ORTHOGRAPHIC ENRICHMENT FOR ARABIC GRAMMATICAL ANALYSIS

Emad Mohamed

Submitted to the faculty of the University Graduate School
in partial fulfillment of the requirements
for the degree
Doctor of Philosophy
in the Department of Linguistics, College of Arts and Sciences,
Indiana University
July 2010

Accepted by the Graduate Faculty, Indiana University, in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Doctoral Committee

Sandra Kuebler, Ph.D.

Markus Dickinson, PhD

Stuart Davis, PhD

Michael Jones, PhD

Date of Dissertation Defense - June 16, 2010

Acknowledgements

One page is not enough to mention all those people to whom I am indebted. If I had the choice, I would dedicate one page to my adviser, Dr Sandra Kuebler, who succeeded in turning an old-fashioned translator without any technical knowledge to a computational linguist. It was not easy on her, I know, and I acknowledge her efforts and time. Dr Markus Dickinson's help and guidance as well as his insightful comments helped make me a better linguist, and I cannot thank him enough. I am also indebted to Dr Michael Jones, from whom I learned my first lessons in statistics and experiment design, and who agreed to serve on my committee. I also thank Dr Stuart Davis for agreeing to serve on my committee and for his comments and helpfulness.

Although Dr Amr Sabry, professor of computer science at Indiana University, did not read this thesis, my discussions with him on several matters led to solutions to many of the problems I encountered, and I am grateful to him. I am also grateful to my friend Yasser Chuttur for the many interesting discussions and sharing his technical knowledge.

I am especially grateful to my daughter Rahaf for bearing two student parents, and for not complaining, but even bigger thanks go to my wife Bakinaz who, in spite of being a graduate student herself, took better care of me than any housewife would. I know I will continue to thank her for ever.

I have to mention here that my PhD was funded by the Egyptian people through the Egyptian Ministry of Higher Education, and I am grateful to every man, woman and child in Egypt.

ORTHOGRAPHIC ENRICHMENT FOR ARABIC GRAMMATICAL ANALYSIS

Emad Mohamed

The Arabic orthography is problematic in two ways: (1) it lacks the short vowels, and this leads to ambiguity as the same orthographic form can be pronounced in many different ways each of which can have its own grammatical category, and (2) the Arabic word may contain several units like pronouns, conjunctions, articles and prepositions without an intervening white space. These two problems lead to difficulties in the automatic processing of Arabic. The thesis proposes a pre-processing scheme that applies word segmentation and word vocalization for the purpose of grammatical analysis: part of speech tagging and parsing. The thesis examines the impact of human-produced vocalization and segmentation on the grammatical analysis of Arabic, then applies a pipeline of automatic vocalization and segmentation for the purpose of Arabic part of speech tagging. The pipeline is then used, along with the POS tags produced, for the purpose of dependency parsing, which produces grammatical relations between the words in a sentence. The study uses the memory-based algorithm for vocalization, segmentation, and part of speech tagging, and the natural language parser MaltParser for dependency parsing. The thesis represents the first approach to the processing of real-world Arabic, and has found that through the correct choice of features and algorithms, the need for pre-processing for grammatical analysis can be minimized.

Table of Contents

Chapter 1: Introduction

Arabic is a Semitic language whose exact classification in the Semitic family is uncertain. It has been classified as either South Semitic or Central Semitic (Faber, 1997: 12-13). It is thus related to such languages as Hebrew, Syriac, and Tigrinya. Arabic has the largest number of speakers among Semitic languages with a total of 206 million speakers as a first language and 246 million speakers as a second language. (Ethnologue, 1999). Arabic is also one of the six official languages of the United Nations.

In this introduction, I introduce the Arabic orthography and the difficulties associated with processing Arabic, i.e. the automatic segmentation, vocalization and grammatical analysis of the language. While I focus on Arabic in this thesis, many of the problems and solutions presented herein, e.g. word segmentation, may also apply to other non-Arabic languages that use the Arabic script including Farsi, Pashto, and Urdu as well as the other Semitic languages that have similar writing systems like Hebrew, in which the vowels are not usually written.

The rest of this chapter introduces the theme and methods of the thesis. Section 1.1 introduces the Arabic orthography and the challenges it poses to computational processing, and section 1.2. outlines the objective, methods and organization of the thesis.

1.1. The Arabic Orthography

The Arabic script is not limited to the Arabic-speaking world, but can be found in many parts of the world due to the spread of Islam (Bellamy, 1989). Persians, Afghans, and Pakistanis, among others, use the Arabic script for writing their languages.

The Arabic script is an abjad i.e. a writing system in which only the consonants are written, while the vowels are not (Rogers, 2004: 115). Arabic may not be a strict abjad since some vowels, namely the three long vowels, are usually written. 22 of the 28 letters of Arabic have four forms each: isolated, final, initial and medial. "The forms of the initial and medial letters are much reduced, in some cases to such an extent that there is no much resemblance between them and the isolated and final forms." (Bellamy, 1989). Table 1.1 lists the Arabic characters in their isolated forms along with their transliteration in the Buckwalter transliteration scheme used throughout this thesis. The notes column gives some information about the letter, and an empty note indicates that the sound is very similar to its Latin transliteration (see section 1.2.2.1. below).

Arabic	Buckwalter	Notes
أ	>	Hamza (Glottal Stop)[1]
إ	<	Hamza (Glottal Stop)
آ	|	Hamza (Glottal Stop)
ء	`	Hamza (Glottal Stop)
ئ	}	Hamza (Glottal Stop)
ؤ	&	Hamza (Glottal Stop)
ب	b	
ت	t	
ة	p	Taa marbuta: pronounced as t in connected speech and as h in

[1] The rules for writing the hamza are very complicated, and which shape it takes depends on subtle phonological differences that are difficult even for native speakers.

		pauses
ث	v	Pronounced as *th* in the English word *thin*
ج	j	
ح	H	Voiceless Pharyngeal fricative
خ	x	Pronounced as *ch* in the German word *Achtung*.
د	d	
ذ	*	Pronounced as *th* in the English word *then.*
ر	r	Pronounced as Scottish r.
ز	z	
س	s	
ش	$	Pronounced as English *sh*
ص	S	Emphatic form of s. Emphatic sounds are pharyngealized versions of their unemphatic counterparts (Watson, 2002:44)
ض	D	Emphatic form of d.
ط	T	Emphatic form of t.
ظ	Z	Emphatic form of z.
ع	E	Voiced pharyngeal fricative.
غ	g	Voiced uvular fricative: pronounced as French r.
ف	f	
ق	q	Uvular plosive
ك	k	
ل	l	
م	m	
ن	n	
هـ	h	
و	w	Can either be the glide w or the long vowel in boot.
ي	y	Can either be the glide y or the long vowel in keen.
ى	Y	Pronounced like the final a in lemma.

Table 1.1: The Arabic letters in their isolated forms

Six letters of Arabic do not have either initial or medial forms, and are always followed by space whenever they occur. These letters are ا د ذ ر ز و (Buckwalter: A, d, z, r, z, w). When any of these letters occurs in a word, it is not connected to the following letter as is the case with other letters. Figure 1.1 is the Arabic word *Aldlyl* (Eng. the evidence). The letters in the word are connected to each other except for the first and third letters, which are non-connecting character. Arabic is written from right to left, but the transliteration proceeds from left to right.

الدليل

Figure 1.1 An Arabic word with connecting and non-connecting letters

There are only 15 basic forms in Arabic letters, and letter pairs are distinguished by dots. The forms denoting *b*, *t*, *v*, *y*, and *n*, for example, can only be distinguished by dots that vary in their position (above or below the letter), and in their number (one, two or three) as shown in Figure 1.2.

بـ تـ ثـ يـ نـ

Figure 1.2: The Arabic letters b, t, v, y, and n: many Arabic letters can only be distinguished by the number and position of the dots

1.1.1. Problems Associated with the Arabic Orthography:

The problems of the Arabic script are the main focus of this thesis. While two of these problems have dedicated chapters that study them and their effect on Arabic computational linguistics in detail, I will give a short description of the most prominent ones here.

(i) Lack of Short Vowels

Arabic is usually written without the short vowels, and it is up to the writer to include those vowels which she deems necessary to help the reader make sense of the text. Some texts are fully diacritized, e.g. the Quran and children's books, but the majority show little or no vocalization. A ramification of this is that a form like *ktb* can have as many as five different pronunciations, each of which is associated with a meaning of its own: *kataba* (Eng. He wrote), *kattaba* (Eng. he made somebody write), *kutiba* (Eng. it was written), *kuttiba* (Eng. he was made to write) and *kutub* (Eng. books). It has to be noted, however, that each of these is related to the concept of writing included in the root *k t b*. Short vowels have both derivational functions and inflectional functions. For example, the noun *ktb* (*kutub*) has internal vocalization that distinguishes it from the verb *ktb* (*kataba*), but can also be assigned case through word-final short vowel. The same word *kutub* is *kutubu* in the nominative case, *kutuba* in the accusative case, and *kutubi* in the genitive case.

(ii) Complex script that combines tokens, stems and inflections in one orthographic unit

The Arabic orthographic unit, a unit delimited by white space, usually carries more than one token. A form like *wsyktbwnhA*, depicted in Figure 1, for example, carries a conjunction *w*, a future particle *s*, a verbal token *yktbwn*, and a feminine singular third person object pronoun *hA*. The verbal token is made of a verb *ktb*, a masculine present 3rd person inflection *y* and a plural indicative inflection *wn*. For many processes in natural language processing, this mandates word segmentation. For example, we need to separate the preposition from the noun in order to obtain separate noun phrases and prepositional phrases. The chapter on word segmentation deals with the problem of word segmentation.

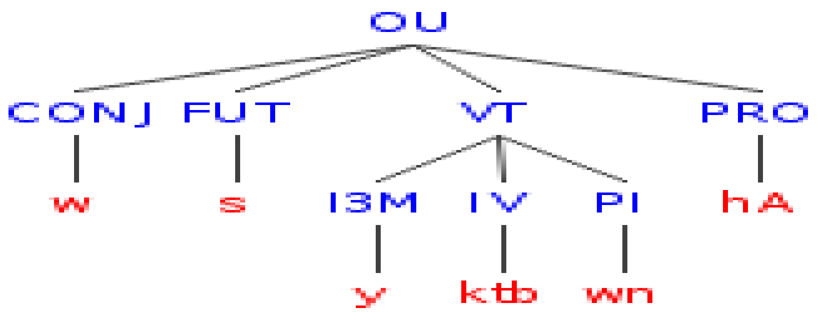

Figure 1.3: The structure of an Arabic orthographic unit

(iii) Orthographic variation

Due to the spread of Arabic in a large geographical area, and because of the local colloquial variations, there are some orthographic variations usually thought of as suboptimal orthography but are nonetheless used by educated and not so highly educated speakers alike. Most of these involve the use, or non-use, of certain diacritics. These do, however, have the potential of affecting the computational processing of Arabic. This is very similar to the use of *it's* in English, where it can mean either *it is*, *it has* or *its*. While the form *it's* is two-way ambiguous in the standard form of English, its substandard use adds a third layer of ambiguity. The case of suboptimal orthography in Arabic is associated with letters: there are three letter groups in Arabic in which the group members are used interchangeably, although they form different and distinct letters in the standard language:

(a) The hamza group: in this group, writers may use any of (A, <, >, |), whose Arabic forms are in Figure 1.4, to mean the same or different things. Given the form *Aktb,* for example, it can be either *Aktb* or *>ktb*, with the former being an imperative masculine singular (Write!) and the latter being an imperfect verb inflected for the first person singular (I write) in the standard orthography.

آ إ أ ا

Figure 1.4: the hamza group

(b) The *h* group comprises two letters: word-final *h* and word-final *p*, whose Arabic forms are shown in Figure 1.5. These two letters look almost the same as the *p* only has two additional dots above it. *p* is a feminine singular marker while *h* is a 3rd person masculine object or possessive pronoun. A word like *mktbh*, which means library in the standard orthography, can also mean *his office* in the substandard orthography.

ه ة

Figure 1.5: The *h* group can only be distinguished by dots

(c) the *y* group. This comprises two letters: *y* and *Y* in word-final positions, depicted in Figure 1.6. The use of these word-final letters varies geographically, as well as at the individual level, but they can be confusing as *Y* is usually a feminine marker while *y* can be a possessive pronoun, a derivation suffix, or the first person singular pronoun. In the standard orthography, *sknY* can only mean "*an abode*", while in the substandard orthography it adds on the meaning "*my abode*".

ي ى

Figure 1.6: The *y* group

Another form of suboptimal orthography, and a more serious one, is what Buckwalter (2004) terms **free concatenation of words**. When a word ends with any of the letters (A, d, *, r, z, w, p), writers feel free not to leave white space between words, and each one of these words can be multiply complex itself. To check how frequent the phenomenon is, I searched Google for the concatenated words وزيرالخارجية (Eng. the-secretary-of-state), and the search turned 56,300 results (4/4/2010). This indicates that the free concatenation of the term is frequent, albeit less so than the standard وزير الخارجية , which has the two words separated by white space, whose frequency is 3,670,000. This free concatenation can contain more than two words, with the only condition that each word ends with one of the non-connecting letters mentioned above. For example, وزيرالخارجيةالمصري (the-Egyptian-secretary-of-state) returned 8 results on Google on the same day.

These different forms of suboptimal orthography cause problems for word segmentation, word vocalization, POS tagging, and parsing. A major problem here is that it is not possible to tell which form of orthography is used, except in very few resources. Each writer will have her own style, and unless there is an editor who forces the standard, like in many newspapers, one ends up with much variation. While this variation does not cause problems for the reader, it can cause many processing problems in the computational treatment of Arabic.

(iv) Diglossia

A diglossia exists when a society has two linguistic norms each of which has its own function and context of use. Ferguson (1959) defines diglossia as follows:

> DIGLOSSIA is a relatively stable language situation in which, in addition to the primary dialects of the language (which may include the standard or regional standards), there is a very divergent, highly codified (often grammatically more complex) superposed variety, the vehicle of a large and respected body of written literature, either of an earlier period or in another speech community, which is learned largely by formal education and is used for most written and formal spoken purposes but is not used by any sector of the community for any ordinary conversation.

Kaye and Rosenhaus (1997) note that Arabic diglossia is an ancient phenomenon dating from the pre-Islamic period, and that the Arabic situation can best be characterized as a continuum from the most formal classic Arabic, CA, to the most colloquial dialect. Modern Standard Arabic (MSA), which is the modern form of the classical language, is a uniting factor between the different Arab countries whose dialects are sometimes mutually unintelligible. According to Kaye and Rosenhaus:

> Generally, MSA is used in the written texts, sermons, university lectures, mostly political speeches and news broadcasts, while colloquial Arabic is used conversing with family or friends, also in radio and TV soap operas. Since there is no clear delimitation between MSA and colloquial Arabic, native speakers often mix the two to various degrees using the so-called "middle" language.

The diglossia phenomenon causes problems for learners of the Arabic languages. According to Abboud and McCarus (1999: V):

> The Arab does not keep MSA and his own dialect separate, but mixes them according to the degree of technical complexity of his subject, the degree of formality of the occasion, etc. When speaking his dialect, he will bring in MSA in varying degrees, and when speaking MSA he may introduce colloquialisms into it if it does not impair understanding on the part of the listener. For a non-Arab to be said to "know Arabic" he or she must master both MSA and any colloquial dialect.

(v) The different levels of the language:

Due to the long history of the Arabic language, it displays variation, both lexically and syntactically, throughout the ages, but users often mix all those levels together. This may lead to ambiguity, especially when a word has taken on a new meaning, but the writer uses it in an old one. This is usually the case in religious discourse and in works of literature.

Lipinsky (2001: 77: 81) classifies Arabic into (1) Pre-Classical Arabic, (2) Classical Arabic, (3) Neo-Arabic, (4) Modern Arabic.

(1) *Pre-Classical Arabic* is described to some extent by early Arabic philologists and remains in some inscriptions dating back to the period from the 2nd century BCE through the 3rd century ACE.

(2) *Classical Arabic* is the language of pre-Islamic poetry that was standardized in the Abbassid period (7th and 8th centuries). This is the form of language that was used by men of letters of all dialects, which is evidence of diglossia as early as the 6th century. The first Arabic grammar was based on this variety of Arabic and it remains the standard grammar up till the present.

(3) *Neo-Arabic* is the Arabic from the 8th century ACE until now. This stage is characterized by the appearance of case endings from nouns, adjectives, and verbs, and thus a more rigid word order. The dual form disappears completely in nouns, verbs, adjectives and pronouns.

(4) *Modern Arabic* is the term used for the current dialects and is spoken by over 300 million people. The current regional dialects are not descendants of classical Arabic, but of different old regional and tribal dialects. No one of the dialects achieved official status except for Maltese, and with the spread of literacy, Modern Literary Arabic has become the most common medium for writing, as it is used today for almost all kinds of writing and some formal speech.

The differences between the Classical variety and Modern Standard Arabic (Neo-Arabic) "are infinitesimal compared with the changes in the European languages over the same period." (Haywood and Nahmad, 1965: 2). Haywood and Nahmad attribute this to the proposition that the Arabic language was hallowed and was not "permitted to change to any marked extent". Consequently, a grammar that was written in the 6th century "still applies largely to modern written Arabic" (ibid).

With the spread of electronic publishing, and the fact that newspapers are now more interactive than they ever used to be, a large portion of the Arab population now has the chance to publish their own comments. In a newspaper directed to the intellectuals, the Egyptian newspaper Alshorouk, columnists use the standard language, but readers' comments are mostly of the middle language type. The following example, a reader's comment, mixes all the levels of the Arabic language, uses colloquialisms, and shows diglossia as well as suboptimal[2] orthography[3]

البرادعي لن يكون رئيسا حتى ولو أعدنا نكتب و نتكلّم للصبح، الكلام اللي البرادعي قاله عن شروط الترشيح والاطار العام لايدلوجيته عن المستقبل المصري تعطي انطباعا ان هذا الرجل هو قفزه كبيره للأمام في الحقيقه التي يجب أن نعترف بيها كلنا أن الشعب المصري غير جاهز أو مؤهل لمثل هذه القفزه، البرادعي بيقول كلام جميل جدا بس عشرة في الميه بس من شعب مصر بيفهمه، و..سماسرة الحزب الوطني بيقولوا كلام وحش جدا لكن للأسف هوه ده الكلام اللي تسعين في الميه من شعب مصر يقدر يفهمه. النهضه محتاجه شغل كتير يا جدعان مش ممكن نعمله في عشرة ولا عشرين سنه، الشخصية المصريه تم تدميرها وتهميشها واضطهادها على مر سنين طويله، وعلشان نعالجها محتاجين برضه سنين طويله، البناء الحقيقي والمشروع القومي الي بيجمع المصريين حوله لازم يكون هو بناء الشخصيه المصريه المعاصره و المسأله دي صعبه و الدليل ظاهر جدا لو أي واحد فيكوا حاور عسكري أمن مركزي قبل التجنيد (لأن بعد التجنيد ثقافته بتكون زادت شويه) وسأله عن حقوقه وحقوق الآخرين، وواجباته وواجبات الآخرين تجاه الوطن هتعرفوا الحقيقه الي عليها السواد الأعظم من شباب مصر في القرى والأقاليم، احنا ليه مش عاوزين نقتنع اننا ممكن نتعب ونضحي من غير ما نحصد، بس ممكن نسيب الحصاد لأولادنا يتمتّعوا بيه ويزرعوا همه زرعه جديده لأولادهم يحصدوها وهكذا. اذا لم نستطيع أن نفهم ذلك فهذه هي الأنانيه المجرّده .

