applied to different languages. We have kept the mechanisms as similar as possible for constructing and designing the experiments in both languages English and Arabic in order that the differences detected will arise not from the method used to collect and annotate the data. However, since each language has different features that need to be addressed carefully the experiment must take into consideration the language differences and examine precisely the behaviour of the proposed techniques.

This chapter discusses the dataset preparation, implementation of the experiments and the results of the validation process of the proposed system judgment compared to human judgment, based on the accuracy of the result measured by precision and recall rates. See Figure 5.1, which is the expanded view of the TEQV workflow from Figure 4.1. The experiment is structured according to the general architecture presented in Chapter 4.

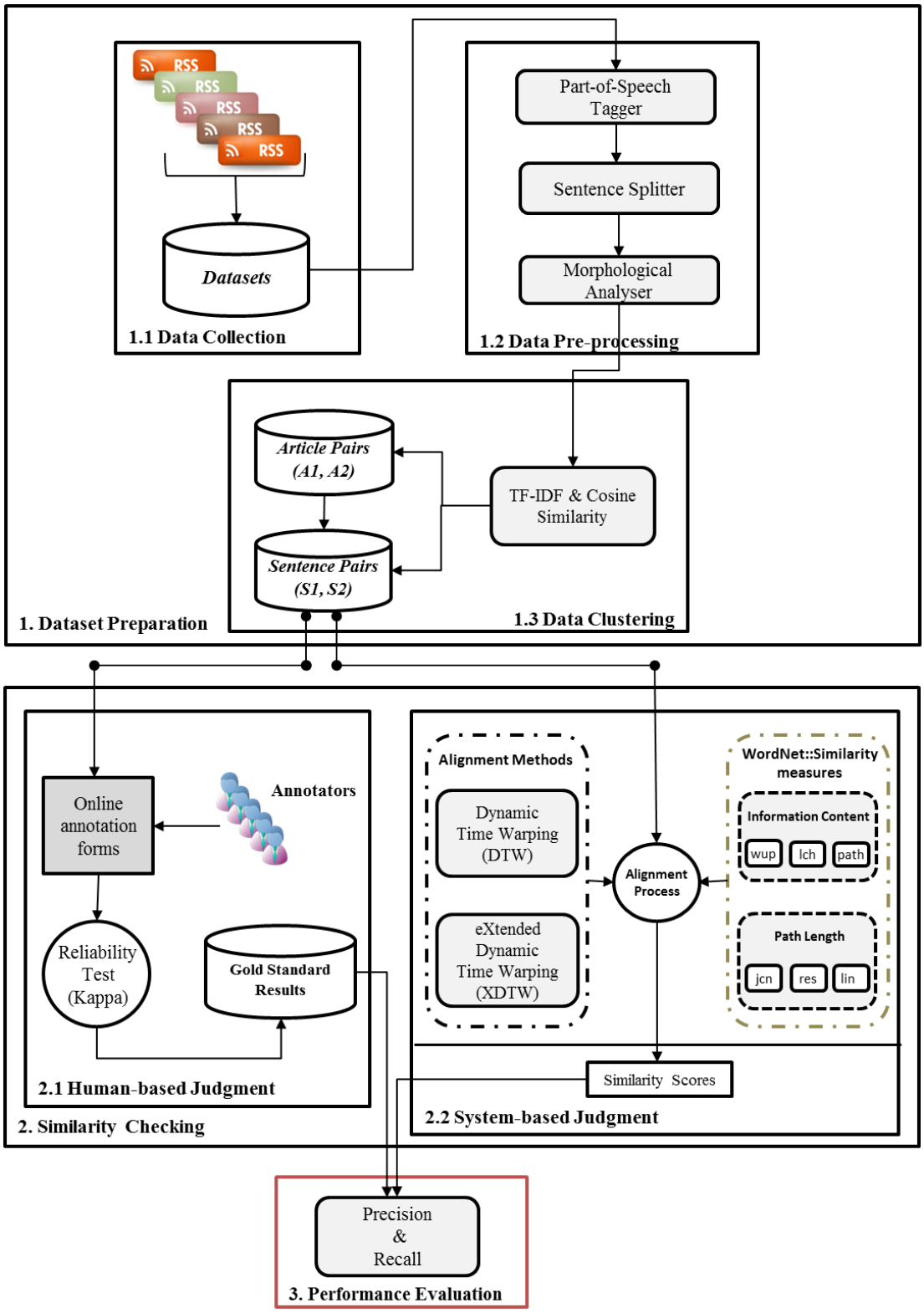


Figure 5.1: Expanded view of TEQV workflow for Figure 4.1.

1. **Dataset Preparation (Section 5.2)**: datasets are created for both languages, English and Arabic. The required datasets are collected, preprocessed and clustered from main articles into sentence pairs, which will be used later for human and system based judgment experiments. This stage of the process potentially produces very large numbers of pairs – there is no real limit to the amount of data that can be collected and classified by automated processes of the kind described in Section 5.2. For the purposes of this thesis, however, we need the data to be annotated by human subjects; and we further need the data that has been annotated by our subjects to be roughly balanced between examples that do and do not contain paraphrases, because otherwise it is easy for a classifier to score highly just by picking tha majority class. We therefore processed the data collected at this stage by assigning TF-IDF cosine scores to individual articles, and then to pairs of sentences that come from articles that scored highly at this stage, to obtain a set of pairs that were reasonably similar but which were roughly evenly balanced between pairs that contained paraphrases and ones that did not. The initial round of collecting articles produced around 9.8K Arabic and 30K English sentences. It is clearly infeasible to compare every possible pair using TF-IDF cosine scores, since that would be nearly 100 million pairs for Arabic and 900 million pairs for English. We therefore initially compared the articles that they appeared in, on the grounds that similar articles would be most likely to contain similar sentences, and then matched pairs from within these. This led to a collection of around 3000 pairs. We ranked these again by TF-IDF cosine score, and then by manual inspection found the score at which the split was roughly 50:50 between sentences that contained paraphrases and ones that did not. This produced a set of 300 pairs, which was a reasonable amount of data to ask our human subjects to annotate. These 300 pairs are the ones that were used in the experiments described below.

2. **Similarity Checking (Section 5.3)**: the 300 pairs are checked for similarity between the sentence pairs by human- and system-based judgment. The datasets were annotated by human subjects, as described in Section 5.3.1. The English and Arabic datasets are also tested by the system using the DTW and its extension (XDTW) algorithms with a range of WordNet similarity functions to obtain similarity scores. Using Arabic WordNet

poses a number of problems, since the text we are working with includes inflected forms of words (including words with a variety of clitics attached) whereas Arabic WordNet is based on root forms. We used the well-known Arabic morphological analyser SAMA, as embodied by the PyAramorph implementation, but this raised a fresh set of problems, in that the output of SAMA provides 'names' of root forms which are not an exact match for the names that are used in Arabic WordNet. These issues are discussed in Section 5.3.2

3. **Experimental Results and Analysis (Chapter 6)**: the similarity scores that have been obtained from system-based judgments, and the 'Gold Standards' that have been obtained from human-based judgments, are compared and tested using the precision and the recall evaluation measures for both English and Arabic.

5.2 Dataset preparation

The processes in this phase of the experiment revolve around preparing English and Arabic datasets to be used later in the similarity measure systems. They are depicted in Figure 5.2, and each process will be explained in detail below.

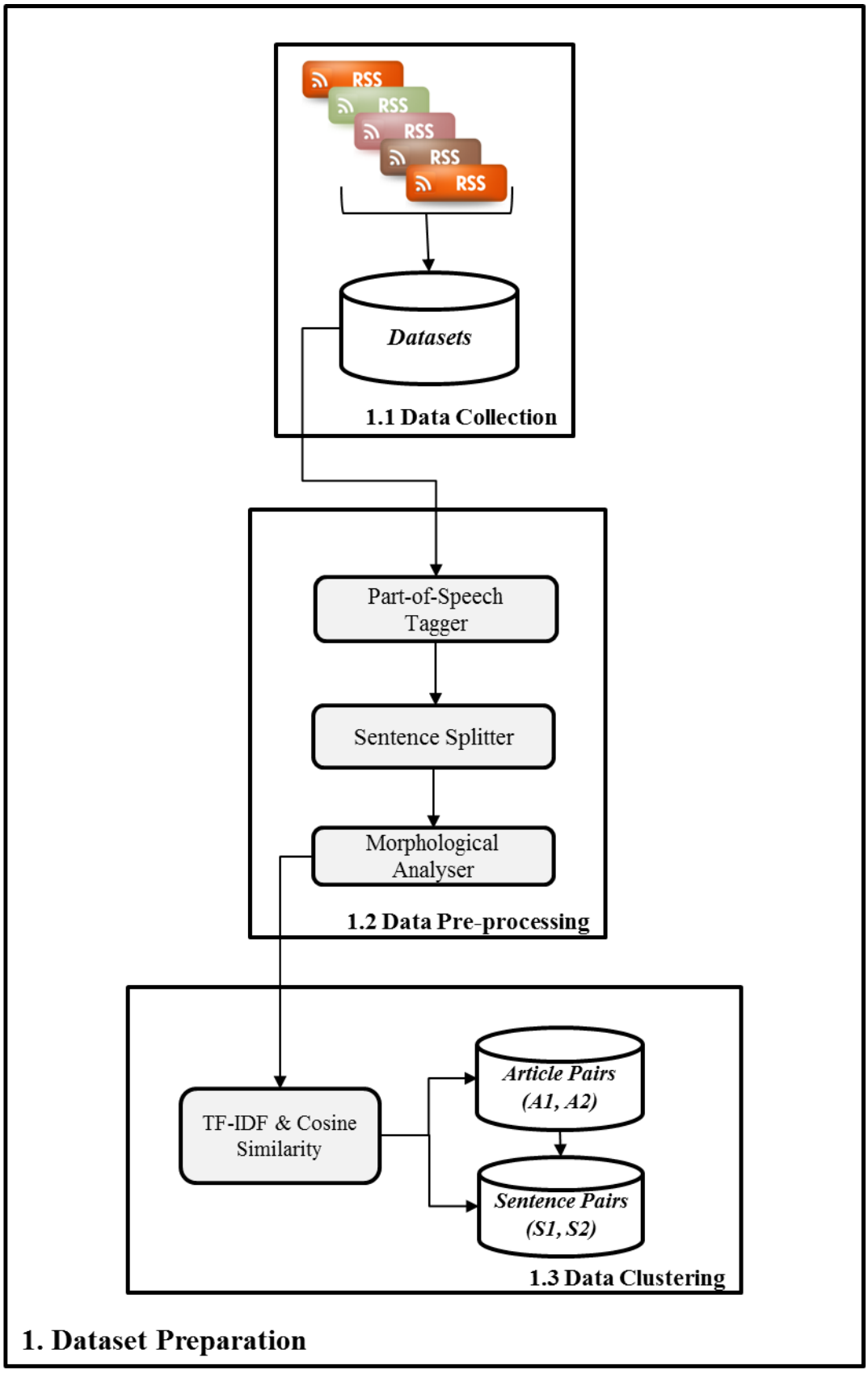


Figure 5.2: Box 1 of dataset preparation in Figure 5.1.

5.2.1 Data Collection (box 1.1 from Figure 5.2)

Existing paraphrase datasets for English and Arabic are not suitable for this research, as pointed out in Section 4.2.1, due to the different mechanisms used for collecting these datasets and lack of availability of some of them. The datasets created here are built upon specified machinery in order to fit the criteria for this system. We chose not to manually select sets of sentence pairs, partly because doing so is a lengthy and tedious process, but more importantly because hand-coded datasets are likely to embody biases introduced by the developer.

Therefore, the aim of building a comparable corpus is to collect pairs of sentences which are likely to contain paraphrase fragments by using the articles extracted automatically from an online newswire. To achieve this aim, RSS feeds have been used, because they have a set structure, which is easy to use and to access providing specific data such as title, date, time, and summary. In addition, a regular expression classifier has been used to clean all unwanted contents from html-coded pages (e.g. comments, HTML tags and other non-human readable elements). The detailed sources of collecting these datasets are now explained as following.

English Dataset Sources

For English newswires, the following feeds were selected: BBC English `http://www.bbc.co.uk/news`, The Guardian News `www.theguardian.com/uk`, Independent News `www.independent.co.uk/news/uk/rss`, Reuters `uk.reuters.com/news/uk`, and Express News `feeds.feedburner.com/daily-express-uk-` news websites as sources for English data.

Arabic Dataset Sources

For Arabic newswires, the following feeds were selected: BBC Arabic `www.bbc.com/arabic`, Sky News Arabia `www.skynewsarabia.com/web/home`, Al Jazeera `http://www.aljazeera.net/`, AL Riyadh `www.alriyadh.com/`, Alsharq `http://www.al-sharq.com/`, Okaz `www.okaz.com.sa/new/rss/f_rss.xml` websites as sources for Arabic data.

5.2.2 Data Pre-processing (box 1.2 from Figure 5.2)

Step 1: Part of Speech POS Tagging

In Figure 5.2 we carry out POS tagging **before** sentence-splitting. This may seem counter-intuitive, but it is essential for Arabic, since full stops in Arabic text tend to mark segments that are more like paragraphs than simple sentences. In order to split Arabic texts into sections each of which conveys a single idea (i.e. into segments that correspond to sentences in English), it is essential to find places where conjunctions are being used to separate such sections. This is made more complicated by the fact that Arabic conjunctions are written as clitics attached to the following word, and hence cannot be identified until after stemming/morphological analysis has been carried out. Tagging thus necessarily precedes sentence splitting for Arabic. The order in which these two steps are carried out in English makes very little difference, so we do tagging and then sentence splitting for both languages.

English POS tagging

For English, the POS tagging in the NLTK[1] toolkit has been used. The POS tagger has been tested on a number of sentence pairs from the datasets and it is reliable in performing the required process.

Arabic POS tagging

The challenges in Arabic are different to English due to its different morphology, lexicon and syntax. Thus, Arabic requires a different POS tagger from that used for English. Two state-of-art POS taggers for Arabic were found embedded in the toolkits of AMIRA (Diab 2009) and MADA[2] (Habash et al. 2009). These toolkits achieve state-of-the-art accuracy in Arabic tagging. However, we use Ramsay and Sabtan (2009)'s Maximum-likelihood (MXL) tagger in the current experiment for the reasons explained in the following paragraph.

This tagger was originally trained on the Quran, on which it obtained similar accuracy to AMIRA and MADA. It was updated by (Alabbas and Ramsay 2012b) to work on Modern Standard Arabic (MSA). Using this tagger allows for better flexibility in the control of the tag sets than the other taggers, which have built-in

[1] `http://www.nltk.org/api/nltk.tag.html`

[2] This acronym comes from "Morphological Analysis and Disambiguation for Arabic".

tagsets. Moreover, it is written in Python, and hence it is easy to integrate it with the rest of the architecture, whereas the other 'black box' taggers are difficult to integrate in Python (Alabbas and Ramsay 2012a).

Alabbas and Ramsay (2012b) carried out a number of experiments using a combination of MXL, MADA, and AMIRA. In order to do this, a common tagset had to be used for the three taggers where in every case the coarser of the MADA and AMIRA tagsets was used. For instance AMIRA has five tags for verbs (VB, VBG, VBD, VBN, VBP), which where mapped to the three tags for verbs (IV, PV, CV) used by AMIRA, whereas MADA used eight tags for particles, which were mapped to a single tag as used by AMIRA. Table 5.1 shows the merged tagset.

ABBREV	IV	PREP
ADJ	NOUN	PRON
ADV	NOUN_PROP	PUNC
CONJ	NUM	PV
CV	PART	REL_ADV
DEM_PRON	POSS_PRON	REL_PRON
INTERROG_PRON		

Table 5.1: MXL tags

(5.1) shows an Arabic sentence that was tagged by the MXL tagger. In Table 5.2 the actual results are presented.

Example 5.1.

Arabic:	شارك ستة جراحين في العملية
BW:	$Ark stp jrAHyn fy AlEmlyp.
English gloss:	Participated six surgeons in theprocess.

Input sentence: $Ark stp jrAHyn fy AlEmlyp.	
MXL Tagger	[('$Ark', 'PV'), ('stp', 'NOUN'), ('jrAHyn', 'NOUN'), ('fy', 'PREP'), ('AlEmlyp', 'DET- NOUN'), ('.', 'PUNC')]

Table 5.2: XML output for the Arabic sentence.

Before running POS tagging, all Arabic articles were first translated into Buckwalter translation, which is a strict transliteration of Modern Standard Arabic orthographical[3] symbols using only 7-bit ASCII characters. However, to be

[3]Orthography is a learnable human technology consisting of 1) a set of characters and 2) conventions for using them to make language "visible".

suitable for this system, a translator was developed based on the Buckwalter translator but with simple modifications and differences. The modification and normalisation processes are discussed below:

1. The aleph with hamza above (أ), aleph with hamza below (إ), and alef (ا) are all transformed into A. In order to maintain consistency it is better to drop hamza from all text to reduce noise and data sparsity.The reason for this is that the newswire sources used are written by different editors, not all of whom use hamza in their writing.

   Example (5.2) contains a sentence pair from different newswire sources (S_1) from BBC Arabic, and (S_2) from Al Jazeera websites.

   Example 5.2.

   (S_1) دعت وزارة الدفاع ا لا مريكية القراصنة ا لا مريكيين لا ختبار مدى قوة شبكتها المعلوماتيه

   (S_2) دعت وزارة الدفاع ا لأ مريكية القراصنة ا لأ مريكيين لإ ختبار مدى صلابة شبكتها المعلوماتيه

   As seen in (5.2), there are differences in matching the two sentences because sentence (S1) does not include hamza and sentence (S2) does include it. Therefore, the hamza is removed from the translation to avoid this problem.

2. All tashkeel letters (diacritical marks) اَ، اِ، اُ، اً، اٍ، اٌ، اّ، آ are removed. These modifications were made in order to avoid causing a contradiction with the special characters. Most people do not write the diacritics, so on those occasions when they do write them it is just confusing, as shown in the example below:

   Example 5.3.

   S_1: و اوضح قائلاً ان عربسات ليست جسماً صناعياً ذكياً فحسب

   BW: w AwDH qA}lAF An ErbsAt lyst jsmAF SnAEyAF *kyAF fHsb

   English gloss: And he-explained that-Arabsat is-not only a-body artificially intelligent

   S_2: و اوضح قائلا ان عربسات ليست جسما صناعيا ذكيا فحسب

   BW: w AwDH qA}lA An ErbsAt lyst jsmA SnAEyA *kyA fHsb

   English gloss: And he-explained that-Arabsat is-not only a-body artificially intelligent

As seen in (5.3), there are differences in matching the two sentences because sentence (S_2) does not include diacritical marks and sentence (S_1) does include them. To avoid this problem we decided to remove the diacritical marks.

3. All numbers written in Arabic (٠، ١، ٢، ٣، ٤، ٥، ٦، ٧، ٨، ٩) are transformed into 'Arabic numbers' (0, 1, 2, 3, 4, 5, 6, 7, 8, 9). The Buckwalter translation does not include the numbers. Thus, we add it to the translator to avoid the problems that occur from these numbers in the programming.

Step 2: Sentence Splitter

This step is important in order to split the text into manageable sizes of sentences, which can look at inside it for paraphrases. Thus, the obvious thing to do is identifying which full stops mark abbreviations and use them as shown in the following sections.

English Splitter

To split the articles into sentences and then into words in English we simply use NLTK tokeniser for Natural language processing[4]. The NLTK sentence splitter depends largely on looking for punctuation marks, but is sensitive to cases where full stops are being used in abbreviations ("Mr. Smith came into the room."). So we do not find any difficulties when splitting the articles into sentences. However, it is useful to use the splitter after tagging in Arabic, because the sentence is often very long and we need to split it into more than one sentence by looking at the conjunctions if it gives a complete sentence (as discussed in the next subsection).

[4] `http://www.nltk.org/api/nltk.tokenize.html`

Arabic Splitter

Arabic sentences can be extremely long – more like a paragraph made of numerous sentences split by using conjunctions than a 'A set of words that is complete in itself, typically containing a subject and predicate, conveying a statement, question, exclamation, or command, and consisting of a main clause and sometimes one or more subordinate clauses' [5]. Hence splitting sentences by full stops is not enough. This means that sentence splitters, such as the standard NLTK splitter and the Arabic splitter SAFAR (Software Architecture For Arabic language pRocessing) (Souteh and Bouzoubaa 2011) that work largely by identifying full stops are not suitable for our purpose since they leave very long sentences untouched.

Due to this lengthy nature in the Arabic language, a special sentence splitter was created. This sentence splitter was designed to improve the quality of the splitting process by determining a set of cases in order to divide Arabic sentences into acceptable chunks as follows:

1. Remove all these Arabic punctuation marks =, [,], {, }, -, " ", ؛, (,), :, ..., ،

2. Two cases must be taken into account when the next word is a dot (full stop), which is quite ambiguous:

   (i) If the dot is part of an abbreviation (person, location, organization, currency, etc.), it does not represent the end of a segment. Table 5.3 shows some Arabic abbreviations, such as:
   د. طارق الحبيب عالج أمراض متعددة
   Dr. Tariq Alhabib treated multiple diseases.

   (ii) If the dot is followed by numbers such as 0.2% or 5.6, then the dot does not represent the end of a segment.

3. Arabic tends to have very long sentences joined by conjunctions, and it would be good to split such long sentences into their constituents. Not all conjunctions link sentences – conjunctions can, for instance, be used to make complex NPs out of simpler ones. The strategy we follow is to split at conjunctions unless the conjunction is between two nouns or two verbs,

[5] `https://en.oxforddictionaries.com/definition/sentence` definition of an English sentence

Arabic abbreviation	Name	English abbreviation
د.	دكتور / Doctor	Dr.
أ.د.	استاذ دكتور / Professor	Prof.
م.	مهندس / Engineering	Eng.
ص.ب.	صندوق بريد /Post Office	P.O.
س.ت.	سجل تجاري / Trading Record	T.R.
أ.ش.أ	وكالة أنباء الشرق الأوسط / Middle East News Agency	MENA
ر.س.	ريال سعودي / Saudi Royal	S.A.R.

Table 5.3: Some Arabic abbreviations.

since in these cases doing so does not give a complete sentence. In other cases of conjunctions we do generally obtain complete sentences, as shown in these two examples.

Example 5.4.

Arabic:	احمد يبيع و يشتري في السيارات المستعملة و يشتري والده الاثاث القديم
BW:	Alqdym AlAvAv wAldh y$try w AlmstEmlp AlsyArAt fy y$try w ybyE AHmd
English gloss:	old furniture his father buys and used cars in buys and sells Ahmad
Translation:	Ahmad buys and sells used cars and his father buys old furniture

Selecting the first conjunction and in segment (5.4) will give only the verb 'buys, since it links two VPs (i.e., *'buys'*, *'sells'*), which is not a complete sentence. In contrast, selecting the second conjunction and will give two complete sentences *Ahmad buys and sells used old cars* then *'his father buys old furniture'*, since it links the two sentences.

Example 5.5.

Arabic:	ذهب احمد و محمد الى صالة الالعاب الرياضية و ذهبت اختهم سارة الى المكتبة
BW:	Almktbh AlY sArh Axthm *hbt w AlryADyh AlAlEAb SAlp AlY mHmd w AHmd *hb
English-gloss:	the-library to their-sister went and the-gym to went Mohammad and Ahmad
Translation:	Ahmad and Mohammad went to the gym and their sister went to the library

Selecting the first conjunction and in segment (5.5) will give only the proper noun *Ahmad*, since it links two NPs (e.g., *'Ahmad' and 'Mohammad'*). In contrast, selecting the second conjunction will give two complete sentences *'Ahmad and Mohammad go to the gym'* then *'their sister goes to the library'*, since it links between two sentences.

As noted above, simply splitting on full stops, as done by the standard NLTK sentence splitter and the sentence splitter from SAFAR (Souteh and Bouzoubaa 2011) misses these cases; but at the same time splitting at every conjunction leads to numerous false positives, as in 5.4 and 5.4. The principles outlined above were captured in a set of regular-expression-like patterns for identifying places where conjunctions were being used as sentence boundaries. These patterns make use of the POS tags of the surrounding words, which is why we cannot do sentence splitting before we do tagging. These rules were developed following inspection of a set of 20 articles, and were then informally evaluated on a disjoint set of 10 articles. F-measure on these articles was compared with manual annotation by the author. The annotation cannot be guaranteed to be unbiased, but it does suggest that this splitter is fit for the current task. The test set contained 176 sentences. The splitter found 159, of which 15 were not in the test set as annotated by the author. There were thus 144 true positives (159-15) and 32 false negatives (176-144), so the precision was $144/159 = 0.91$ and the recall was $144/176 = 0.88$, giving an F-measure of 0.85. Given that the aim of the splitter was to break the very long sentences that are found in Arabic texts into chunks that are small enough to be compared, precision is more important for our work than recall, and a value of 0.91 for precision suggests that the splitter is good enough to be used for our task. This is not to say that it would be suitable for other tasks, but it is adequate in the present context where we are simply trying to break the text into manageable pieces within which we can look for paraphrases.

Step 3: Morphological Analysis

The third stage of pre-processing is morphological analysis; this step is used to realise if the two different forms of the same word are actually similar. But it also introduces an additional level of ambiguity, which raises a problem. At a morphological level, the same word with different forms may have different meanings in the context, which causes ambiguity problems, see examples (5.6)

and (5.7).

English Morphological Analyser

The NLTK morphology analyser[6] was used to return the root form of a word. Morphology uses a combination of inflectional ending rules and exception lists to handle a variety of different possibilities as explained in Chapter 4. A word may have different meanings depending on the context at the syntax and semantic level. An example of a problem that occurs at the semantic level is as follows:

'The fisher went to the bank'.