Buckwalter transliteration:

AlbrAdEy ln ykwn r}ysA HtY wlw >EdnA nktb wntkl~m llSbH‹ AlklAm Ally AlbrAdEy qAlh En $rwT Altr$yH wAlATAr AlEAm lAydlwjyth En Almstqbl AlmSry tETy AnTbAEA An h*A Alrjl hw qfzh kbyrh ll>mAm fy AlHqyqh Alty yjb >n nEtrf byhA klnA >n Al$Eb AlmSry gyr jAhz >w m&hl lmvl h*h Alqfzh‹ AlbrAdEy byqwl klAm jmyl jdA bs E$rp fy Almyh bs mn $Eb mSr byfhmh‹ w … <qr> Almzyd..smAsrp AlHzb AlwTny byqwlwA klAm wH$ jdA lkn ll>sf hwh dh AlklAm Ally tsEyn fy Almyh mn $Eb mSr yqdr yfhmh. AlnhDh mHtAjh $gl ktyr yA jdEAn m$ mmkn nEmlh fy E$rp wlA E$ryn snh‹ Al$xSyp AlmSryh tm tdmyrhA wthmy$hA wADThAdhA ElY mr snyn Twylh‹ wEl$An nEAljhA mHtAjyn brDh snyn Twylh‹ AlbnA' AlHqyqy wAlm$rwE Alqwmy Aly byjmE AlmSryyn Hwlh lAzm ykwn hw bnA' Al$xSyh AlmSryh AlmEASrh w Alms>lh dy SEbh w Aldlyl ZAhr jdA lw >y wAHd fykwA HAwr Eskry >mn mrkzy qbl Altjnyd (l>n bEd Altjnyd vqAfth btkwn zAdt $wyh) ws>lh En Hqwqh wHqwq Al|xryn‹ wwAjbAth wwAjbAt Al|xryn tjAh AlwTn htErfwA AlHqyqh Aly ElyhA AlswAd Al>EZm mn $bAb mSr fy AlqrY wAl>qAlym‹ AHnA lyh m$ EAwzyn nqtnE AnnA mmkn ntEb wnDHy mn gyr mA nHSd‹ bs mmkn nsyb AlHSAd l>wlAdnA ytmt~EwA byh wyzrEwA hmh zrEh jdydh l>wlAdhm yHSdwhA whk*A. A*A lm nstTyE >n nfhm *lk fh*h hy Al>nAnyh Almjr~dh.

The text above has the following characteristics:

[2] The term *suboptimal orthography* may not be accurate, as the optimal orthography is much less in common use than the suboptimal one. I will continue to use the term becaue it has been used in the literature.

[3] Source: http://www.shorouknews.com/Columns/column.aspx?id=161102

(1) Use of colloquial vocabulary: for example, the author uses the relative pronoun *Ally*, which is only used in colloquial Arabic. It can also be noticed that the spelling of this relative pronoun is not standardized as it is sometimes written with one *l*, and sometimes with 2 *l*'s. This is an example of mixing the different levels of the language as well as of diglossia.

(2) Use of colloquial pronunciation: the author uses the form >*EdnA*, which is the Egyptian spoken form of *qEdnA* (Eng. we sat down). This use causes ambiguity since >*EdnA* is also a legal Arabic word (Eng. we repeated). This is an example of mixing the various levels of the language.

(3) Use of classical Arabic: the author uses expressions from Classical Arabic such as: *AlswAd Al>EZm*. (Eng. the great majority)

(4) Sometimes the hamzas are used, and sometimes they are not, and an alif is used instead. This is an example of suboptimal orthography.

(5) All the masculine singular pronouns, *h*, are used correctly, but not all the feminine marker symbols, *p*, which are sometimes written as *h*. This is also an example of suboptimal orthography.

1.2. The Thesis

The term definitions, methodologies, and limitations of this thesis are listed as follows:

1.2.1. Definition of Terms

For the purpose of this thesis, the following terms are used in the meanings defined herein, although they may have other meanings elsewhere:

Orthographic Unit: the term Orthographic Unit (OU) denotes a whitespace delimited graphic unit in written Arabic. This may or may not be equivalent to the term **word** as used in the linguistic literature. The term **word** is sometimes used to mean an orthographic unit, and sometimes used to mean **token**, a syntactically independent segment in an orthographic unit.

Segmentation: The term segmentation is used here to denote the setting of boundaries between the various components of the Orthographic Unit, each of which is termed a segment. A segment can be a prefix, a stem, or a suffix. A segment may also be a clitic or an inflectional affix. Affixes are limited to inflectional affixes as derivational affixes are usually of the infix type, and require a treatment of the root and pattern (templatic) morphology. Templatic morphology is beyond the scope of this thesis.

Vocalization: The term vocalization is used here to mean the restoration of the Orthographic Unit-internal short vowels and the consonant doubling diacritic, which are often missing in naturally occurring written Arabic. The task does thus not include the restoration of either case markers or mood markers, which are also represented as short vowels, since these are Orthographic Unit-final rather than internal. The decision not to include case endings and mood markers is based on the observation that Modern Standard Arabic is a language that obeys a specific word order pattern, and that case and mood markers are redundant (Drozdik, 2001: 193-205).

Orthographic Enrichment: The term *Orthographic Enrichment* means adding the missing elements to the orthography, and, in the context of this thesis, denotes both segmentation and vocalization. To be fully enriched, the word *wAlmSrywn* (Eng. and the Egyptians) has to be *wa+Alo+miSoriy~+wn,* with all of the short vowels, the consonant doubling marker, and the boundaries between the different segments marked.

Grammatical Analysis: The term *Grammatical Analysis* in the context of this thesis means both part of speech tagging, which is word-level characterization of the grammatical categories in the language, and parsing, which examines the relations between the words, or the tokens in the case of Arabic, within the sentence.

1.2.2. Thesis Objective

The thesis thus seeks to answer one question: Can orthographic enrichment help Arabic grammatical analysis. To answer this question, other questions need to be answered first:

- How can we perform orthographic unit segmentation of Arabic, a morphologically rich language, that is suitable for grammatical analysis?

- How can we perform vocalization in a Semitic language whose orthography is impoverished? What are the best settings? How much context do we need? Does word segmentation help vocalization?
- Can Arabic part of speech tagging be performed without using gold standard word segmentation? Does segmentation help POS tagging? If yes, which type of segmentation? Does vocalization, whether gold standard or automatic, help POS tagging?
- What is the effect of orthographic enrichment, or the lack thereof, on Arabic dependency parsing? What is the effect of using non-gold standard enrichment and POS tags on the parsing task?

1.2.3. Data

The thesis draws on two sorts of data belonging to roughly the same genre albeit with two different annotation schemes: (a) the Penn Arabic Treebank in the chapter on segmentation, vocalization, and part of speech tagging, and (b) The Prague Arabic Dependency Treebank in the chapter on dependency parsing.

(a) The Penn Arabic Treebank

The Penn Arabic Treebank (Maamouri and Bies 2004) is the main source of data for this thesis. It was compiled by the Linguistic Data Consortium, University of Pennsylvania. The project started in the Fall of 2001 "with the objective of performing human and computer annotations of a large Arabic machine-readable text corpus"[4]. The corpus contains POS tagging, segmentation, tokenization, and vocalization as well syntactic trees of 800,000 tokens from three sources, Agence France Press (AFP), Alhayat newspaper, and the Ummah newspaper. The topics covered are mostly of the economic, political, and sports genres.

The treebank was first annotated automatically using the Buckwalter Arabic Morphological Analyzer (BAMA), then the annotation was passed on to the human annotators to check the quality. The Buckwalter Arabic Morphological Analyzer has three files: Prefix file (99 entries), suffix file (618 entries) and stem file (82158 entries). It also has compatibility tables used for controlling prefix combinations (1648) entries, stem suffix combination (1285 entries) and prefix suffix combinations (598 entries). BAMA uses these combinations to produce all possible analyses of an input word, without trying to perform any disambiguation. The annotation procedure goes as follows:

- Use the Buckwalter Arabic Morphological Analyzer (BAMA) (Buckwalter: 2002) to produce a candidate list of words and their POS tags.
- Annotators go through the list and select the correct analysis of several analyses provided.
- Clitics are split off the words automatically based on the part of speech tags to create treebank compatible tokens since no parsing can be performed without splitting the syntactically significant tokens first.

[4] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2003T06

- The data produced by the step above is parsed using Dan Bikel's parsing engine for Arabic (Bikel, 2004), and then passed to human annotators for correction.

It should be noted that the creators of the Arabic treebank used a variant of the English Penn treebank annotation in order to make the Arabic treebank as compatible as possible with the English one with the purpose of making use of all the tools available instead of creating new tools. This sometimes led them to deviate from the Arabic grammatical tradition (Maamouri and Bies 2004).

(b) The Prague Arabic Dependency Treebank

The Prague Arabic Dependency Treebank (PADT) (Haijc et al 2004) is built on the same data sources as the Penn Arabic Treebank, i.e. newswire corpora, and there is some overlap between the data in both treebanks. It differs in that it applies three levels of annotation, and that it uses dependency syntax rather constituent analysis. The three levels of analysis used in PADT are: (1) Morphological analysis, (2) Analytical analysis, and (3) tectogrammatic analysis. The PADT is used only in chapter 5 on dependency parsing, and full details of this data source are given therein.

The variety of Arabic used in these two sources is Modern Standard Arabic in a limited domain. The data comes from a collection of newswire text mostly from the political, economic, and sports genre. While there may not be a difference in terms of grammatical structures between the genres, the differences in the lexical items may be impediment to extending the results of this thesis beyond the newswire texts. Different domains may require different settings, but the limitedness of data does not allow us to investigate the questions of domain adaptation.

1.2.2.1. Transliteration

Transliteration is the practice of transcribing a word or text written in one writing system into another writing system (Kashani et al, 2006). Transliteration is thus a mapping between two writing systems, possibly of different orthographies. Research on Arabic transliteration consists in trying to map the many possible (Latin) variants of an Arabic word, which is complicated by the lack of vowels, and the unavailability of some sounds across languages. For example, English lacks the guttural sounds that are found in Arabic, and Arabic lacks the P sound in English. Kashani et al report a total of 87 different - and official- transliterations for the name (معمر القذافي), Libya's strongman, including Qathafi, Kaddafi, Qadafi, Gadafi, Gaddafi, Kathafi, Kadhafi, Qadhafi, Qazzafi, Kazafi, Qaddafy,Qadafy, Quadhaffi, Gadhdhafi, al-Qaddafi, Al-Qaddafi, and Al Qaddafi.

Most transliteration research targets either Machine Translation or Information Retrieval cf. e.g. (Abduljaleel and Larkey (2003), Freeman et al 2006, and Stalls and Knight 1998), and while these are very important issues, for Arabic transliteration to be effective for presenting Arabic to non-Arabs, the mapping has to be one-to-one and reversible, and this is what the Buckwalter transliteration system, which is used in the Arabic Treebank, offers.

The Arabic Treebank and the Prague Arabic Dependency Treebank, as well this thesis, use the Buckwalter transliteration scheme, and this carries two advantages: (a) it uses ASCII, and this makes it easy for readers to make sense of the Arabic examples without having to know the Arabic orthographic system, and (b) it is a 1 to 1 lossless mapping between Arabic characters and the Latin ones. The Buckwalter transliteration scheme transliterates Arabic as is without including any interpretations or analyses that are not usually found in the original. It is thus completely reversible. The Buckwalter Arabic transliteration scheme is, however, not without problems. One of the problems is that it is difficult to read since some of the characters used have a different pronunciation in the English language. For example, *v* is used to transliterate the Arabic letter ث, pronounced like the first sound of the word three, but this is not intuitive. The other problem is that the Buckwalter scheme uses some punctuation marks to represent Arabic letters while those marks themselves can be part of the Arabic text. An example of this is the { and } symbols which are sometimes found in the original Arabic text as punctuation, and if we use the Buckwalter scheme to transliterate them, we obtain Arabic alphabetic characters instead of punctuation marks.

To give an example of the Buckwalter transliteration scheme, consider the proper noun معمر القذافي above. The Buckwalter scheme transliterates it as *mEmr Alq*Afy* without trying to insert vowels or performing any normalization. For example, if there is a misspelling in the Arabic, it will remain in the romanized version. Also, if there is any vocalization in the original, it will cross over to the romanized version without any modification. It is this mapping that guarantees complete reversibility, and it can thus be used to honestly represent Arabic.

One of the peculiar decisions made in the transliteration and the morphological analysis in general in the Arabic treebank is the distinction between the wasla alif and the regular alif although the Arabic orthography does not make this distinction except in the Qur’an.

1.2.3. The Algorithm

Throughout this thesis, I will make use of the Memory-Based Learning algorithm as I model the problems of Arabic orthography as classification tasks.

Memory-based learning is based on the hypothesis that when people are faced with a new cognitive task, they handle it based on similarity with instances stored in memory "rather than on the application of mental rules extracted from earlier experiences” (Daelemans et al, 2009: 20). The approach has been used in different fields and has been given different names such as similarity-based learning, example-based learning and instance-based learning (ibid). One of the reasons for the success of this lazy learning algorithm is that natural language exhibits a high percentage of sub-regularities or irregularities, which cannot be distinguished from noise. Eager learning paradigms smooth over all these cases while memory-based learning still has access to the original instance. Thus, if a new instance is similar enough to one of these irregular instances, it can be correctly classified as such. Additionally, MBL is a paradigm that is capable of handling symbolic features with a high number of different feature values. This allows us the use of complete context words as features.

A memory-based system comprises two components: (a) a learning component which depends on storage, or memory, and (b) a testing components which tests the new instances against the stored examples based on similarity. Essential to this process is a similarity metric that measures the distance between the new example and the stored examples. The example in the memory that has the shortest distance to the new example decides the class of the new example.

To give a concrete example, I will present the task of choosing a new book for Rahaf, a 5-year old girl. I have picked a book, and I want to know in advance whether Rahaf will like it. I know from experience that Rahaf received five books as gifts before, and I will list them according to their features. There are four features that describe the books:

- The size of the book, and this has the values big and small.
- The book publisher with the values wd, for Walt Disney, and nj, for Nick Jr.
- The has_monster feature which asks whether the book has monster stories, and it has the values yes or no.
- The language feature, which has two values: English or Arabic.

In order to cast the Rahaf Book problem as a classification task, I create vectors of equal lengths, each representing one of the five books, as depicted in Table 1.2. The last column in the table is the class which tells us whether Rahaf liked the book, and it naturally has two values: likes_it or dislikes_it.

	size	**Publisher**	**has_monsters?**	**language**	**likes_it?**
1	big	wd	no	english	likes_it
2	big	nj	no	arabic	dislikes_it

3	small	wd	yes	arabic	likes_it
4	small	nj	yes	arabic	dislikes_it
5	small	wd	yes	english	likes_it

Table 1.2: The Rahaf Book problem features

Now that we have our experience stored in our memory, we are faced with the situation in which we must buy a new book for Rahaf for her birthday. We pick a book of a big size, published by Nick jr., with some monster stories, and written in Arabic, and we need to decide whether Rahaf will like the book. To do this, we measure the distance between the new book and the books in the memory.

I turn the characteristics of the book which I am planning to buy into a vector of similar nature, with the exact same order, and we obtain the vector:

big nj yes arabic

The only difference between the new book vector and the ones in our data set is that the new book does not have a class associated with it (likes_it, dislikes_it), and this is exactly what I want to measure. Any measurement requires a distance metric, and for this example, I use the Overlap Metric (also known as the Manhattan Distance and the City Block distance). The Overlap Metric is shown in Figure 1.6 where Δ(X, Y) is the distance between instances X and Y, represented by *n* features and δ is the distance per feature.

$$\Delta(X,Y) = \sum_{i=1}^{n} \delta(x_i, y_i)$$

where:

$$\delta(x_i, y_i) = \begin{cases} abs(\frac{x_i - y_i}{max_i - min_i}) & \text{if numeric, else} \\ 0 & \text{if } x_i = y_i \\ 1 & \text{if } x_i \neq y_i \end{cases}$$

Figure1.6: The Overlap Metric

The distance between two vectors is the summation of the distances between vector elements. So, to measure the distance between the new book vector, and the first book in our data set, we find that:

- Both books are big, so the size distance is 0.
- The publisher is different in each case, so the publisher distance is 1.
- One of the books has monster stories, and the other does not, so the monster difference is 1.
- The language of the new book is Arabic, while the language of book 1 in the dataset is English, so the language distance is 1.

Summing all these element distances, we find the difference between the two vectors to be 3, but we still have to find the difference between the new instance and every book in the dataset. Table 1.3 shows that the vectors with the minimal distance are those representing books no. 2 and no. 4, and since the class assigned to both of these is dislikes-it, we conclude that Rahaf will most probably not like the new book.

	size	**publisher**	**has_monster?**	**language**	**distance**
1	0	1	1	1	3
2	0	0	1	0	1
3	1	1	0	0	2
4	1	0	0	0	1
5	1	1	0	1	3

Table 1.3: Distances between the test instance and the stored instances

If, however, there were many instances in the dataset with the same distance to the new instance, but with different classes assigned, then we may decide to take the majority class among these. We may also decide to take the majority class of the instances within a number of examples, and not a single example.

While the example above is illustrative, it only represents the naive implementation of memory-based learning IB1 algorithm, which is rather inefficient since the new instance has to be compared against every example in the dataset no matter how large that is.

I use the TiMBL implementation of memory-based learning which has the following advantages:

a. TiMBL implements a number of distance metrics, each of which can be useful in specific settings. For example, the Levenshtein distance can be useful when we want to collapse feature values that have substrings in common. In dealing with linguistic values, we may want to tell the classifier that both *play* and *plays* represent the same word.

b. TiMBL indexing makes it very efficient, as it does not store all examples in memory, but rather collapses similar examples and performs a fair amount of bookkeeping for this purpose.

TiMBL also implements the IGTREE algorithm, which is faster than the IB1 algorithm outlined in the example above, sometimes at the cost of accuracy. The IGTREE algorithm stores all the information in a decision tree structure in which "instances are stored as paths of connected nodes which contain classification information. Nodes are connected via arcs denoting feature values" (Daelemans et al, 2009: 31). Information Gain determines in which order feature-values are added as arcs in the tree.

Essential to the quality of the classification process is a feature weighting mechanism which assigns different weights to features according to how much they contribute to the classification. If we look at the naive example above, we will find that we can reach the decision of whether to buy the book for Rahaf or not, based only on the publisher feature: every time a book is published by Walt Disney, Rahaf likes it, and every time a book is published by Nick Jr., Rahaf does not like it. It may even be the case that it is Walt Disney against all other publishers and not specifically against Nick Jr. per se. TiMBL assigns weights to features through a number of weighting mechanisms, of which two are used in this thesis: Gain Ratio and the Modified Value Difference Metric.

1.2.3.1. Weighting by Information Gain / Gain Ratio

Information gain weighting looks at each feature in isolation to measure its contribution to the discovery of the class. To do so, it measures the difference between the classification with and without the feature. In our example above, Information gain looks at the feature PUBLISHER to determine how much its existence helps in reaching the decision of whether Rahaf will like or dislike a certain book. It does so for every individual feature, and then assigns weights according to this relative importance. IG is defined as:

$$w_i = H(C) - \sum_{v \in V_i} P(v) \times H(C|v)$$

where *C* is the set of class labels, *H(C)* is the entropy of the class labels, and *Vi* is the set of values for feature *i*. The probabilities are estimated from relative frequencies in the training set (Daelemans et al, 2009: 28)

One problem with information gain is that it tends to over-estimate the weight of features with large numbers of values. If one of the features, for example, is CUSTOMER_SOCIAL_SECURITY_NUMBER, this feature will have so many different values, and will have a high information gain value, although this feature is not expected to add anything to the classification accuracy, and will not generalize to any new examples. To remedy this problem, Quinlan (1993) has introduced a normalized version called Gain Ratio, which is simply Information Gain divided by *si(i)*, which is the entropy of the feature values. The Information Gain formula is shown in shown in Figure 1.7.

$$w_i = \frac{H(C) - \sum_{v \in V_i} P(v) \times H(C|v)}{si(i)}$$

$$si(i) = - \sum_{v \in V_i} P(v) \log_2 P(v)$$

Figure 1.7: Gain Ratio

1.2.3.2. Weighting by MVDM

The features in linguistics are usually of a symbolic nature, and Information Gain and Gain Ratio restrict the similarity or dissimilarity to exact match, without considering feature values, but this may not be what we want. Modified Value Difference Metric (Stanfill and Waltz, 1986 and Cost and Salzberg 1993) looks at the similarity between values of each feature in terms of their co-occurrence with the class. For example, if the classifier's task is to determine whether a certain text was written by a man or a woman, and in the training set, the two POS tags DET and VERB usually co-occur with the class WOMAN, while the NOUN value usually co-occurs with the class MAN, then VERB is more similar to DET than it is to NOUN. The equation for MVDM is shown in Figure 1.8. For the distance between two values *v1*, *v2* of a feature, we compute the difference of the conditional distribution of the classes *Ci* for these values.

$$\delta(v_1, v_2) = \sum_{i=1}^{n} |P(C_i|v_1) - P(C_i|v_2)|$$

Figure1.8: The Modified Value Difference Metric

1.2.4. Assumptions and Limitations

The thesis depends on treebank data, either from the Penn Arabic Treebank or the Prague Arabic Dependency Treebank, and assumes that the Arabic in both is representative of naturally occurring Arabic, and no attempt will be made to go beyond the treebank data. Although naturally occurring Arabic may be different, in many important respects, from the ATB and PADT Arabic, these are the only annotated data source available.

Another assumption that follows from the assumption above is that I will use only data in the standard orthography, and I will not treat orthographic variation in this thesis. The reason for this is that the non-standard issues in the orthography are varied, and they deserve an investigation of their own.

A third assumption is that I will take sentences as is and will not try to divide the text myself, i.e. I will assume gold standard sentence markers. This is especially important in dealing with the dependency parsing data in which some sentences are over 300 tokens long. Most of these sentences can, or should, be divided into smaller sentences, but I do not attempt this in the thesis.

Depending on the treebank data also means that some of the characteristics of Arabic (diglossia, the different levels of the language) will not be treated in this dissertation since these are not represented in the data.

1.2.5. Thesis organization

The thesis comprises 6 chapters as follows:

Chapter 1: Introduction. The introduction introduces the Arabic language, and the features that make it both hard and interesting from a computational perspective. The chapter also introduces the data and algorithms used throughout the thesis as well as the organization.

Chapter 2: Arabic Word Segmentation. This chapter handles the complex nature of the Arabic script and outlines a scheme for disentangling the various parts of the Arabic word, be they lexical, inflectional, or clitical. This process is the first step in the automatic enrichment of the language and is necessary for later chapters that handle POS tagging and parsing.

Chapter 3: Vocalization. This chapter outlines a scheme for short vowel restoration in Arabic. This is the second step in the automatic enrichment of the Arabic orthography and it can be useful in disambiguating an otherwise poly-ambiguous script. The chapter also includes a final section on the interaction between segmentation and vocalization and whether these two enrichment processes can help each other.

Chapter 4: Part of Speech Tagging. This chapter outlines assigning morphosyntactic categories to Arabic words. In the assignment of such categories, the chapter examines the effect of segmentation and vocalization, and whether they can help improve the tagging process. This chapter is the first step in grammatical analysis, and the methods developed in it will help also in the chapter on parsing.

Chapter 5: Dependency Parsing. This chapter introduces a real-world experiment in parsing Arabic in a way that does not assume anything but a string of words. The chapter applies word segmentation, vocalization, stemming, and part of speech tagging before it passes the text to the parser.

Chapter 6: Conclusion. The conclusion sums up the results of the thesis and how it has contributed to understanding how Arabic can be processed computationally.

Chapter 2: Arabic Orthographic Unit Segmentation

2.1. Introduction

In this chapter, I start my investigation of orthographic enrichment by studying the structure of the Arabic word and why word segmentation is necessary for the computational processing of Arabic. The need for segmentation arises from the fact that Arabic is a morphologically rich language whose orthography is even more complicated by the addition of clitics. The Arabic orthographic unit, for example: *wllmhndsAt* (Eng. and for the female engineers), contains a conjunction, a preposition, a definite article, a stem, and a feminine plural marker. This is not the only form involving the stem *mhnds* (Eng. engineer) as the form inflects for both number and gender, and the same stem can be found in hundreds of words. This can cause data sparseness, and word segmentation has the objective of combating this sparseness by reducing morphologically complex units to simpler ones. While *mhndsAt*, *mhnds*, *mhndswn*, *mhndsAn* and *mhndsp* may be treated as different words, segmentation reduces them all to the stem *mhnds*, and some number and gender inflections. Word segmentation can thus help in such tasks as lexical acquisition, part of speech tagging, syntactic parsing, and information retrieval. In collocation extraction, for example, the collocation *ElAmp tjAryp* (علامة تجارية) (Eng. trademark) may occur in the forms *ElAmAt tjAryp*, *AlElAmAt AltjAryp*, *ElAmAthA AltjAryp*, *ElAmtAn tjArytAn*, as well as many other forms, and only through segmentation can we find that they are forms of the same word.

This chapter is divided into three major parts: (1) definition of terms, and (2) computational approaches to Arabic word segmentation, and (3) my own approach to word segmentation and tokenization. In part one, I define the terms necessary to understand the structure of Arabic written forms including such terms as orthographic unit, clitics and suffixes, segmentation, and tokenization. In part two, I review the literature on Arabic word segmentation and tokenization. I then present my own approach to the problem.

2.2. Definition of Terms

In order to avoid ambiguity, I will use the following terms in the meanings associated with them in the definitions, although some of them may have definitions outside the context of this thesis.

Orthographic Unit[5]: an orthographic unit (OU) is a written form delimited by whitespace. This orthographic unit may or may not be equivalent to the term *word*. In Arabic grammar books (e.g. Al-Hamalawy, 1998) a word is usually used to mean a stem plus affixes, although the written form in Arabic usually contains also clitics. The orthographic unit will then include those clitics which may be excluded by the traditional definition. The OU *lmSr* (Eng. for Egypt) thus counts as one orthographic unit, but as two linguistic words. The term word is often also used to mean a space-delimited unit notwithstanding.

Clitic, Affix and Stem. To understand the structure of the Arabic orthographic unit, it is necessary to introduce clitics and affixes. The difference between clitics and affixes is important since it is at the heart of the distinction between segmentation and tokenization, where in segmentation, I set boundaries between all the units in the orthographic unit, while in tokenization, I consider only clitics. Crystal (2008:75),

[5] I repeat the definition in chapter 1 here for the purpose of clarity, and since new information is added.

defines clitics as a term used in grammar to refer to a form which resembles a word, but which cannot stand on its own as a normal utterance, being phonologically dependent upon a neighboring word (its host) in a construction. Clitics can be further divided into proclitics, those that attach to the following words, and enclitics, those that attach to the preceding word. The Summer Institute of Linguistics (SIL) glossary defines a clitic as "a morpheme that has syntactic characteristics of a word, but shows evidence of being phonologically bound to another word" (www.sil.org).

The **stem** is the basic element in the orthographic unit, and it carries the lexical content.

Proclitics: Up to three proclitics can occur in the orthographic unit. The first proclitic is usually a member of the set {f, w} (Eng. and, then). Both of these function as conjunctions and are mutually exclusive. The orthographic unit *wktAb* thus means *and-a-book.* The second proclitic set has only one member, the affirmative particle *l* (Eng. definitely, certainly). The third proclitic is one of the set {k, l, b, w} (Eng. like, for, with, by). Although there are many other prepositions, these three are the only clitical ones.

Enclitic: Enclitics are either possessive pronouns or object pronouns. Possessive pronouns in Arabic are attached to the end of the noun. Possessive pronouns inflect for person, gender and number. Possessive pronouns and object pronouns have the same forms, and are thus ambiguous. The difference between these two functions of the same form can be established through the type of the base form. If the base form is verbal, the object pronoun interpretation is the correct one, if the pronoun is attached to a noun, it is interpreted as possessive. When the base form itself is ambiguous, we have a complex situation. To illustrate, the base form *ktb* can be either a noun meaning *books* or a verb meaning *(he)-wrote*, and the clitic pronoun *nA* can either mean *us* or *our*. The orthographic unit *ktbnA* is assigned the segmentation *ktb+nA* in both cases, but it is ambiguous between *our books* and *he wrote us* (among other interpretations).

The Base Form. The base form is a complex of a stem and inflections. Noun inflections indicate number and gender. Number and gender affixes can be only seen when the noun is non-masculine singular, which is the unmarked form. The feminine marker in Arabic is the *taa marbuta*, p in Buckwalter transliteration, the feminine marker changes into *t* when the number is dual, and disappears completely when the number is plural. There are two dual markers in Arabic: *An* in the nominative case and *yn* in the accusative and genitive cases. The number system in Arabic is complicated, and often there is no plural marker at all. In the construct state, which is the Arabic form of compound nouns, the plural and dual markers are reduced to the first letter of the affix (Haywood and Nahmad, 1965: 63). For example, in *mdyrw AlSrkp* (Eng. managers of the company), the *w* in *mdyrw* is short for the plural marker *wn*.

Segmentation. For the purpose of this thesis, segmentation is the process of analyzing an orthographic unit into its constituent segments. A segment can be either an affix or a clitic, or it can be the stem itself. Segmentation does not make a difference between affixes and clitics, and its sole purpose is to delimit the different components of the orthographic unit, each of which is called a segment. A segment cannot be more than one simple unit. Given the orthographic unit *lsyArAthm* (Eng. For their cars), a segmentation process will return *l+syAr+at+hm,* without indicating that *l* is a clitic while *At* is an inflectional suffix.

Tokenization. For the purpose of this thesis, tokenization is taken to mean splitting off those elements of the orthographic unit that have a syntactic functions. Each resulting unit is called a **token**. In the word *lsyArAthm* in 2.2.3 above, *l* is a token, but *At* is not. The orthographic unit will then be tokenized as *l+syArAt+hm*, but segmented as *l+syAr+At+hm*.

2.2.1. Segmentation vs. Tokenization

Word segmentation means setting the boundaries between every segment in the word, be it inflectional or otherwise. For example, the orthographic unit *wkmhndsAtnA* (Eng. and like our female engineers) is segmented as *w+k+mhnds+At+nA*, and I call each unit in the orthographic unit a segment. The word thus has 5 segments of which two are proclitics, *w* and *k*, one is the stem, *mhnds*, one is an inflectional suffix denoting feminine plural, *At*, and the final one, *nA*, is an enclitic which serves as a first person plural possessive pronoun (Eng. our). Tokenization, on the other hand, is the process by which clitics are separated from the stem and its inflectional affixes. The orthographic unit is thus tokenized as *w k mhndsAt nA*, where it has 4 tokens since inflectional suffixes count as part of the stem. Tokenization is thus potentially subsumed under segmentation since by performing segmentation, we are also performing tokenization, but the opposite is not true. To give an example of the structure of a tokenized orthographic unit, the verb *wsyktbwnhA* (Eng. and they will write it) is tokenized as in Figure 2.1(a) and segmented as in in Figure 2.1(b). The labels are explained in Appendix A.

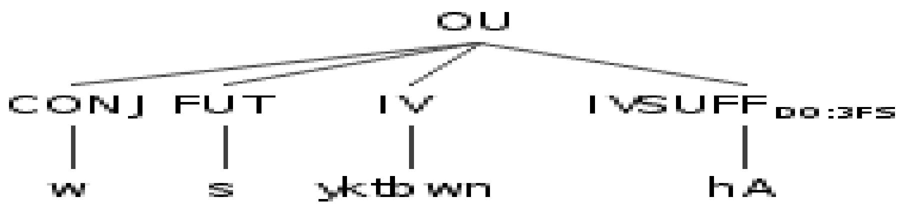

Figure 2.1(a): Tokenization of the verbal OU *wsyktbwnhA*. The POS tags are listed for the sake of description and are not part of the tokenization process.

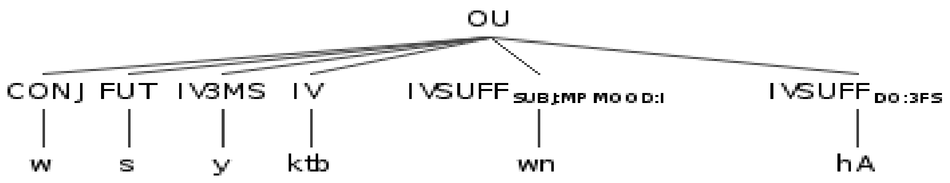

Figure 2.1(b): Segmentation of the verbal OU *wsyktbwnhA*

The distinction between segmentation and tokenization is important since each has its own use in Arabic computational processing, and since the decision of whether to treat segments or tokens has ramifications for all the processes in language analysis. For example, in performing part of speech tagging, I may decide to assume tokens or segments as input, and the tagset will be different in each case. The tagset is much smaller in size in the case of tokenization since inflectional affixes will not require their own tags. When inflectional affixes are assigned their own tags, this leads to more complex tags for the base form. In parsing, tokenization is the default procedure since syntactic relations are expressed through clitics rather than affixes, which are morphological.

2.3. Arabic Non-Stem Segments: A Linguistic Overview

Arabic non-stem segments include both affixes and clitics, and these are not usually easily distinguishable from each other or from the original letters of the stem in the written form due to the multiple functions they perform (i.e. they are ambiguous by nature). The following is a comprehensive list of non-stem segments, their functions, orthographic variations, and ambiguities:

- Hamza (أ), > in Buckwalter transliteration. The Hamza can be either a clitic or a prefix, and can also be part of the stem. For example, the orthographic unit *>mr* (أمر) is at least three way ambiguous as the Hamza can be part of the word, functioning as an imperfective first-person singular marker, in which case it is an inflectional prefix meaning *I* and the orthographic units thus means *I pass*, or it can be a clitical question word, in which case the OU means *Did he pass?*

- Alif (ا) (A in Buckwalter transliteration) is a suffix that serves as the nominative dual marker in the construct state, or an indefinite accusative case marker. For example *mst$ArA* can either mean *the two consultants of* or *a consultant*.

- The definite article Al (ال). The Arabic definite article is a prefix that attaches to both nouns and adjectives. The definite article serves a syntactic function as it distinguishes between the attributive adjective and the predicate adjective. For example, *AlktAb mmtAz* means *the book is excellent* while *AlktAb AlmmtAz* means *the excellent book*. The ATB treats the definite article as an inflection, and does not split it off in tokenization.

- (ات). (At). The feminine plural marker. This is a suffix that attaches to nouns and adjectives and turns them into the feminine gender. When the original noun ends in a singular feminine marker, the singular marker is removed. For example, the plural of *sydp* (Eng. Lady) is *sydAt*. However, not all nouns ending in this suffix are feminine. This is especially the case with inanimate nouns whose plurals usually end in *At* regardless of the gender. These two letters can also be part of the stem, and this may cause segmentation errors, for example in the word *$tAt* (Eng. diaspora).

- (ان) (An). This suffix is a dual marker used with both the indicative verb and the nominative noun, for example, *ktAbAn* means *two books* and *yktbAn* means *they both write*. These two letters can also be part of the stem, and may cause segmentation errors like proper name, *EvmAn*, and the noun *>hsAn* (Eng. benevolence).

- b (ب). This clitic is a preposition that roughly translates into *with* or *by*. It can also be the beginning of a word. The orthographic form *bAsm* can either be made of a preposition + noun, *b+Asm*, (Eng. in the name of), or can be a single unit proper noun. Ambiguous words that begin with this preposition/letter are very common.

- t (ت). This can be either a prefix or suffix. As a prefix, *t* is an imperfect verb marker for either the 3rd person feminine or the second person singular. *tktb* is thus ambiguous between *she-writes* and *you-write*. As a suffix, it can be a perfect verb suffix meaning *she* or *you*, or a perfect verb subject suffix meaning *I*. The form *ktbt* can thus mean *I wrote*, *she wrote*, or *you wrote*. It can also be a singular feminine marker used with nouns when a possessive pronoun is attached as in *mhnds+t+hm* (Eng. *their female engineer*). In this case, it is an orthographic variation of the feminine marker *taa marbuta* (p in Buckwalter transliteration).

- taa marbuta (ة). The singular feminine marker (p in Buckwalter transliteration). This suffix turns a masculine noun into a feminine one: *mhnds* means *a male engineer*, *mhndsp* means *a female engineer*. Many forms ending in this suffix do not have masculine counterparts, for example, *syArp* means *a car* and *snp* means *a year*, without masculine forms for either of them. It may be a hard decision to treat these as segments since segmentation in this case is not useful as this suffix always occurs word-finally and cannot be confused with any other suffixes or stem-final characters in this position. What makes the segmentation decision even harder is that some forms with and without the singular feminine marker can be used for other purposes than the masculine / feminine distinction. For example, the final *p* is used with the initial derivational prefix *m* to denote a place: *mktbp* (Eng. library) is derived from the root *ktb* (Eng. to read) to denote a place of writing, but there is also the (masculine) form *mktb*, which means *an office* or *a department*. Performing segmentation of the *p* is usually problematic, but the ATB segments it, except in some proper nouns. This does not seem to be the correct decision since it obfuscates many useful distinctions.

- s (س). The future marker. This proclitic always precedes one of the imperfect verb inflections. *syktb* means *he will read*. It can sometimes be part of the word and cause ambiguity. The orthographic unit *syry* can either mean *he will see* or *walk*, an imperative verb, based on whether the *s* is a clitic or part of the stem, which can only be determined contextually.

- f (ف) is a proclitic that functions either as a conjunction, meaning *then*, or as a subordinate particle introducing consequences in conditional sentences. It can potentially cause segmentation errors due to its ubiquity in word-initial positions as either a clitic or as part of the stem. The orthographic unit *fDl* can either mean *virtue*, as a single segment OU, or *then-he-went-astray*, if the *f* is a proclitic. Only contextual clues can help disambiguate the form.