The term bank could refer to an edge of a river or to a financial institute.

Arabic Morphological Analyser

Arabic has a rich morphology, based on the roots, dependent on vowel changes, which makes morphological analysis very complex (Froud et al. 2013), as shown in Figure 5.4. Arabic is also an inflectional language, where inflection is a process that adds affixes to a word to produce several forms of the same word (Alabbas 2013a). Therefore, the Pyaramorph analyser was used, which is a Python reimplementation of the Buckwalter Arabic Morphological Analyser (BAMA) (Buckwalter 2004). Pyaramorph contains a large set of tables to represent information about roots and affixes to propose potential morphological analysis of Arabic word forms.

The problems that have been encountered when using Pyaramorph were:

1. It is based on a fixed vocabulary. Thus, quite a few words are missing and for some words their roots cannot be found. For example: 'بازغة / bAzgp', which means 'shining', is not covered in Pyaramorph. It could be used as in sentence 5.6:

[6]http://www.nltk.org/howto/wordnet.html

Example 5.6.

Arabic:	رايت الشمس بازغة
BW:	bAzgp Al$ms rAyt
English gloss:	shining the-sun I-saw
Translation:	I saw the sun was shining

2. Arabic is ambiguous, and it is not the job of a morphological analyser such as 'Pyaramorph' to choose between the alternative analyses. This leads to considerable ambiguity, for instance: كتب 'ktb', as seen in Figure 5.3.

In Figure 5.4, the word درس 'drs' with different diacritic marks produces different meanings which leads to ambiguity.

```
analysis for: كتب ktb

  solution: (كَتَبَ kataba) [katab-u_1]
       pos: katab/VERB_PERFECT+a/PVSUFF_SUBJ:3MS
     gloss: ___ + write + he/it <verb>

  solution: (كُتِبَ kutiba) [katab-u_1]
       pos: kutib/VERB_PERFECT+a/PVSUFF_SUBJ:3MS
     gloss: ___ + be written;be fated;be destined + he/it <verb>

  solution: (كُتُب kutub) [kitAb_1]
       pos: kutub/NOUN
     gloss: ___ + books + ___
```

Figure 5.3: Using Pyaramorph to analyse 'ktb'.

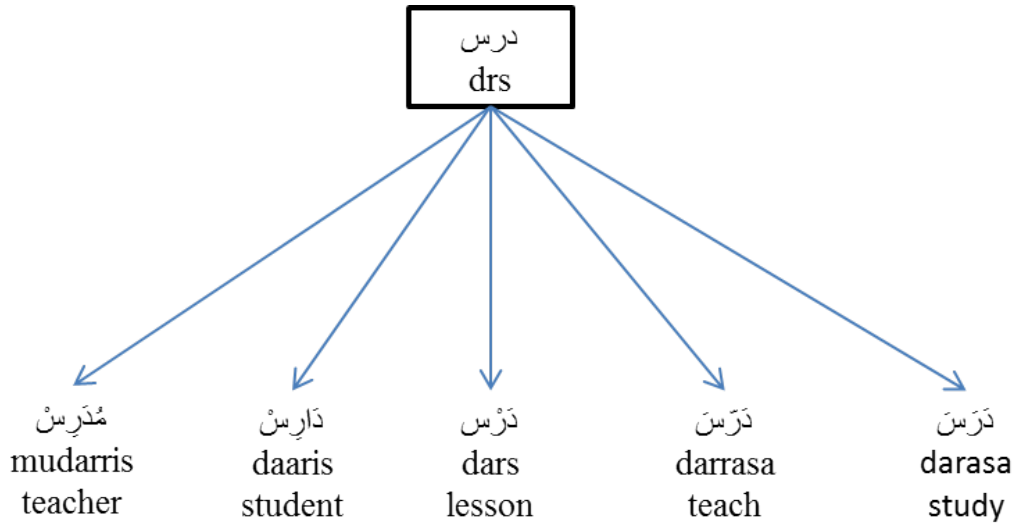


Figure 5.4: Ambiguity caused by the diacritics (one word with multiple meanings).

The representation of this word درس 'drs' in various sentences with different meaning is given in examples (5.7) and (5.8).

Example 5.7.

Arabic:	هو دَرَسَ في الجامعة
BW:	AljAmEp fy darasa hw
Translation:	He studies at the university

Example 5.8.

Arabic:	هو دَرَّسَ في الجامعة
BW:	AljAmEp fy darša hw
Translation:	He teaches at the university

This ambiguity causes the potential problems when we calculate the similarity. We will discuss this in more detail in Section 5.3.2.

5.2.3 Data Clustering (box 1.3 from Figure 5.2)

We have now preprocessed our initial large dataset. The next move is to try to find a balanced subset to give to the annotators. To do this we started by clustering the datasets into pairs of articles and afterward into sentence pairs that are likely to contain candidate paraphrases. To achieve this, two standard similarity techniques were used: cosine similarity and tf-idf vector, as mentioned in Section 4.3. These techniques were applied to English and Arabic datasets separately. This resulted in a large quantity of data for each language. Then, a sensible threshold had to be determined by manual investigation that could be used to filter the sentences and select pairs of sentences that were plausibly related. A specific threshold was required because the annotators who were asked to mark up sentence pairs are volunteers, and had to be given the minimum amount of work to keep them motivated. Therefore, the annotators were provided with a sensible amount of data to annotate instead of thousands of unrelated sentences pairs.

The Performance of the Clustering Process

The datasets collected from the different newswire sources were very large, and we wanted to give our annotators a reasonable number to deal with, so we had to make a selection. This selection needed to be balanced, i.e., to contain about 50% of similar cases and 50% of dissimilar ones. We therefore ordered the entire dataset for each language in terms of their cosine scores, and took the top 500 sentence pairs for further investigation. We carried out an informal annotation of these 500 sentence pairs, in order to establish the cosine similarity, and TF-DF vector threshold at which about 50% were similar and 50% were dissimilar. Tables 5.4 and 5.5 show the proportion of sentences that were similar at a range of cosine scores. From these tables, it emerges that a cosine score of about 0.6 will give a roughly even split between similar and dissimilar cases.

Using this threshold enabled us to select a subset of 300 sentence pairs which were roughly balanced from the original dataset (i.e, 2.9k English sentence pairs, and 3k Arabic sentence pairs) to give to our annotators. Reducing the original set in this way meant that we were able to get the data annotated by all five annotators for each language Without overwhelming them. It is this data that was used in the experiments described below.

Threshold	Proportion of similar cases
≥ 0.9	80%
≥ 0.8	73%
≥ 0.7	65%
≥ 0.6	57%
≥ 0.5	48%

Table 5.4: The proportion of similar cases for different thresholds in English.

Threshold	Proportion of similar cases
≥ 0.9	75.4%
≥ 0.8	66%
≥ 0.7	57%
≥ 0.6	48%
≥ 0.5	40%

Table 5.5: The proportion of similar cases for different thresholds in Arabic.

5.3 Similarity Checking

The datasets produced by the first stage of the experiment were then used for checking the degree of similarity between the sentence pairs according to human- and system-based judgment. The results of human-based judgment were used to create a 'Gold Standard' for English and Arabic languages by annotating their datasets. The English and Arabic datasets were then measured by the systems to create the similarity scores, as discussed in the next section.

5.3.1 Human-Based Judgment

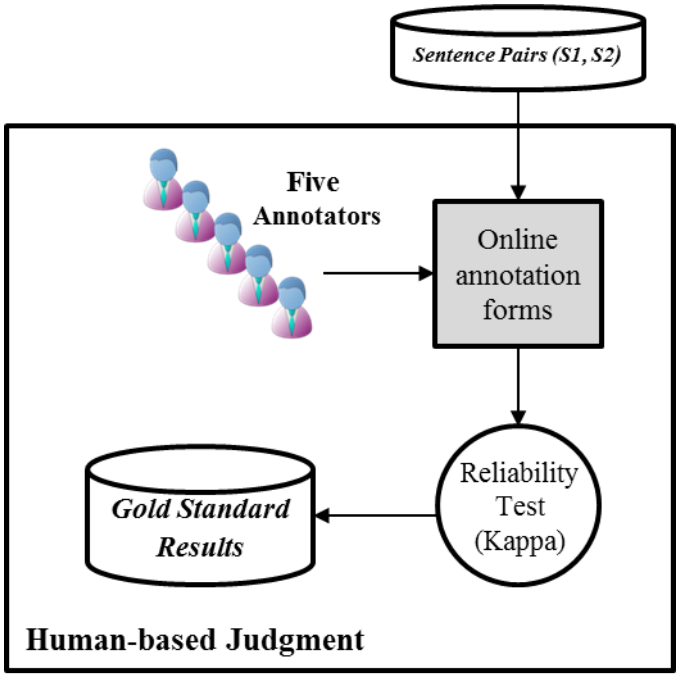


Figure 5.5: Box 2.1 of similarity checking in Figure 5.1 for human-based judgment.

The datasets were assessed by human judgment by annotating them, and then the reliability of the annotators using inter-annotator-reliability for both languages was measured (see Figure 5.5).

The aim of the human judgment was to define the 'Gold Standard'. To achieve this, the online annotation tool was developed and distributed to the annotators for assessing the sentence pairs, and a statistical measure was then used to assess the reliability of agreement among the annotators called Fleiss′s kappa measure, as explained in Section 4.4.2.

English Dataset Annotations

Five expert and non-expert volunteer annotators, all English native speakers, were asked to annotate the 300 pairs of sentences by choosing'YES' or 'NO', meaning that they agreed the two sentences were the same meaning or that they differed significantly. These pairs cover a number of subjects such as politics, business, sport and general news. Those annotators followed nearly the same

annotation guidelines as those for building the TEQV task dataset (see Section 2.3.2).

Since the evaluators were distributed widely, an online annotation form was developed, shown in Figure 5.6, which made their task easier. The form showed the annotator pairs of sentences and asked them ***Do these two sentences have the same meaning?*** to mark up this pair.

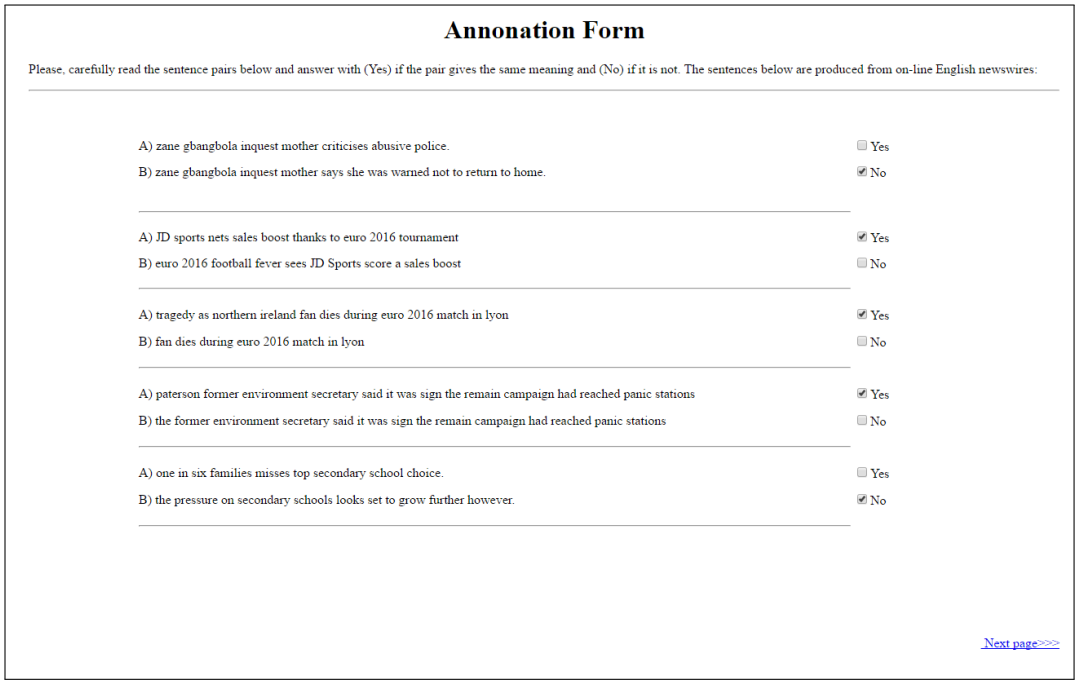


Figure 5.6: English annotation form.

When all the annotators had finished and submitted their results, another test was conducted on their reliability as annotators. In order to detect anyone who had not done the task correctly, and hence remove their judgments from the Gold Standard. The inter-annotator-reliability was calculated using Equation 4.3.

	Annotators ID					Mean
	ANT1	ANT2	ANT3	ANT4	ANT5	
Kappa for ANT's coannotators	0.52	0.55	0.58	0.40	0.57	0.52

Table 5.6: Reliability measures of English annotators.

The key observation from Table 5.6 is that most kappa values are in the range $(0.50 - 0.59)$, which is included within the range $(0.41 - 0.6)$ that Landis and Koch (1977) and Altman (1990) refer to as a moderate level of agreement

(see Section 4.4.2). The divergence between the kappa, including the annotator kappa and the kappa of their coannotators is comparatively slight, except for ANT4. Since the average of both kappa rates for all annotators was 0.52, this represents a moderate level of agreement among the annotators, i.e., Table **??**, has two annotators (ANT3, and ANT5) whose kappa rates are higher than those of their coannotators. The other annotators have kappa rates lower than those of their coannotators, but these differences are slight. The findings of these interrateagreement rates suggest that all the annotators were reliable and their annotated dataset can be used in this work except ANT4, whose result is not accurate and out of range. It was therefore decided to remove this annotator from the list and recalculate with four annotators, which raised the average kappa to 0.56.

Arabic Dataset Annotations

Five expert and nonexpert volunteer annotators, all Arabic native speakers, were asked to annotate the 300 pairs of sentences chosen with 'YES' or 'NO', meaning that they agreed the two sentences were the same meaning or that they differed significantly. These pairs cover a number of subjects such as politics, business, sport and general news. Those annotators followed nearly the same annotation guidelines as those for building the TEQV task dataset (see Section 2.4.2). An online annotation form was created for Arabic sentence pairs, as shown in Figure 5.7, which made their task easier by distributing to the annotators. Again, the form showed the annotators Arabic sentence pairs, and asked them ***Do these two sentences have the same meaning?*** to mark up.

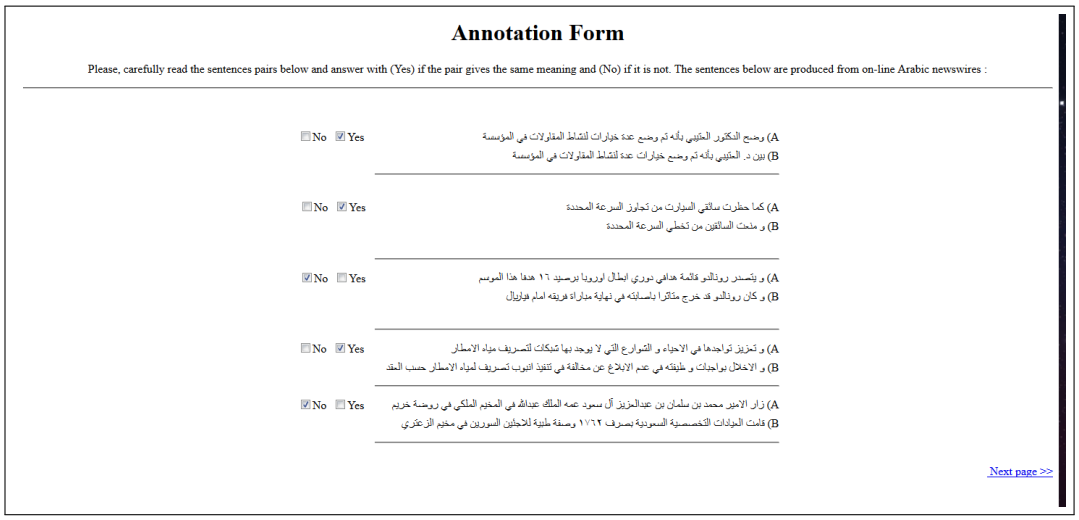


Figure 5.7: Arabic annotation form.

The inter-annotator-reliability was calculated using Equation 4.3 to measure the reliability of annotators. The reason of using this test is to detect anyone who had not done the task correctly and hence remove their judgments from the Gold Standard. The result of this experiment is equal to 0.44, which indicates a moderate agreement.

	Annotators ID					Mean
	ANT1	ANT2	ANT3	ANT4	ANT5	
Kappa for ANT's co-annotators	0.41	0.49	0.41	0.42	0.47	0.44

Table 5.7: Reliability measures of Arabic annotators.

The key observation from Table 5.7 is that most kappa values are in the range $(0.40 - 0.50)$, which is also within the range $(0.41 - 0.60)$ that refer to as a moderate level of agreement (see Section 4.4.2). But, it is clearly that Arabic kappa is lower than English. The divergence between the kappa, including the annotator kappa and the kappa of their co-annotators is comparatively slight. Since the average of both kappa rates for all annotators was 0.44, this represents a moderate level of agreement among the annotators, i.e., $0.40 \leq \text{kappa} \leq 0.59$. Table 5.7 has two annotators ANT2 and ANT5 whose kappa rates are higher than those of their co-annotators. The other annotators have kappa rates lower than the kappa rates of their co-annotators, but these differences are slight. The findings of the inter-rate-agreement rates suggest that all the annotators were

reliable and their annotated dataset can be used in this work.

Note that the English speakers were much more consistent and the Arabic speakers were less so. That was expected since Arabic is more ambiguous compared to English. The task is hard for Arabic speakers, and therefore the system developed here is expected to produce less accurate results in Arabic than in English (if it is hard for native speakers, as suggested by the lack of agreement, then it is likely to be hard for a computer). Some cases of sentence pairs were hard for the Arabic speakers to take a decision on. As shown in the following examples.

1. The omission of short vowels:

   Example 5.9.

S_1:	فقد سمعته قائلا انا برئ
BW:	br} AnA qA}lA smEth fqd
Translation:	I heard him saying I am innocent
	Or
	He lost his reputation saying I am innocent

   In (5.9), the sentence pair is ambiguous. The phrase (فقد سمعته, fqd smEth) has two meanings: *and so I had heard him* 'f qd smEth' or *he lost his reputation* 'fqd smEth'. Because of this ambiguity, the annotators found it hard to distinguish the exact meaning.

2. Sentence pair with ambiguity structure:

   Example 5.10.

S_1:	شاهدت الولد بالنظارة
BW:	bAlnZArp Alwld $Ahdt
Translation:	I saw the boy with the glasses

   In this case the pair of sentences is ambiguous. Sentence (S_1) has two meanings, "*The boy has glasses*" or "*I used the glasses to see the boy*" (the seeing was done with glasses).

3. Some ambiguity arises because one of the sentences is a generalisation of the other rather than being a true paraphrase – some of the annotators marked these positively and some negatively.

Example 5.11.

S_1:	الرئيس الامريكي باراك اوباما يزور عددا من الدول الاوروبية لتعزيز العلاقات
BW:	AlElAqAt ltEzyz AlAwrwbyp Aldwl mn EddA yzwr AwbAmA bArAk AlAmryky Alr}ys
Translattion:	President US Barack Obama is-visiting a-number of countries European to-strengthen relations
S_2:	الرئيس اوباما يزورهولندا لتعزيز العلاقات الاوروبية
BW:	AlAwrwbyp AlElAqAt ltEzyz hwlndA yzwr AwbAmA Alr}ys
Translation:	President Obama is visiting Netherlands to strengthen relations with European

Example 5.12.

S_1:	تنزيل ماسنجر فيسبوك و تحديث تطبيقات الهواتف
BW:	AlhwAtf tTbyqAt tHdyv w fysbwk mAsnjr tnzyl
Translation:	Messenger Facebook has been download and updates applications phone
S_2:	تحميل فيسبوك ماسنجر و تحديث المكالمات بالفيديو
BW:	bAlfydyw AlmkAlmAt tHdyv w mAsnjr fysbwk tHmyl
Translation:	Facebook Messenger has been download and updates video call

The problem seems to be that some annotators were cautious about the generalisation to customisation. In sentence pairs (5.11) Netherlands is included in the term of European. The same problem occurs in sentence pair (5.12), where video calling is a part of phone applications.

4. Some meanings of the abbreviations are not known or unfamiliar.

Example 5.13.

S_1:	اعلنت ناسا عدم توافر اللعبة الاشهر حاليا بيكمون جو لرواد محطة الفضاء
BW:	AlfDAG mHTp lrwAd jw bykmwn HAlyA AlA$hr AllEbp twAfr Edm nAsA AElnt
Translation:	NASA announced that the current famous game Pokémon GO is not available to the astronauts
S_2:	اعلنت الإدارة الوطنية للملاحة الجوية والفضاء حظر لعبة بيكمون جو لرواد الفضاء
BW:	AlfDAG lrwAd jw bykmwn lEbp HZr wAlfDAG Aljwyp llmlAHp AlwTnyp AlAdArp AElnt
Translation:	National Aeronautics and Space Administration announced the banning of Pokémon GO.

Example 5.14.

S_1:	الناتو يعلن تضامنه بقوة مع تركيا في حملتها على تنظيم الدولة الاسلامية
BW:	AlAslAmyp Aldwlp tnZym ElY HmlthA fy trkyA mE bqwp tDAmnh yEln AlnAtw
Translation:	NATO announce their strong commitment with Turkey in their campaign on Islamic State Organization
S_2:	حلف شمال الاطلسي يقدم الدعم السياسي لحملة تركيا على مسلحي تنظيم الدولة الاسلامية
BW:	AlAslAmyp Aldwlp tnZym mslHy ElY trkyA lHmlp AlsyAsy AldEm yqdm AlATlsy $mAl AHlf
Translation:	The North Atlantic Treaty Organization offers political support to Turkey′s campaign on militants of Islamic State Organization

The problem in (5.13) seems to be that some people are unfamiliar with meanings of some abbreviations (ناسا NASA, which is an abbreviation of الإدارة الوطنية للملاحة الجوية والفضاء The National Aeronautics and Space Administration). The same problem arises in Figure (5.14) where the word 'الناتو' is an abbrevation of the حلف شمال الاطلسي, which is not known to some people.

5. Some diseases have multiple names, but only familiar names are used:

Example 5.15.

S_1:	ينتشر مرض النقرس عند الرجال أكثر من السيدات
BW:	AlsydAt mn Akvr AlrjAl End Alnqrs mrD ynt$r
Translation:	Gout disease spreads in men more than women
S_2:	ينتشر داء الملوك عند الرجال أكثر من السيدات
Bw:	AlsydAt mn Akvr AlrjAl End Almlwk dAG ynt$r
Translation:	Kings disease spreads in men more than women

In the sentence pair (5.15) we noticed that some people only know the common name of a disease النقرس, but داء الملوك most of the people do not know it.

In this section we annotated the sentence pairs and measured the reliability of annotators, depending on the human judgments to create 'Gold standard'. In the next section we will explain how to measure the similarity based on the system-based judgment to create the similarity score

5.3.2 System-Based Judgment

The aim of system-based judgment is to automatically calculate the similarity scores between pairs of sentences. To achieve this, alignment methods have been used to align fragments of text between sentence pairs that are considered as plausible candidate paraphrases. The DTW algorithm was used for aligning the words while maintaining their original order, and the XDTW algorithm was used to allow transposition operations between adjacent words. These algorithms measure the minimum cost distance of operations (Insert, Delete, Exchange, and Swap) for converting one string to another. These operations used different cost functions in the same way as illustrated in the discussion of spelling correction in Section 4.5.3, depending upon the POS tags and the degree of similarity between the words in a sentence pair.

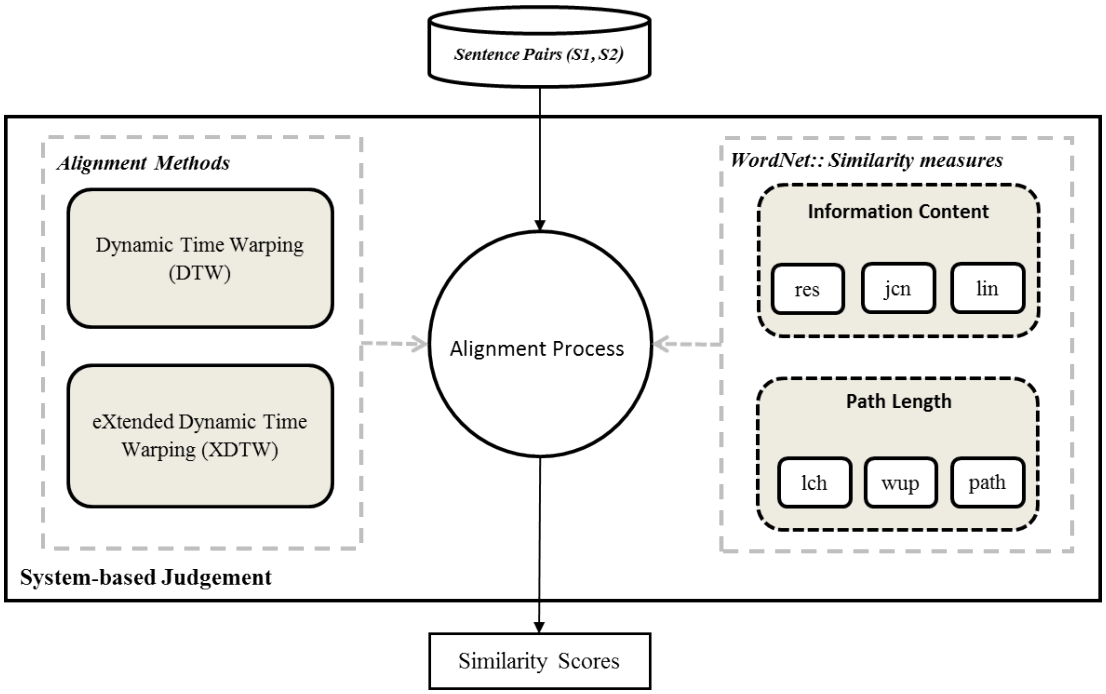


Figure 5.8: Box 2.2 of similarity checking in Figure 5.1 for system-based judgment.

English Dataset Similarity Measurement

To measure the degree of similarity between words in pairs of sentences in English, the DTW algorithm and its extension XDTW were used to transform one string into another. The cost functions for the operations Insert and Delete depend on

the POS tagging and for Exchange and Swap depend on the degree of similarity of the words involved. Variable cost for Exchange and Swap operations was used, which means that the cost of Exchange should be lower for similar words than for dissimilar ones.

A range of different WordNet similarity measures were used in the DTW and XDTW algorithms as cost functions for the Exchange and Swap operations, as shown in Figure 5.8. Three of the six WordNet semantic similarity measures are based on the content of information: Resnik (**res**), Lin (**lin**) and Jiang & Conrath (**jcn**), with the other three based on the length of the path: Leacock & Chodorow (**lch**), Wu & Palmer (**wup**) and Shortest Path (**path**) (see Section 4.6.4).

(5.16) shows how these algorithms measure the minimum cost of operations (Insert, Delete, and Exchange) to convert one string into another. In Figure 5.9, the similarity calculations process is depicted based on two algorithms; one represents the baseline (i.e., the DTW algorithm alone), and another algorithm is DTW with a range of WordNet similarity measures. We assume that for the baseline INS=2, DEL=2, and XCH=3, whereas when using the WordNet similarity measure we normalise the similarity measures to give a cost for XCH that lies between 0 and 3, as shown in Figure 5.9.

Example 5.16.

(S_1) Minimum wage to rise to £6.50 an hour

(S_2) Minimum wage to increase to £6.50 an hour

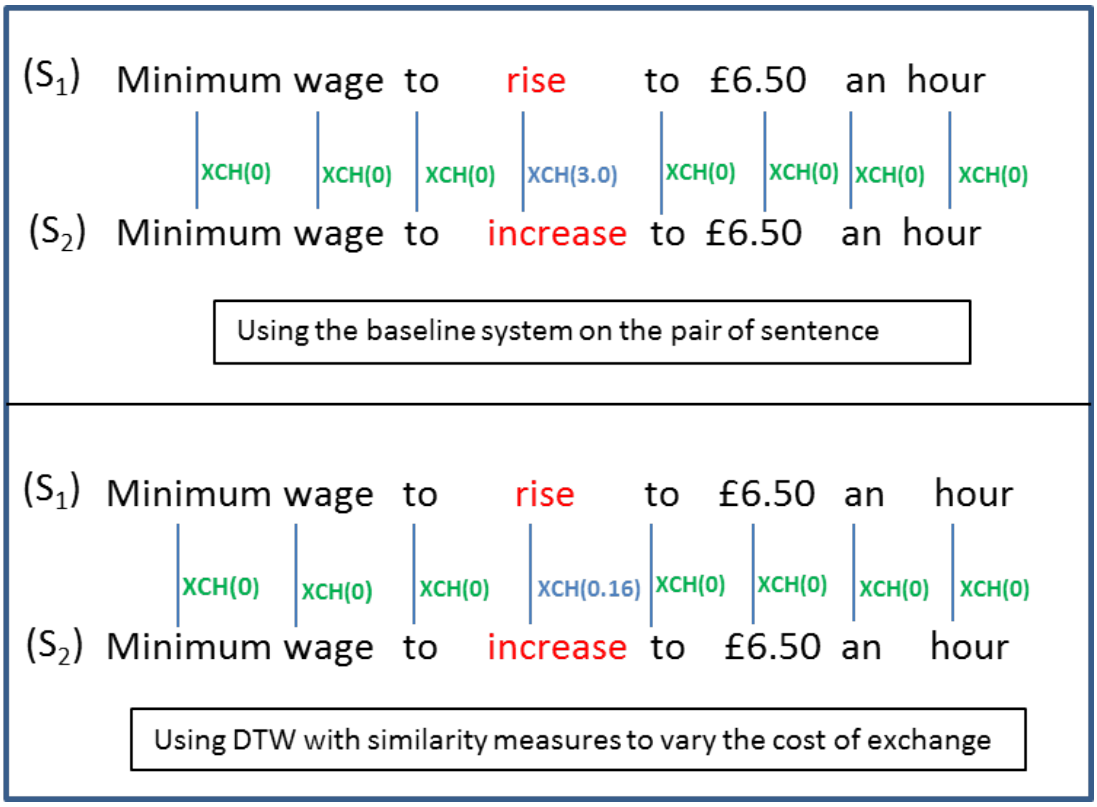


Figure 5.9: Using the baseline system and DTW with similarity measures on a pair of sentences.

The difference between the two sentences in (5.16) is shown in using the words *'rise'* and *'increase'*, which by logical sense are similar due to the relatedness of their words and meaning. So, to calculate the similarity scores between them in (5.16), we applied the two algorithms.

When we used the baseline, the cost of exchanging is 3. To calculate the similarity score, we used the formula (4.13) in Chapter 4 by taking the cost of exchanging and divided by the worst possible cost function, then subtracted the result from 1, which is leading to an overall score of (1- (3/24)) = 0.87.

In the DTW algorithm with similarity measures the cost for the Exchange operation will be determined by the value obtained from WordNet Similarity measures. In example 5.16, the cost of exchanging *'rise'* and *'increase'* is 0.16, leading to an overall score of $(1-(0.16/24)) = 0.99$. In other words, the fact that the two words to be exchanged are similar leads to an increase in the score of the overall sentences pair as seen in Figure 5.9, and this is a promising idea.

Moreover, if we take another sentence pair, as shown in Example 5.17, the

word selection is quite different (e.g., *'increase'* and *'cancel'*) as they give completely different meanings. If we tackle this with the baseline, then we will obtain a fixed cost result of value 3, which indicates to use a single XCH operation. However, if we use the WordNet similarity measures, we will obtain a high cost of 2.3 (see Figure 5.10), and that leads to an overall score of $(1 - (2.3/24) = 0.90)$. This Example shows that the distance between two words *'increase'* and *'cancel'* is affected by the cost function of the exchange operation. This is another promising idea in adopting WordNet similarity measures to aid in detecting the sentence similarity by meaning.

Example 5.17.

(S_1) Google decided to increase production of self-driving cars

(S_2) Google decided to cancel production of self-driving cars

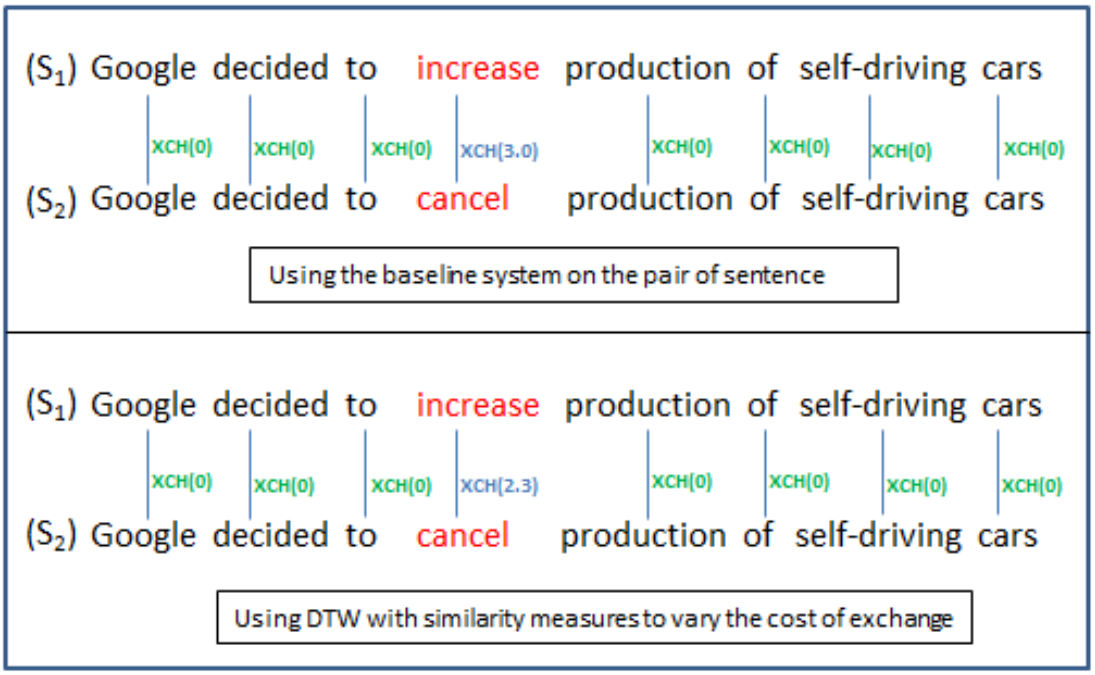


Figure 5.10: Using the baseline system and DTW with similarity measures on a pair of sentences.

However, this method does not always produce desirable results, as seen for instance in (5.18), where the obtained cost of substituting *'increase'* and *'decrease'* is 0.33, as shown in Figure 5.11, leading to an overall score of $(1 - (0.33/24)) = 0.98$. Therefore WordNet similarity measures are not always reliable and can be misleading in cases such as this.

Example 5.18.

(S_1) Google decided to increase production of self-driving cars

(S_2) Google decided to decrease production of self-driving cars

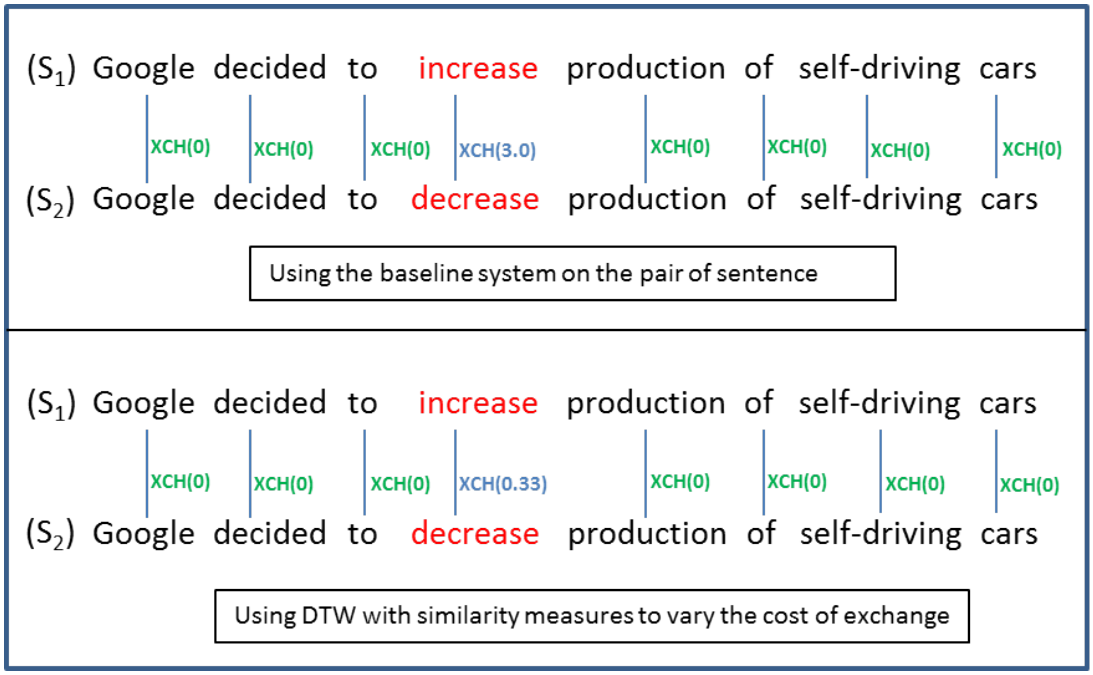


Figure 5.11: Using the baseline system and DTW with similarity measures on a pair of sentences.

In contrast, the pair of sentences in (5.19) contains the same words but in a different order. This will provide another challenge for the DTW algorithm in calculating the similarity score as it will calculate more operations of insertion and deletion as seen in Figure 5.13. This can be resolved by extending DTW (XDTW) algorithm to include another operation, which is 'SWAP'. This algorithm is more suitable for matching the adjacent words for substitutability between them (see Figure 5.12). Assuming that for the baseline INS = 2, DEL = 2, XCH = 3, and SWAP = 0.5, whereas when using the WordNet similarity measure we normalise the similarity measure to give a cost for XCH that lies between 0 and 3. In (5.19), we obtained a similarity score 0.5 when using XDTW, which is less than the cost of the similarity score 4.0 result of INS and DEL that using DTW. Notice that XDTW costs less than DTW because the swapping operation has been used, as shown in Figure 5.12. In addition, it is observed that changing the word order makes a difference in English, e.g., where it changes the statement

into a question. Also, in example5.20 changing one word 'sleeping' by 'dozing' in the sentence make a difference on the cost of exchanging, when using DTW and XDTW with the similarity measures as shown in 5.12.

Example 5.19.
(S_1) he is sleeping
(S_2) is he sleeping

Example 5.20.
(S_1) he is sleeping
(S_2) is he dozing

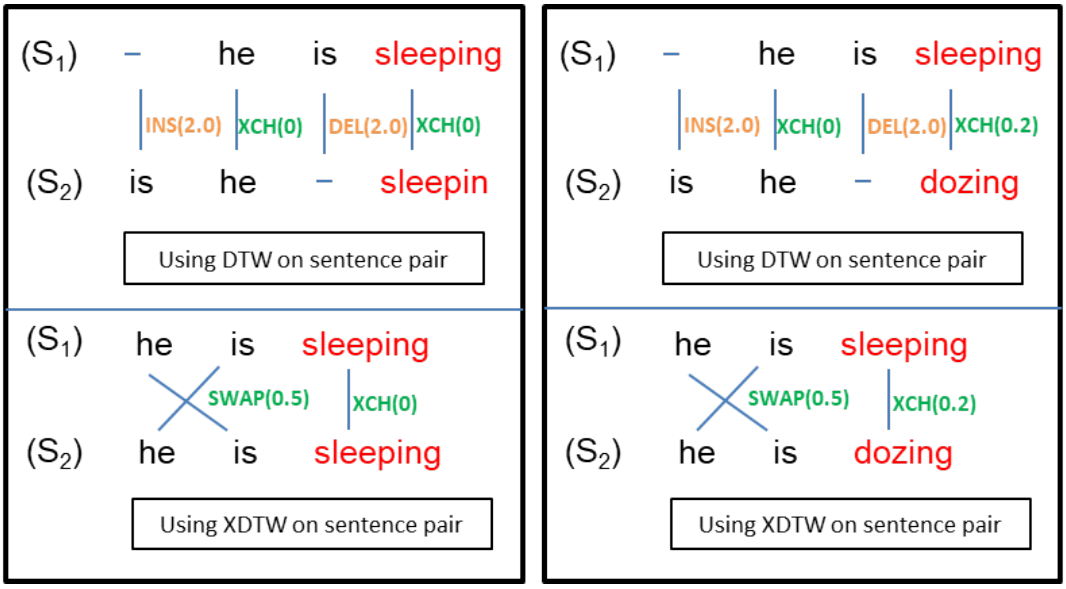


Figure 5.12: Using the DTW and XDTW algorithems on sentence pairs with swapped words.

As we mentioned before in Chapter 4, the cost of Insert and Delete depend upon different POS tags and Exchange depends upon the degree of similarity between them. So, adding an adverb or an adjective has less effect on the meaning of a sentence than adding a noun or verb. In (5.21) the adjective has been inserted, and in (5.22) the noun has been inserted as shown in Figure 5.13.

Example 5.21.
(S_1) The man is running
(S_2) The old man is running

Example 5.22.

(S_1) The man is running

(S_2) The man is running races

Example 5.23.

(S_1) The man is running

(S_2) The old man is sprinting

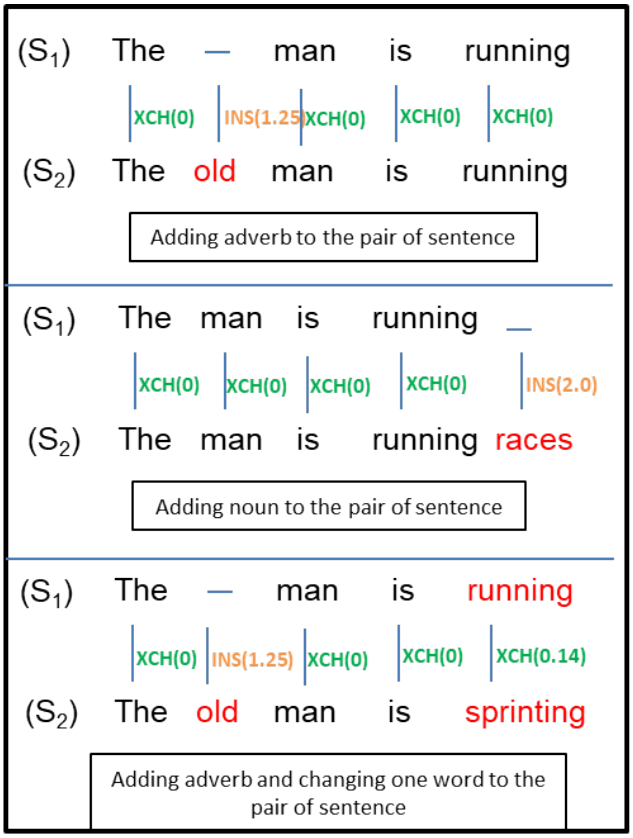


Figure 5.13: Adding words with different POS tags to a sentence results in different changes to the score.

In Examples 5.21 and 5.22 the two pairs of sentences are not the same. However, we notice that inserting the modifier (adjective) has less effect on the meaning of a sentence than adding a noun, where S_1 and S_2 in (5.21) cost 1.25 when we insert the adjective, while in (5.22) the cost is 2, which reflects the fact that adding a noun makes more difference to the meaning than adding an adjective. We notice in (5.23) that is changing one word 'running' by 'sprinting' make a difference in the cost of XCH when using DTW and XDTW with similarity measures as shown in 5.13.

Table 5.8 shows the absolute scores for the various measures, divided into three groups. The first group contains examples 5.16, 5.17, and 5.18. In this group, we notice that using DTW and XDTW with different similarity measures affected the cost of exchange but that there was no difference between using the two algorithms. This is unsurprising: there are no inversions of words in this group, so you would not expect DTW and XDTW to produce different results. In contrast, in the second group, we notice that for examples 5.19 and 5.20 DTW and XDTW produce different scores, since the order of *he* and *is* has been reversed. In 5.20 *sleeping* has also been replaced by *dozing*, so the different similarity measures also have an effect. The third group contains examples 5.21, and 5.22 where we notice that using different similarity or alignment algorithms does not affect the cost of insert or delete. In example 5.23, as with example 5.20, we have a combination of insertion/deletion along with a change of one word, so that we get different scores when using different similarity measures. We will investigate the relative effectiveness of these measures on more realistic examples in Chapter 6.

Example	DTW						XDTW					
	wup	lch	path	jcn	res	lin	wup	lch	path	jcn	res	lin
(5.16)	0.16	0.34	0.28	0.5	0.66	0.66	0.16	0.34	0.28	0.5	0.66	0.66
(5.17)	2.3	2.9	2.5	3.4	4.0	4.0	2.3	2.9	2.5	3.4	4.0	4.0
(5.18)	0.33	0.55	0.43	0.71	0.85	0.83	0.33	0.55	0.43	0.71	0.85	0.83
(5.19)	4.0	4.0	4.0	4.0	4.0	4.0	0.5	0.5	0.5	0.5	0.5	0.5
(5.20)	2.2	2.42	2.25	2.5	2.7	2.71	0.7	0.77	0.72	0.8	0.84	0.84
(5.21)	1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25
(5.22)	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0
(5.23)	1.39	1.7	1.45	1.95	2.05	2.05	1.39	1.7	1.45	1.95	2.05	2.05

Table 5.8: Scores for the six WordNet similarity measures for English.

Arabic Dataset Similarity Measurement

The same process was followed for Arabic. To measure the similarity between two sentences the DTW and its extension XDTW algorithms were used with a range of different WordNet similarity measures to convert one string to another. However, for Arabic sentences we have used Arabic WordNet (AWN), which provides a version of the standard Princeton English WordNet (EWN) as discussed later in more detail in Section 5.3.2.

The following examples[7] show how these algorithms measure the minimum cost of operations (Insert, Delete, Exchange, and Swap) to convert one string into another. When the baseline is used in (5.24), We assume that for the baseline INS = 2, DEL = 2, and XCH = 3, whereas when using the WordNet similarity measure we normalise the similarity measure to give a cost for XCH that lies between 0 and 3. Thus, the cost of exchanging يناقشون/'ynAq$wn' ,and 'ybHvwn'/يبحثون is 3.0. This leads to an overall score of similarity $(1 - (3/24)) = 0.87$, while we obtained the cost of exchanging them of 1.5 when using DTW with the similarity measure[8] as seen in Figure 5.14, and this leads to an overall score of similarity $(1 - (1.5/(1.5 * 24))) = (1 - 0.041) = 0.96$

Example 5.24.

S_1: و خبراء دوليون يناقشون مستقبل الاعلام في المملكة

BW: Almmlkp fy AlAElAm mstqbl ynAq$wn dwlywn xbrAG w

S_2: و خبراء دوليون يبحثون مستقبل الاعلام في المملكة

BW: Almmlkp fy AlAElAm mstqbl ybHvwn dwlywn xbrAG w

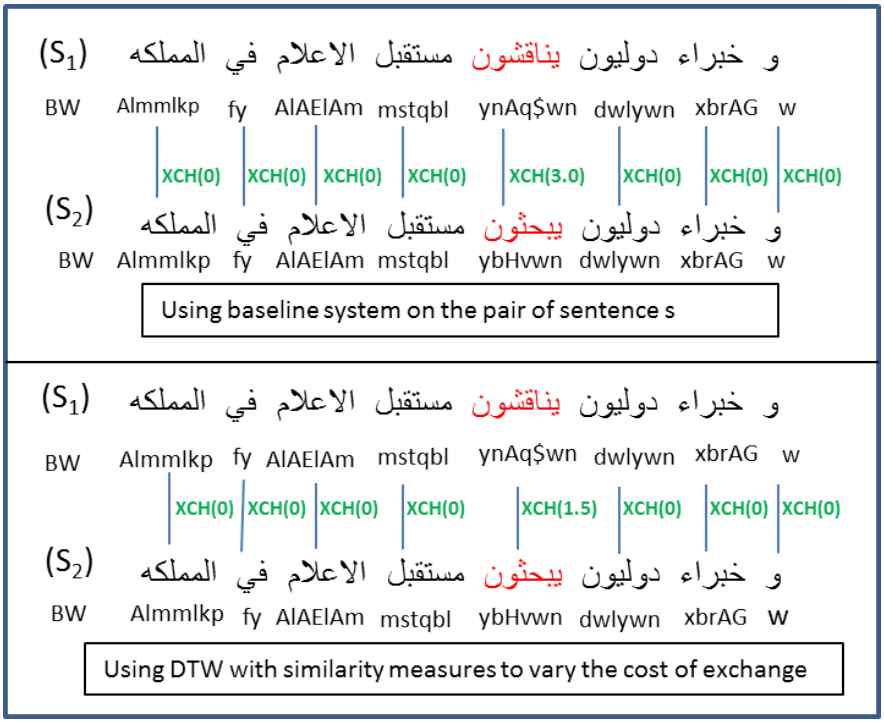


Figure 5.14: Using the baseline system and DTW with similarity measures.

[7] Arabic sentence pairs are taken from our corpus.

[8] Wu and Palmer (wup)

In contrast, the sentence pair of (5.25) contains the same words but in a different order. So, when using the DTW algorithm with similarity measures, the cost of inserting and deleting 'جوجل'/'jwjl' is 4 to convert one string to another, the similarity score is $(1 - (4/27) = 0.85)$. However, using XDTW with similarity measures is more flexible, which is dealing with the adjacent words by adding a new operation 'Swap' to the string edit distance. Assuming that is the baseline INS=2, DEL=2, XCH=3 and SWAP=0.5, whereas when using the WordNet similarity measure we normalise the similarity measure to give a cost for XCH that lies between 0 and 3. Thus, the cost of swapping 'جوجل'/'jwjl' and 'تمكنت'/'tmknt' is 0.5, leading to an overall $(1 - (0.5/27)) = 0.98$, which means using Swap gave us a better result of similarity score than using INS and DEL as shown in Figure 5.15. We noticed that changing the word order has less effect on the meaning in Arabic, where it generally makes no difference at all. However, in English as we have mentioned before in sentence pair 5.19, changing the word order makes a difference where it changes the statement into a question. In addition, we observed that the two words ابتكار/ AbtkAr and اختراع/ AxtrAE have the same meaning. So, the cost of exchanging is 0 and so the similarity score between them is 0 as well. In contrast, the Example 5.26 we noticed that the cost of exchange the two words ابتكار/ AbtkAr and اكتشاف/ Akt$Af is different, so it will be affected on the result of the similarity score.

Example 5.25.