- k (ك) is always a clitic, but it can either be a proclitic functioning as a preposition (Eng. like, as), or an enclitic where it functions as an object pronoun or a possessive pronoun, in both cases for the second person singular. Whether the pronoun is masculine or feminine depends on the vocalization, which is usually missing. As an example of the prepositional use, the orthographic form *kmAl* can either mean *perfection*, where all the letters are part of the stem, or *like money*, where the first letter is a preposition. The orthographic unit *drsk* can mean either *he-studied-you*, or *your-lesson* depending on whether *drs* is a verb, or a noun.

- kmA (كما) serves one of two functions: (a) direct object pronoun for the second person dual, or (b) possessive pronoun for the second person dual. Which of these functions it serves depends on the grammatical category of its host. *ktbkmA* can either mean *your-books*, if *ktb* is a noun, or *he-wrote-you*, if *ktb* is a verb.

- km: (كم). This enclitic functions either as a possessive pronoun or a direct object for the second person masculine plural.

- kn (كن). This enclitic functions either as a possessive pronoun or a direct object for the second person feminine plural.

- l (ل). This is either a proclitic or a prefix. As a proclitic, it can either be (a) a preposition (Eng. to, for, in order to), (b) a confirmation particle (roughly translates to *definitely*, *certainly*). As an inflectional prefix, it can only be the definite article when it follows the preposition *l*. In Arabic orthography, *l+Al* is always written as *ll*. Although the Penn Arabic Treebank does not follow this rule, this does not pose a problem as the conversion can be performed deterministically through a simple rule. This distinction affects tokenization as the ATB style tokenization does not split off the definite article. There is also the possibility that the letter is part of the stem, in which case it must not be split off.

- n (ن). When not part of the stem, *n* is an inflectional affix that can be either (a) a plural prefix for the imperfective tense (translates into *I*), or (b) a feminine plural suffix for the imperfective tense. *ktbn* thus means *They(feminine)-have-written*.

- nA (نا) can be either inflectional or clitical. When it functions as an inflectional suffix, it attaches to the imperfective verb and marks the first person plural. As an enclitic, it can either be a possessive pronoun for the first person plural (Eng. our), or a direct object for the first person plural (Eng. us). It is often the case that the context is needed to disambiguate this segment.

- h (ه). This is an enclitic that can mean (a) *he* (in certain construction), (b) *him*, and (c) *his*. The distinction is dependent on contextual clues.

- hA (ها). This is an enclitic that can mean (a) *she* (in certain construction), (b) *her* (accusative or genitive), and (c) *her* (possessive). The distinction is dependent on contextual clues.

- hmA (هما). This is an enclitic that can mean (a) *they* (dual, in certain construction), (b) *them* (dual), and (c) *their* (dual). The distinction is dependent on contextual clues.

- hm (هم). This is an enclitic that can mean (a) *they* (plural, in certain construction), (b) *them* (plural), and (c) *their* (plural). The distinction is dependent on contextual clues.

- hn (هن). This is an enclitic that can mean (a) *they* (plural feminine, in certain constructions), (b) *them* (plural feminine), and (c) *their* (plural feminine). The distinction is dependent on contextual clues.

- w (و) When not part of the stem, the *w* functions as a proclitic of conjunction, equivalent to the English *and*. It can also be used as a preposition in such phrases as *wAllh* (Eng. by God). It can also be used as an inflectional suffix indicating nominal plural in the construct state.

- wA. (وا). An inflectional suffix attached to verbs that marks the masculine plural. When used with imperfect verbs, it indicates a non-indicative mood.

- wn. (ون). This inflectional suffix has two functions: (a) a nominal masculine plural marker, and (b) a 3rd person masculine plural marker for imperfect verbs in the indicative mood.

- y. (ي). This can be either an inflectional affix or a clitic. As an inflectional affix, it can be (a) an imperfect verb prefix for the 3rd person masculine, (b) a noun plural suffix in the accusative and genitive construct states. As a clitic, *y* is either (a) a personal pronoun meaning *I*, or (b) a possessive pronoun meaning *my*. It can also be a derivational suffix that transforms nouns into adjectives, for example, *byrwt* (Eng. Beirut) and *byrwty* (Eng. Beirutian).

2.4. Previous Work in Word Segmentation

The first approaches to Arabic word segmentation were rule-based. Darwish (2002) used a list of stems and a list of prefixes and suffixes, which he derived from an early version of the Penn Arabic Treebank, containing 560,000 words of AFP newswire text, to strip (i.e. remove) prefixes and suffixes from the stem within a word. Darwish assumed that a word would carry at most one suffix and one prefix, and to operationalize his idea, he grouped combinations of affixes together. For example, the OU *wbHsnAthm* is not treated as *w+b+Hsn+At+hm*, but as *wb+Hsn+Athm*. Darwish's rules scan word beginnings and endings and if a beginning and/or an ending matches one of the (composite) affixes in the affix list, the affix is stripped, and if the remaining part of the word is found in the stem corpus, then this is considered the correct segmentation. Darwish's method can be viewed as more of stemming than tokenization/segmentation since it does not take care of the different segments. Darwish reports and accuracy of 92.7% on a 9606 word corpus.

Lee et al (2003) use a trigram language model and treat Arabic words as a sequence of zero or more prefixes followed by a stem followed by zero or more suffixes. They use two manually annotated corpora of 10,000 and 110,000 words for training, from which they extract all the possible affixes. The two corpora sizes aim at measuring the effect of the training size. In computing the probability of possible segmentations, they consider only the segmentations that yield affixes found in the affix table. The algorithm by Lee et al has four steps: (1) Compute all possible segmentations of the word depending on the table of prefixes and suffixes. (2) Compute the trigram language model score of each segmentation. If the stem is unknown, an UNKNOWN class is used in the probability estimation. (3) Obtain the top N highest scoring segmentations, and (4) filter out illegal segmentations based on hand-written rules and the table of prefixes and suffixes. Since the corpus they use is small, many of the stems are unknown, a problem that is solved by using their small manually annotated corpus in segmenting a large, 155 million word unsegmented corpus, and re-estimating the probabilities. Lee et al report a word error rate reduction due to their use of the large unsegmented corpus of 38% for the segmenter developed from the 10,000 words, and 32% for the segmenter developed from 110,000 word corpus. The unsupervised acquisition of stems adds more stems to their annotated corpus, which relieves the data sparseness problem. Lee et al (2003) do not handle Arabic infixes, which are a completely different problem. Lee et al report an accuracy of 97% on their corpus, which seems to be an early version of the Penn Arabic Treebank.

The work most similar to the current one is by Diab et al (2004) in which they use an IOB tagging approach to word tokenization. IOB tagging treats sequences as Inside, Outside, or the Beginning of a desired unit. For the Arabic orthographic unit *Almdrs* (Eng. the teacher), for example, *A* is tagged as the beginning of the prefix, *l* as inside the prefix, and *m* as the beginning of the lexical token. In their approach, a word consists of letters and each letter is assigned a tag of B-Prefix, I-Prefix, B-word, I-word, B-Suffix, and I-Suffix. Diab et al use as features the context of +/- 5 letter features as well as the classifier's previous decisions in the feature set. Their approach is then a per-letter classification approach in which only the letters within the word itself constitute the features used by their machine leaner. Diab et al use the Penn Arabic Treebank in its state of 2004 (140,000 words). Diab et al use Support Vector Machines and report an accuracy of 99.77% and an F score of 99.12. It is not clear whether Diab et al. calculate the accuracy on orthographic units, on tokens, or on IOB tags[6]. The problem with this evaluation is that it does not give a clear picture of how the system performs on words since in a word like *wbhsnAthm*, 8 out of the 9 characters can be assigned the correct IOB tags, but the word would still be ill-tokenized, as a partially correctly tokenized word is an ill-tokenized one.

Letter	**IOB Tag**
w	B-PRE1
b	B-PRE2
H	B-WORD
s	I-WORD
n	I-WORD

[6] There is also the possibility that the F score was calculated on the number of tokens returned, although this does not seem to be very likely.

A	I-WORD
t	I-WORD
h	B-SUFF
m	I-SUFF

Table 2.1: IOB tokenization example from Diab et al (2005)

I will not follow the Diab et al tokenization design since I do not treat segmentation as a separate process, since, for the purpose of this thesis, all I need is demarcations for segment boundaries that correspond to the POS demarcations in the Penn Arabic Treebank, which makes most of the distinctions introduced in Diab et al (2004) irrelevant for the current task. In the word in Table 2.1, it is of no consequence whether a certain letter is the beginning of a word or the beginning of a prefix since this information will not be used in the POS tagging and parsing stages. Also, prefixes and suffixes are limited in number and can be grouped in a list if need be. In our approach, the word *wbHsnAthm* will be segmented as *w+b+Hsn+At+hm* without referring to their IOB tags since the token boundaries can be easily identified after segmentation. It is also worth mentioning that my style of word segmentation is different from that by Diab et al, as they do not consider inflectional affixes as separate segments, while for me, those are segments that need to be identified due to their independent POS tags.

Habash and Rambow (2005) use a morphological analyzer to produce all the possible tokenizations of a word, then use different classifiers to choose the best solutions among those provided in the morphological analysis step. The classifiers use ten features for including the parts of speech provided by the morphological analyzer, gender, person, number, voice and aspect, along with binary features: Does the word have a conjunction? Does the word have a determiner? Is there a cliticized particle? And: Is there is a pronominal clitic? Habash and Rambow use an SVM approach for classification, and report a token accuracy of 99.6% and a word accuracy of 99.3%.

The work by Diab et al and Habash and Rambow suggests that Arabic tokenization is not a hard task, and this may also be true for segmentation. There are, however, still questions that need to be answered, especially since all the work on segmentation so far has focused on the problem per se, and not on how segmentation can be used to improve the processing of Arabic. As the purpose of this thesis is to use segmentation for orthographic enhancement with grammatical analysis being the objective, the thesis will try to find the optimal way to perform segmentation suitable for this task, and examine how the quality of segmentation affects the quality of processes depending on it, e.g. part of speech tagging and parsing.

2.5. The Current Study

2.5.1. Data, Methods, and Evaluation

The data for the experiments presented here is extracted from the Penn Arabic Treebank (ATB). The ATB is a collection of newswire stories segmented by the Buckwalter morphological analyzer, which gives multiple solutions, then checked by human annotators to select the best analysis. The treebank lists all the solution and marks the correct one with a star. The data are taken specifically from the POS section in the Arabic

Treebank, rather than the syntax section since the latter does not have full segmentation information (see section 1.2.2. for a description of the Penn Arabic Treebank).

The ATB segmentation scheme, and the POS tags based on it, are sometimes confusing from a linguistics perspective. One clear example of this is the feminine singular marker (ة), p in Buckwalter transliteration. The feminine marker's most natural function is to distinguish between the feminine and masculine singular. For example, *Tbyb* means *a physician* and *Tbybp* means *a female physician*, but this is not always the case as there are many words in Arabic which are feminine in form, but with no masculine counterpart. Some of these are *HyAp* (Eng. life), *mdrsp* (Eng. school), *dymqrAtyp* (Eng. democracy), and *mmArsp* (Eng. practice). The ATB segments these and others like them as *X*+*p*, although *X* may not be meaningful. For example, *HyAp* is segmented as *HyA+p* and *mdrsp* as *mdrs+p*, even though the *p* should not be treated as a segment. More serious are cases in which the form without the *p* can mean something different. While *dymqrAtyp* means democracy, the *p*less form means *democratic*, and definitely not the masculine form of *democracy*. Another example is the form *mdynp* (Eng. city), whose *p*less form means *indebted*.

Another peculiarity is the treatment of proper nouns. Proper nouns in Arabic are regular nouns used to name people or organizations and do not have any formal distinguishing markers such as capitalization. The treatment of such proper nouns is not consistent in the ATB: Proper nouns with the definite article are sometimes segmented as Al+NOUN_PROP and sometimes as just NOUN_PROP with the POS tags DET+NOUN_PROP and NOUN_PROP respectively. For example, *Alq*Afy* (Libya's strong man) is a proper noun that is sometimes segmented as *Al+q*Afy* and sometimes *Alq*Afy*, without segmentation. Another example, involving the feminine marker rather than the definite article, is *zngAnp* which is a Persian masculine name segmented in the ATB as *zngAn+p* and tagged as NOUN_PROP+NSUFF_FEM_SG, although the *p* does not perform any feminine-marking functions.

Data

For all the experiments below I use a combination of two sections of the ATB (P1V3 and P3V1)[7] distributed in a 5-fold cross validation setting where the data is divided into 5 sections and each one of them plays the role of the testing fold while the four others act as the training section, i.e. there are 5 runs of the experiment. I use the devocalized version since this is a better representative of real-world Arabic, which is mostly unvocalized. Segmentation using vocalized data will be presented in section 3.4 within the context of the interaction between segmentation and vocalization.

In the data, the average number of segments per orthographic unit is 1.67 and the majority of orthographic unit tokens (97%) have less than 4 segments each. Orthographic units of length 1 constitute 50.67% of all orthographic units, followed by OU's of length 2 (33.40%), and length 3 (14.37 %).

Word types[8], unique words, show a different pattern as the average number of segments per orthographic unit type is 2.197. 93.61% of all word types are less than 4

7 Although other volumes of the ATB exist, I have noticed that there a significant amount of overlap between them. In my choice of the data, I had to make sure that there is no overlap between the training and the test sets. These two volumes proved to meet the criterion.

8 The distinction between types and tokens will be used throughout this thesis. A type is a unique unit,

segments. Word types of length 1 constitute 22.51 %, word types of length 2 constitute 41.81%, word types of length 3 constitute 29.26%, and word types of length 4 constitute 6.23%. This means that mono-segmental orthographic units are more frequent than bi-segmental and poly-segmental ones.

Since, in our definition, a word is an orthographic unit delimited by white space, certain units, not identified by linguists as affixes, are considered affixes in our analysis. The following is an exhaustive list:

1. Conjunctions: I treat the conjunctions *w* and *f* as prefixes since they are proclitics that form part of the orthographic unit. Other conjunctions that are not attached to the word are not treated as prefixes.
2. Prepositions: I treat the prepositions *b*, *k*, *l*, and *w* as prefixes.
3. I do not treat the nisba *y* as a suffix since it is derivational. The nisba *y* transforms a noun into an adjective, for example, *mSr* (Eng. Egypt) turns into *mSry* (Eng. Egyptian). The Arabic Treebank does not segment the nisba *y* and treats the noun or adjective formed by it as a separate stem.
4. The comparative and superlative hamza is not treated as a prefix. This is an artifact of the ATB annotation style. Superlative and comparative adjectives are simply POS-tagged as ADJ in the ATB, and are not distinguished from normal adjectives.
5. The place marker *m* is not treated in the ATB as a prefix. This is the right decision since it is derivational rather than inflectional. I follow the ATB in not treating it

while a token is an instance of that unit. For example, in the English word *fluffy* there are three tokens of the type *f*. The word has six tokens and only 4 types: *f*, *l*, *u*, and *y* (Matthews, 2007: 409).

as a prefix.

6. The ATB does not handle infixes, and I follow the ATB in this respect.

Classification

For the segmentation of Arabic Orthographic Units, I use the memory-based algorithm (see chapter 1.2.3) in a per letter classification task in which I turn each word into a number of vectors, equivalent to the number of characters per word, and assign one of two classes (+ and -) for each vector. If a letter in the word is the end of a segment, its vector receives the + sign, otherwise, it takes a - sign. I use the context of 5 characters before and 5 characters after the letter in a sliding window to represent the context of the letter. When there are not enough letters for the context, the place-holder _ is used. By way of illustration, the word *wAlmhndswn* (Eng. and the engineers) is represented by the vectors in table 2.3. Since the first letter, *w*, does not have any preceding characters, the vector starts with 5 placeholders, while the last letter, *n*, is preceded by 5 characters and followed by 5 placeholders since it is not followed by any characters.

_ _ _ _ _ **w** A l m h n +

_ _ _ _ w **A** l m h n d -

_ _ _ w A **l** m h n d s +

_ _ w A l **m** h n d s w -

_ w A l m **h** n d s w n -

w A l m h **n** d s w n _ -

A l m h n **d** s w n _ _ -

l m h n d **s** w n _ _ _ +

m h n d s **w** n _ _ _ _ -

h n d s w **n** _ _ _ _ _ -

Table 2.3: A vector representation of *wAlmhndswn.* Focus characters are in bold

The per letter classification approach is projected to be useful even if we naively assume that each word has only one possible segmentation, for two reasons: (1) It is hard to come up with all the possible words, due to the morphological complexity of Arabic in which a word like *kataba* (Eng. wrote) occurs in over 1000 forms. Using per letter classification abstracts beyond the word level and allows for partial matching, and (2) in the data, there is an average of 8.68% unknown words per fold, which means that the maximum accuracy I obtain in this data is 91.32% if I use a look-up table.

2.5.2. Segmentation Experiments and Results

2.5.2.1. Experiments

Within the unvocalized settings, I have run a number of experiments, varying the features used in terms of the size of the lexical context and its associated POS tags to find the best settings for Arabic word segmentation. The POS tags represent grammatical information that can provide information about the context, and the lexical context models collocational information. Three experiments stand out since they involve adding features that made a contribution to the quality of segmentation: (a) the Basic Experiment (b) Basic+PreviousTag, (c) Basic+PreviousTag+POS. I have also tried to expand the feature set by adding more lexical content and more part of speech tags, but each feature that was added slightly reduced the accuracy of the segmenter. In all the experiments, the best results were obtained with the IB1 algorithm with similarity computed as weighted overlap, relevance weights computed with gain ratio, and the number of *k* nearest neighbors (or in TiMBL's case, nearest distances) equal to 1.

(a) **The Basic Experiment**: The Basic experiment is the simplest of all experiments in that it requires no information beyond the orthographic unit itself. Each word is made into a number of vectors equal to the number of letters it has. Each vector is made up of the focus letter, the five preceding characters, and the five following characters. Since ambiguous words constitute only 2.77% of the data, I expect word-internal context to yield good results.

(b) **Basic+PreviousDecision**: This is a two-stage experiment in which I run the Basic experiment first, then take the classes predicted by the learner and add them to the training set along with the original gold standard classes. The purpose of this is to add more information to the feature vector, and to let the learner learn from its mistakes. To illustrate, let us assume that the word *wAlmhndswn* above was segmented by our basic segmenter introduced above as *w+Al+mhnds+w+n*, where the rightmost *w* is classified as a segment end and receives a "+" class. I can use this information to tell the classifier where it made mistakes. The new feature vector for this word is shown in Table 2.4.

```
+ _ _ _ _ _ w A l m h n +
- _ _ _ _ w A l m h n d -
+ _ _ _ w A l m h n d s +
- _ _ w A l m h n d s w -
- _ w A l m h n d s w n -
- w A l m h n d s w n _ -
- A l m h n d s w n _ _ -
```

+ l m h n d s w n _ _ _ +

+ m h n d s w n _ _ _ _ -

-h n d s w n _ _ _ _ _ -

Table 2.4: Within-word context + previous decisions

The first feature of each vector is now the class previously assigned, and in the second vector from the bottom, the class previously assigned is different from the correct class: *w* was previously assigned a segment-initial class (+) whereas it is not.