S_1: و تمكنت جوجل من ابتكار عدسة ذكية لتصحيح النظر
BW: AlnZr ltSHyH *kyp Edsp AbtkAr mn jwjl tmknt w

S_2: و جوجل تمكنت من اختراع عدسة ذكية لتصحيح النظر
BW: AlnZr ltSHyH *kyp Edsp AxtrAE mn tmknt jwjl w

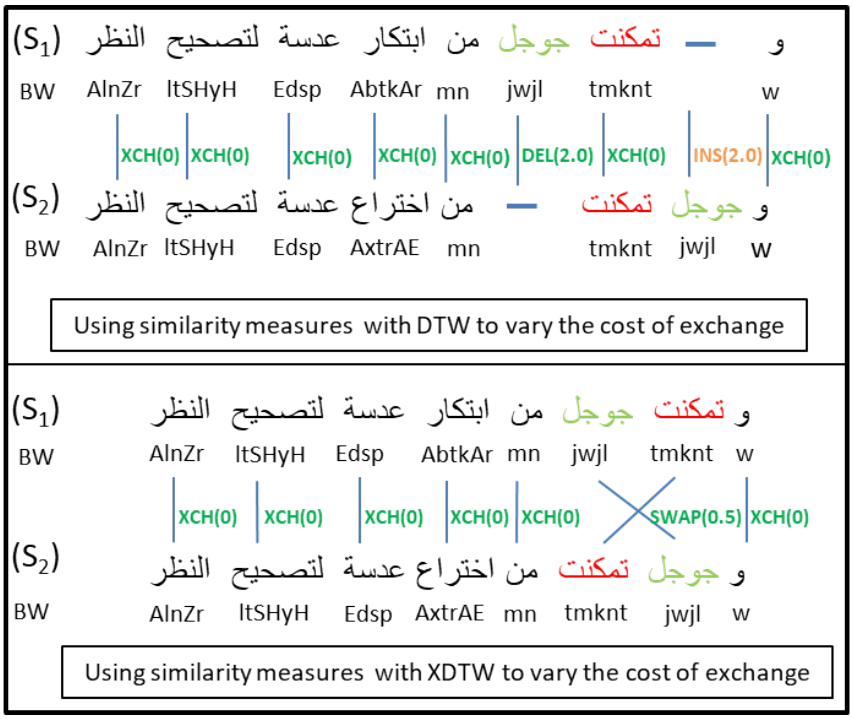


Figure 5.15: Using baseline, DTW, and XDTW to (5.25).

Example 5.26.

S_1: و تمكنت جوجل من ابتكار عدسة ذكية لتصحيح النظر

BW: AlnZr ltSHyH *kyp Edsp AbtkAr mn jwjl tmknt w

S_2: و جوجل تمكنت من اكتشاف عدسة ذكية لتصحيح النظر

BW: AlnZr ltSHyH *kyp Edsp Akt$Af mn tmknt jwjl w

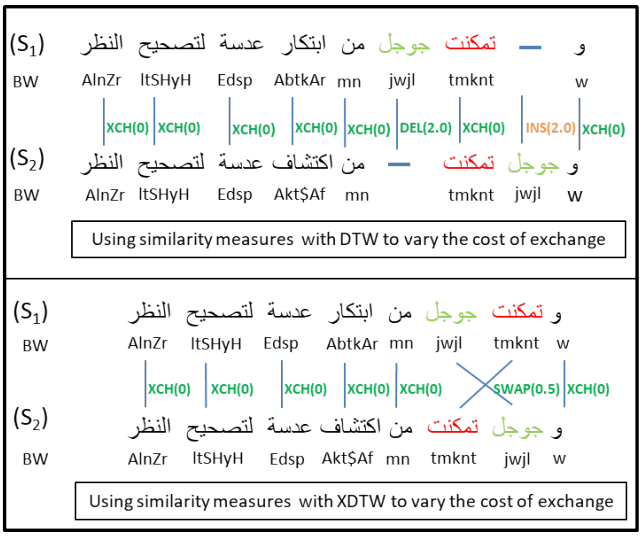


Figure 5.16: Using baseline, DTW, and XDTW to (5.26).

As we have mentioned before in Chapter 4, adding an adverb or an adjective has less effect on the meaning of a sentence than adding a noun or verb. As shown in the sentence pairs 5.27, when an adjective has been inserted the cost was 1.25. Whereas, when a verb has been inserted then the cost would be 2 as shown in (5.28). We notice in (5.29) that is changing one word «تركض» by «تعدو» make a difference in the cost of XCH when using DTW and XDTW with similarity measures, also wit adding an adjective to a sentence make the cost of transfer one sentence to another is less than the cost of inserting verb and high than adding an adjective. See Figure 5.17.

Example 5.27.

S_1: ذهبت الفتاه للمدرسه

BW: llmdrsh AlftAh *hbt

S_2 : ذهبت الفتاه المجتهده للمدرسه

BW: llmdrsh Almjthdh AlftAh *hbt

Example 5.28.

S_1: ذهبت الفتاه للمدرسه

BW: llmdrsh AlftAh *hbt

(S_2) ذهبت الفتاه للمدرسه تركض

BW: trkD llmdrsh AlftAh *hbt

Example 5.29.

S_1: ذهبت الفتاه للمدرسه تركض

BW: llmdrsh AlftAh *hbt

S_2 : ذهبت الفتاه المجتهده للمدرسه تعدو

BW: llmdrsh Almjthdh AlftAh *hbt

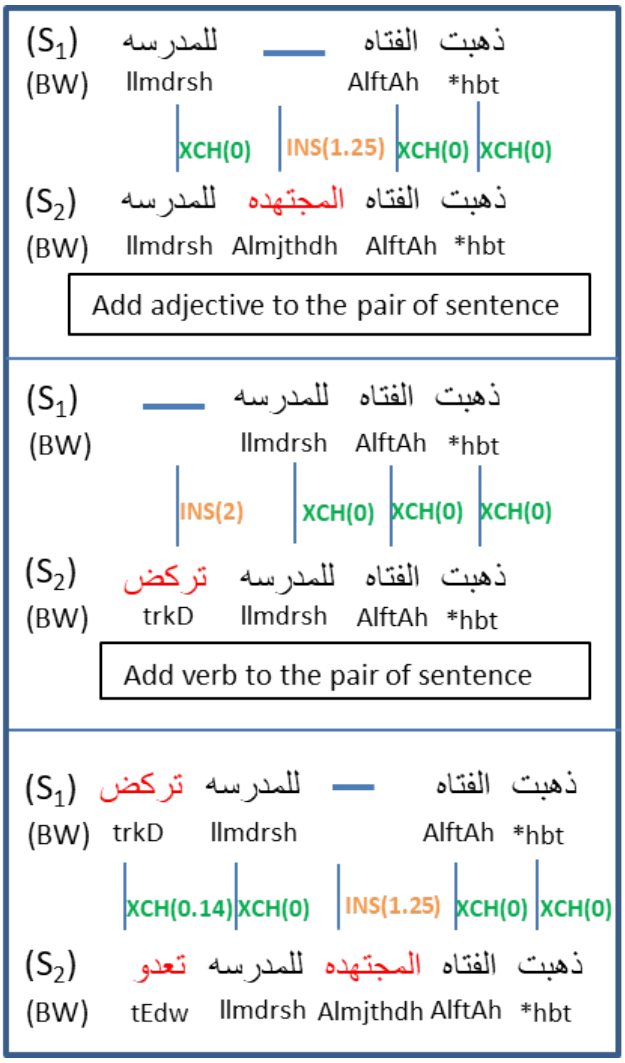


Figure 5.17: Adding words with different POS tags to a sentence results in different changes to the score.

Example	DTW						XDTW					
	wup	lch	path	jcn	res	lin	wup	lch	path	jcn	res	lin
(5.24)	1.5	2.0	1.6	2.3	3.0	3.0	1.5	2.0	1.6	2.3	3.0	3.0
(5.25)	4.0	4.0	4.0	4.0	4.0	4.0	0.5	0.5	0.5	0.5	0.5	0.5
(5.26)	5.1	5.6	5.3	5.9	6.3	6.3	1.6	2.1	1.8	3.0	2.3	2.3
(5.27)	1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25	1.25
(5.28)	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0
(5.29)	1.39	1.5	1.42	1.85	3.25	3.25	1.39	1.5	1.42	1.42	1.85	3.25

Table 5.9: Scores for the six WordNet similarity measures for Arabic.

Table 5.9 shows the results of using DTW and XDTW algorithms with a range of WordNet similarity measures to compute the similarity score. Again,

these scores are the raw output of similarity measures: we report on more detailed experiments to determine the effectiveness of the various measures in Chapter 6.

As with the English examples, this table is divided into three groups. In the first group (example 5.24), the only change is that one word, يناقشون/'ynAq$wn' , has been replaced by another, 'ybHvwn'/يبحثون. Consequently the costs obtained used different similarity measures are different, but using XDTW rather than DTE has no effect. In the second group the word order has been changed, so that XDTW produces a lower score than simple DTW; in addition, in 5.26 one word has been changed for another, so that in addition to the difference between the scores from DTW and XDTW the different similarity measures also lead to different scores. Finally in the third group (5.27, 5.28, 5.29) a word has been added or deleted, with 5.29 also including an exchange of two words. We will investigate the relative effectiveness of these measures in Chapter 6.

We used Arabic WordNet (AWN) as a lexical resource in Arabic (see Section 5.3.2). However, there are two problems with the Arabic version of WordNet (AWN): (i) it is comparatively sparse in comparison to the standard Princeton English WordNet (EWN); and (ii) the obvious route into it is via word roots, which are not easily obtainable from written forms. We investigate ways in which the Buckwalter morphological analyser can be used to overcome these problems as shown in the next sections.

Using AWN with Inflected Arabic

Using AWN raises the problem that Arabic words can take many forms with different structures. As we mentioned in Chapter 4 there are three phenomena of word form:

Inflectional morphology: take a single word and produce different version from it, which means changing the grammatical function of a word, but not the core meaning such as (study/studies/studying/studied, cat/cats, happy/happier/happiness, and so on). For example

Example 5.30.

Arabic:	هو دَرَسَ في الجامعة
BW:	AljAmEp fy darasa hw
Translation:	He studied at the university

Example 5.31.

Arabic:	هم دَرَسُوا في الجامعة
BW:	AljAmEp fy darasuwA hm
Translation:	They studied at the university

Derivational morphology: word with a new meaning and typically a new category is formed, which means changing the meaning of the base such as (construct/construction, write/writer, and so on). For example

Example 5.32.

Arabic:	هو كَتَبَ الرسالة
BW:	AlrsAlp kataba hw
Translation:	He write the letter

Example 5.33.

Arabic:	هو كَاتِبْ الرسالة
BW:	AlrsAlp kaAtib hw
Translation:	He (is) the writer (of) the letter

Cliticisation: is an operation that does not create new words, but combines two morphemes together in one word, which is written with no space between them e.g., وكتبهم (wkutubahm) / and their books.

Example 5.34.

Arabic:	كُتُبْ في المكتبة
BW:	Almktbp fy kutub
Translation:	Books (are) in the library

Example 5.35.

Arabic:	وَكُتُبُهُمْ في المكتبة
BW:	Almktbp fy wakutubhum
Translation:	And their books (are) in the library

To deal with these phenomena, we have to know the structure of Arabic words as described in Section . In Section 5.3.2we will discuss AWN. AWN is a lexical database resource for Arabic, which is indexed by roots; we therefore have to find a route from surface forms into AWN. To do this we tried to integrate the Arabic morphological analyser Pyaramorph with AWN to tackle the inflected Arabic words, as illustrated in Sections 5.3.2, and5.3.2.

Arabic WordNet (AWN)

Arabic WordNet (AWN) is a freely available lexical resource for Arabic designed by Black et al. (2006); it provides a version of the standard Princeton English WordNet (EWN), by linking Arabic roots to EWN synsets and then relying on EWN to provide hypernym relations and other links. The words are grouped together to build the synonyms, called 'synsets', each synset therefore represents a single sense or concept. The synsets of AWN correspond to English EWN. Synonym is the main WordNet relation between the words, whereas hypernym/ hyponymy are the most frequent relations between the synsets.

Generally, AWN provides a number of tables: form, word, and link tables in which each table contains details to represent words in AWN according to a specific structure based on each table. The tables are structured in spread sheet format and explained below by testing the root درس (`drs`):

The Forms Table in AWN The forms table, "form.csv", contains links between "roots" and "names of words".

authorshipid	type	value	wordid
162594	root	درس	darasa_1
162596	root	درس	darasa_3
162597	root	درس	darasa_4
162598	root	درس	darasa_5
162599	root	درس	darasa_6
162600	root	درس	darasa_7
162601	root	درس	darasa_8

Table 5.10: Form table in AWN.

The name of a word denotes a cluster of 'name of word' and 'root' which share the same pattern, root, meaning, and POS tag.

The Words Table in AWN The word table "word.csv" contains links between "names of words" and "names of synsets". Some are names of English words, connected to names of synsets in EWN, and some are names of Arabic words with Arabic names for synsets.

authorshipid	synsetid	value	wordid
162594	darasa_v1AR	دَرَسَ	darasa_1
162595	darasa_v2AR	دَرَسَ	darasa_2
162596	taEal~ama_v2AR	دَرَسَ	darasa_3
162597	darasa_v3AR	دَرَسَ	darasa_4
162598	darasa_v4AR	دَرَسَ	darasa_5
162599	Hal~ala_v1AR	دَرَسَ	darasa_6
162600	naZara__v1AR	دَرَسَ	darasa_7
162601	taEal~ama_v1AR	دَرَسَ	darasa_8

Table 5.11: Word table in AWN.

The Links table in AWN The link table, "link.csv", contains synsets through relations such as 'equivalence', 'hyponym', 'similar', etc. It connects sense items to other sense items, e.g., AWN synset connects to a EWN synset.

Authorshipid	link1	link2	type
145823	darasa_v1AR	study_v2EN	equivalent
145824	darasa_v2AR	nalyse_v2EN	equivalent
145825	darasa_v3AR	study_v5EN	equivalent
145826	darasa_v4AR	study_v3EN	equivalent
145827	daroda$a_v1AR	chew_the_fat_v1EN	equivalent

Table 5.12: Link table in AWN.

As shown in Table 5.11, the initial form of words in AWN are their original roots. AWN takes the root as the canonical form of a word. This poses two problems.

- numerous different words are likely to have the same canonical form.
- a single word can have numerous inflected forms, each of which can have a variety of proclitic and enclitic items attached to them. A regular verb

such as 'درس'/'drs' can have more than 30 distinct undiacriticised forms, corresponding to nearly 70 distinct diacriticised forms (various combinations of number, gender, tense, voice and mood), and each of these could be preceded by a clitic conjunction and followed by a clitic pronoun.

We will address the second of these issues first: the solution to this will lead to a partial solution of the first.

Using Pyaramorph with AWN Names

The problems indicated above are well-known, and several tools for carrying out morphological analysis of Arabic have been developed (Beesley 1996; Beesley and Karttunen 2003; Habash et al. 2009; Kiraz 2000; Ramsay and Mansour 2011). The Standard Arabic Morphological Analyser (SAMA), which is an update of the Buckwalter Morphological Analyser (BAMA) (Buckwalter 2004), is very widely used, either by itself or as a component of some other tool such as MADA or AMIRA. SAMA has a number of advantages:

- It is reasonably fast. The underlying algorithm simply splits words into chunks, looks these up in a set of tables, and verifes that the various elements that are found are indeed compatible, rather than applying a set of rules governing spelling changes at morpheme boundaries, as proposed by Beesley (1996), Barzilay and Lee (2003), and Ramsay and Mansour (2011).
- It has a large lexicon: roughly 40K words
- Lee has provided a free Python version of Buckwalter′s algorithm along with the standard BAMA tables[9], which makes it easy to combine with the rest of the tools used in this project. This implementation also works with the updated SAMA tables, for which we have a license. We will refer below to the combination of Lee′s implementation of the underlying algorithm and the SAMA tables as PYA.

The output of PYA is not, unfortunately, directly usable as a route into AWN. Figure 5.18 shows the output of PYA for يدرسونه (`ydrswnh`). AWN contains the root درس (`drs`) (actually it contains 38 instances of this root, since

[9] //bitbucket.org/alexlee/pyaramorph

there are numerous words with this root: see below). The first solution returned by PYA contains three items that each contain this root, namely the fully diacriticised version of the input string (`yadorusuwnahu`), the 'name' of the word (`daras-u_1`), and the segmented tagged set of morphemes that make it up (`ya/IV3MP+dorus/VERB_IMPERFECT+uwna/IVSUFF_SUBJ:MP`). The first of these is clearly no use as a way of accessing the root درس (`drs`) in AWN. The letters د (d), ر (r), س (s) are all present, but so are lots of other letters, and picking the relevant ones out of this sequence is just as difficult as picking them out of the original string. The last one does at least isolate the root: we have adapted PYA so that the root is included inside curly braces { }, so it is easy to see which element of the set of morphemes that make up the input string is the root.

```
>>> print analyser.lookup("ydrswnh")
    solution: (يَدْرُسُونَهُyadorusuwnahu) [daras-u_1]
     pos: ya/IV3MP+{dorus/VERB_IMPERFECT}+uwna/IVSUFF_SUBJ:MP_MOOD:I+hu/IVSUFF_DO:3MS
    gloss: they (people) + study;learn + it/him

    solution: (يُدَرِّسُونَهُyudar~isuwnahu) [dar~as_1]
     pos: yu/IV3MP+{dar~is/VERB_IMPERFECT}+uwna/IVSUFF_SUBJ:MP_MOOD:I+hu/IVSUFF_DO:3MS
     gloss: they (people) + teach;instruct + it/him
```

Figure 5.18: Output of PYA for "ydrswnh".

Butjustbusing the root found by PYA throws away information. Consider the word والدارسات (`wAldA- risAt`). The sequence of morphemes that PYA returns for this is `wa/CONJ + Al/DET + dAris/NOUN + At/NSUFF_FEM_PL`. If we throw away all the vowels from the stem, we get a root that can be found in AWN, but we will have lost the fact that this word has the long (A) as its first vowel, so we will get all the entries for words that contain this sequence of consonants.

There are two ways we can proceed at this point. (i) The table of roots in AWN links roots to names of words: درس (`drs`), for instance, is linked to the names `daAris_1`, `daAris_2`, `darasa_1`, `darasa_3`, `darasa_4`, `darasa_5`, `darasa_6`, `darasa_7`, `darasa_8`, `daros_1`, `dar asa_1`, `dar asa_2`, `diraAsap_1`, `drAsp_1`, `madorasap_1`, `madorasap_2`, `madorasap_3`, `madorasap_4`, `tadaArasa_1`, `tadoriys_1`, `madorasap_1`, `madorasap_2`, `madorasap_3`, `madorasap_4`, `tadaArasa_1`, `tadoriys_1`. Several of these contain other consonants, and these should be excluded. Among the remainder there are some that have the long A as the first vowel, so it would be sensible to choose these ones.

To do this, we have to overcome the fact that there are some differences between the way that PYA and AWN treat diacritics in word names: in particular, when the semi-consonants w, y and A are being used as vowels then AWN names for words include the associated short vowel as well, to emphasise that the semi-consonant is indeed being used as a vowel in this context and is not part of the root. The required changes are fiddly, but it is not technically difficult to patch them, and in this way we can get straight from the roots that PYA returns to AWN names of words.

This will not always work. The root that we can obtain from the PYA analysis is the undiacriticised version of the stem of the written form. Consider الدرس (Aldrs) in Figure 5.19. PYA returns `Al/DET+daros/NOUN` for this, suggesting that the stem is `daros`. AWN, however, does not have a word whose name looks like this: درس (`drs`) is the broken plural of a word whose singular is `daros`, and the AWN name for this is `daros_1`. Thus while there are many cases where the AWN name looks very like the PYA stem, this will not always work. In particular, it will not work for broken nouns or for weak verbs.

This suggests (ii) using the PYA name for the word. PYA names do look very like AWN names: the PYA name for the word realised as الكتب (`Alktb`) is `kitAb_1`, and one of the AWN names for words whose root is كتب (`ktb`) is also `kitAb_1`. If all PYA names were identical to AWN names, then this would be an obvious way to proceed. Unfortunately, while PYA names often look quite like AWN names, they are not always identical. The PYA name for the first analysis in Figure 5.19 is `daras-u_1`, while the AWN name for the same word is `darasa_1`. In particular, PYA names often include extra morphological information (the -u here, for instance, indicates that the diacriticised form of the active present tense for this verb has u as its second vowel), and AWN names often have a final vowel, since they are more like the traditional standard forms discussed above, along with the presence of short vowels to mark cases where a semi-consonant is being used as a vowel.

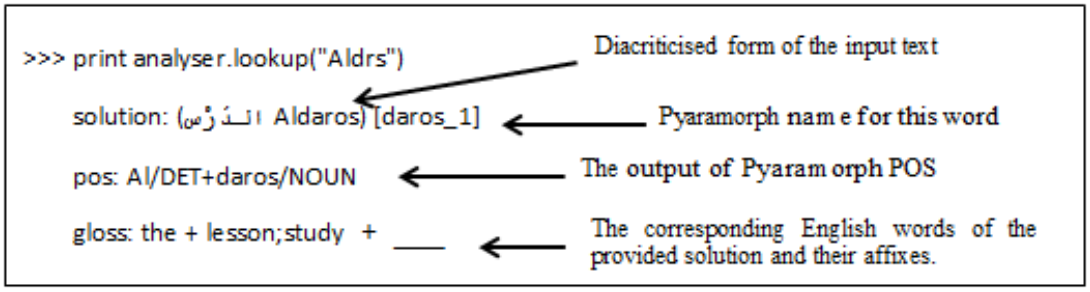


Figure 5.19: Output of PYA for "Aldrs".

Nonetheless, a modest amount of preprocessing of names makes it possible to construct fairly reliable links between PYA names and AWN names, and hence to use the output of PYA as input to AWN.

Using PYA Glosses to extend AWN

There is, however, a further issue to consider. As noted above, AWN has around 2.7K distinct roots, which map to about 16.8K named words. PYA has a lexicon of 40K words. There are thus nearly 25K words that PYA can analyse, but which are not represented in AWN. Some of these are fairly standard words: PYA will analyse the noun الدارستان (`AldArstAn`) as a word whose name is `dAris_1`, with the gloss `student;studying;researcher`. AWN has no word with a name like this, so we cannot use AWN to ascertain that الدارستان (`AldArstAn`) has an interpretation meaning student[10].

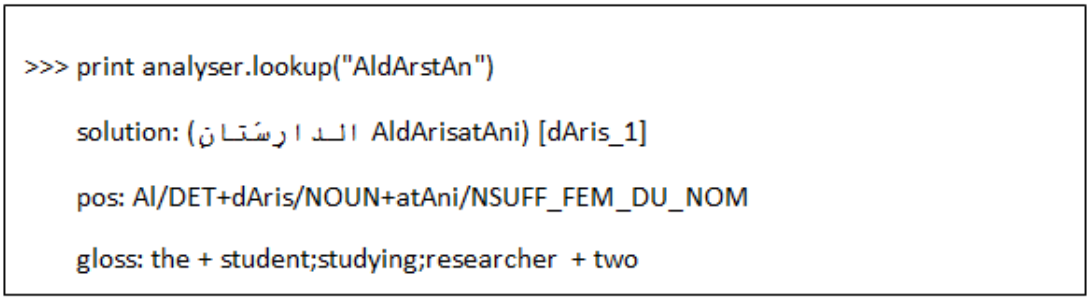


Figure 5.20: Output of PYA for "AldArstAn".

[10]Given that, as described above, it is not always straightforward to find words with a given root in AWN we did a reverse search for words in AWN that link to the English word student. The only such word is called `TaAlib_1`, which is represented by the root طلب (`Tlb`). AWN does not contain an entry for الدارستان (`AldArstAn`)

There is no way into AWN for such words الدارستان (`AldArstAn`). However, AWN is itself largely a way of getting into EWN from Arabic roots. Recall that over half the hypernym links in AWN are actually links between synsets in EWN, with AWN providing a way of getting from Arabic roots to Arabic words, and from Arabic words to Arabic synsets, and then from Arabic synsets to EWN synsets. If we can get directly from PYA output (i.e., using PYA gloss) into EWN then we can do the kinds of reasoning supported by EWN without going through AWN at all. Going directly to EWN may thus, for some applications, be *more* useful than using AWN.

The PYA gloss is a set of English words intended to give some indication of the meaning of the Arabic word. It is a 'brief notation of the meaning of a word' rather than a definition, but the words that it contains should have at least something to do with its meaning. So if any of the words in the gloss have EWN synsets, then it is reasonable to suppose that these can be used as synsets for the Arabic word.

We have adapted PYA so that the part-of-speech tag and the gloss for the stem are marked by curly braces, so that it is easy to identify them. There are, however, some minor complications to be handled before we can use them to find EWN synsets. In particular, they may be disjunctive (as in `student; studying; researcher` as shown in Figure 5.21), and they may contain multiword phrases.

Dealing with disjunction is simple: we simply split the gloss into its elements and treat each of them as a possible interpretation, i.e., as a source of EWN synsets. Multiword phrases are slightly more problematic, e.g., the phrase '*give birth*' is linked to two synsets with the definitions *cause to be born* and *create or produce an idea*. The PYA output for تنجب (`tnjb`) includes an analysis with the gloss *give birth*, which seems likely to correspond to at least the first of the EWN synsets. We therefore use multiword glosses directly as keys into EWN. We will refer to the combination of PYA and English synsets for glosses as PYA+EWN.

PYA+EWN provides us with a route into EWN for Arabic words that are missing from AWN or that have such irregular forms that the PYA output cannot be matched to the AWN name. There are 38.1K distinct glosses in PYA (a number of words have no gloss), of which 16K link to EWN synsets (of these, 1K are multiword expressions). This provides a very substantial increase over the coverage in AWN, at the risk that a PYA gloss may not be an accurate reflection of the meaning of the Arabic word.

How big is this risk? To estimate this, we looked at all the cases where AWN and PYA+EWN both provide an answer and checked whether the AWN answers are included in the PYA+EWN answers. Doing this was not straightforward. As noted before, there are two routes into AWN: either through the root, or through the name of the word. It makes no sense to do the comparison using roots, since AWN roots can correspond to long words, e.g., درس (drs) is, reasonably enough, the root for words called drs_1,…, drs_8. But it is also the root for words called madorasap_1, , madorasap_4, so to see whether PYA+EWN produced the same result for this word we would have to test it with something more like مدرسة (mdrsp). We therefore took the AWN names for words and undiacricitised them and used these as the test set. There are some problems with this, since the undiacritised version of an AWN name is not guaranteed to be a well-formed Arabic word, let alone a well-formed Arabic word with the expected part-of-speech tag, but it did at least enable us to carry out a fair sized evaluation.

The outcome of this experiment is as follows: as noted above, there are 2728 distinct roots in the AWN dictionary, linked to 16998 word names. When we convert word names to candidate words (e.g., converting the name madorasap_4 to the word mdrsp) we end up with 7907 distinct words. Of these, PYA is able to produce analyses for 4273. These are the words that are worth looking at further: there are 7253 EWN senses associated with these words in the AWN. PYA+EWN retrieves 4169 (57.2%) of these. In other words, PYA+EWN recovers most of the hand-coded entries in AWN, so it is reasonable to suppose that if the AWN tables were extended then PYA-AWN would also recover most of these. Thus **PYA+EWN provides an automated way to extend AWN**.

There does not seem to be much of a pattern to the missing examples. Some arise because the two lexicons have different spellings of the same item, e.g., AWN has نقانق (nqAnq) as a word meaning sausage and PYA as مقانق (mqAnq), or have used different transliterations of the semi-consonants. Some arise because PYA has two-word glosses where AWN maps the same word to an EWN synset which can be reached by a single English word, e.g., PYA maps عم (Em) to a word whose gloss is paternal uncle, where AWN maps it to the EWN synsets uncle.n.01 and uncle.n.02. And some are just missing. Nonetheless, the fact that PYA+EWN produces significant overlap with AWN for words where both produce something suggests that the output of PYA+EWN is a useful extension of AWN for words that are not in AWN.

If we use the glosses, the fact that PYA produces multiple analyses is compounded by the fact that each gloss may be disjunctive (so that there are several English words to check) and that each of these may itself lead to several EWN synsets. For الدارس (AldArs), for instance, we get `student.n.01, scholar.n.01, perusal.n.01, research_worker.n.01`. If the gloss was a fair summary of the meaning of the Arabic word, then these are all sensible synsets, but again there are rather a large number of them.

What should we do about this? The first move is to accept the output from AWN itself if there is any. As noted above, PYA+EWN does include the AWN output in the majority of cases, but it does also produce numerous extra EWN synsets. AWN produces 1.69 synsets for each of the words where they both produce something, PYA+EWN produces 17.8 synsets. This arouses partly because PYA itself produces 2.85 synsets analyses per input form, and partly because the gloss for each of these produces multiple synsets. We can thus reduce the number of synsets produced by PYA-EWN by taking the first analysis returned by PYA and using all the synsets that EWN provides for the gloss for that analysis; or by accepting all the analyses provided by PYA but just taking the first EWN synset for each of these; or by just taking the first PYA analysis and the first EWN synset for that analysis. These all cut the number of synsets returned by PYA-EWN,

The second and third strategies (taking the first analysis returned by PYA and then returning all possible synsets for that analysis or taking all analyses returned by PYA but then only taking the first choice from the gloss) cut the number of synsets returned per word without damaging the overall recall too much. Taking all PYA analyses and all elements of the gloss, but only accepting the first EWN synset for each, significantly decreases the recall, to the point where using PYA+EWN does not seem like an effective strategy.

We have seen that there are a number of ways of getting from an Arabic written form into WordNet. Look the written form up in AWN, use PYA to find its root and look that up in AWN, look up its glosses in EWN. These are summarised in Figure 5.21.

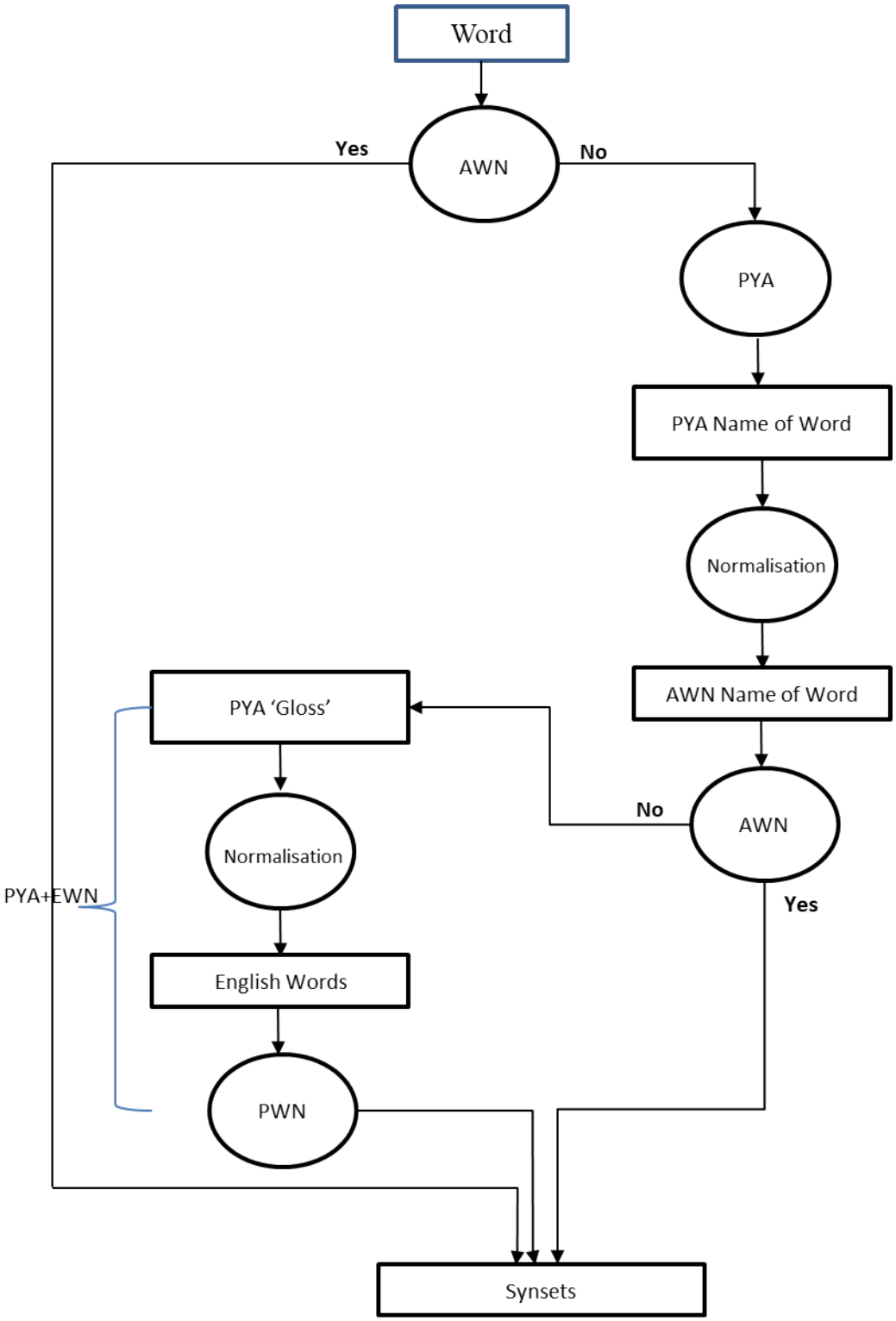


Figure 5.21: The workflow of extracting the synsets from AWN and PYA.

The experiments in Chapter 6 are based on the algorithm outlined in Figure 5.21, using the strategy that involves taking all the PYA analyses and using the first choice from the gloss for each of these and then taking all the English synsets for that gloss. This produces quite large sets of synsets for comparison, but it fits neatly with the view that this strategy carries out a form of mutual disambiguation.

Chapter 6

Experimental Results

6.1 Overview

In Chapter 5, we described how to prepare the dataset e.g. how to collect the data, preprocess and cluster from main articles into sentence pairs to be ready for the next step. Recall from Section 5.1 that we started by collecting a large number of English and Arabic articles, which we reduced to a set of 300 pairs that were roughly balanced between positive and negative examples. The experimental results in this chapter make use of these two sets of 300 pairs. Experimental implementation (similarity judgments) takes the sentence pairs and checks for similarity between them by human and systembased judgment. The results obtained from the humanbased judgments are used to create 'Gold Standards' for both languages. The results obtained from system judgments using the DTW, and its extension XDTW, algorithms with a range of WordNet similarity measures are then used to produce similarity scores. In this chapter, the results of a series of experiments will be evaluated in terms of precision and recall measures by comparing the Gold Standards with similarity scores to see which system behaviour is closest to the human judgment for both English and Arabic (as described in Section 6.2). Afterwards, because the differences in the precision rates between the systems are fairly slight, we will carry out further statistical analysis to see whether these differences are significant or whether they are accidental (as described in Section 6.3).

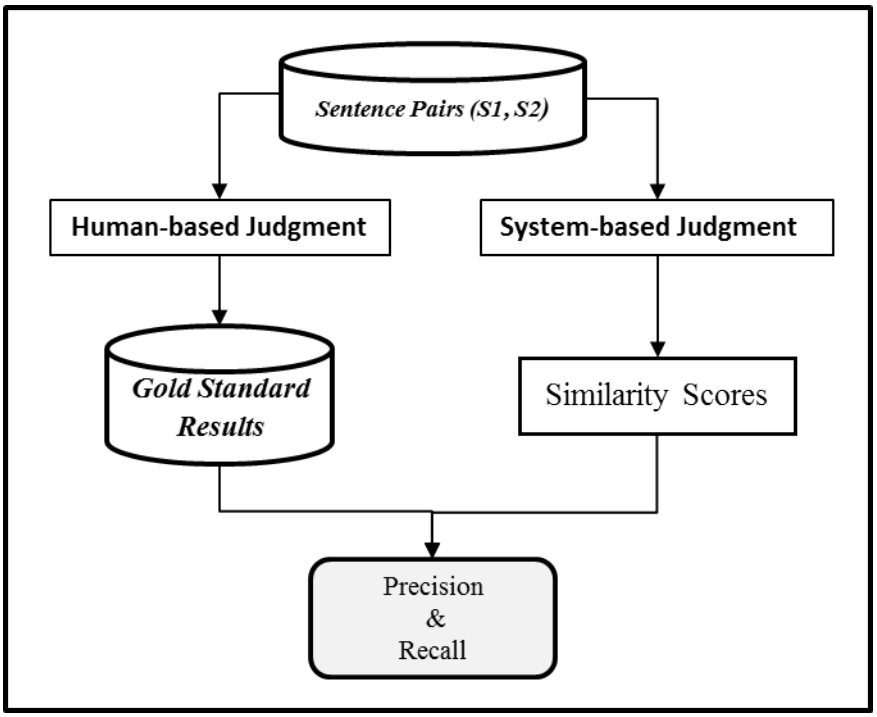


Figure 6.1: Experimental results.

6.2 Precision and Recall measures

In the following experiments we will investigate whether using WordNet similarity measures wup, lch, path, jcn, res, and lin with the alignment methods is better than using the baseline system for both Arabic and English language. The baseline system is the DTW or XDTW algorithm alone without using WordNet similarity measures. The reason for using the baseline is to verify that using WordNet similarity measures does indeed produce improvements over the ordinary baseline system.

The results of these experiments have been evaluated in terms of precision and recall as described in Chapter 4 (see Equations 4.16 and 4.17 in Section 4.6) for each dataset English and Arabic. The precision can be calculated. However, the recall we can only estimate as discussed below.

Precision and recall can be compared for different similarity measures. All similarity measures give us a score between 0 and 1.

We need to find a fair way to compare the results obtained by using different similarity measures. The problem is that the behaviour of the various measures depends on the threshold that we choose when deciding whether a sentence pair

should be deemed to be textually equivalent. There are two ways in which we could proceed:

1. We could try picking a single cosine threshold for all measures to see how they each perform at that threshold. However, they may not give directly comparable scores. To take an extreme case, S_1 might give all its scores between 0 and 0.5, while S_2 might give all its scores between 0.5 and 1. We could not compare S_1 and S_2 by choosing 0.5 as a threshold, since S_1 would say that no pairs were equivalent and S_2 would say that they all are. Hence, to make a fair comparison, we have to pick a different threshold for each measure. Suppose S_1 assigns 0.46 to 10% of cases and S_2 assigns 0.88 to 10% of cases. Then using threshold $T_1 = 0.46$ and $T_2 = 0.88$ would be fair, since in both cases 10% of pairs would be accepted.

2. We would then want to see which of them picked the better 10% of pairs; we could pick a fixed percentage of the entire dataset, and see how well each measure performed for a given percentage. This provides a fair comparison; we might need to choose different thresholds to obtain the same percentage, but once we have done that then we can systematically compare the performance of the different measures. It should be recalled that we selected 10% of the original data for annotation, so picking 10% of the annotated data will give us 1% of the entire dataset, while 20% will give us 2% and so on.

Figure 6.2 shows 9 clusters for Arabic using the XDTW alignment algorithm on the X axis that represents the thresholds. The first cluster contains 30 sentence pairs (1% of the full dataset), the second cluster contains 60 sentence pairs (2% of the full dataset), and so on. The Y axis represents the precision rate: as expected, the precision goes down as we include a larger percentage of the possible pairs. Figures for the other systems (Arabic DTW, English DTW, and English XDTW) are presented in Appendix A.

There are two major observations that can be made on the basis of Figure 6.2. The first is that the baseline consistently produces lower precision than anything else; in other words, exploiting the cost of exchanges in WordNet has a beneficial effect. The second observation is that the differences in the precision rates between the systems are fairly slight. The red and purple columns (wup and path) seem to be the best, but the difference between them and the others

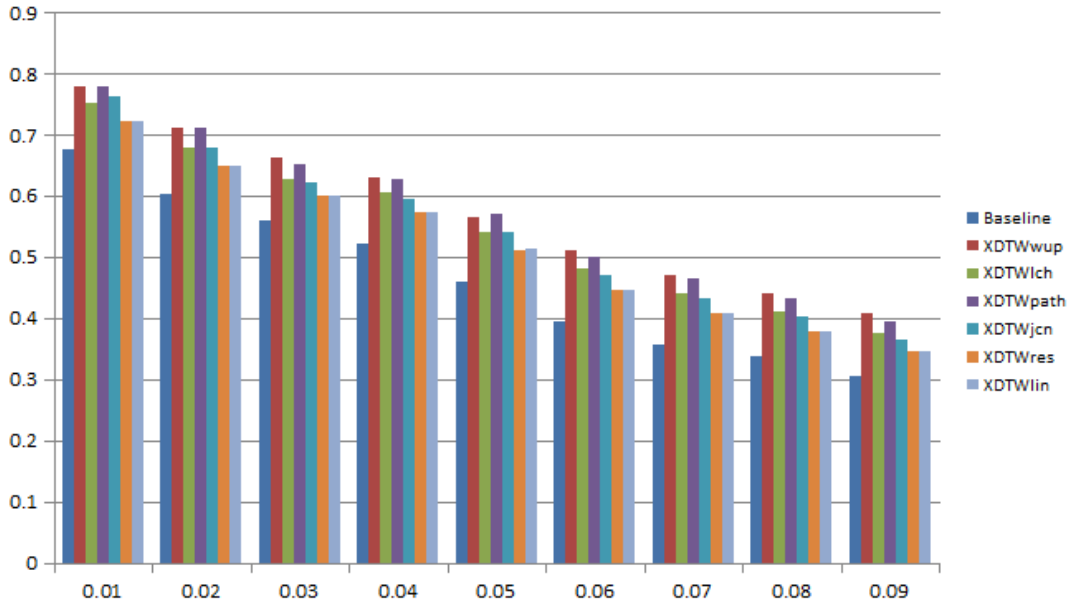


Figure 6.2: The precision of the Arabic XDTW system.

is fairly small and it is hard to be sure that this is a reliable distinction.

The precision has been calculated using the equation 4.21 in Section 4.7. Thus, we can measure the precision on each cluster: the precision for the column (baseline) in the first cluster is 0.67. Given, as we have just seen, that there 30 sentence pairs in the first cluster, we know that there are $0.67 * 30 \approx 20$ positive instances in this cluster.

The precision for the column (wup) in the first cluster at threshold 1 is 0.77. Given, as we have also seen, that there are 30 sentence pairs in this cluster, we know that there are $0.77 * 30 \approx 23$ positive instances in this cluster. In the second cluster at threshold 2 the precision is 0.71, and the number of positive cases is $0.71 * 60 \approx 43$ correct ones, and so on.

We are, also interested in recall values as we change the threshold. However, as noted above, we only gave our annotators 10% of the original dataset, and hence we do not know how many positive examples there are in total. To estimate this, we plot the number of positive instances against the threshold for the cases we do have annotated. If we can fit a curve to this (as seen in Figure 6.3), then we can extrapolate this to estimate the positive examples in the entire dataset.

Figure 6.3 shows how for Arabic XDTWwup the absolute number of correct pairs varies as we increase the percentage of positive pairs that we accept up to 10% of the original set. If we accept the curve that has been fitted to the actual

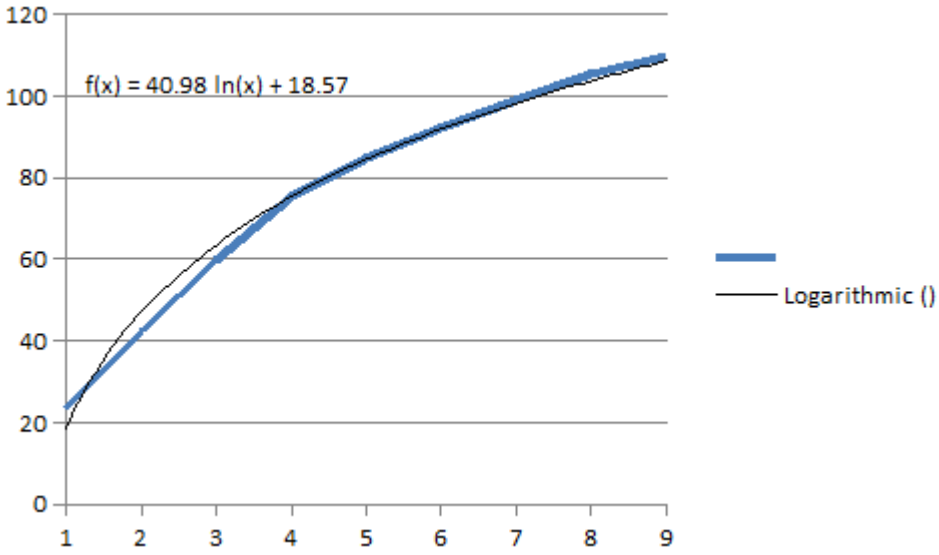


Figure 6.3: The recall of the Arabic XDTWwup.

data, we can estimate that there will be $41.0 \ln(100)+18.5$, i.e. around 207 positive examples in the entire dataset. Assuming that this curve is a reasonable fit for the entire dataset, this means that the recall varies from around $23/207 = 10.5\%$ at the threshold which selects 1% of all instances as candidates to $110/207 = 50.6\%$ at the threshold that selects 10% of all instances as candidates. These values are necessarily estimates, since our annotators were only given 10% of the original data to mark up, but the curve is a reasonable fit for the data which was annotated, so it is plausible that it will give us fairly reliable figures for the overall distribution.

The key result of this section is that the differences between the various measures that used EWN and AWNbased costs for exchange and the baseline are substantial (of the order of 11% percentage points, or around 14% proportionally).

However, the differences in the precision rates between the various measures are fairly slight, therefore it seemed appropriate to carry out a more detailed statistical analysis to see whether these differences are significant or whether they may just be noise.

6.3 Further Statistical Analysis

In the experiments presented in Section 6.2, we used thresholds on the annotated data and on the outputs of the classifiers to compare the overall level of agreement between the human annotators and the classifiers. To probe more deeply, we looked at the individual scores for each sentence pairs.

In order to do this, we considered each sentence pair as a datapoint and looked at the correlation between the average scores assigned by the annotators to that datapoint and the scores assigned by the similarity measures. This enabled us to look in greater detail at the relations between the various similarity measures. The scores that were assigned by the annotators are in a range from 0 to 1: this value comes from dividing the number of Yes's (positive) for each sentence pair marked by the annotators by the total number of annotators. To normalise the value of the similarity scores to be between 0 to 1, we took the score of each sentence pair and divided it by the maximum number of all sentence pairs (see Table 6.1). We would like to know whether the scores assigned by the similarity measures are correlated with the scores that are assigned by the annotators.

Pair No.	Annotators					Average	Scores of similarity measures					
	Ant1	Ant2	Ant3	Ant4	Ant5		wup	lch	path	jcn	res	len
P_1	Yes	Yes	Yes	Yes	Yes	1	0.82	0.79	0.81	0.78	0.76	0.76
P_2	Yes	Yes	Yes	Yes	No	0.8	0.72	0.69	0.81	0.78	0.66	0.66
P_3	Yes	Yes	Yes	Yes	Yes	1	0.67	0.64	0.64	0.61	0.59	0.59
P_4	No	No	No	No	No	0	0.65	0.62	0.63	0.6	0.58	0.58
P_5	Yes	Yes	Yes	Yes	No	0.8	0.47	0.45	0.47	0.44	0.41	0.41

Table 6.1: Some examples of sentence pairs with annotation and the scores of similarity measures.

We aimed in the statistical analysis to answer the experimental questions listed below by analysing the results of similarity scores acquired from the baseline DTW and XDTW systems for both English and Arabic compared to the Gold Standard acquired from the annotators. The experimental questions that guided the experiment design and implementation are:

EQ1. Are the values assigned by the baseline and the various WordNet Similarity measures when applied with the two alignment algorithms correlated with the Gold Standard judgments made by the annotators on the two datasets?

EQ2. Is the correlation for each similarity measures and each alignment algorithms significantly greater than for the corresponding baselines? We can

see from Figure 6.2, for instance, that the score for each of the similarity measures are higher than that for the baseline for XDTW applied to Arabic. What we do not know is whether they are significantly better or whether the differences could just be accidental.

EQ3. Are the differences in the correlation between the different similarity measures significant? While we can see from Figure 6.2 that wup scores better on this data than lch, we cannot tell without further analysis whether this difference is significant or whether it may just be accidental.

EQ4. How do the similarity measures and alignment algorithms compare when applied to the two languages?

The sections which follow will explain the statistical analysis of the score obtained by the various systems. A set of analytical techniques will be used to examine the overall performances as there are significant differences in the utility of the various similarity measures.

6.4 Statistical Analysis Workflow

As shown in Section 6.2 the differences in the precision rates between the systems are fairly slight, which indicates the importance of carrying out a detailed statistical analysis to see whether they were more than simple accidental differences. Based on the experimental questions stated above, we examined and analysed our results from the experiments. The analysis of these results will be used to answer the four experimental questions that we have stated earlier in Section 6.3. The general workflow of this statistical analysis is shown in Figure 6.4. At various points we needed to assess whether different tests produce statistically significant differences.

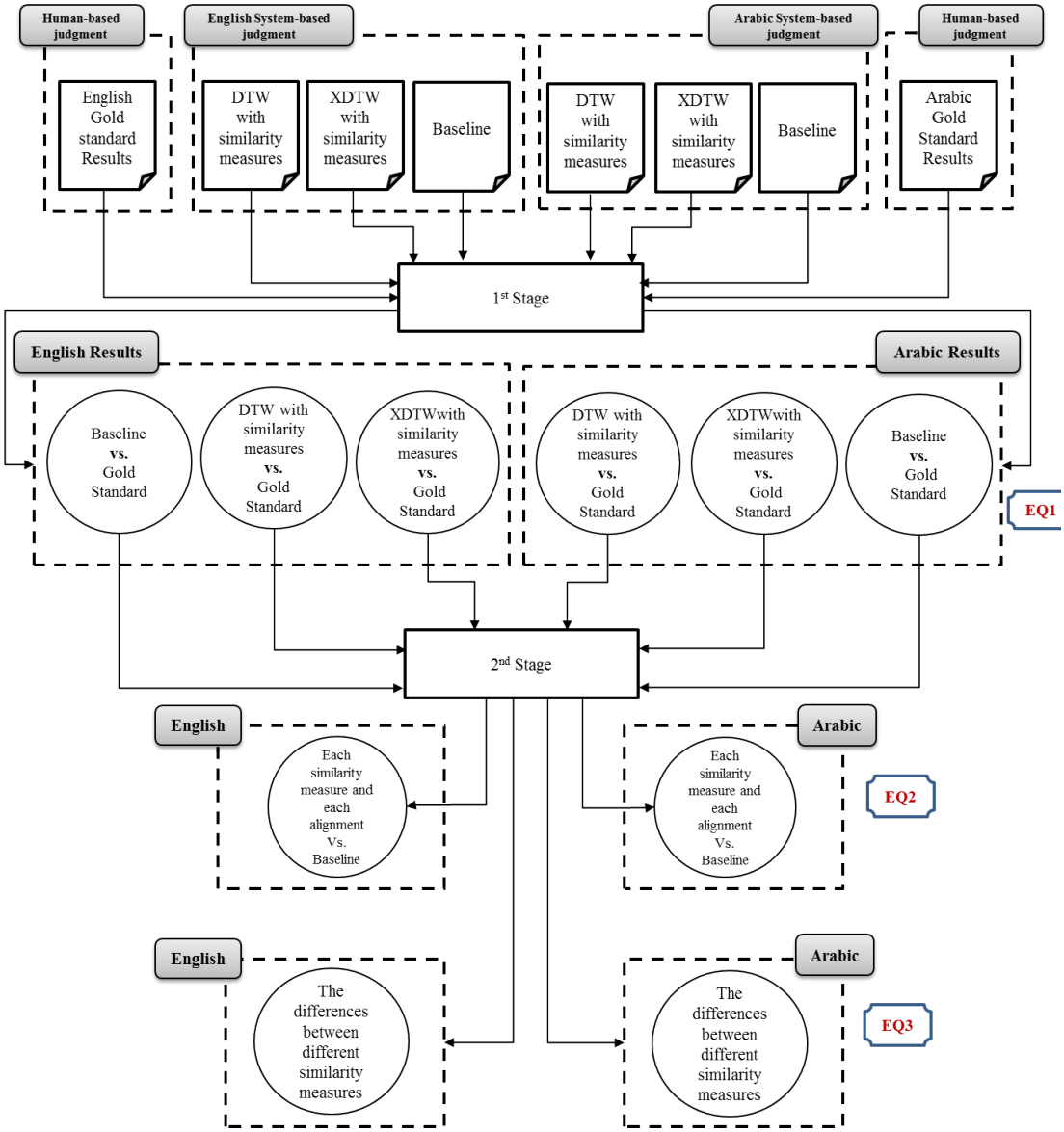


Figure 6.4: Workflow of the Statistical Analysis Process.