(c) Basic+PreviousDecision+POS: For some words, several segmentations are possible, based on the grammatical category of the word. This is usually the case when a word starts with a letter that can be either a prefix or part of the stem. For example, the word *wDE* can be *w+DE* (CONJ+VERB) or *wDE* (NOUN/VERB). So, knowing the part of speech of the word can help in deciding on the segmentation. For this reason, I have added POS tags to the feature vectors. As can be seen from the previous example, it is probably the focus word tag, and not any other context tags, that is useful. I used the Whole Word Tagger (see section 4.4.1.2) to assign the POS tags since I have no access to the segments at this point. The Whole Word Tagger assigns POS tags to complete words without performing any segmentation first. For example, the word *wAlmhndswn* would be input to the tagger as is with its POS tag as CONJ+DET+NOUN+MASC_PL_NOM, i.e. it consist of a conjunction followed by a definite noun (Det+Noun), followed masculine plural marker in the nominative case.

2.5.2.2. Results and Discussion

Table 2.5 presents the results of the segmentation experiments across 5 folds of cross validation.

Experiment	**Basic**	**Basic+PreviousDecision**	**Basic+PreviousDecision+POS**
Accuracy	98.15%	98.21%	98.23%

Table 2.5: Segmentation accuracy across 5 folds

As can be seen from table 2.5, the Basic experiment scores an accuracy of 98.15% on words. Adding the previous decision to the feature vectors results in a gain of 0.5%, and adding the part of speech tag results in an increase of 0.81%. While this gain is not huge in either case, it shows that identifying the grammatical category can help in such a task. The limited gain can also be due to the fact that the result of the Basic experiment is high (98.15%), and it is hard to get better results[9].

With the number of orthographic units in each fold averaging 96,649, the average number of errors per fold in the basic experiment is 1788.6 compared to 1715 in the best scoring experiment. This means that adding previous decisions and part of speech tags results in an error reduction of 4.26%.

[9] Adding the POS tags directly to the basic experiment leads to a drop of 0.4% from the Basic accuracy (tested on one fold).

This does not mean, however, that this extra information is always useful. In fact, this extra information introduces its own errors, and the improvement in accuracy can simply be characterized in the difference between the numbers of errors in each setting. In fold 1, for example, the best scoring experiment, Basic+PreviousDecision+POS has 136 orthographic units correctly segmented which are incorrectly segmented in the Basic experiment. The Basic experiment has 74 orthographic units correctly segmented that are incorrectly segmented in the best scoring experiment. Those errors are distributed among grammatical categories, and they do not seem to have distinguishing characteristics in either setting.

The most important features measured by gain ratio, as given by TiMBL, are the previous decision, followed by the letter following the focus letter, followed by the focus letter itself, and then the last letter of the vector. The POS feature turned out to be the least important of all the 13 features (it adds 0.022% on average). This means that word segmentation can be largely viewed as both lexical and local. Previous decisions have also been reported as useful by Diab et al (2004).

Part of the reason why the task is easy is that the ATB is not very ambiguous as far as word segmentation is concerned. In fact, the average number of segmentations per orthographic unit in the ATB is 1.01, which is by no means high[10]. This may be due to the nature of the ATB, which is all news stories from a limited domain, with unknown words across folds averaging 8.68%. To reach a proper, more realistic evaluation of segmentation, I have decided to use three methods: (1) Accuracy Rate on Segmentables,

[10] This is not to be confused with the average number of segments per OU, which is 1.67.

(2) Accuracy Rate on Ambiguous Words, and (3) Accuracy Rate on Known vs. Unknown Words.

Accuracy on Segmentables

Segmentables are words, or units, that can be naturally segmented while non-segmentables are units that can not be segmented. For example, punctuation marks such as (.:,) cannot have multiple segments, just like numbers like 1, 2, or 543 cannot. For this reason, a more realistic measure of accuracy needs to exclude non-segmentables from evaluation.

Segmentables%	Segmentable Accuracy	Overall Accuracy
97.96%	98.19%	98.23%

Table 2.6: Accuracy on segmentables

In the data set I use, segmentables constitute 97.96% of all orthographic units across the five folds, and the average accuracy on these is 98.19% in the best scoring experiment above. This is slightly lower than the average overall accuracy of 98.23% (see Table 2.6). While non-segmentables can theoretically be ill-segmented, for example if the classifier segments 235 as 23+5, this does not happen in the experiments presented here, and the accuracy on non-segmentables is 100%.

Accuracy on Ambiguous Orthographic Units

The data set used for the current experiments contains 595 word types (1.06% of all word types) that are ambiguous with respect to segmentation, i.e. each of these words has more than one possible segmentations in the training set. Only five of these words have 3 segmentations each while all the others have only two. An example is the word *ysyr,* which can be either a one-segment adjective (Eng. easy) or a verb inflected for the singular 3rd person masculine in the present tense *y+syr* (Eng. he walks). This situation is possible when a word starts with one of the common prefixes, like *y*, *w*, and *n*, and can only be disambiguated through the context.

In this section, I look at the accuracy obtained on those words to see how well the segmenter handles ambiguity. This is very important since different genres may have different segmentations for the same word.

Ambiguous OU's %	Accuracy	Overall Accuracy
2.77%	86.79%	98.23%

Table 2.7: Accuracy on ambiguous orthographic units.

The accuracy on ambiguous words, as we can see in Table 2.7, is significantly lower than overall accuracy. Ambiguous words constitute 2.77% of all words across the five folds and the accuracy on these averages 86.79%, which is considerably lower than the general accuracy of 98.23%. Analyzing Fold 1, I find that the classifier favored the more frequent variation in 90.08% of the ambiguous word types, which means that frequency plays a major role in determining the solution, and even in the cases where the solution selected by the classifier is less frequent, it is because part of this solution is more frequent. For example, the form *tErD* is more frequent than *t+ErD*, but *+Erd* is much more common, and this may be the reason this solution is chosen, since the classifier works on letters, not whole words.

Handling Out-of-vocabulary Orthographic Units

The real value of a system is in its ability to generalize beyond its training data, and for me to test this, I need to test the performance of the segmenter on previously unknown data.

The average number of unknown words across the five folds is 8.68%, and the average accuracy on these is 82.22%, which is significantly lower the accuracy on known words, which averages 99.75%. Examining the nature of unknown words reveals that the majority of these are of the noun class. Nouns constitute 44.28% of all unknown words followed by Proper Nouns, which constitute 21.32% and adjectives (11.55%). These nominal categories are the ones with inconsistencies in the segmentation annotations, especially the proper nouns. These annotation inconsistencies in the training data may have prevented the classifier from proper generalization.

Known	Unknown
99.75%	82.22%

Table 2.8: Known vs. unknown words

As can be seen from table 2.8, there is a discernible difference between the accuracy on known words and the accuracy on unknown words (99.75% vs. 82.22%). In real-world situations, the percentage of unknown words can be higher if there is a difference between the training genre and the test genre.

2.5.3. Error Analysis

In an examination of one fold for error analysis, I have found that words with specific POS tags are more likely to be ill-segmented than others. Table 2.9 presents percentages of the 10 most frequent tags of segmentation errors.

#	**Tag**	**Percentage**
1	NOUN_PROP	41.12 %
2	CONJ+NOUN_PROP	3.66 %
3	DET+NOUN	2.80 %
4	NOUN	2.67%
5	IV3FS+IV	2.42%
6	PREP+NOUN	2.17 %
7	DET+NOUN_PROP	1.74 %
8	PV	1.55%
9	NOUN+CASE_INDEF_ACC	1.55 %
10	NOUN+NSUFF_FEM_SG	1.37%

Table 2.9: POS tags of most common errors

In light of the discussion of the segmentation scheme of in the ATB throughout this chapter, it is not surprising that Proper Nouns are responsible for the largest share of errors in segmentation due to inconsistency in annotation and the absence of any formal markers that distinguish Proper Nouns.

Also, some non-stem segments are more likely to cause segmentation errors than others. The following is a list of the 10 most frequent affix/clitic-induced errors that occur at least 100 times each in the 5 folds:

(1) The feminine singular marker *p* is segmented where it should not. This is usually the case in proper nouns like *AlHyAp* where the annotators chose not to split the marker off.

(2) *A* is segmented where it should not. Word-final *A* is often confused with the accusative marker *A*.

(3) *y* is treated as part of the stem while it should be segmented. This is due to the ambiguity between a derivational suffix, and inflectional affix and an enclitic.

(4) The segment *A* is treated as part of the stem. This is the reverse of number 2.

(5) *y* is treated as an affix while it is part of the stem. This is the reverse of number 3.

(6) The feminine singular marker is not segmented where it should. This is the reverse of number 1.

(7) *h* is treated as a segment where it is part of the stem.

(8) *h* is treated as part of the stem where it would be segmented. This is the reverse of number 7.

(9) *yn* is not treated as a segment where it should.

(10) *t* is treated as a segment where it is part of the stem.

2.5.4. Tokenization

So far, I have performed word segmentation, which does not differentiate between flective affixes and clitics in that they are all treated equally. In parsing Arabic, and in some forms of Arabic POS Tagging, I need to identify and separate those elements that have a syntactic function. For example, in the following sentence, 9 of the 26 orthographic units have to be tokenized for syntactic analysis. These include conjunctions, future markers, prepositions, and possessive pronouns (underlined in the example):

وقالت المصادر إن النيابة ستدرس القضية خلال الأسبوع الجارى للتحقق من وجود مخالفات من عدمه، وأنه سيتم استدعاء الوزير ورؤساء المدن كمتهمين، حال ثبوت المخالفات فى حقهم

wqAlt AlmSAdr <n AlnyAbp **stdrs** AlqDyp xlAl Al>sbwE AljArY **lltHqq** mn wjwd mxAlfAt mn **Edmh** ،**w>nh** **sytm** AstdEA' Alwzyr **wr&sA'** Almdn **kmthmyn** ،HAl vbwt AlmxAlfAt fY **Hqhm**.

Rather than treat tokenization as a separate process, I treat it as a post-processing step after segmentation. The process involves removing inflectional segment boundaries rather than inserting token boundaries. To illustrate, segmentation yields the following output:

w+qAl+t Al+mSAdr <n Al+nyAb+p s+t+drs Al+qDy+p xlAl Al+>sbwE Al+jArY l+l+tHqq mn wjwd mxAlf+At mn Edm+h ،w+>n+h s+y+tm AstdEA' Al+wzyr w+r&sA' Al+mdn k+mthm+yn ،HAl vbwt Al+mxAlf+At fY Hq+hm.

By processing the segmented output through rules, I obtain the following tokenized text:

w+qAlt AlmSAdr <n AlnyAbp s+tdrs AlqDyp xlAl Al>sbwE AljArY l+ltHqq mn wjwd mxAlfAt mn Edm+h ،w+>n+h s+ytm AstdEA' Alwzyr w+r&sA' Almdn k+mthmyn, HAl vbwt AlmxAlfAt fY Hq+hm.

What makes tokenization amenable to post-processing is that clitics can hardly be confused with affixes. Proclitics are members of the list [b, s, f, k, l, w] and enclitics are members of the list [h, hA, hmA, hm, hn, ny, nA, k, kmA, km, kn]. The only exception is the prefix *l*, the definite article following the preposition *l*. This can be easily accounted for since it is regular.

There is, however, one suffix that can serve as both an enclitic and an inflectional suffix with exactly the same distribution: *y*. This suffix can be a derivational suffix, in which case nothing needs to be done about it, but it can also be a possessive pronoun (first person singular possessive pronoun), in which case it is a clitic. It can also be a plural marker, or a dual marker with a different pronunciation that is not reflected in the orthography, in which case it is inflectional rather than clitical. The orthographic unit *ktAby* can thus mean either *my book*, or *the two books of*, or in a less limited genre than the newswire: *my two books*. This affects tokenization since if the *y* was inflectional, it should remain as part of the base form, but if it were a pronoun, it should be split off for tokenization. If it were derivational, however, it would not be marked as a segment at all. This information can only be derived contextually as the orthographic form itself, in the absence of vocalization, does not provide enough information.

In order to solve this problem, I used the part of speech as assigned by the segmentation- based tagger (see section 4.4). The tokenizer is thus a function that takes as input segmented words and their respective parts of speech tags and returns a tokenized form of the orthographic unit. Given the orthographic unit *ktAby* and the tag NOUN+POSSESSIVE, the tokenizer returns *ktAb+y*, but given the same orthographic unit with the part of speech tag ADJECTIVE or NOUN+DUAL, the tokenizer returns *ktAby*, i.e. it returns the orthographic unit without any change.

2.5.4.1. Tokenization Evaluation

Averaged over five folds, tokenization scores an accuracy of 99.36% in the experiment that uses the best scoring features, i.e. the word context plus the previous decision plus whole word part of speech tagging. This shows that tokenization, and the general case of segmentation, is a stable process. Tokenization accuracy is more than 1 percentage point higher than segmentation accuracy. This is not surprising since in tokenization we remove the inflectional distinctions, some of which may lead to errors in segmentation.

While correct tokenization requires the availability of POS tags for the enclitic *y*, the assignment of the correct part of speech to, let alone the segmentation of, *y* is far from trivial, as *y* is one of the most ambiguous segments, in terms of grammatical category, with 17 tags attested in the Penn Arabic Treebank. The clitic *y* is correctly segmented in only 55.67% of all cases across the five folds, and its POS tagging accuracy is merely 49.89%. This poor performance does, however, not affect segmentation or tokenization accuracy due to the fact that the frequency of this segment is very low, at 0.058% of all segments in the data set used in this study.

The reason for the scarcity of *y* as a first person pronoun can be attributed to the nature of the Penn Arabic Treebank, which is all newswire of limited genre variation. In such a corpus, one does not expect to find many first person references, but this can be a problem in such genres as political speeches, in which first person references abound, and can thus negatively affect the performance of a segmenter trained on the Penn Arabic Treebank.

The tokenization results reported here are slightly better than those reported by Habash and Rambow (2005) (99.36% vs. 99.3%), although a direct comparison is not possible since the data set is not the same. However, it shows that tokenization through post-processing of segmentation is a viable process.

2.5.5. Instance sampling for segmentation

In performing segmentation, the machine learner has to decide for each letter whether it is a segment boundary or not. This means that most letters will receive the "-" class indicating that the letter is not a segment boundary and thus constitutes a negative example. This results in a skewed distribution between positive and negative examples. In such a situation, random sampling can be used to even out the differences. By random sampling I mean that I randomly choose some of the negative example, according to a predefined ratio, while the number of positive examples remains constant.

Random sampling has been shown to work when the data is skewed. Wunsch et al (2009) show that random sampling improves the quality of co-reference resolution on German data from an F-score of 0.541 to 0.608 for memory-based learning, from 0.561 to 0.611 for decision tree learning, and from 0.511 to 0.584 for maximum entropy learning. The co-reference resolution task is characterized by a majority of negative examples as most words in the contexts are not antecedents of the pronoun in question. This bears some resemblance to the segmentation task in which most of the examples are also negative.

In the segmentation data, the ratio of positive examples to negative examples is 1 to 5.3. I ran experiments on ratios ranging from 1 to 1 to 5.3. Table 2.7 shows the results on the experiments. The results are reported on only one fold.

Ratio	**Accuracy**
1 to 1	99.339
1 to 2	99.457
1 to 3	99.524
1 to 4	99.557
1 to 5	99. 568

Table 2.10: Instance sampling ratios and accuracies

The results presented in Table 2.10 show that an even ratio of positive and negative examples may not be enough for obtaining the desired accuracy. The more negative examples that I add to the training set, the better the accuracy becomes. One possible explanation is that the ratio of positive examples to a negative example is not very small, and is within the safe limits. This can be contrasted to the task of co-reference resolution in which the ratio can be as extreme as 1: 48 (Ng and Cardie, 2002).

2.5.6. Training Size Effect on Segmentation

To test the effect of the training size on segmentation accuracy, I trained the segmenter on various sizes of data, starting from 10,000 words and up to the full size of the training set in fold one (386,596 orthographic units) and test on the test set of fold 1 (96647 OU's). The 10,000 words training set represents a relatively small starting point since our full training set is more than 38 times as large. After that, I start with 1/10 of the data size, and keep adding an extra 1/10 every time.

Table 2.11 and figure 2.2 display the results obtained. With the training set of 10,000 words, I reach an accuracy of 98.08%. This goes up to 98.20% when I use 1/10 of the training size. The accuracy then keeps fluctuating until it reaches 98.34% when I use the full training set, which is the exact same accuracy I obtain by using two tenths of the full training size.

Training Size	**Word Accuracy**
10000 OU's	98.08

1/10	98.20
2/10	98.34
3/10	98.31
4/10	98.23
5/10	98.22
6/10	98.21
7/10	98.25
8/10	98.29
9/10	98.30
Full (386,596 OU's)	98.34

Table 2.11: Training size effect on segmentation

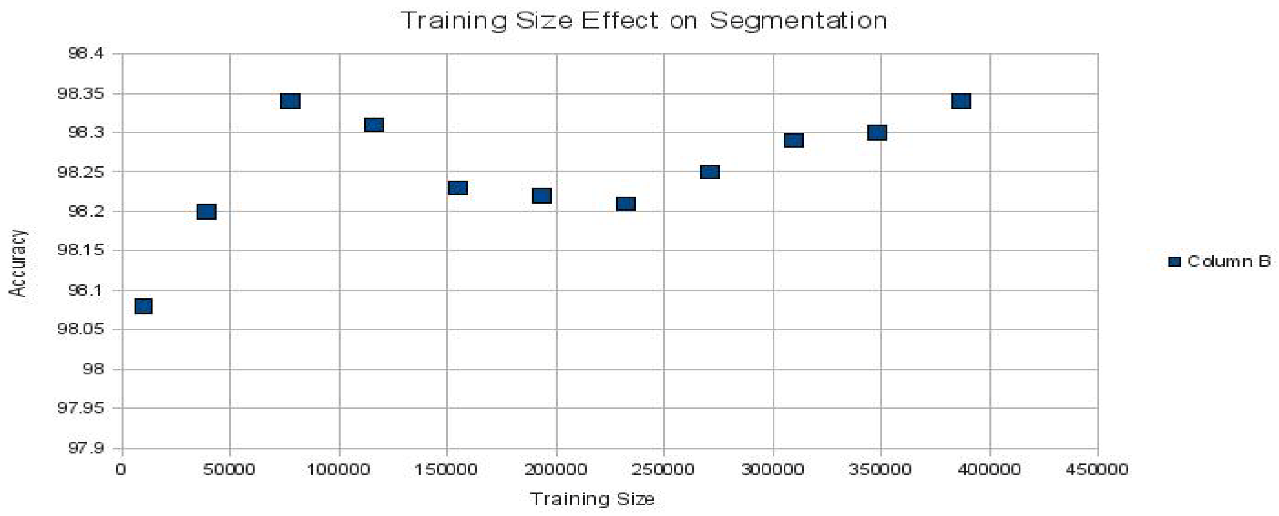

Figure 2.2: Data size effect on segmentation accuracy

The size effect experiments indicate that segmentation requires only a small data size, and that increasing the data in the training set does not necessarily improve accuracy. The numbers plateau at around 2 tenths of the data, and adding more data may not help improve the accuracy of segmentation.

From now on, I will be using the segmentation obtained by training on the full training set in each fold for the further processes of part of speech tagging and parsing.