6.5 Testing Normality

The standard statistical tests for studying the results of different classifiers assume that the data is normally distributed. To ensure that the tests we used were appropriate for our result sets, we investigated their normality by following the steps stated below:

1. Check normality using a normality test such as the KolmogorovSmirnov test, the ShapiroWilk test and the QQ Plot test, then:

a. If the data is normally distributed, then use Pearson's correlation.

b. If the data is not normally distributed, then use Spearman's correlation, which is a nonparametric measure between two variables.

When testing the dataset for normality, we will be interested in the numerical KolmogorovSmirnov and ShapiroWilk methods, and the graphical QQ Plot method, as shown in Sections 6.5.1 and 6.5.2, respectively.

6.5.1 The KolmogorovSmirnov Test and the ShapiroWilk Test for Normality

The KolmogorovSmirnov test and the ShapiroWilk test are commonly used as tests of normality (Chakravarti et al. 1967). The ShapiroWilk test is more appropriate for small sample sizes.

Systems	KolmogorovSmirnov[1]			ShapiroWilk		
	Statistic	df	Sig.	Statistic	df	Sig.
BaseLine	.135	300	.000	.924	300	.000
DTWwup	.111	300	.000	.939	300	.000
DTWlch	.106	300	.000	.940	300	.000
DTWpath	.119	300	.000	.938	300	.000
DTWjcn	.116	300	.000	.936	300	.000
DTWres	.125	300	.000	.936	300	.000
DTWlin	.119	300	.000	.936	300	.000

Table 6.2: The KolmogorovSmirnov test and the ShapiroWilk test applied to Arabic_DTW

Table 6.2 shows that the data for all systems is not normally distributed. If the Sig. value of the two tests are greater than 0.05, then the data is normal. If it is below 0.05, then the data is not normally distributed. Since the significance values for both test for all measures are less than 0.0005 (i.e. less than 0.001 to 3 significant figures) it is clear that this data is not normally distributed. The other system tables (English_DTW, English_XDTW, and Arabic_XDTW) are presented in Appendix C.

[1]Lilliefors Significance Correction

6.5.2 QQ plotting Test for Normality

Because the significance tests for normally distributed data are considerably simpler and more reliable than those for data which is not so distributed, we decided to confirm the results of the KolmogorovSmirnov test by using QQ plots. If the data are normally distributed, the data points will be close to the diagonal line. If the data points stray from the line in an obvious nonlinear fashion, then the data are not normally distributed. The results of this normality test for Arabic DTW using WordNet similarity measures are presented in Figure 6.5, and the results for the other cases are presented in Appendix B. Because all these tests showed that the data is not normally distributed, a nonparametric approach was applied.

For this study, the tests have shown that the data is not normally distributed. As a result, the nonparametric Spearman's correlation coefficient test was applied. This test was designed by Spearman as a measure of the strength of an association between two variables (Spearman 1987). Spearman's correlation r_s is constrained between -1 and 1.

When r_s is close to ± 1 there is a strong relationship between two variables, whereas if r_s is close to 0, then the relationship between the two variables is weak (see Section 6.6.1). The strength of the correlation is described as follows:

$\pm$ 0.00–0.19 "very weak"

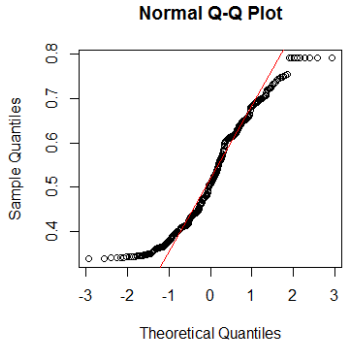
$\pm$ 0.20–0.39 "weak"

$\pm$ 0.40–0.59 "moderate"

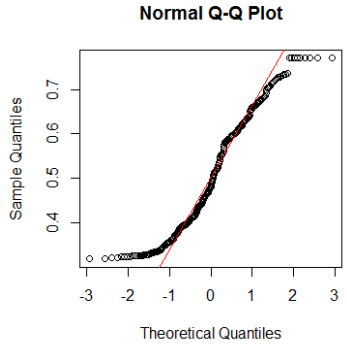
$\pm$ 0.60–0.79 "strong"

$\pm$ 0.80–1.0 "very strong"

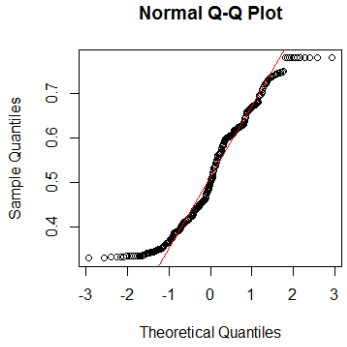
It is standard practice to use levels of significance at 0.05, 0.01 and 0.001, so that the test result for a $p$value is significant if $p < 0.05$, highly significant if $p < 0.01$, and very highly significant if $p < 0.001$.



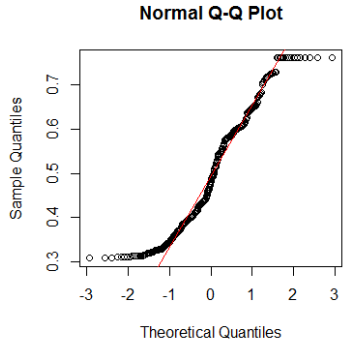
(a) Normality test based on Arabic DTWwup datasets; the figure indicates nonnormality.



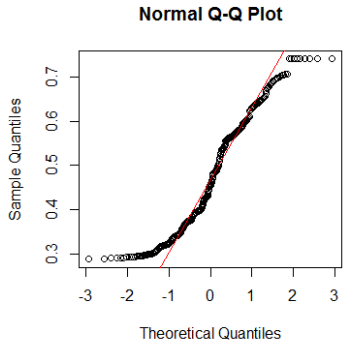
(b) Normality test based on Arabic DTWlch datasets the figure indicates nonnormality.



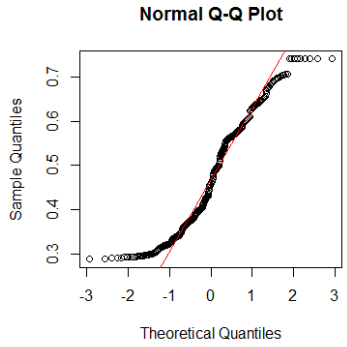
(c) Normality test based on Arabic DTWpath datasets; the figure indicates nonnormality.



(d) Normality test based on Arabic DTWjcn datasets; the figure indicates nonnormality.



(e) Normality test based on Arabic DTWres datasets; the figure indicates nonnormality.



(f) Normality test based on Arabic DTWlin datasets; the figure indicates nonnormality.

Figure 6.5: QQ Plot tests of Arabic DTW for the wup, lch, path, jcn, res and lin similarity measures.

6.6 The Statistical Analysis Correlation Tests

To answer the questions EQ1, EQ2 and EQ3 that we have stated earlier in Section 6.3, we need to specify a set of hypotheses to initiate the statistical process. We have three main hypotheses corresponding to the three questions EQ1, EQ2 and EQ3.

EQ1: Are the values assigned by the baseline and the various WordNet similarity measures when applied with the two alignment algorithms correlated with the Gold Standard judgments made by the annotators on the two datasets?

Hypothesis 1: The values that are assigned by the various WordNet Similarity measures when applied with the two alignment algorithms and the baseline are correlated with the Gold Standard.

Null hypothesis H_0: $r_S^{I,L} = 0$, where I stands for the alignment algorithm (DTW or XDTW), L stands for the language (English or Arabic), and S stands for the WordNet similarity measures (wup, lch, path, jcn, res or lin), and the baseline. The null hypothesis states that there is no correlation between the Gold Standard (GS) and different WordNet similarity measures jcn, lich, line, path, res and wup.

Alternative Hypothesis H_1: $r_S^{I,L} <> 0$, the alternative hypothesis states that there is a correlation between the Gold Standard and different WordNet similarity measures.

The results in Table 6.3 were obtained by running Spearman′s rankorder correlation to determine the relationship between the different similarity measures for the alignment algorithm DTW and the GS in English. r_S was strong and positive in wup, lch, path, and jcn. Whereas r_S was moderate and positive in res, lin, and baseline. This correlation $p$value was statistically highly significant in wup and path, whereas only significant in (lch).

The results in Table 6.4 were obtained in the same wasy as those in Table 6.3 but looking at the relationship between the different similarity measures for XDTW and the GS rather than DTW. r_S was strong and positive in wup, lch, path, and jcn. Whereas r_S was moderate and positive in res, lin, and baseline. This correlation $p$value was statistically highly significant in wup and path, whereas significant in lch and jcn.

$r_S^{I,L}$	Correlation value	$p$value
$r_{\text{wup}}^{\text{DTW,En}}$(wup, GS)	0.680	0.01
$r_{\text{lch}}^{\text{DTW,En}}$(lch, GS)	0.630	0.05
$r_{\text{path}}^{\text{DTW,En}}$(path, GS)	0.676	0.01
$r_{\text{jcn}}^{\text{DTW,En}}$(jcn, GS)	0.608	-
$r_{\text{res}}^{\text{DTW,En}}$(res, GS)	0.573	-
$r_{\text{lin}}^{\text{DTW,En}}$(lin, GS)	0.578	-
$r_{\text{baseline}}^{\text{DTW,En}}$(baseline, GS)	0.539	-

Table 6.3: The correlation between DTW with similarity measures and the GS for English.

$r_S^{I,L}$	Correlation value	$p$value
$r_{\text{wup}}^{\text{XDTW,En}}$(wup, GS)	0.691	0.01
$r_{\text{lch}}^{\text{XDTW,En}}$(lch, GS)	0.644	0.05
$r_{\text{path}}^{\text{XDTW,En}}$(path, GS)	0.686	0.01
$r_{\text{jcn}}^{\text{XDTW,En}}$(jcn, GS)	0.618	0.05
$r_{\text{res}}^{\text{XDTW,En}}$(res, GS)	0.588	-
$r_{\text{lin}}^{\text{XDTW,En}}$(lin, GS)	0.588	-
$r_{\text{baseline}}^{\text{XDTW,En}}$(baseline, GS)	0.539	-

Table 6.4: The correlation between XDTW with similarity measures and the GS for English.

$r_S^{I,L}$	Correlation value	$p$value
$r_{\text{wup}}^{\text{DTW,Ar}}$(wup, GS)	0.535	0.05
$r_{\text{lch}}^{\text{DTW,Ar}}$(lch, GS)	0.505	-
$r_{\text{path}}^{\text{DTW,Ar}}$(path, GS)	0.529	0.05
$r_{\text{jcn}}^{\text{DTW,Ar}}$(jcn, GS)	0.467	-
$r_{\text{res}}^{\text{DTW,Ar}}$(res, GS)	0.402	-
$r_{\text{lin}}^{\text{DTW,Ar}}$(lin, GS)	0.402	-
$r_{\text{baseline}}^{\text{DTW,Ar}}$(baseline, GS)	0.391	-

Table 6.5: The correlation between DTW with similarity measures and the GS for Arabic.

As above, the results in Table 6.5 were obtained by running Spearman's rankorder correlation to determine the relationship between the different similarity measures in the alignment algorithm DTW and the GS, but this time in Arabic. r_S was moderate and positive in wup, lch, path, jcn, res, and lin, r_S was weak and positive in baseline. This correlation $p$value was statistically significant in wup and path.

Again, the results in Table 6.6 were obtained by running Spearman′s rankorder correlation for XDTW, rather than DTW, and the GS in Arabic. r_S was moderate and positive in wup, lch, path, jcn, res, and lin. r_S weak and positive in baseline. This correlation $p$value was statistically highly significant in wup and path, and significant in lch. All the tables 6.3, 3.4, 6.5, and 6.4 contain '-', which means that the p-value is not significant.

Notice that the alternative hypothesis H_1 is accepted in some cases as shown in Table 6.4. Figure 6.2 suggested that using similarity measures in the alignment algorithm will produce improvement in the results; these tests confirm for both English and Arabic that this improvement was not accidental. These results led us to do more tests to see if these correlations between the similarity measures with the GS and the baseline with the GS are significant or not, as shown in the

$r_S^{I,L}$	Correlation value	$p$value
$r_{\text{wup}}^{\text{XDTW,Ar}}$(wup, GS)	0.558	0.01
$r_{\text{lch}}^{\text{XDTW,Ar}}$(lch, GS)	0.525	0.05
$r_{\text{path}}^{\text{XDTW,Ar}}$(path, GS)	0.552	0.01
$r_{\text{jcn}}^{\text{XDTW,Ar}}$(jcn, GS)	0.499	-
$r_{\text{res}}^{\text{XDTW,Ar}}$(res, GS)	0.430	-
$r_{\text{lin}}^{\text{XDTW,Ar}}$(lin, GS)	0.430	-
$r_{\text{baseline}}^{\text{XDTW,Ar}}$(baseline, GS)	0.391	-

Table 6.6: The correlation between XDTW with similarity measures and the GS for Arabic.

second experiment. The third experiment shows if the differences between these correlations for different similarity measures are significant.

We now move on to question EQ2.

EQ2: Is the correlation for each similarity measure and each alignment algorithm significantly greater than the baseline? We can see from Figure 6.2, for instance, that the score for each of the similarity measures is higher than that for the baseline for the XDTW applied to Arabic. What we do not know is whether they are significantly better or whether the differences could just be accidental.

Hypothesis 2: The correlation between each similarity measure and the Gold Standard is significantly greater than the correlation between the baseline and the Gold Standard.

In this hypothesis to assess the significance of the difference between two correlation coefficients, we used Fisher′s $z$test (Fisher and Yates 1938).

Null hypothesis H_0: $F_S^{I,L} = 0$, where F stands for Fisher's test, I stands for the alignment algorithms DTW and XDTW, L stands for the language (English or Arabic), and S stands for the WordNet Similarity measures wup, lch, path,

jcn, res, and lin. The null hypothesis states that there is no significant difference between each similarity measure with Gold Standard and the baseline with the Gold Standard.

Alternative hypothesis H_1: $F_S^{I,L} <> 0$, the alternative hypothesis states that there is a significant difference between each similarity measure with the Gold Standard and the baseline with Gold Standard.

Fisher's $z$test	$z$value	$p$value
$F_{\text{wup}}^{\text{DTW,En}}(r_{\text{wup}}^{\text{DTW,En}}, r_{\text{baseline}}^{\text{DTW,En}})$	2.76	0.00
$F_{\text{lch}}^{\text{DTW,En}}(r_{\text{lch}}^{\text{DTW,En}}, r_{\text{baseline}}^{\text{DTW,En}})$	1.69	0.09
$F_{\text{path}}^{\text{DTW,En}}(r_{\text{path}}^{\text{DTW,En}}, r_{\text{baseline}}^{\text{DTW,En}})$	2.76	0.00
$F_{\text{jcn}}^{\text{DTW,En}}(r_{\text{jcn}}^{\text{DTW,En}}, r_{\text{baseline}}^{\text{DTW,En}})$	1.26	0.20
$F_{\text{res}}^{\text{DTW,En}}(r_{\text{res}}^{\text{DTW,En}}, r_{\text{baseline}}^{\text{DTW,En}})$	0.60	0.54
$F_{\text{lin}}^{\text{DTW,En}}(r_{\text{lin}}^{\text{DTW,En}}, r_{\text{baseline}}^{\text{DTW,En}})$	0.69	0.49

Table 6.7: Results of Fisher′s $z$test between DTW with similarity measures and baseline for English.

Figure 6.2 suggested that using the similarity measures in the alignment algorithm will produce more accurate results; this test aims to confirm that this improvement is not accidental.

The results of significance testing have shown that the correlation between the different similarity measures in the alignment algorithm and the baseline is significant, as shown in tables 6.7, 6.8, 6.9 and 6.10. This means that the alternative hypothesis has to be accepted. Figure 6.2 suggested that using similarity measures in the alignment algorithm will produce improvement in the results; these tests confirm for both English and Arabic that this improvement is not accidental.

In tables 6.7, 6.8, 6.9 and 6.10, the similarity measures wup and path indeed suggest that they are better than other measures, i.e. that it is worth choosing

Fisher's $z$test	$z$value	$p$value
$F^{\text{XDTW,En}}_{\text{wup}}(r^{\text{XDTW,En}}_{\text{wup}}, r^{\text{XDTW,En}}_{\text{baseline}})$	3.01	0.00
$F^{\text{XDTW,En}}_{\text{lch}}(r^{\text{XDTW,En}}_{\text{lch}}, r^{\text{XDTW,En}}_{\text{baseline}})$	1.98	0.04
$F^{\text{XDTW,En}}_{\text{path}}(r^{\text{XDTW,En}}_{\text{path}}, r^{\text{XDTW,En}}_{\text{baseline}})$	2.09	0.00
$F^{\text{XDTW,En}}_{\text{jcn}}(r^{\text{XDTW,En}}_{\text{jcn}}, r^{\text{XDTW,En}}_{\text{baseline}})$	1.45	0.14
$F^{\text{XDTW,En}}_{\text{res}}(r^{\text{XDTW,En}}_{\text{res}}, r^{\text{XDTW,En}}_{\text{baseline}})$	0.88	0.37
$F^{\text{XDTW,En}}_{\text{lin}}(r^{\text{XDTW,En}}_{\text{lin}}, r^{\text{XDTW,En}}_{\text{baseline}})$	0.88	0.37

Table 6.8: Results of Fisher's $z$test between XDTW with similarity measures and baseline for English.

Fisher's $z$test	$z$value	$p$value
$F^{\text{DTW,Ar}}_{\text{wup}}(r^{\text{DTW,Ar}}_{\text{wup}}, r^{\text{DTW,Ar}}_{\text{baseline}})$	2.24	0.02
$F^{\text{DTW,Ar}}_{\text{lch}}(r^{\text{DTW,Ar}}_{\text{lch}}, r^{\text{DTW,Ar}}_{\text{baseline}})$	1.74	0.08
$F^{\text{DTW,Ar}}_{\text{path}}(r^{\text{DTW,Ar}}_{\text{path}}, r^{\text{DTW,Ar}}_{\text{baseline}})$	2.14	0.03
$F^{\text{DTW,Ar}}_{\text{jcn}}(r^{\text{DTW,Ar}}_{\text{jcn}}, r^{\text{DTW,Ar}}_{\text{baseline}})$	1.14	0.25
$F^{\text{DTW,Ar}}_{\text{res}}(r^{\text{DTW,Ar}}_{\text{res}}, r^{\text{DTW,Ar}}_{\text{baseline}})$	0.16	0.87
$F^{\text{DTW,Ar}}_{\text{lin}}(r^{\text{DTW,Ar}}_{\text{lin}}, r^{\text{DTW,Ar}}_{\text{baseline}})$	0.16	0.87

Table 6.9: Results of Fisher's $z$test between DTW with similarity measures and baseline for Arabic.

one of the better ones to test in the next hypothesis, thus we are interested in determining if they are significantly better or not. This leads us to question EQ3:

EQ3: Are the differences in the correlation between the different similarity measures significant? While we can see from Figure 6.2 that wup scores better

Fisher's $z$test	$z$value	$p$value
$F_{\text{wup}}^{\text{XDTW,Ar}}(r_{\text{wup}}^{\text{XDTW,Ar}}, r_{\text{baseline}}^{\text{XDTW,Ar}})$	2.64	0.00
$F_{\text{lch}}^{\text{XDTW,Ar}}(r_{\text{lch}}^{\text{XDTW,Ar}}, r_{\text{baseline}}^{\text{XDTW,Ar}})$	2.07	0.03
$F_{\text{path}}^{\text{XDTW,Ar}}(r_{\text{path}}^{\text{XDTW,Ar}}, r_{\text{baseline}}^{\text{XDTW,Ar}})$	2.54	0.01
$F_{\text{jcn}}^{\text{XDTW,Ar}}(r_{\text{jcn}}^{\text{XDTW,Ar}}, r_{\text{baseline}}^{\text{XDTW,Ar}})$	1.65	0.09
$F_{\text{res}}^{\text{XDTW,Ar}}(r_{\text{res}}^{\text{XDTW,Ar}}, r_{\text{baseline}}^{\text{XDTW,Ar}})$	0.57	0.56
$F_{\text{lin}}^{\text{XDTW,Ar}}(r_{\text{lin}}^{\text{XDTW,Ar}}, r_{\text{baseline}}^{\text{XDTW,Ar}})$	0.57	0.56

Table 6.10: Results of Fisher's $z$test between XDTW with similarity measures and baseline for Arabic.

on this data than lch, we cannot tell without further analysis whether this difference is significant or whether it may just be accidental.

Hypothesis 3: The differences of the correlation between different similarity measures are significant.

Null hypothesis H_0: $F_S^{I,L}$, where I stands for the alignment algorithms (DTW or XDTW), L stands for the language (English or Arabic), and S stands for WordNet Similarity measures (wup, path, res or lin). The null hypothesis states that there is no significant correlation between different similarity measures.

Alternative Hypothesis H_1: $F_S^{I,L} <> 0$, which means that there is a significant difference between two correlation coefficients.

The results of the significance tests showed that the correlation between the different similarity measures in the alignment algorithm is significant as shown in tables 6.11, 6.12, 6.13 and 6.14. It is statistically significant at $p = 0.03$ with wup similarity measure, and $p = 0.04$ with the path similarity measure for both languages. These results lead us to the next experimental question.

Fisher's $z$test	$z$value	$p$value
$F^{\text{DTW,En}}_{\text{wup, res}}(r^{\text{DTW,En}}_{\text{wup}}, r^{\text{DTW,En}}_{\text{res}})$	2.07	0.03
$F^{\text{DTW,En}}_{\text{wup, lin}}(r^{\text{DTW,En}}_{\text{wup}}, r^{\text{DTW,En}}_{\text{lin}})$	2.76	0.00
$F^{\text{DTW,En}}_{\text{path, res}}(r^{\text{DTW,En}}_{\text{path}}, r^{\text{DTW,En}}_{\text{res}})$	1.98	0.04
$F^{\text{DTW,En}}_{\text{path, lin}}(r^{\text{DTW,En}}_{\text{path}}, r^{\text{DTW,En}}_{\text{lin}})$	2.67	0.00

Table 6.11: Results of Fisher's $z$test between DTW with different similarity measures for English.

Fisher's $z$test	$z$value	$p$value
$F^{\text{XDTW,En}}_{\text{wup, res}}(r^{\text{XDTW,En}}_{\text{wup}}, r^{\text{XDTW,En}}_{\text{res}})$	2.14	0.03
$F^{\text{XDTW,En}}_{\text{wup, lin}}(r^{\text{XDTW,En}}_{\text{wup}}, r^{\text{XDTW,En}}_{\text{lin}})$	2.14	0.03
$F^{\text{XDTW,En}}_{\text{path, res}}(r^{\text{XDTW,En}}_{\text{path}}, r^{\text{XDTW,En}}_{\text{res}})$	2.02	0.04
$F^{\text{XDTW,En}}_{\text{path, lin}}(r^{\text{XDTW,En}}_{\text{path}}, r^{\text{XDTW,En}}_{\text{lin}})$	2.02	0.04

Table 6.12: Results of Fishers $z$test between XDTW with different similarity measures for English.

EQ4: How do the similarity measures and alignment algorithms compare when applied to the two languages?

To answer this question we need to prove that the XDTW is better suited for Arabic than it is for English. In order to do this, for each similarity measure we will calculate the ratio between its DTW and XDTW scores and take the average for each language.

Theit average improvement that XDTW produces over DTW for Arabic is 1.06, the average improvement for English is 1.01. While these figures are suggestive, it is simply not possible to run any significance test to see whether the improvement for Arabic is significantly better than for English as the two datasets,

Fisher's $z$test	$z$value	$p$value
$F^{\text{DTW,Ar}}_{\text{wup, res}}(r^{\text{DTW,Ar}}_{\text{wup}}, r^{\text{DTW,Ar}}_{\text{res}})$	2.08	0.03
$F^{\text{DTW,Ar}}_{\text{wup, lin}}(r^{\text{DTW,Ar}}_{\text{wup}}, r^{\text{DTW,Ar}}_{\text{lin}})$	2.08	0.03
$F^{\text{DTW,Ar}}_{\text{path, res}}(r^{\text{DTW,Ar}}_{\text{path}}, r^{\text{DTW,Ar}}_{\text{res}})$	1.98	0.04
$F^{\text{DTW,Ar}}_{\text{path, lin}}(r^{\text{DTW,Ar}}_{\text{path}}, r^{\text{DTW,Ar}}_{\text{lin}})$	1.98	0.04

Table 6.13: Results of Fishers $z$test between DTW with different similarity measures for Arabic.

Fisher's $z$test	$z$value	$p$value
$F^{\text{XDTW,Ar}}_{\text{wup, res}}(r^{\text{XDTW,Ar}}_{\text{wup}}, r^{\text{XDTW,Ar}}_{\text{res}})$	2.07	0.03
$F^{\text{XDTW,Ar}}_{\text{wup, lin}}(r^{\text{XDTW,Ar}}_{\text{wup}}, r^{\text{XDTW,Ar}}_{\text{lin}})$	2.07	0.03
$F^{\text{XDTW,Ar}}_{\text{path, res}}(r^{\text{XDTW,Ar}}_{\text{path}}, r^{\text{XDTW,Ar}}_{\text{res}})$	1.97	0.04
$F^{\text{XDTW,Ar}}_{\text{path, lin}}(r^{\text{XDTW,Ar}}_{\text{path}}, r^{\text{XDTW,Ar}}_{\text{lin}})$	1.97	0.04

Table 6.14: Results of Fishers $z$test between XDTW with different similarity measures for Arabic.

while these datasets that collected under similar conditions are not demonstrably comparable. It does look as though XDTW is useful for Arabic, and is less obvious that it is so for English, but we cannot be certain that this is not just an accidental effect.

Chapter 7

Conclusion and Future Work

7.1 Research Questions and Research Tasks Revisited

In presenting this thesis, we sought to investigate and establish an effective way of detecting paraphrases between two text fragments. Specifically, the aim of this research is to investigate to what extent the standard techniques perform differently when applied to different languages. Accordingly, the mechanisms employed were kept as similar as possible in both languages (English and Arabic). In Chapter 1, the following research questions were raised:

Research Question 1: Can TEQV techniques that work for English be equally well applied to Arabic?

Research Question 2: Is it possible to make the widely-used string edit distances algorithm more robust and reliable in the face of free word-order languages?

Research Question 3: How well does Arabic WordNet (AWN) support semantic similarity measures?

In order to answer these questions, the following research tasks were carried out:

Research Task 1: To collect and annotate comparable English and Arabic datasets.

Research Task2: To develop an implementation of the standard string edit distance algorithm that can cope with free word-order languages by adding an extension allowing transposition of adjacent items..

Research Task3: To integrate Arabic WordNet (AWN) and Pyaramorph in order to obtain potential synsets of an input Arabic word.

Research Task4: To apply and evaluate different variants of the system to see how this affects performance in detecting paraphrases in the fragment texts.

In the following, the research tasks set out above are discussed in more detail.

Research Task 1

In order to examine a TEQV system for any language, an appropriate dataset is needed. The dataset preparing process was split into four sub-tasks:

1. **Collecting Data**: data was amassed by building a comparable corpus of articles that had been automatically acquired from different newswire sources using online RSS feeds. The reason for using RSS feeds was that they provide a simple way to collect articles from multiple sources in a structured manner. Additionally, the RSS technique provides a large number of potential articles reporting the same event or topic on a specific day; it is likely that such articles contain semantically similar sentences.

2. **Pre-processing the Data Collection**: in order to prepare the collected data from sub-task 1, a pre-processing pipeline was applied to normalize the data and prepare them for later processing, i.e. clustering and similarity judgment. The NLP techniques used in this phase are: tokenization and sentence splitting, POS tagging, and morphological analysis.

3. **Data Clustering**: These subtasks are divided into two main steps:

   (a) **Articles clustering**: cluster sets of pairs of similar articles by using simple similarity techniques, i.e. cosine similarity and TF-IDF vectors.

   (b) **Sentences clustering**: cluster set of pair of related sentences from the previously paired articles by using the same simple similarity techniques used in the previous step. The reason for carrying out this clustering is that similar sentences are likely to contain candidate paraphrases.

It is necessary to filter the data to obtain a balanced corpus (sentence pairs) and therefore a threshold of 0.6 was used for classification between positive and negative cases. This threshold was used to find the pairs that were plausibly related for submission to the annotators to mark up and to the system to measure the similarity scores.

4. **Similarity judgements**: this sub-task considers the dataset annotation process. The dataset has been annotated in two ways:

   (a) **Human-based judgment**: assessed by human annotators to generate the Gold Standard dataset. To examine the reliability of the annotators we used inter-rater-reliability (IRR) metrics. Both datasets were used English and Arabic.

   (b) **System-based judgments**: the dataset was aligned through utilisation of alignment methods (i.e. Dynamic Time Warping) to measure the similarity scores between two sentences (sequences), which required converting one sequence to another.

Generally, to ensure the equality of the datasets the same preparation procedures for both English and Arabic datasets were applied. The collection and preparation mechanisms for constructing and designing the experiments in both languages (English, and Arabic) were kept as similar as possible in order to eliminate any differences that might arise from the collection and annotation of the data, thereby allowing improved detection of the differences arising from the linguistic features related to these languages. For Research Task1, the complete details of the conception and implementation for all four of the sub-tasks, explained above, are introduced in Chapter 4. Chapter 5 describes the application of these sub-tasks to English and Arabic. As noted in Chapter 1, Research Task 1 is crucial to answering all the research questions, since without a suitable dataset it is not possible to compare the effectiveness of the various algorithms across the two languages.

Research Task 2

The existing string edit distance algorithm is improved, updated and extended to make it more robust and more effective, by allowing the transposition of adjacent items as an edit operation swap, which partially tackles the problem of free

word order languages. This enhanced version was applied to the DTW algorithm, and we created what we term the eXtended Dynamic Time Warping (XDTW) algorithm. XDTW has been illustrated by using a spelling correction example in Chapter 4. XDTW operations are enhanced to produce cost-effective results when compared to DTW operations by transferring one string to another by calculating the minimum number of basic operations (insert, delete, exchange) and adding another operation (swap). The implementation of the proposed algorithm XDTW for both English and Arabic is explained in depth in Chapter 5. The results in Chapter 6 show that this makes a modest but useful contribution to the accuracy, thus answering Research Question 2. In particular, the increase in accuray achieved by using XDTW for Arabic is greater, at 6%, than for Englishj, at 1%. This reflects the fact that Arabic word-order is freer than English, so that mechanisms for coping with free word-order are more useful for Arabic than for English.

Research Task 3

A major difficulty experienced during the research was that Arabic poses a number of problems that are not present in most other languages. For example, it is a morphologically rich language. In order to overcome this problem in the current project, an Arabic morphological analyser (Pyaramorph) was integrated with AWN. AWN is smaller than the lexical databases of EWN, where AWN is sparse and many words are missing, therefore, the reason of such integration is to ensure that the richness of the lexical resources is available to the Arabic version of TEQV system, at least partially. However, Pyaramorph has certain limitations: (i) in many cases, it returns multiple answers, and (ii) it is based on a fixed vocabulary, and somewhat archaic lexicon, and hence lacks entries for many words. Despite these factors, it is widely used because it has large of vocabulary, is good at dealing with irregular forms, and is freely available. The analyser tool from the Pyaramorph package was utilised. This task contributes to answering Research Question 3, since without the use of Pyaramorph it would only be possible to use AWN with a very small subset of the words in the Arabic texts.

Research Task 4

Finally, the comparative effectiveness of WordNet similarity measures was assessed. To do so, a number of evaluations were carried out to examine the effectiveness of the different similarity measurements. The similarity scores of these methods were compared to the Gold Standard by using their precision rates. Then, a statistical analysis of the precision results of the conducted experiments was carried out, as described in Chapter 5, to determine how the precision varied as the conditions themselves were varied. The results show that using WordNet similarity measures produces better results than the baseline. Of these, the wup WordNet measure is the best similarity measure among the six adopted measures to be integrated with the DTW and XDTW systems. These results answer Research Question 1 by showing that AWN **is** useful for finding paraphrases in Arabic, but less so than for English. Using AWN with wup as the similarity measure improves the accuracy by a factor of 1.09, whereas using EWN with the same similarity measure improves the accuracy for English by around 1.5. We surmise that this is partly because AWN is sparser than EWN, and partly because of the extra ambiguity that arises when working undiacriticsed Arabic text.

7.2 Future Work

There are a number of possible directions for future research and also some remaining open questions related to this field, that are worthy of further investigation. For example:

1. Further experimental investigations are needed to extend the TEQV system by adding a step between data pre-processing and clustering. This step is for inference rules, which play an important role in TEQV.

2. The datasets that were collected consist of 600 sentence pairs for both English and Arabic. The datasets have been created through two-ways decisions as initial dataset, and there is potential to examine TEQV with enhanced datasets with larger context and three-way decisions e.g. yes, no, and unknown.

3. We intend to use our TEQV system to improve the quality of text summarisation and plagiarism detection systems, since such techniques have not been investigated carefully for Arabic.

Ultimately, we highlight the importance of this particular work in the TEQV area. It is a very challenging field, in particular for Arabic where we were faced with different linguistic obstacles for both lexical and semantic levels.

We believe that any future attempts in this regard for languages other than English will elicit interesting features for the whole TEQV community, because they will spotlight the linguistic challenges associated with some of these languages, such as Arabic in the case of this particular research. In addition to the specific contributions stated in this thesis, we are confident that the aim of this study has been accomplished. Achieving the stated goals will offer further significant opportunities to investigate and enhance various real-world challenges, such as document summarisation, machine translation, information retrieval, information extraction, question and answering, text-to-speech generation and plagiarism detection systems. Consequently, this study aimed to partially bridge the gap between the available TEQV techniques for Arabic and those that have been undertaken for other languages such as English.

Appendix A

The histograms of Arabic and English systems

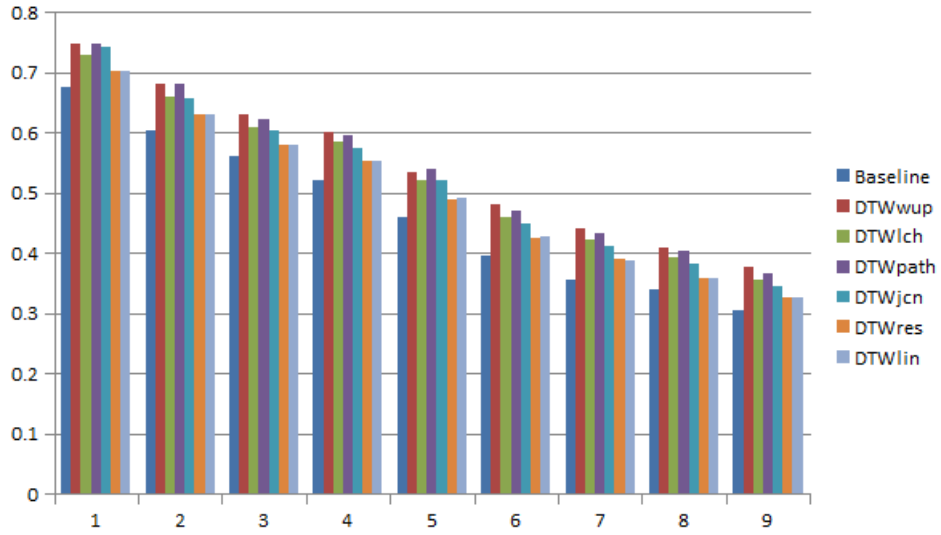


Figure A.1: Arabic DTW

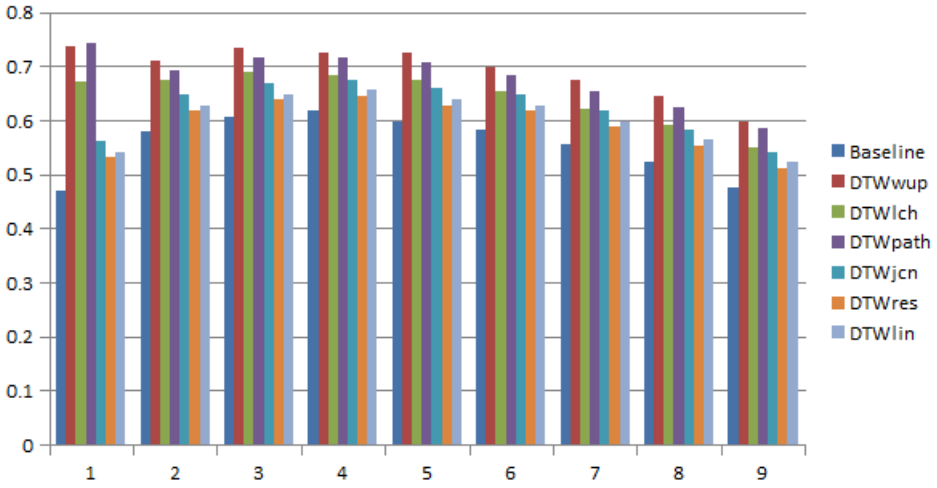


Figure A.2: English DTW

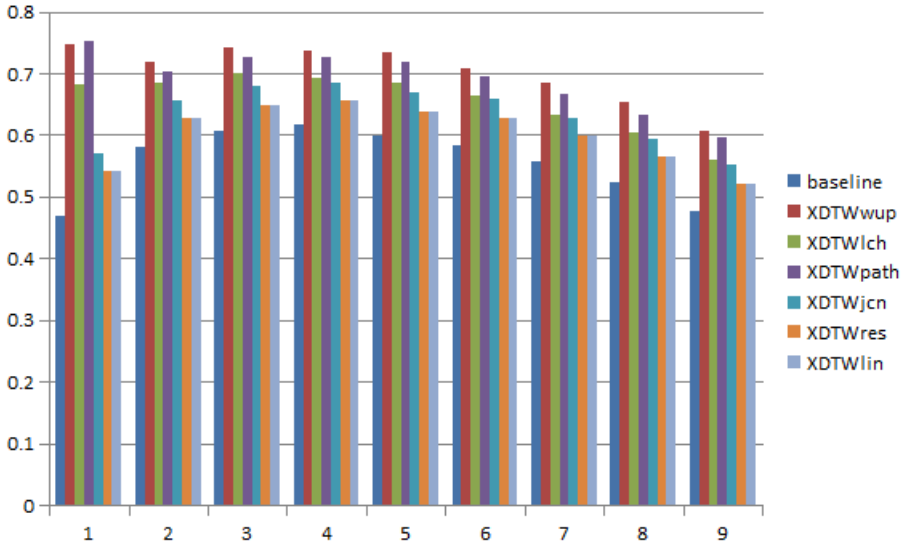
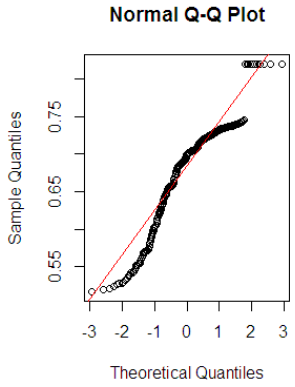


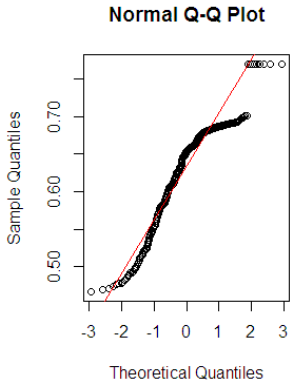
Figure A.3: English XDTW

Appendix B

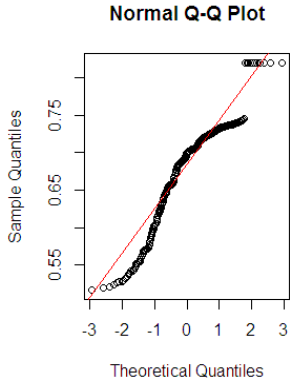
Q-Q plotting tests



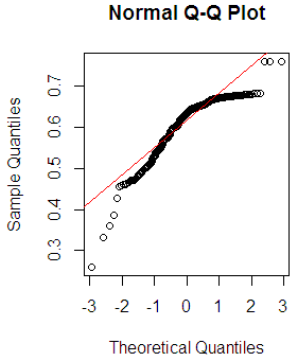
(a) Normality test based on English DTW-wup datasets; the figure indicates non-normality.



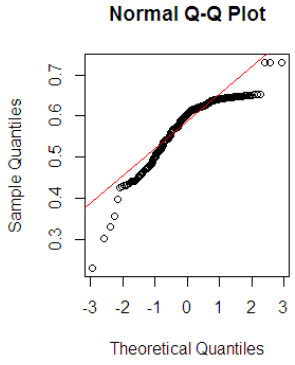
(b) Normality test based on English DTW-lch datasets, the figure indicates non-normality.



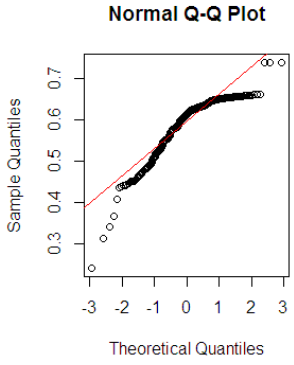
(c) Normality test based on English DTW-path datasets; the figure indicates non-normality.



(d) Normality test based on English DTW-jcn datasets; the figure indicates non-normality.

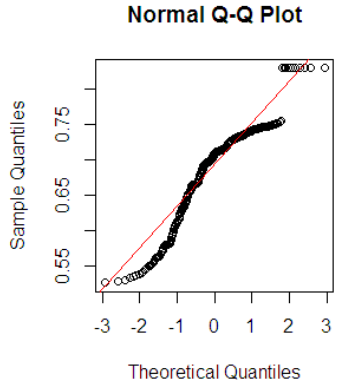


(e) Normality test based on English DTW-res datasets; the figure indicates non-normality.

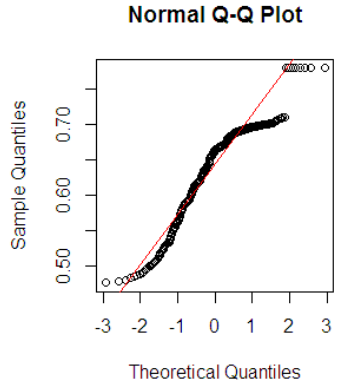


(f) Normality test based on English DTW-lin datasets; the figure indicates non-normality.

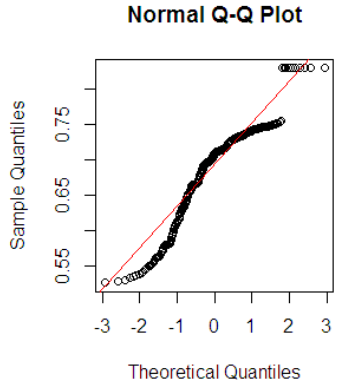
Figure B.1: English DTW



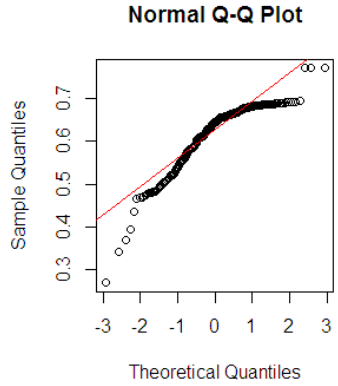
(a) Normality test based on English XDTW-wup datasets; the figure indicates non-normality.



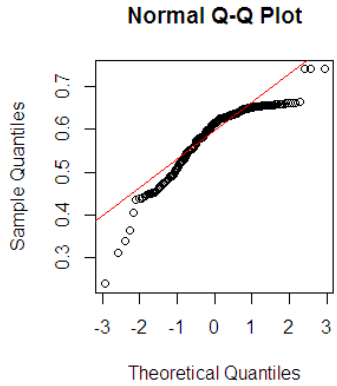
(b) Normality test based on English XDTW-lch datasets the figure indicates non-normality.



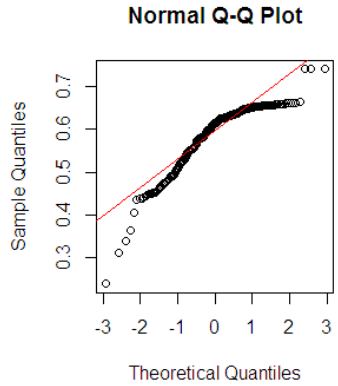
(c) Normality test based on English XDTW-path datasets; the figure indicates non-normality.



(d) Normality test based on English XDTW-jcn datasets; the figure indicates non-normality.

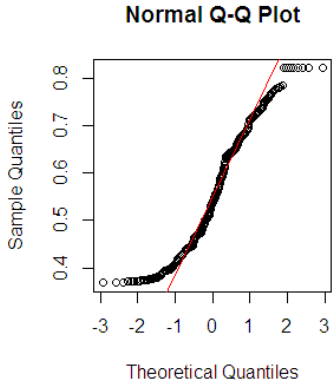


(e) Normality test based on English XDTW-res datasets; the figure indicates non-normality.

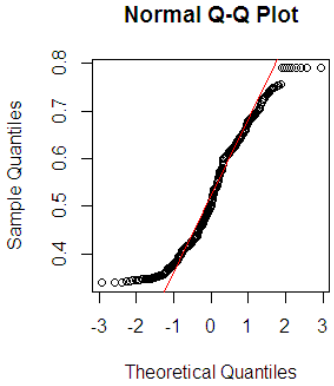


(f) Normality test based on English XDTW-lin datasets; the figure indicates non-normality.

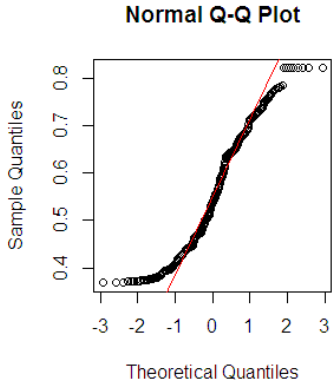
Figure B.2: English XDTW



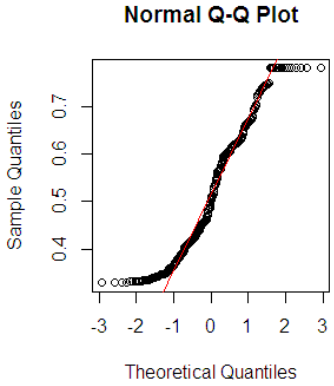
(a) Normality test based on Arabic XDTW-wup datasets; the figure indicates non-normality.



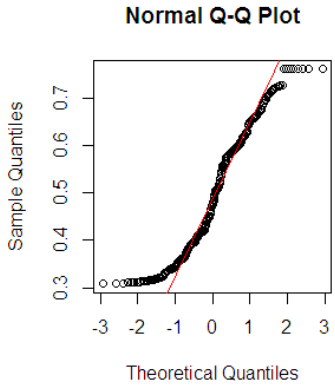
(b) Normality test based on Arabic XDTW-lch datasets, the figure indicates non-normality.



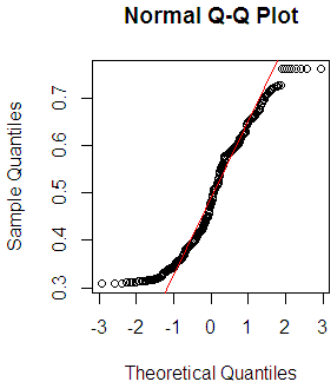
(c) Normality test based on Arabic XDTW-path datasets; the figure indicates non-normality.



(d) Normality test based on Arabic XDTW-jcn datasets; the figure indicates non-normality.



(e) Normality test based on Arabic XDTW-res datasets; the figure indicates non-normality.



(f) Normality test based on Arabic XDTW-lin datasets; the figure indicates non-normality.

Figure B.3: Arabic XDTW

Appendix C

The Kolmogorov-Smirnov and Shapiro-Wilk Tests

	Kolmogorov-Smirnov[1]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
BaseLine	.167	300	.000	.765	300	.000
DTWwup	.137	300	.000	.924	300	.000
DTWlch	.144	300	.000	.924	300	.000
DTWpath	.146	300	.000	.927	300	.000
DTWjcn	.144	300	.000	.884	300	.000
DTWres	.144	300	.000	.884	300	.000
DTWlin	.144	300	.000	.884	300	.000

Table C.1: English_DTW table.

	Kolmogorov-Smirnov[1]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
BaseLine	.167	300	.000	.765	300	.000
XDTWwup	.137	300	.000	.924	300	.000
XDTWlch	.144	300	.000	.924	300	.000
XDTWpath	.146	300	.000	.927	300	.000
XDTWjcn	.144	300	.000	.884	300	.000
XDTWres	.144	300	.000	.884	300	.000
XDTWlin	.144	300	.000	.884	300	.000

Table C.2: English_XDTW table.

[1]Lilliefors Significance Correction

	Kolmogorov-Smirnov[1]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
BaseLine	.135	300	.000	.924	300	.000
XDTWwup	.111	300	.000	.939	300	.000
XDTWlch	.106	300	.000	.940	300	.000
XDTWpath	.119	300	.000	.938	300	.000
XDTWjcn	.116	300	.000	.936	300	.000
XDTWres	.125	300	.000	.936	300	.000
XDTWlin	.119	300	.000	.936	300	.000

Table C.3: Arabic_XDTW table.