2.6. Conclusion

I have experimented with Arabic orthographic unit segmentation using TiMBL, a memory-based learner, and found that the best results can be obtained by incorporating previous decisions and part of speech tags from a Whole Word Tagger. I have also shown that while the task is seemingly easy (as the accuracy reaches 98.23%), accuracy on unknown orthographic units is much lower than that on known orthographic units.

Ambiguous words also score lower than unambiguous words in terms of accuracy. The distinction between known and unknown, and ambiguous and unambiguous is indispensable for real world situations, especially when the segmenter is used with text of a different genre than that of the training set.

While tokenization is different from segmentation, in that it considers only syntactically functional segments, I have shown that tokenization can be derived from segmentation through simple rules with accuracies surpassing those reported in the literature. Some linguistic processes, such as syntactic parsing, require tokenization, which involves some morphological processing, and it is very practical to have a single segmenter that can perform both segmentation and tokenization accurately.

I have also found that increasing the training size does not help improve segmentation results, and that a training set as small as 10,000 words is enough to obtain similar accuracy to that obtained by a segmenter trained on 400,000 orthographic units. Instance sampling also proved unhelpful for segmentation.

In the error analysis, I have found that the most obvious segmentation problem lies with Proper Nouns since many of these are foreign names, and since the segmentation scheme adopted for Proper Nouns is not consistent throughout the Penn Arabic Treebank. This problem can only be solved through a careful cleanup of the data, which is an arduous task.

In the chapters on grammatical analysis, part of speech tagging and dependency parsing, I will use the segmenter developed here to prepare the data as required: segmentation for POS tagging, and tokenization for parsing.

Chapter 3: Vocalization

3.1. Introduction

The Arabic orthography comprises 28 letters of which three are vowels and 25 are consonants. Arabic is generally known to have at least 6 vowel sounds. The orthography has characters only for the 3 long vowels while the 3 short ones are usually neglected, although in some contexts, some or all of the vowels may be included as diacritics. The lack of short vowels, and the agglutinative nature of the orthography render it poor in that it is highly ambiguous on the one hand as a single spelling usually denotes multiple words that differ in their grammatical categories and pronunciations, and very complex on the other hand since it can be difficult to distinguish between what is part of the word, and what is not (in the case of prefix conjunctions for example). Chapter 2 handled the complexity of the white-space delimited unit. The purpose of this chapter is to examine how this poor orthography can be enriched with the usually missing vowels, with the ultimate goal of using this enrichment for aiding grammatical analysis in later chapters of this thesis.

The rest of the chapter is divided into two sections: section 3.2 is a linguistic introduction to vocalization outlining the functions of the short vowels both syntactically and morphologically as well as the effect of short vowels on reading comprehension. Section 3.3 treats the problem of vocalization, the insertion of the missing vowels, from a computational perspective: it reviews the related studies about short vowel restoration then introduces my own approach. Section 3.4 examines the interaction between segmentation and vocalization with the purpose of finding whether performing any of these two processes can help improve the other. Section 3.5 is a conclusion that summarizes the findings of the chapter.

3.2. Vocalization: a linguistic introduction

An unvocalized word is potentially ambiguous. As an example, Table 3.1 presents the word *mSr* (مصر) which consists of three consonants, but can nonetheless take on at least five different pronunciations in Modern Standard Arabic. Each pronunciation has its own grammatical category and meaning.

Pronunciation	**Grammatical Category**	**Meaning(s)**
miSor	NOUN, PROPER_NOUN	Country, Egypt
maS~ra	VERB	Egyptianize
muS~ira	PASSIVE VERB	to be Egyptianized
muSr	NOUN	intestine
muSir~	ADJECTIVE	persistent

Table 3.1: The different pronunciations and meanings of the word *mSr*

As can be seen from Table 3.1, the lack of short vowels results in ambiguity in both meaning and grammatical category, and makes it difficult to determine which one of the five words above the orthographic form *mSr* represents without utilizing contextual clues. In order to overcome this problem of lack of short vowels, the orthography needs to be enriched with the usually missing diacritics that represent the short vowels.

It should be noted, however, that the morphological complexity of the orthographic unit, even in the absence of vocalization, can help disambiguate the unit and provide enough context for vocalization. While the orthographic unit *mSr* is ambiguous, the unit *bmSr*, which consists of the preposition *b* and *mSr* is hardly so since prepositions almost exclusively precede nouns, which makes the verb and the adjective reading very unlikely.[11]

3.2.1. The Functions of Vocalization

Vocalization serves linguistic functions necessary for understanding the Arabic language. The major three functions of vocalization are: (1) lexical derivation, (2) lexical disambiguation, and (3) case assignment. The vocalization task thus helps in morphological and syntactic analysis, but the assignment of vowels to consonants can also be merely orthographic, i.e. a specific vowel can be arbitrarily assigned to a specific consonant within the orthographic unit.

3.2.1.1. Lexical derivation

Much of Arabic word formation is carried out through templatic morphology. Templatic morphology treats words as consonantal skeletons, usually made up of three or four consonants with vowels added in between those consonants to form new words. To illustrate this, the consonantal skeleton *k t b*, also known as the root of the word, represents the concept of writing, but whether this writing is a verb or a noun is determined by which vowels are inserted between the consonants. The following table presents some derivations, their templates, and grammatical categories. An upper case V denotes a long vowel while a lower case v indicates that a short vowel is used, C stands for Consonant, and CC means that the consonant is geminated.

[11] Adjectives often, but not always, precede nouns in Arabic. In those cases in which adjectives precede nouns, it is possible for a preposition to attach to the adjective rather than the noun. An example of this is the phrase *bHlw AlklAm* (Eng. with-sweet words).

vocalized form	**Morphological template**	**Grammatical category and meaning**
Kataba	CvCvC	He wrote
kAtaba	CVCvC	To correspond with
Kutiba	CvCvC	It was written
Kattaba	CvCCVv	He made him write
Kuttiba	CvCCvC	He was made to write
kitAb	CvCVC	A book
kutub	CvCvC	Books
kuttAb	CvCCVC	Writers, school

Table 3.2: Templatic morphology: Arabic words are formed through vowels inserted between consonants

As we can see from Table 3.2, derivation is carried out mainly through vowels, whether short or long. Long vowels are usually written in the orthography, but short vowels are rarely written even when short vowel absence leads to ambiguity. The common belief is that the short vowels are restored through context.

3.2.1.2. Lexical, morphological and syntactic disambiguation

While morphological derivation can create new words, these created words sometimes share the same unvocalized form. It is common even for words that do not share the same derivation to have the same written form. For example, the unvocalized form *fDl* can have several morphological analyses, and unlike the word *mSr* discussed above, some of these analyses span token boundaries, and only vocalization can help disambiguate them:

a) *faDula*: intransitive verb (Eng. to be better)

b) faDol: NOUN (Eng. virtue)

c) *faDal~a*: two tokens, the conjunction *f* and the intransitive verb *Dal~a* (Eng. and then he went astray)

The analysis in c differs substantially not only lexically, but also morphologically and syntactically as it represents two tokens are glued together due to the nature of the orthography. This situation is very likely to arise when the orthographic unit starts with a letter that can also be a proclitic like *b*, *w*, *f*, and *k*.

3.2.1.3. Case and mood assignment

In most instances, case assignment is displayed with short vowels. The short vowel on the last letter of the noun indicates whether the noun is in the nominative, accusative, or genitive case. The noun *kitAb* can thus be *kitAbu*, in the nominative case, *kitAba* in the accusative case, and *kitAbi* in the genitive case. The imperfective verbal mood is also shown through the short vowels.

There are, however, instances in which case assignment is done through suffixes. Case assignment through suffixes is more likely to be found in naturally occurring Arabic, and is part of the non-vocalized orthography. For example, in plural and dual nouns and adjectives, case is shown though inflectional suffixes. In the plural, a noun ending in *–wn* is nominative, while a noun ending in *–yn* is either genitive or accusative. Thus, *mhndswn* (Eng. engineers) and *mhndsyn* are the same noun in different cases. In the dual, *-An* is the nominative marker, and *–yn* is the accusative and genitive marker. So, *mhndsAn* (two engineers) is a nominative dual noun, while *mhndsyn* is ambiguous: either a dual non-nominative, or a plural non-nominative. The pronunciation, i.e. vocalization, is different in both cases: when the word *mhndsyn* is dual, it is vocalized as *muhanodisayoni* while the vocalization is *muhanodisyna* in the plural.

3.2.2. The Arabic Diacritics

Arabic short vowels, as well as the gemination marker, are written in the Arabic orthography as diacritics placed above or below the consonant. Table 3.3 shows the diacritics, with their shapes and names as they appear on the Arabic letter s (س). Each one of these diacritics has its own functions:

سَ	سُ	سِ	سـ’	ساً
Fatha	Damma	Kasra	Sukun	Fathataan
سٌ	سٍ	سَ	سُّ	سِّ
Dammatan	Kasrataan	Shadda with fataha	Shadda with damma	Shadda with kasra

Table 3.3: The Arabic diacritics as they appear on the letter (س) (Buck. s)

3.2.2.1. Fatha. The fatha takes the shape of a horizontal bar above the affected consonant and is pronounced as the letter *u* in the English word *cup.* The fatha has syntactic, derivational and inflectional functions as well as arbitrary lexical functions. For example:

- Perfective and imperfective verbs whose first consonant is fatha-vocalized are active verbs. The verbs *kataba* (Eng. he wrote) and *naktubu* (Eng. we write) are thus active verbs.
- The past and present participles are distinguished, in many cases, by the vowel on the pre-final consonant. A fatha on the pre-final consonant indicates a past participle. For example, *muqaw**i**m* means *resisting*, but *muqaw**a**m* means *resisted*.
- The fatha is the accusative marker on singular and broken plural nouns. It is also the genitive marker on diptote nouns (nouns that have only two cases).

3.2.2.2. Damma. Some of the functions of the damma, besides its arbitrary distribution, are the following:

- An imperfect or perfect verb beginning with a damma-vocalized consonant is in the passive voice. *kutiba* and *yuktabu* are perfect and imperfect passive verbs respectively.
- An imperfect verb ending in damma is in the indicative mood.
- A noun ending in the damma is in the nominative mood.

3.2.2.3. Kasra: The kasra is a horizontal bar placed under the consonant. Apart from being arbitrarily assigned to a lexical item and not being associated with any function, the kasra has derivational and inflectional functions. For example:

- It distinguishes between the present participle, and the past participle. The present participle has a kasra on the pre-final consonant.
- A kasra on the final consonant marks the genitive case. Verbs do not receive the

kasra in any of their moods.

3.2.2.4. Shadda. The shadda is the consonant gemination marker. Besides being arbitrarily assigned, the shadda performs derivational functions. It is a derivational infix serving functions such as:

- Transitivization. It turns an intransitive verb into a transitive one, and the transitive into a ditransitive. For example, *kataba* (Eng to write) becomes *kat~aba* (Eng. to make someone write). These forms are usually semantically associated with causativization.
- Emphasis and Repetition. For example, *qaTaEa* (Eng to cut) and *qaT~aEa* (Eng. to cut repeatedly or into small pieces).

The shadda does not occur on its own as a diacritic, but has to occur with another vowel. This leads to the existence of three distinct combinations: shadda with fatha, shadda with damma, and shadda with kasra.

3.2.2.5. Sukun. The sukun is a circle placed above a consonant to indicate that the consonant does not have a vowel associated with it, i.e. it is not vocalized like the *p* in the English word *apt*. In the spoken standard language, all short vowel case markers are naturally pronounced as sukun when the word is the last unit in an utterance.

3.2.2.6. Fatahataan: (2 fathas): Case marker: Accusative marker for indefinite singular and broken plural nouns and adjectives.

3.2.2.7. Dammataan (2 dammas): Case marker: Nominative marker for indefinite singular and broken plural nouns and adjectives.

3.2.2.8. Kasrataan: (2 kasras) Case marker: Genitive marker for indefinite singular and broken plural nouns and adjectives.

While the lack of short vowels and the gemination marker may be an obvious shortcoming in the orthography, the Arabic writing system is regular otherwise. The combination of the *abjad,* the consonantal orthographic system, and the diacritics makes Arabic writing a perfect system for representing the sounds of the language. According to Bellamy (1989),

> When fully pointed and vocalized, the Arabic alphabet provides a nearly perfect phonemic transcription of the sounds of the classic Arabic language. It has thus avoided the problems of irregular spellings that afflict languages such as English and French, in which the development of the script has not kept pace with that of the language.

If we look closely at the Arabic abjad, we can see that only the root units are written, even if root units were long vowels, but "the inflectional morphology is written only to the degree that it contains consonants" (Rogers, 2005: 140). Many researchers in Semitic languages believe, according to Rogers (2005: 114-5) that the abjad is suitable for the Arabic morphology since Arabic combines both inflectional and root-based morphology, and the deletion of vowels makes it very easy to discern the content of the lexical item and find relations between the different derivations. For example, when we consider the words: *ktb*, *yktb*, *mktb*, *kAtb*, and *ktAb*, we can immediately find the lexical relationship between them once we remove what is called in Arabic *the extra letters*.

Sometimes the argument of the suitability of the abjad system is stated as though inflectional morphemes are of no consequence. Most people, however, feel that there is some significance in the difference between *I will die* and *I have died*. Rather than saying that the inflectional information is not so important, perhaps it would be closer to the truth to say that it is likely to be more redundant or more easily recovered from the context. In this sense the Semitic abjad emphasizes the lexical and less redundant parts of words (ibid).

One reason for the usability of the abjad system, that I have not seen mentioned in research, is that the Arabic language as we have it today is a collection of tribal dialects, and those dialects have different pronunciations of words and the differences are usually ones of short vowels. When we look at an Arabic dictionary like *Lisan ul-Arab* (Ibn Manzur; 1968), we can find many words whose different pronunciations depended on what short vowels the consonantal skeleton has. This continues on today, with Arabic dialects differing in short vowel allocation. The following are representative examples:

- The word سلحفاة (Eng. turtle) has three pronunciations differing only in short vowels: *sulaHofAp*, *suloHafAp* and *siloHafAp*
- The word رسغ (Eng. hand joint) has three pronunciations differing only in short vowel: *rusog*, *risog*, and *rusug*.
- The word الأربعاء (Eng. Wednesday) has three pronunciations differing only in short vowels: *Alo>arob**i**EA'*, *Alo>arob**a**EA'*, and *Alo>arob**u**EA'*.

Adopting an abjad writing system may have been a way to overcome this problem, especially that Arabic has only three short vowels.

3.2.3. How Vocalized is Naturally Occurring Arabic?

While it is true that naturally occurring Arabic is, generally speaking, unvocalized, it is an oversimplification to assume that vocalization is non-existent or non-significant. In fact, the importance of vocalization varies by such factors as the text genre, and the writer's feeling of how ambiguous a word is.

In a 12,396,661 word corpus from the AlHyAp newspaper (The Arabic GigaWord Corpus), there were 60,180 words with some sort of vocalization (0.49%), with the shadda alone accounting for 61.42% of those diacritics - followed by the damma with 18.94%. The diacritics seem to have been used because the writers felt that they needed to disambiguate the chosen words. The most notable of the disambiguation tasks is the difference between passive and active verbs, in which the damma plays the major part. For example, *kataba* means *to write*, while *kutiba* means *to be written*. In a non-exhaustive search of the damma use in that section of the AlhyAp newspaper corpus, it turns out that this may be the sole purpose of its use.

While only a little below 0.5% of the words in the *AlHyAp* newswire corpus have some form of vocalization, other text genres exhibit varying ratios. In the religious genre, for example, one would expect a higher ratio of vocalized or partially vocalized words. The Qur'an is fully vocalized, and so is the Bible, but this is the not the case with other books in the genre. A study of the factors behind (partial) vocalization in naturally occurring Arabic text is interesting, but is beyond the purpose of this study.

3.2.4. Vocalization and Reading Comprehension

Abu-Rabia and Abu-Rabia and Siegel (Abu-Rabi and Siegel 1995, Abu-Rabia 1997, Abu-Rabia 1998, Abu-Rabia 1999, and Abu-Rabia 2001) conducted a series of studies on the effect of vocalization on reading comprehension in both Arabic and Hebrew. In all these studies, it was found that there is a significant difference in reader comprehension between vocalized and unvocalized text (with favor given to vocalized text), regardless of the level of the subjects (skilled or unskilled readers), the reading material (newspapers, literary texts, and religious texts), or the age of the subjects (children and adults). Abu-Rabia (2001) also found that while the sentence context had a positive effect on reading comprehension in unvocalized text, no such effect was found in vocalized texts.

3.3. Computational Approaches to Vocalization

3.3.1. Task Definition

The vocalization task described in this dissertation involves the restoration of word-internal vowels, and does not involve the restoration of case and mood endings, even when those are expressed in terms of short vowels. Given a word like *Almhnds* (Eng. the engineer), the task will seek to obtain as an output *Alomuhanodis*, while the last letter of the word, which is the case carrier, will remain unspecified.

The reason for this is that case restoration is a complex task by and of itself, and it requires syntactic analysis, and even with the use of gold standard tokenization, part of speech tags, and syntactic trees, Habash et al (2007) reported an accuracy of 95.8%. This renders case assignment useless for the purpose of real-world analysis of Arabic for our purposes here. Case assignment using automatic data could be a point of further research.

Another reason is that case assignment is itself not easily accessible for native speakers of Arabic (Holes, 2004). Ibrahim Anis (1958)[12], a distinguished Arab linguist, even proposed that case and mood endings are not necessary as he stated:

> It seems that providing the word ends with vowels was one of distinguishing marks of junction in the speech, both in poetry and prose. Whenever a speaker makes a pause or concludes his sentence, he has no need of those vowels, he stops at the last word of his utterance with what is known as sukun. It may be inferred from this premise that the basic rule for all inflective words is to end in this sukun, and the speaker has to resort to the vowelling of words but in (the case of) a phonetic necessity called forth by the (word) junction.

While this task definition excludes case assigned through short vowels at word ends, there are situations in which case is assigned word-internally. When an orthographic unit ends with a possessive pronoun token, the short vowel indicating case is placed on the last letter of the noun token, and not on the last letter of the orthographic unit. For example, when the preposition *b* is attached to a noun, it assigns the noun a genitive case and the short vowel *i* is attached to the end of the noun. The word *say~aratuhum* (in the nominative case) becomes *say~aratihim* (in the genitive case) in the orthographic unit *bisay~aratuhum* (Eng. in their car). It can also be noticed that there is vowel harmony by which the possessive pronoun changes pronunciation from *hum* to *him,* and this pronunciation has to be reflected

[12] In Drozidik (2001).

in the writing system through the assignment of the diacritics. In this chapter, vocalization targets whole words, and I will treat word-internal case errors as regular errors, and not as case errors, but the discussion will shed some light on their frequency and effect on vocalization.

3.3.2. Related Work

The first approaches to the vocalization of Arabic define the problem as word-based, i.e. the task is to determine for each word the complete diacritized form. Gal (2002) used a bigram Hidden Markov Model for diacritizing both Arabic and Hebrew where the hidden states were vowel-annotated words and the observations were vowel-less words. Gal used the Bible and the Qur'an as data sources. Gal achieves a word error rate (WER) of 14% on Arabic and a WER of 19% on Hebrew. Gal attributes the better results of Arabic, in spite of the fact that the Hebrew data was three times larger than the Arabic data, which contained 90,000 words, to the fact that in Arabic there are only three missing diacritics while the Hebrew task involves the restoration of twelve vowels. Gal used 90% of the data for training and 10% for testing in both Arabic and Hebrew. Gal's error analysis shows that the errors result mostly from unknown words.

Kirchhoff et al. (2002) design a vocalization module for use in a speech recognition system. They the LDC CallHome corpus of Egyptian colloquial Arabic, which is a collection of telephone conversations between family and friends. Their system uses a unigram model extended by a heuristic for unknown words, which retrieves the most similar unlexicalized word and then applies edit distance operations to turn it into the unknown word. They reach a WER (for vocalization) of 16.5% on conversational

Arabic.

Nelken and Shieber (2005) use the Penn Arabic Treebank (Part 2, 144199 words divided into 90% for training and 10% for testing) and tackle the problem with weighted finite state transducers. The system uses a cascade of transducers for the following: (a) a trigram language model of Arabic diacritized words from which to learn the most probable word sequence that could have generated the unvocalized text, (b) a spelling transducer that transduces the word to its constituent letters, (c) a diacritic drop transducer that replaces all occurrences of the short vowels with empty strings (the transducer also handles the various forms of the glottal stop), and (d) a clitic transducer that uses data from the Buckwalter Arabic Morphological analyzer to learn the most probable vocalization of affix combinations. For known words, morphological units are used for retrieving the vocalization while unknown words are diacritized based on the sequence of characters using a 4-gram character model. Nelken and Shieber reach a WER of 23.61% when case restoration is considered. The system achieves a WER of 7.33% on word-internal vocalization without considering case.

Zitouni et al. (2006) treat the problem of vocalization as one of sequence classification, and use per letter classification instead of words. For each letter, they decide which vowel, if any, should be assigned to that specific letter. Zitouni et al use Maximum Entropy modeling for classification. The features used by Zitouni et al include lexical features, segmentation features, and POS features, as well as the previously assigned two diacritics. The lexical features include a window of seven characters (current character, three preceding and three following characters), the current word in a window of 5 words, as well as the position of the current character (beginning or end of the word). The segment information uses only automatic segmentation with the features being the current segment and its word segment context in a window of five segments. The POS features include segment tags generated by a Maximum Entropy model that achieves an accuracy of 96%. Zitouni et al use the Penn Arabic Treebank P3V1 (340,281 word) with 85% for training and 15% as a development/test set. Zitouni et al reach a WER of 17.3% on restoration including case endings and 7.2% when case endings are omitted.

Habash and Rambow (2007) perform a full vocalization including case endings and nunation. They use the Buckwalter analyzer (Buckwalter, 2002) to obtain all possible morphological analyses, including all diacritics (the same system they used in segmentation). Then they train individual classifiers to disambiguate between these analyses. Residual ambiguity is resolved via an n-gram language model. Habash and Rambow reach a WER of 5.5% using the same data and division into training and development/test set as Zitouni et al. (2006). When they include case endings, the WER reaches 14.9%.

Shaalan et al. (2009) compare a lexicon-based approach with an approach using word bigram statistics and a Support Vector Machine (SVM) classifier. The SVM approach uses features from automatic segmentation, POS tagging, and chunk parsing. Shaalan et al. show that the best results for vocalization are reached by combining all three approaches. Shaalan et al reach a WER of 12.16% on the task including case restoration.

3.3.3. Data, Methods, and Evaluation

3.3.3.1. Data

I use the same data in the previous chapter on segmentation (Chapter 2) with the same division into five folds for cross validation. The ATB comes fully vocalized with word-internal and case vocalization. The vocalization in the Penn Arabic Treebank has some peculiarities that are worth mentioning here:

1. The definite article *Al* ends with a consonant that must be vocalized, but it's not vocalized in the ATB. This may be an artifact of using the Buckwalter Morphological analyzer since it separates the definite article with a + sign. In naturally occurring Arabic, *Al* has two states (1) qamari *Al*, which is followed by one of the set (>, <, b, g, H, j, k, w, x, f, E, q, y, m, h), in which case the *l* has to be vocalized with a sukun (o in Buckwalter transliteration), (2) shamsi *Al*, i.e., *Al* followed by any other letter, in which case the *l* becomes silent, and the first letter of the stem is geminated. For example, *Al+nAs* is pronounced as *An-nAs*. This pronunciation is reflected in the writing system. While the conversion from the ATB style to the naturally occurring style is easy, it may be more complicated in the rare occasions in which *Al* is part of the stem, and when the segment boundaries are not marked. This is especially important when using the ATB to train a vocalizer that would vocalize non-ATB text.
2. In naturally occurring Arabic, when the prepositions *ElY* and *<lY* are connected with pronouns, the final *Y* changes into *y* and is assigned the sukun vocalization. In the ATB, the prepositions are separate tokens and are not assigned the sukun. A word like Elyhm should be vocalized as *Ealayohimo*,

but is vocalized as *Ealayhim*.

3. Double vocalization. Arabic has three long vowels: *A*, *y*, and *w*. When a consonant is followed by a long vowel, some people still write the equivalent short vowel and some do not. For example, the word *qAl* may also be written *qaAl*, The ATB is not consistent in terms of double vocalization: when the long vowel is *A*, the ATB does not provide the short vowel, but when the long vowel is *y* or *w*, the short vowel is always written.

I have not tried to change any of these peculiar characteristics of ATB Arabic, although they may need to be changed when handling naturally occurring Arabic.

3.3.3.2. Methods

I treat vocalization as a per letter classification problem in which the classifier has to decide, for each character, whether it should be assigned a short vowel class. The classifier has to choose one of eight different classes, here arranged in terms of frequency of occurrence in the ATB:

- **-** : occurs in 55.61% of all classes and indicates that the consonant is not assigned any vowel class. This is usually the case with the second letter in the definite article *Al*, and with word endings since we do not assign case.
- **a** occurs in 17.66% of all classes and represents the fatha.
- **i** occurs in 12.52% of all classes and represents the kasra.
- **o** occurs in 6.54% of the time and represents the sukun.

- **u** occurs in 4.98 % of all classes and represents the damma
- **~a** occurs in 2.012% of all classes and represents the combination of gemination and fatha.
- **~i** occurs in 0.53% of all classes and represents the combination of gemination and kasra.
- **~u** in occurs 0.13% of all classes and represents the combination of gemination and damma.

The classification is carried out by the memory based learner TiMBL. Details of the learner have been discussed in chapter 1. The best results are obtained for all the experiments below with the IB1 algorithm with similarity computed as weighted overlap, i.e. with a standard city block metric as distance measure. Relevance weights are computed with gain ratio, and the number of *k* nearest neighbors (or in TiMBL's case, nearest distances) is set to 1. The latter setting is noteworthy in that it signals that only the closest training examples provide reliable information for classifying a character. Parameter settings were determined in a non-exhaustive search.

3.3.3.3. Evaluation

In evaluating all the experiments below, I use word accuracy as the sole evaluation metric. For vocalization, a word is correct if and only if all the vowels assigned by the classifier are correct. A partially correct word is counted as wrong. For example, if *mhnds* is vocalized as *muhanodas* instead of *muhanodis*, it will be considered incorrect even though the difference is only in one character, and although the vocalization proposed by the classifier is a valid one (albeit different in meaning). There are four measures of evaluation, each geared toward a specific purpose:

(a) **General Accuracy**: The number of correctly vocalized words in the test set divided by the total number of words in the test set. General accuracy is what is usually reported in previous studies, but it does not give a good indication of how well a certain vocalization system will work on different kinds of data such as cases in which there is not enough overlap between the training set and the test set, and the case in which there is a large number of ambiguous words. General accuracy numbers also ignore the fact that not all orthographic units can be assigned vocalization. For these, the evaluation measures below can give better estimations.

(b) **Accuracy on Vocalizables**: The number of correctly vocalized vocalizables divided by the total number of vocalizables. Vocalizables are words that can be vocalized. Non-vocalizables are words, or orthographic units, that do not accept vocalization. To illustrate, numbers and punctuation marks can have no vocalization, and this is the reason they are excluded from evaluation in this measure. Including unvocalizables in evaluation tends to give more optimistic results, as the accuracy on these should be 100% whether or not we use a vocalizer.

(c) **Accuracy on Unknown versus Known Words**: Unknown words are those that are in the test set, but not in the training set, while known words are those in the test set that are also in the training set. The purpose of this evaluation measure is to test the classifier's ability to generalize beyond the training data.

(d) **Accuracy on Ambiguous Words**: Ambiguous words are those that can, according to the training set, have more than one possible vocalizations. Both ambiguous and unambiguous words are by necessity known words since unknown words cannot be characterized as ambiguous. The purpose of this measure is to test how the classifier handles ambiguity, and this can be useful in estimating how a vocalizer trained on some genre can generalize to another genre.

3.3.4. Vocalization Experiments:

I have run a number of experiments in non-exhaustive search to find the best settings for vocalization. The following represent the most important ones.

(1) **Characters Only as Features, (C)**

The C experiment uses only characters only feature vectors, without any regard to the context beyond the word itself. In a per letter classification task, the vectors look like those in Table 3.4 which vectorizes the word *Almthdp* (The-United), where each letter is in the context of the five preceding letters, and the five following letters. When the word does not have enough context before or after the focus letter, the underscore _ replaces the missing character. The last character represents the intended vowel.

```
_ _ _ _ _ A l m t H d -
_ _ _ _ A l m t H d p -
_ _ _ A l m t H d p _ u
_ _ A l m t H d p _ _ ~a
```

_ A l m t H d p _ _ _ i

A l m t H d p _ _ _ _ a

l m t H d p _ _ _ _ _ -

Table 3.4: Within-word context

(2) **Characters and Word Context (C+W)**

In this experiment, I add the context words to the vector. The purpose of adding the context words is based on the idea that the context can help in disambiguation since the context provides syntactic as well as collocational information for ambiguous words. The lexical context in this experiment includes, for each focus character, the two previous words, the focus word, the two following words, in addition to the character context in experiment C above. The word *Almthdp* is represented as vectors in Table 3.5.

wAl<mArAt AlErbyp AlmtHdp wAlErAq wsytwjh _ _ _ _ _ A l m t H d -

wAl<mArAt AlErbyp AlmtHdp wAlErAq wsytwjh _ _ _ _ A l m t H d p -

wAl<mArAt AlErbyp AlmtHdp wAlErAq wsytwjh _ _ _ A l m t H d p _ u

wAl<mArAt AlErbyp AlmtHdp wAlErAq wsytwjh _ _ A l m t H d p _ _ ~a

wAl<mArAt AlErbyp AlmtHdp wAlErAq wsytwjh _ A l m t H d p _ _ _ i

wAl<mArAt AlErbyp AlmtHdp wAlErAq wsytwjh A l m t H d p _ _ _ _ a

wAl<mArAt AlErbyp AlmtHdp wAlErAq wsytwjh l m t H d p _ _ _ _ _ -

Table 3.5: Lexical and Character Context

(3) **Characters and Stemmed Word Context (C+SW)**

This experiment uses the same set of features above, but instead of words, stems are used. The idea behind using stems is combating data sparseness. Arabic is a morphologically complex language with a large array of suffixes and prefixes. Reducing words to the stem can provide context without exploding the number of feature values.

I use a light stemmer that strips combinations of prefixes and suffixes without using any dictionary of stems. The stemmer starts with the longest span of possible affixes then goes inwards recursively. To illustrate, the stemmer checks if the word starts with the combination Conjunction+Preposition+DefiniteArticle+Stem. This is realized in Arabic through the complex prefixes *wbAl*, *wkAl*, *wll*, *fbAl*, *fkAl*, and *fll* followed by a string of characters. If a word does not match this criterion, the stemmer checks whether words it starts with the complex prefix Preposition+DefiniteArticle, which is realized in Arabic in the prefixes *bAl*, *kAl*, and *ll*. If the word does not meet this requirement, the stemmer checks whether it starts with the definite article. Similar processes apply for suffixes. The stemmer does not strip off the feminine singular marker, and when a word ends with a feminine plural marker, it is normalized to the singular form. The reason for this is that in many case the singular feminine marker may affect the vocalization. For example, if we remove the marker, *HyA* (a verb) and *HyAp* (a noun) will both be normalized to *HyA*, although the vocalization is different in both cases.

To illustrate, let us examine how the stemmer handles the word "*llmnTqp*" (Eng. to the region):

(a) The word does not start with Conjunction+Preposition+DefiniteArticle, pass to the next stage.

(b) The word starts with *ll*, which is a preposition plus the definite article, strip *ll*.

(c) The stemmer has now produced *mnTqp*. Since *ll* can only be followed by the stem with only inflectional suffixes, check whether it has inflectional suffix.

(d) The word ends with *p*, which is the feminine singular marker, keep the marker.

(e) The stemmer produces *mnTqp*

(f) No further operations are possible since *p* is always a single, unaccompanied, suffix.

The same lexical context in table 3.5 is now stemmed as shown in Table 3.6. For example, the first word in the vector, *wAl<mArAt*, is now *<mArp*, stripped of the conjunction *w*, the definite article, and the feminine plural marker *At* has been transformed to the feminine singular marker *p*. The feminine marker has not been stripped since it may be original in the stem, which is the case here.

<mArp Erbyp mtHdp ErAq sytwj _ _ _ _ _ A l m t H d -

<mArp Erbyp mtHdp ErAq sytwj _ _ _ _ A l m t H d p -

<mArp Erbyp mtHdp ErAq sytwj _ _ _ A l m t H d p _ u

<mArp Erbyp mtHdp ErAq sytwj _ _ A l m t H d p _ _ ~a

<mArp Erbyp mtHdp ErAq sytwj _ A l m t H d p _ _ _ i

<mArp Erbyp mtHdp ErAq sytwj A l m t H d p _ _ _ _ a

<mArp Erbyp mtHdp ErAq sytwj l m t H d p _ _ _ _ _ -

Table 3.6: Stemmed lexical context

(4) **Characters and Stemmed Word Context with Previous Decisions (C+SW+PD)**

This is a two-stage process that first runs the experiment with stemmed words and character context in the first stage, and, in the second stage, I add the decision reached by the classifier in stage 1, i.e. the class assigned, to the feature vector. Table 3.7 presents the same word *Almthdp* within its context and the previous decisions made. In this specific example, all the classes were correctly assigned in the previous experiment, which is often the case.

- <mArp Erbyp mtHdp ErAq sytwj _ _ _ _ _ A l m t H d -

- <mArp Erbyp mtHdp ErAq sytwj _ _ _ _ A l m t H d p -

u <mArp Erbyp mtHdp ErAq sytwj _ _ _ A l m t H d p _ u

~a <mArp Erbyp mtHdp ErAq sytwj _ _ A l m t H d p _ _ ~a

i <mArp Erbyp mtHdp ErAq sytwj _ A l m t H d p _ _ _ i

a <mArp Erbyp mtHdp ErAq sytwj A l m t H d p _ _ _ _ a

- <mArp Erbyp mtHdp ErAq sytwj l m t H d p _ _ _ _ _ -

Table 3.7: Stemmed lexical context + previous decisions

3.3.5. Results and Discussion

Table 3.8 presents the results for the four experiments above averaged across five folds of cross validation.

Experiment	**WAR**
C	92.31%

C+W	92.62%
C+SW	93.06%
C+SW+PD	93.28%

Table 3.8: Lexical information as features

As can be seen from table 3.8, using characters alone as features is fairly effective in deciding the vocalization class as it yields a Word Accuracy Rate (WAR) of 92.31%. This means that vocalization can generally be determined based on local context. The extra-word context does, however, improve vocalization accuracy as adding the word context (C+W) increases the WAR to 92.62%. When we stem the context words in the C+W experiment, we obtain a better accuracy: a WAR of 93.06%, which means that stems are more effective than words as a context for vocalization. When we take the results of this experiment, and add it to the training set (C+SW+PD), we gain an extra 0.22% as the accuracy increases to 93.28%.

The addition of the context and the previous decisions leads to accuracy improvement in every positive class from the C experiment to the C+SW+PD experiment. Table 3.9 compares the two settings on every class in the vocalization experiments:

Class	**C**	**C+SD+PD**
a	96.90%	97.44%
i	97.00%	97.66%
u	92.90%	94.25%
~a	96.00%	96.86%
~u	80.36%	84.53%
~i	89.02%	91.30%
o	96.78%	97.23%
-	99.03%	99.08%

Table 3.9: Accuracy improvement between C and C+SW+PD

The only class whose accuracy decreases is the "-", which indicates that no vowel is assigned at this position, and the class gaining the most is the **~u** class, which is the class with the lowest initial results. This means that with the improvement, there is a slight increase of the possibility to assign a vowel class where no vowel class is required, but this over-vocalization is minimal.

In the C+SW+PD experiments, the most important feature, judging by gain ratio, is the previous decision. The second most important feature is the character following the focus character. This is natural since when a long vowel, or a word end, follows, this results in the assignment of the "-" class, which is the majority class. The third most important feature is the focus character itself, and this can be attributed to similar reasons as those associated with the pre-focus character since many characters simply do not accept vocalization classes.

3.3.5.1. Evaluation on Unknown vs. Known Words

The evaluation above was general in that it did not distinguish between known and unknown words. In this section, we provide numbers pertaining to the classifier's ability to generalize to unknown words. Table 3.10 presents the evaluation on known vs. unknown words across five folds of cross validation in the best scoring experiment in this chapter, C+SW+PD, where UWA is the Unknown Word Accuracy and KNA is the Known Word Accuracy. The average percentage of unknown words per fold in the vocalization experiments is 9.62%[13] with a standard deviation of 0.56.

[13] The percentage of unknown words varies between vocalized and unvocalized words.

Experiment	UWA	KNA	General Accuracy
C+SW+PD	57.82%	96.59%	93.28%

Table 3.10: Accuracy on known vs. unknown words

We can see from Table 3.10 that there is a significant difference between known words and unknown words, with known words scoring an accuracy of 96.59% versus 57.82% on unknown words.

Part of the difficulty on unknown words may be that many of them are proper nouns. Proper nouns constitute 21.32% of all unknown words across the five folds. Many, if not most, of these are foreign names that do not follow any rules, and thus cannot be generalized.

The numbers on unknown vs. known words suggest that vocalization accuracy is a function of the overlap between the training set and the test set. If the two sets have enough words in common, then there is a high chance of obtaining good quality vocalization, whereas if the ratio of common words is low, then the quality of vocalization may not be as good.

The fact that known word accuracy is high, regardless of the context, may be due to the low ambiguity rate in the Arabic Treebank (1.07 vocalizations per word). This leads us to examining performance on ambiguous words.

3.3.5.2. Evaluation on Ambiguous Words

The data set used in this chapter has a vocalization ambiguity rate of 1.07, with 5.14% of the word types being ambiguous, i.e. each has more than one vocalization. 86.3% of the ambiguous word types have two vocalizations each, 12.15% have three vocalizations each, 1.35% have four vocalizations each, and only 5 word have five vocalizations each (0.17%).