References

Aggarwal, C. C. and Reddy, C. K. (2013). *Data clustering: algorithms and applications*. CRC press, Boca Raton, London, New York. 67

Alabbas, M. (2011). ArbTE: Arabic Textual Entailment. In *RANLP Student Research Workshop*, pages 48– 53. 42

Alabbas, M. (2013a). A Dataset for Arabic Textual Entailment. In *Proceedings of Recent Advantages in Natural language processing (RANLP)*, pages 7–13. 106

Alabbas, M. (2013b). *Textual entailment for Modern Standard Arabic*. PhD thesis, University of Manchester, UK. 24, 45

Alabbas, M. and Ramsay, A. (2011). Evaluation of dependency parsers for long Arabic sentences. In *International Conference on Semantic Technology and Information Retrieval (STAIR)*, pages 243–248. IEEE. 53

Alabbas, M. and Ramsay, A. (2012a). Combining black-box taggers and parsers for Modern Standard Arabic. In *Computer Science and Information Systems (FedCSIS), 2012 Federated Conference on*, pages 19–26. IEEE. 46, 100

Alabbas, M. and Ramsay, A. (2012b). Improved POS-tagging for Arabic by combining diverse taggers. In *Artificial Intelligence Applications and Innovations*, pages 107–116. Springer. 99, 100

Alabbas, M. and Ramsay, A. (2013). Optimising Tree Edit Distance with Subtrees for Textual Entailment. In *RANLP*, pages 9–17. 42

Althobaiti, M., Kruschwitz, U., and Poesio, M. (2014). AraNLP: A java-based library for the processing of arabic text. In *Proceedings of the 9th International Conference on Language Resources and Evaluation, LREC*, pages 4134–4138. 58

Altman, D. G. (1990). *Practical statistics for medical research*. CRC press. 112

Attia, M. (2012). *Ambiguity in Arabic computational morphology and syntax: A study within the lexical functional grammar framework*. LAP LAMBERT Academic Publishing, Saarbrücken. 19, 50, 52

Azmi-Murad, M. and Martin, T. (2004). Using fuzzy sets in contextual word similarity. In *Intelligent Data Engineering and Automated Learning–IDEAL 2004*, pages 517–522. Springer. 40

Bach, N. X., Le Minh, N., and Shimazu, A. (2014). Exploiting discourse information to identify paraphrases. *Expert Systems with Applications*, 41(6):2832–2841. 34

Banerjee, M., Capozzoli, M., McSweeney, L., and Sinha, D. (1999). Beyond kappa: A review of interrater agreement measures. *Canadian Journal of Statistics*, 27(1):3–23. 71

Bannard, C. and Callison-Burch, C. (2005). Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 597–604. Association for Computational Linguistics. 36, 39, 45, 46

Baptista, M. (1995). On the nature of pro-drop in capeverdean creole. *Harvard Working Papers in Linguistics*, 5:3–17.

Bar, K. (2013). *Deriving Paraphrases for Highly Inflected Languages, with a Focus on Machine Translation.* PhD thesis, Tel Aviv University. 35, 45, 46, 69

Bar, K. and Dershowitz, N. (2012). Deriving paraphrases for highly inflected languages from comparable documents. In *Proceedings of COLING 2012*, pages 185–200. 30, 35

Bar, K. and Dershowitz, N. (2014). Inferring paraphrases for a highly inflected language from a monolingual corpus. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 254–270. Springer. 64

Barrón-Cedeño, A., Vila, M., Martí, M. A., and Rosso, P. (2013). Plagiarism meets paraphrasing: Insights for the next generation in automatic plagiarism detection. *Computational Linguistics*, 39(4):917–947. 62, 63, 70

Barzilay, R. and Lee, L. (2003). Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 16–23. Association for Computational Linguistics. 34, 35, 45, 67, 69, 135

Barzilay, R. and McKeown, K. R. (2001). Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th annual meeting on Association for Computational Linguistics*, pages 50–57. Association for Computational Linguistics. 35, 41, 45

Barzilay, R., McKeown, K. R., and Elhadad, M. (1999). Information fusion in the context of multi-document summarization. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 550–557. Association for Computational Linguistics. 38

Beesley, K. R. (1996). Arabic finite-state morphological analysis and generation. In *Proceedings of the 16th conference on Computational linguistics-Volume 1*, pages 89–94. Association for Computational Linguistics. 135

Beesley, K. R. and Karttunen, L. (2003). Finite-state morphology: Xerox tools and techniques. *CSLI, Stanford*. 135

Bhagat, R. and Hovy, E. (2013). What is a paraphrase? *Computational Linguistics*, 39(3):463–472. 32

Bhagat, R., Hovy, E., and Patwardhan, S. (2009). Acquiring paraphrases from text corpora. In *Proceedings of the fifth international conference on Knowledge capture*, pages 161–168. ACM. 40

Bhagat, R. and Ravichandran, D. (2008). Large Scale Acquisition of Paraphrases for Learning Surface Patterns. In *Proceedings of ACL-08: HLT*, pages 674–682. 40, 45

Black, W., Elkateb, S., Rodriguez, H., Alkhalifa, M., Vossen, P., Pease, A., and Fellbaum, C. (2006). Introducing the Arabic WordNet project. In *Proceedings of the third international WordNet conference*, pages 295–300. 20, 133

Brockett, C. and Dolan, W. B. (2005). Support vector machines for paraphrase identification and corpus construction. In *Proceedings of the 3rd International Workshop on Paraphrasing*, pages 1–8. 34, 35, 45, 69

Bublitz, W. and Norrick, N. R. (2011). *Foundations of pragmatics*, volume 1. Walter de Gruyter, Berlin, Boston. 29

Buckwalter, T. (2004). Buckwalter Arabic Morphological Analyzer Version 2.0. Linguistic Data Consortium, University of Pennsylvania, 2002. LDC Cat alog No.: LDC2004L02. Technical report. 20, 106, 135

Burrows, S., Potthast, M., and Stein, B. (2013). Paraphrase acquisition via crowdsourcing and machine learning. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 4(3):43. 38

Callison-Burch, C. (2008). Syntactic constraints on paraphrases extracted from parallel corpora. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 196–205. Association for Computational Linguistics. 41, 45

Celikyilmaz, A., Thint, M., and Huang, Z. (2009). A graph-based semi-supervised learning for question-answering. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2*, pages 719–727. Association for Computational Linguistics. 38

Chakravarti, I., Laha, R., and Roy, J. (1967). Kolmogorov-Smirnov (KS) test. In *Handbook of methods of applied Statistics*, volume 1, pages 392–394, New York. John Wiley. 152

Chalabi, A. (2004). Elliptic personal pronoun and MT in Arabic. In *JEP-TALN*. `http://www.afcp-parole.org/doc/Archives_JEP/2004_XXVe_JEP_Fes/actes/arabe2004/TAAC17.pdf`. 47

Clinchant, S., Goutte, C., and Gaussier, E. (2006). Lexical entailment for information retrieval. In *European Conference on Information Retrieval*, pages 217–228. Springer. 38

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46. 70

Cohn, T., Callison-Burch, C., and Lapata, M. (2008). Constructing corpora for the development and evaluation of paraphrase systems. *Computational Linguistics*, 34(4):597–614. 36

Connor, M. and Roth, D. (2007). Context sensitive paraphrasing with a global unsupervised classifier. In *European Conference on Machine Learning 2007*, pages 104–115. Springer. 41

Cormen, T. H. (2009). *Introduction to algorithms*. MIT press, Cambridge. 74

Cruse, A. (2011). *Meaning in language: An introduction to semantics and pragmatics*. Oxford University Press, Oxford. 29

Dagan, I., Glickman, O., and Magnini, B. (2006a). The pascal recognising textual entailment challenge. In Dagan, I., Quiñonero Candela, J., Magnini, B., and d'Alché Buc, F., editors, *Machine Learning Challenges: Evaluating Predictive Uncertainty, Visual Object Classification, and Recognizing Textual Entailment, First Pascal Machine Learning Challenges Workshop, MLCW 2005, Southampton, UK, April 11-13, 2005, Revised Selected Papers*, volume 3944 of *LNCS*, pages 177–190. Springer. 19

Dagan, I., Glickman, O., and Magnini, B. (2006b). The pascal recognising textual entailment challenge. In *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment*, pages 177–190. Springer. 30, 45

Daimi, K. (2001). Identifying syntactic ambiguities in single-parse arabic sentence. *Computers and the Humanities*, 35(3):333–349. 47, 50

Damerau, F. J. (1964). A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176. 77, 79

Dasgupta, S., Papadimitriou, C. H., and Vazirani, U. (2006). *Algorithms*. McGraw-Hill, Inc., New York. 74

Deléger, L. and Zweigenbaum, P. (2008). Paraphrase acquisition from comparable medical corpora of specialized and lay texts. In *AMIA Annual Symposium Proceedings*, pages 146–150. American Medical Informatics Association. 43, 45

Deléger, L. and Zweigenbaum, P. (2009). Extracting lay paraphrases of specialized expressions from monolingual comparable medical corpora. In *Proceedings of the 2nd Workshop on Building and Using Comparable Corpora: from Parallel to Non-parallel Corpora*, pages 2–10. Association for Computational Linguistics. 43

Deléger, L. and Zweigenbaum, P. (2010). Identifying paraphrases between technical and lay corpora. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, pages 3537–3541. 43

Denkowski, M., Al-Haj, H., and Lavie, A. (2010). Turker-assisted paraphrasing for english-arabic machine translation. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 66–70. Association for Computational Linguistics. 64

Diab, M. (2009). Second generation AMIRA tools for Arabic processing: Fast and robust tokenization, POS tagging, and base phrase chunking. In *2nd International Conference on Arabic Language Resources and Tools*, volume 110. 99

Dolan, B., Brockett, C., and Quirk, C. (2005). Microsoft research paraphrase corpus. *Retrieved March*, 29:2008. 63

Duboue, P. A. and Chu-Carroll, J. (2006). Answering the question you wish they had asked: The impact of paraphrasing for question answering. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 33–36. Association for Computational Linguistics. 38

Dulucq, S. and Tichit, L. (2003). RNA secondary structure comparison: exact analysis of the Zhang–Shasha tree edit algorithm. *Theoretical Computer Science*, 306(1-3):471–484. 42

Fellbaum, C. (1998). *WordNet*. MIT Press, Cambridge. 20, 86

Fisher, R. A. and Yates, F. (1938). *Statistical tables for biological, agricultural and medical research*. Oliver and Boyd, Edinburgh. 158

Fleiss, J. L. (1971). Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382. 71

Froud, H., Lachkar, A., and Ouatik, S. A. (2013). Arabic text summarization based on latent semantic analysis to enhance arabic documents clustering. *International Journal of Data Mining & Knowledge Management Process*, 3(1):79–95. 106

Ganitkevitch, J., Van Durme, B., and Callison-Burch, C. (2013). PPDB: The Paraphrase Database. In *HLT-NAACL*, pages 758–764. 30, 63

Glickman, O. and Dagan, I. (2003). Identifying lexical paraphrases from a single corpus: a case study for verbs. In *Proceedings of Recent Advantages in Natural language processing (RANLP)*, pages 166–173. 41, 45

Glickman, O. and Dagan, I. (2004). Acquiring lexical paraphrases from a single corpus. *Recent Advances in Natural Language Processing III. John Benjamins Publishing, Amsterdam, Netherlands*, pages 81–90. 30

Grigonytė, G., Cordeiro, J., Dias, G., Moraliyski, R., and Brazdil, P. (2010). Paraphrase alignment for synonym evidence discovery. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 403–411. Association for Computational Linguistics. 39

Habash, N., Rambow, O., and Roth, R. (2009). MADA+ TOKAN: A toolkit for Arabic tokenization, diacritization, morphological disambiguation, POS tagging, stemming and lemmatization. In *Proceedings of the 2nd international conference on Arabic language resources and tools (MEDAR), Cairo, Egypt*, volume 41, pages 102–109. 99, 135

Harris, Z. S. (1954). Distributional structure. *Word*, 10(2-3):146–162. 31

Hasegawa, T., Sekine, S., and Grishman, R. (2005). Unsupervised paraphrase acquisition via relation discovery. In *11th Annual Meeting of the Japanese Association for Natural Language Processing*, pages 1–8. 40

Hashimoto, C., Torisawa, K., De Saeger, S., Kazama, J., and Kurohashi, S. (2011). Extracting paraphrases from definition sentences on the web. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1087–1097. Association for Computational Linguistics. 19, 30, 42

Hearst, M. A. (1992). Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics-Volume 2*, pages 539–545. Association for Computational Linguistics. 42

Heilman, M. and Smith, N. A. (2010). Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1011–1019. Association for Computational Linguistics. 38

Herrera, J., Penas, A., and Verdejo, F. (2007). Paraphrase extraction from validated question answering corpora in Spanish. *Procesamiento del Lenguaje Natural*, 39:37–44. 39

Ho, C., Azmi Murad, M. A., Doraisamy, S., and Abdul Kadir, R. (2014). Extracting lexical and phrasal paraphrases: a review of the literature. *Artificial Intelligence Review*, pages 1–44. 30, 34, 35, 37, 39, 40

Ho, C., Murad, M. A. A., Kadir, R. A., and Doraisamy, S. (2011). Comparing two corpus-based methods for extracting paraphrases to dictionary-based method. *International Journal of Semantic Computing*, 5(02):133–178. 37, 45, 46

Holes, C. (2004). *Modern Arabic: Structures, functions, and varieties*. Georgetown University Press, Washington D.C. 47

Hwang, Y.-S., Kim, Y. K., and Park, S. (2008). Paraphrasing depending on bilingual context toward generalization of translation knowledge. In *Proceedings of the Third International Joint Conference on Natural Language Processing*, pages 327–334. 45, 46

Ibrahim, A., Katz, B., and Lin, J. (2003). Extracting structural paraphrases from aligned monolingual corpora. In *Proceedings of the second international workshop on Paraphrasing-Volume 16*, pages 57–64. Association for Computational Linguistics. 35, 41, 45

Islam, A. and Inkpen, D. (2008). Semantic text similarity using corpus-based word similarity and string similarity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2(2):10:1–10:25. 44

Jago, M. T. (2007). *Formal logic*. Penrith: Humanities Ebooks, Tirril. 18

Jiang, J. J. and Conrath, D. W. (1997). Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of International Conference Research on Computational Linguistics (ROCLING X)*, pages 19–33. 86

Jusoh, S., Masoud, A. M., and Alfawareh, H. M. (2011). Automated text summarization: sentence refinement approach. In *Digital Information Processing and Communications*, pages 207–218. Springer. 38

Kaji, N. and Kurohashi, S. (2005). Lexical choice via topic adaptation for paraphrasing written language to spoken language. In *Proceedings of the International Joint Conference on Natural Language Processing*, pages 981–992. Springer. 38

Keshtkar, F. and Inkpen, D. (2010). A corpus-based method for extracting paraphrases of emotion terms. In *Proceedings of the NAACL HLT 2010 Workshop on Computational approaches to Analysis and Generation of emotion in Text*, pages 35–44. Association for Computational Linguistics. 34, 35, 42

Kiraz, G. A. (2000). Multitiered nonlinear morphology using multitape finite automata: a case study on Syriac and Arabic. *Computational Linguistics*, 26(1):77–105. 135

Kouylekov, M. (2006). *Recognizing textual entailment with tree edit distance: Application to question answering and information extraction*. PhD thesis, University of Trento. 38

Kovacs-Vajna, Z. M. (2000). A fingerprint verification system based on triangular matching and dynamic time warping. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1266–1276. 73

Landis, J. R. and Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174. 112

Leacock, C. and Chodorow, M. (1998). Combining local context and WordNet similarity for word sense identification. In Fellbaum, C., editor, *WordNet: An Electronic Lexical Database*, pages 265–283. MIT Press, Cambridge. 86

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 8, pages 707–710. 73

Lin, D. (1998). An information-theoretic definition of similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 296–304. 86

Lin, D. and Pantel, P. (2001). Discovery of Inference Rules from Text. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 323–328. ACM. 31, 41, 45

Liu, X., Zhou, Y., and Zheng, R. (2007). Sentence similarity based on dynamic time warping. In *International Conference onSemantic Computing (ICSC 2007)*, pages 250–256. IEEE. 43, 45

Lloret, E., Ferrández, O., Munoz, R., and Palomar, M. (2008). A text summarization approach under the influence of textual entailment. In *Proceedings of the 5th International Workshop on Natural Language Processing and Cognitive Science*, pages 22–31. 38

Madnani, N., Ayan, N. F., Resnik, P., and Dorr, B. J. (2007). Using paraphrases for parameter tuning in statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 120–127. Association for Computational Linguistics. 36, 45

Madnani, N. and Dorr, B. J. (2010). Generating phrasal and sentential paraphrases: A survey of data-driven methods. *Computational Linguistics*, 36(3):341–387. 30, 36

Merrison, A. J., Bloomer, A., Griffiths, P., and Hall, C. J. (2013). *Introducing language in use: A course book*. Routledge, London and New York. 27

Mihalcea, R., Corley, C., and Strapparava, C. (2006). Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference*, pages 775–780.

Mount, D. W. (2004). *Bioinformatics: sequence and genome analysis*. Cold Spring Harbor Laboratory Press, Cold Spring Harbor, New York. 73

Müller, M. (2007). *Information retrieval for music and motion*. Springer, Berlin, Heidelberg. 74

Murata, M., Kanamaru, T., and Isahara, H. (2005). Automatic synonym acquisition based on matching of definition sentences in multiple dictionaries. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 293–304. Springer. 39

Negri, M. and Kouylekov, M. (2009). Question answering over structured data: an entailment-based approach to question analysis. In *RANLP*, pages 305–311. 38

Nelken, R. and Shieber, S. M. (2005). Arabic diacritization using weighted finite-state transducers. In *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, pages 79–86. Association for Computational Linguistics. 48

Ou, S. and Zhu, Z. (2011). An entailment-based question answering system over semantic web data. In *Digital Libraries: For Cultural Heritage, Knowledge Dissemination, and Future Creation*, pages 311–320. Springer, Berlin, Heidelberg. 38

Pang, B., Knight, K., and Marcu, D. (2003). Syntax-based alignment of multiple translations: Extracting paraphrases and generating new sentences. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 102–109. Association for Computational Linguistics. 34, 35, 45

Parapar, D., Barreiro, Á., and Losada, D. E. (2005). Query expansion using WordNet with a logical model of information retrieval. In *Proceedings of the IADIS International Conference on Applied Computing*, volume 2005, pages 487–494. 38

Pasca, M. and Dienes, P. (2005). Aligning needles in a haystack: Paraphrase acquisition across the web. In *Natural Language Processing IJCNLP*, volume 3651 of *LNCS*, pages 119–130. Springer. 42, 45

Pedersen, T., Patwardhan, S., and Michelizzi, J. (2004). WordNet::Similarity: measuring the relatedness of concepts. In *Demonstration papers at HLT-NAACL 2004*, pages 38–41. Association for Computational Linguistics. 85

Rada, R., Mili, H., Bicknell, E., and Blettner, M. (1989). Development and application of a metric on semantic nets. *IEEE transactions on systems, man, and cybernetics*, 19(1):17–30. 86

Ramsay, A. and Mansour, H. (2011). Exploiting Hidden Morphophonemic Constraints for Finding the Underlying Forms of 'weak' Arabic Verbs. In *Proceedings of Recent Advantages in Natural language processing (RANLP)*, pages 448–454. 135

Ramsay, A. and Sabtan, Y. (2009). Bootstrapping a lexicon-free tagger for Arabic. In *Proceedings of the 9th Conference on Language Engineering*, pages 202–215. 99

Ratanamahatana, C. A. and Keogh, E. (2005). Three myths about dynamic time warping data mining. In *Proceedings of the 2005 SIAM International Conference on Data Mining*, pages 506–510. SIAM. 73

Ravichandran, D. and Hovy, E. (2002). Learning surface text patterns for a question answering system. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 41–47. Association for Computational Linguistics. 34

Regneri, M., Wang, R., and Pinkal, M. (2014). Aligning predicate-argument structures for paraphrase fragment extraction. In *LREC*, pages 4300–4307. 34, 35, 45

Resnik, P. (1995). Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th international joint conference on Artificial intelligence*, volume 1, pages 448–453. 86

Riezler, S., Vasserman, A., Tsochantaridis, I., Mittal, V., and Liu, Y. (2007). Statistical machine translation for query expansion in answer retrieval. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 464–471. 38, 45

Ringlstetter, C., Schulz, K. U., and Mihov, S. (2006). Orthographic errors in web pages: Toward cleaner web corpora. *Computational Linguistics*, 32(3):295–340. 35

Romano, L., Kouylekov, M., Szpektor, I., Dagan, I., and Lavelli, A. (2006). Investigating a Generic Paraphrase-Based Approach for Relation Extraction. In *Proceedings of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics*, pages 409–416. 38

Rus, V., McCarthy, P. M., Graesser, A. C., and McNamara, D. S. (2009). Identification of sentence-to-sentence relations using a textual entailer. *Research on Language and Computation*, 7(2):209–229. 30

Ryding, K. C. (2005). *A reference grammar of Modern Standard Arabic*. Cambridge University Press, Cambridge. 53, 57

Sakoe, H. and Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49. 21, 62, 73

Salloum, W. and Habash, N. (2011). Dialectal to standard arabic paraphrasing to improve arabic-english statistical machine translation. In *Proceedings of the first workshop on algorithms and resources for modelling of dialects and language varieties*, pages 10–21. Association for Computational Linguistics. 52

Salton, G. and McGill, M. J. (1986). *Introduction to modern information retrieval*. McGraw-Hill, Inc., New York. 67

Schulz, E., Krahl, G., and Reuschel, W. (2000). *Lughah Al-Arabīyah*. Cambridge University Press, Cambridge. 54

Sekine, S. (2005). Automatic paraphrase discovery based on context and keywords between NE pairs. In *Proceedings of the Third International Workshop on Paraphrasing (IWP-05)*, pages 80–87. 40

Shimohata, M. and Sumita, E. (2002). Automatic paraphrasing based on parallel corpus for normalization. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC-02)*, pages 453–457. 39

Shinyama, Y. and Sekine, S. (2003). Paraphrase acquisition for information extraction. In *Proceedings of the second international workshop on Paraphrasing-Volume 16*, pages 65–71. Association for Computational Linguistics. 30, 35, 41, 45, 69

Shinyama, Y., Sekine, S., and Sudo, K. (2002). Automatic paraphrase acquisition from news articles. In *Proceedings of the second international conference on Human Language Technology Research*, pages 313–318. Morgan Kaufmann Publishers Inc. 40, 41, 45

Souteh, Y. and Bouzoubaa, K. (2011). Safar platform and its morphological layer. In *Eleventh Conference on Language Engineering (ESOLEC 2011)*, Cairo, Egypt. 103, 105

Spearman, C. (1987). The proof and measurement of association between two things. *The American Journal of Psychology*, 100(3/4):441–471. 153

Szpektor, I., Shnarch, E., and Dagan, I. (2007). Instance-based evaluation of entailment rule acquisition. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 456–463. 31

Szpektor, I., Tanev, H., Dagan, I., and Coppola, B. (2004). Scaling web-based acquisition of entailment relations. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP-04)*, pages 41–48. 31

Tatar, D., Mihis, A., Lupsa, D., and Tamaianu-Morita, E. (2009). Entailment-based linear segmentation in summarization. *International Journal of Software Engineering and Knowledge Engineering*, 19(08):1023–1038. 38

Uzuner, Ö., Katz, B., and Nahnsen, T. (2005). Using syntactic information to identify plagiarism. In *Proceedings of the second workshop on Building Educational Applications Using NLP*, pages 37–44. Association for Computational Linguistics. 38

Vila, M. and Dras, M. (2012). Tree edit distance as a baseline approach for paraphrase representation. *Procesamiento del Lenguaje Natural*, 48:89–95. 42

Vila, M., Rodríguez, H., and Martí, M. A. (2010). WRPA: A system for relational paraphrase acquisition from Wikipedia. *Procesamiento del lenguaje natural*, 45:11–19. 63

Wali, W., Gargouri, B., and Hamadou, A. B. (2014). Using standardized lexical semantic knowledge to measure similarity. In *International Conference on Knowledge Science, Engineering and Management*, pages 93–104. Springer.

Wali, W., Gargouri, B., and Hamadou, A. B. (2017). Enhancing the sentence similarity measure by semantic and syntactico-semantic knowledge. *Vietnam Journal of Computer Science*, 4(1):51–60.

Wang, R. and Callison-Burch, C. (2011). Paraphrase fragment extraction from monolingual comparable corpora. In *Proceedings of the 4th Workshop on Building and Using Comparable Corpora: Comparable Corpora and the Web*, pages 52–60. Association for Computational Linguistics. 45, 69

Wang, X., Lo, D., Jiang, J., Zhang, L., and Mei, H. (2009). Extracting paraphrases of technical terms from noisy parallel software corpora. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 197–200. Association for Computational Linguistics. 39

Wang, X. and McCallum, A. (2006). Topics over time: a non-Markov continuous-time model of topical trends. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 424–433. ACM. 67

Wu, D. (2010). Alignment. In *Handbook of Natural Language Processing, Second Edition*. CRC Press, Boca Raton, London, New York. 42

Wu, H. and Zhou, M. (2003). Optimizing synonym extraction using monolingual and bilingual resources. In *Proceedings of the second international workshop on Paraphrasing*, pages 72–79. Association for Computational Linguistics. 41

Wu, Z. and Palmer, M. (1994). Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138. Association for Computational Linguistics. 86

Zhang, K. and Shasha, D. (1989). Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing*, 18(6):1245–1262. 42

Zhao, S., Lan, X., Liu, T., and Li, S. (2009a). Application-driven statistical paraphrase generation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2*, pages 834–842. Association for Computational Linguistics. 36

Zhao, S., Liu, T., Yuan, X., Li, S., and Zhang, Y. (2007). Automatic acquisition of context-specific lexical paraphrases. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 1789–1794. 35

Zhao, S., Wang, H., and Liu, T. (2010). Paraphrasing with search engine query logs. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1317–1325. Association for Computational Linguistics. 42, 45

Zhao, S., Wang, H., Liu, T., and Li, S. (2008). Pivot approach for extracting paraphrase patterns from bilingual corpora. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL:HLT-08)*, pages 780–788. 41

Zhao, S., Wang, H., Liu, T., and Li, S. (2009b). Extracting paraphrase patterns from bilingual parallel corpora. *Natural Language Engineering*, 15(04):503–526. 30, 36, 41

Zukerman, I., Raskutti, B., and Wen, Y. (2002). Experiments in query paraphrasing for information retrieval. In *Australian Joint Conference on Artificial Intelligence*, pages 24–35. Springer. 38