In dealing with ambiguous words, I look at each fold in each experiment separately because what may be an ambiguous word in one fold may not be so in another. In doing so, I have found that the average percentage of ambiguous words per fold is 21.31%. This means that while ambiguous words make up only 5.14% of the word types, they constitute more than a fifth of the word tokens across the five folds. Table 3.11 summarizes the results obtained on ambiguous vs. unambiguous words across five folds.

Experiment	**AmbigWordAccuracy**	**UnAmbigWordAccuracy**	**Ambiguous**	**General Accuracy**
C+SW+PD	91.28%	98.03%	21.31%	93.28%

Table 3.11: Accuracy on ambiguous vs. unambiguous words

Table 3.11 shows that there is a considerable difference between ambiguous words and unambiguous words as ambiguous words score an accuracy of 91.28% versus 98.03% for unambiguous words. This shows that if the training set has a large percentage of ambiguous words, these words will be less accurate in terms of vocalization than unambiguous words.

3.3.6. Error Analysis

Although the quality of vocalization is mainly measured through word accuracy or word error rate, the process of vocalization itself is performed on characters. Error Analysis should thus consider the errors made by the classifier at the character, or class, level.

Table 3.12 provides the confusion matrix between the eight classes in the best scoring experiment. Rows present the original gold standard classes, while the columns present the classes assigned by the vocalizer.

	a	**u**	**i**	**o**	**~a**	**~u**	**~i**	**-**
a	97.44	3.15	1.07	1.23	1.03	1.70	1.17	0.29
u	0.77	94.25	0.40	0.14	0.01	2.33	0.25	0.16
i	0.90	1.35	97.64	0.31	0.26	1.55	1.95	0.26
o	0.40	0.18	0.14	97.23	0.41	0.78	0.64	0.17
~a	0.10	0.04	0.04	0.10	96.86	5.23	2.98	0.02
~u	0.01	0.06	0.01	0.01	0.29	84.53	0.74	0.01
~i	0.03	0.03	0.1	0.03	0.82	2.76	91.30	0.01
-	0.35	0.94	0.63	0.96	0.24	1.13	0.98	99.08

Table 3.12: vocalization classes confusion matrix

The best scoring class is the “-”, which denotes that there is no vowel assigned. This is probably due to the predictability of this class as it is reserved for word-final consonants, long vowels, and the *l* in the definite article *Al*.

We can also see that the ***u*** class gets confused with the ***a*** class in 3.15% of the time as they occur in the same contexts and the words in which they appear are usually ambiguous. This is obviously also the case with the geminated version **~*u*** which gets confused with **~a** in 5.23% of the cases. **~*u*** is also the lowest scoring class at 84.53%.

Sometimes, the error may be due to annotation inconsistency rather than to a misclassification by the classifier. This may be due to dialectal variation. For example, the word *HzyrAn* (Eng. June) is vocalized in the ATB either as *HaziyrAn* (34%) or *HuzayorAn* (66%)*,* both of which are possible vocalizations. Another example is the word *dwly* (Eng. international) which can be either *dawoliy* (18%) or *duwaliy* (82%), according to the geographical region, and both are used in the ATB.

Another observation is that some words, especially proper nouns, are left unvocalized in the ATB. I assume that these words did not have a suitable solution provided by the Buckwalter Morphological Analyzer, and were thus left untouched. Some of these words include *qlEp* (Eng. castle), *Aldwly* (Eng. the international), *alhndAwy* (anthroponyms) and *AlHS* (anthroponyms).

As with segmentation, words of specific POS tags contribute more to errors than others. Table 3.13 lists the most common parts of speech of erroneously vocalized words.

Percentage	**Tag**
18.22%	NOUN_PROP
4.98%	NOUN
3.8%	DET+NOUN
3.61%	ADJ+CASE_INDF_ACC
3.11%	SUB_CONJ
3.04%	DET+NOUN_PROP

Table 3.13: POS tags with most vocalization errors

Proper nouns top the list of the POS tags associated with errors with 18.22% of all errors. Proper Nouns also rank sixth when they are preceded by the definite article. The reason for this may be that many proper nouns are foreign words, which is normal in newswire text. For example, the name *Kent* has been vocalized by the vocalizer as *kunot*. Also, many proper nouns are left completely unvocalized, which is an annotation inconsistency.

The subjunctive conjunction tag (SUB_CONJ) ranks fifth on the list of tags for the erroneously vocalized POS classes. This tag is assigned to a limited number of words most of which are very frequent and highly ambiguous. For example, The SUB_CONJ *kamA* (Eng. also, in addition) can also be a noun vocalized as *kam~A* (Eng. an amount).

Another source of errors is words with internal case. An examination of one fold reveals that words with internal case/mood assignment constitute 4.22% of all words in the test set, and the accuracy on these words is 80.45%. 75% of the erroneously vocalized case-internal words fall within an edit distance of 1 from the correct vocalization. A manual examination of 50 errors showed that most of the errors (81.25%) are related to case assignment. Examples of these include *Hukumatah* vocalized as *Hukumatih,* where the former is in the accusative case while the latter is in the genitive case.

3.3.7. Training Size Effect on Vocalization

To measure the effect of the training set size on vocalization, I test on one fold (fold 1) and vary the size of the respective training set starting from 10,000 words, then 1/10 of the data, and increase the amount by one tenth until I reach the full training set used in the previous experiments. Table 3.14 and figure 3.1 detail the relationship between the size of the training set and the accuracy of the vocalization process.

When the training size is 10,000 words, the accuracy on words is 90.97%. It increases to 91.88% with 38,659 words of training (one tenth of the full training size). The increase keeps consistently going up till it reaches 93.49% with 386,590 words in the training set (full size). In fact, there is a correlation of 0.916 between the training size and vocalization accuracy, which suggests a causal relation between the size of the training data and the accuracy of vocalization since the size variation is manipulated rather than observed. Since the data sizes are manipulated, and not naturally observed, we can assume that the gain in accuracy is due to the increased data size.

Figure 3.1 shows that we may still obtain greater accuracy if we increase the data size above the full set used here.

Number of Words	**Accuracy**
10000	90.97%
38659	91.88%
77318	92.23%
115977	92.57%
154636	92.79%
193295	92.92%
231954	93.06%
270613	93.13%
309272	93.25%
347931	93.37%
386590	93.49%

Table 3.14: Training Size Effect on Vocalization

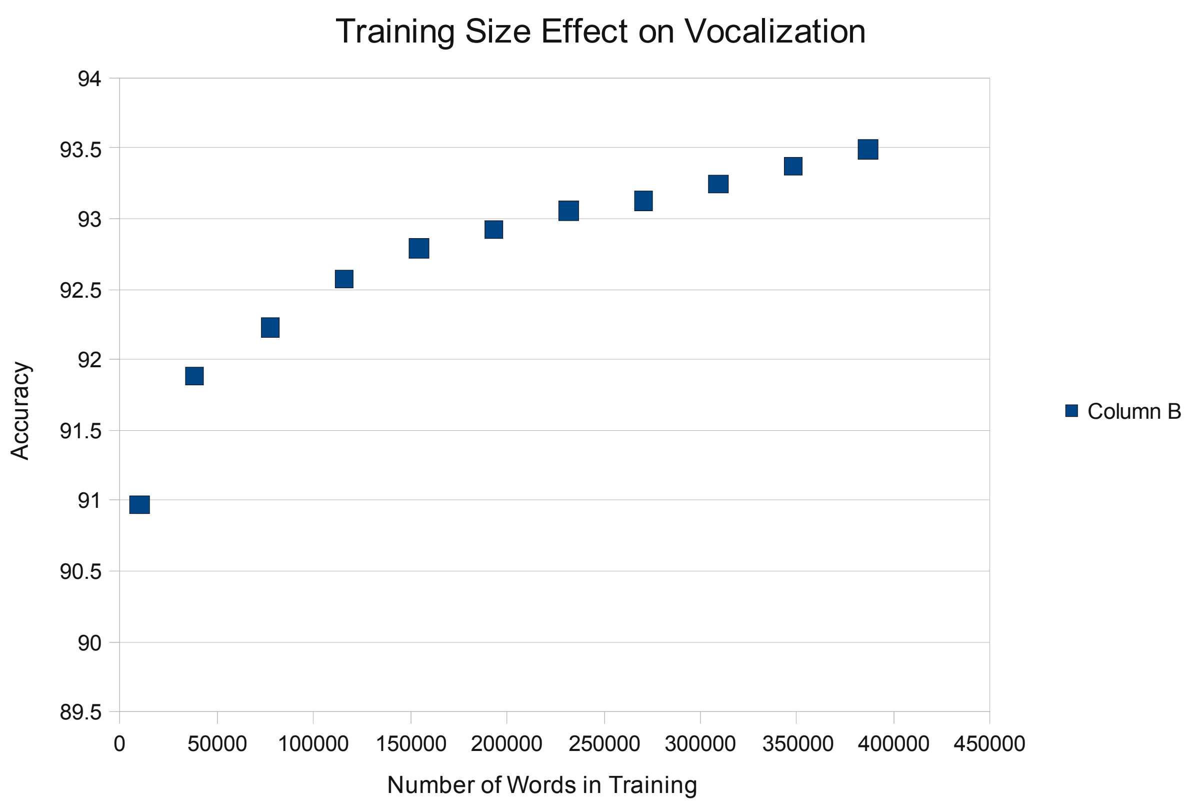

Figure 3.1: Training Size Effect on Vocalization

3.4. The interaction between vocalization and word segmentation

Now that we have studied vocalization and word segmentation, and found the best settings for each, it is time we considered how these two processes interact. In this section, I investigate whether performing vocalization can help word segmentation and vice versa. From a theoretical point of view, vocalization is perceived to help in segmentation, since vocalization helps to disambiguate words. This is usually the case when an Arabic orthographic unit starts with a character that is ambiguous between a prefix/clitic and a word-original character. To illustrate, words beginning with the conjunction *w* are often ambiguous. The words *wSAl* (Eng.love or and+walked bravely), *wdE* (Eng. bid farewell or and+let), *wlm (dine or and+not)*, and *wDw'* (Eng. ablution or and+light) are all ambiguous between the readings in which *w* is a conjunction, and the reading in which *w* is part of the word. Vocalization can help to decide which reading is the intended one in these cases although there are other cases in which it cannot, but there is not a single case in which vocalization leads to more ambiguity than there already is in the orthographic unit.

It is unclear how segmentation can help if we isolate the segments, since isolating segments usually leads to ambiguous units. For example, the segment *Hsn* in the orthographic unit *wbHsnAthm* has only one possible vocalization (*Hasan*) due to the word-internal context constraints, while an isolated *Hsn* can have such vocalization as *Hasan*, *Hasa~na Hasuna*, *Hus~ina*, and *Huson*.

In this section I use one fold as test data (fold 1), and I add information to the best scoring feature set from the vocalization and segmentation experiments I have carried out.

3.4.1. Does word segmentation help vocalization?

In this section, we investigate the effect of word segmentation on word vocalization. To this end, I ran two experiments to investigate the matter:

(a) **Including the carrier segment as a feature in the vocalization vector**. In this experiment, I include the results of word segmentation in the vocalization experiment by including the carrier segment in the vocalization vector. For example, if the word *Almhndswn* (Al+mhnds+wn) is being vocalized, then I include for each letter of the word, the segment in which that letter occurs. Thus, in addition to the best scoring features from the vocalization experiment, a new feature is included. In the word *Almhndswn* above, if the focus letter in the vector is the letter *m*, then the segment *mhnds* is included in the vector. I use only automatic segmentation, since gold standard segmentation cannot be found in naturally occurring Arabic.

(b) **Including the segmentation class as a feature in the vocalization vector**. In this experiment, I do not include the carrier segment, but the class predicted by the classifier when performing the segmentation. For example, in the word *mhnds* above, the classifier assigns class - to the letter *m*, meaning that this letter is not a segment boundary, then **-** is added to the vocalization vector.

The purpose of either experiment is to add more information to the vocalization vector in hope that this information will remove the ambiguity inherent in Arabic words. Table 3.15 presents the results of the three experiments[14].

Experiment	**Accuracy**
C+SW+PD	93.49
C+SW+PD+carrier_segment	93.37
C+SW+PD+seg_class	93.40

Table 3.15: Segmentation effect on vocalization

[14] Experiments that targeted segments, rather than whole words, as the units of the vocalization vector yielded much worse results even with the use of gold standard segmentation. On one fold, the Character Accuracy Rate on segment vectors is 96.6% vs. 97.8% on word vectors. Re-combining the segments into words is expected to yield worse results, and the numbers will be even worse if the segmentation is not perfect. The reason for this is that segments are much more ambiguous than words.

As we can see from table 3.15, adding the carrier segment leads to the accuracy deteriorating from 94.49% in the C+SW+PD experiment to 93.37%, and when I add the segmentation class the accuracy drops to 93.40%. This means that adding this information actually hurts the vocalization process rather than improves it. Vocalization should thus be done in isolation of segmentation. As pointed out in the introduction to this section, segments are ambiguous by nature, and this may be the reason behind their negative effect on vocalization accuracy.

3.4.2. Does vocalization help segmentation?

As stated in the introduction, vocalized data is less ambiguous than unvocalized data, and this should make language processing on vocalized data, including segmentation, more accurate. This hypothesis is corroborated by the fact that performing segmentation on gold standard vocalized data yields better results than performing segmentation on unvocalized data by 1%. (99.26% vs. 98.24%), but gold standard quality vocalizations are very hard to obtain as the best vocalization systems obtain accuracies around 94%. Although this quality is good for the task of vocalization, it will act as a ceiling for the quality of segmentation. This would harm the segmentation process whose quality is in excess of 98%, which makes performing vocalization immediately before performing segmentation a useless step, but we can approach the problem differently by using vocalization as a feature in the segmentation vector. I have conducted two experiments to examine the effect of vocalization on segmentation:

a) **Vocalize then Segment.** The vocalize then segment experiment takes as input automatically vocalized text and produces segmented text based on this vocalized input. This experiment is used as a guideline only since we know beforehand that the segmentation accuracy will be harmed by the relatively low accuracy on vocalization.

b) **Add the vocalization class to the vector of the best scoring segmentation experiment.** Another way to integrate vocalization and segmentation is to include the class of vocalization in the segmentation process. Thus, for every character that is the focus character in the segmentation vector, we also include the class assigned by the vocalization experiment to that specific character.

Table 3.16 presents the results on fold 1 using the best vocalization settings.

Experiment	**Accuracy**
Gold Standard Vocalization	99.26%
Vocalize then segment	91.01%
Add vocalization class to segmentation vector	98.29%
Best scoring unvocalized segmentation experiment	98.34%

Table 3.16: Effect of vocalization on segmentation

When I vocalize the text then segment it, I obtain an accuracy of 91.01%. This performance is very mediocre compared to the experiment run on gold standard vocalization which scores an accuracy of 99.26% and unvocalized data (98.23%). This means that vocalization before segmentation harms performance. This experiment indicates that performing vocalization then segmenting the vocalized data is not the best solution since the errors resulting from automatic vocalization harm word segmentation.

Adding the vocalization class to the best scoring segmentation experiment, the character plus previous decision plus part of speech, yields an accuracy of 98.29% compared to 98.34% on unvocalized data (see chapter 2). The addition of vocalization as a feature in the vector harms segmentation accuracy.

3.5. Conclusion

I have introduced vocalization, the process of restoring short vowels and the gemination marker which are usually missing in naturally occurring Arabic. I have used a memory-based learner, TiMBL, in a per letter classification approach to determine for each letter which vowel should be assigned to it. The best results were obtained by a feature vector that includes the focus letter, the focus stem, the two previous stems, and the two following stems, and with including the classifier's previous decision. The Word Error Rate of the best scoring experiment was 6.72%, with a Known Word Error Rate of 2.66% and an Unknown Word Error Rate of 44.92%. The results on unknown words indicate that vocalization is a difficult task. Even within known words, there is a difference between ambiguous words, those words with more than one possible vocalizations in the training set, and unambiguous words, with ambiguous words scoring an accuracy of 91.27% (a Word Error Rate of 8.73%).

Many errors in the experiments result from the confusion between the diacritics *u* and *a* since these diacritics occur in the same contexts, although they denote different meanings and different morphological and syntactic structures. Other errors result from different vocalizations of the same words in the Penn Arabic Treebank.

I have also examined the effect of the training data size on vocalization, and found that the addition of more training instances can still help vocalization.

I have tried to integrate vocalization and segmentation in order to improve the quality of each one of these processes, but it has been shown that segmentation information does not help vocalization and vocalization information harms segmentation. This means that vocalization and segmentation are two separate and independent processes. The high quality of segmentation may allow the use of the segmentation in the following chapters concerning part of speech tagging and dependency parsing. I will also still explore the use of vocalization in Arabic morphosyntactic processing.

Chapter 4: Part of speech Tagging

4.1. Introduction

Part of speech tagging (POS), **“**the task of labeling (or tagging) each word in a sentence with its appropriate part of speech” (Manning and Schütze, 1999: 341), derives its significance from the large amount of information it gives “about a word and its neighbors”. (Jurafsky and Martin, 2009: 123). Part of speech tagging is an intermediate step that is easier than parsing, but nonetheless can give us useful information. The input to the part of speech tagger is a sequence of words (a sentence) and the output is each word in the sentence along with its grammatical category given the context. For example, given the sentence: “Opposition Republicans plan to fight back against health care legislation and take back congressional seats.”, the POS tagger should give the following analysis:

Opposition/NOUN **Republicans/**PROPER_NOUN **plan/**VERB **to/**TO **fight/**VERB **back/**ADVERB **against/**PREPOSITION **health/**NOUN **care/**NOUN **legislation/**NOUN **and/**CONJUNCTION **take/**VERB **back/**ADVERB **congressional/**ADJECTIVE **seats/**PLURAL_NOUN) **in/**PREPOSITION **November/**NOUN . /FULL_STOP

POS tagging can thus be viewed as a disambiguation task since it has to decide, for example, whether *plan* is a verb or a noun, whether *back* is a verb or an adverb, and whether *seats* is a plural noun or a third person imperfective verb.

Part of speech tagging can be used for determining the pronunciation of a certain word. If we know that *present* is a verb, then we can tell that the stress falls on the second syllable, and this can be very effective in Text to Speech systems used for reading for the visually impaired, and for people who prefer listening to reading. The same is also true for Arabic: if we know that *qsm* is a noun, then the pronunciation (i.e. vocalization) will be '*qisom*' (Eng. department), while as a verb the word is pronounced as *qas~ma* (Eng. to divide). POS tagging has also been used in Word Sense Disambiguation, Information Retrieval, Information Extraction, and a variety of linguistic research (Jurafsky and Martin, 2009: 124).

Morphologically rich languages like Written Modern Standard Arabic (Arabic henceforth) offer some challenges to natural language processing systems due to the many forms a word can take, which leads to data sparseness. Most current part of speech taggers, for example, depend on supervised machine learning techniques in which the tagger learns from training sets which contain a fair amount of words and their associated parts of speech. When the morphology is rich, the tagger will be faced by many forms of the same word that do not repeat enough for the tagger to learn the pattern. As an example, the Arabic verb *ktb* (Eng. to write) can be found in over 400 different forms. The verb *ktb* can be realized as *ktb*, *ktbt*, *ktbn*, *ktbnA*, *ktbA*, *ktbwA* in the past tense alone, and each of these can be preceded by proclitics and followed by pronouns that are parts of the words, and no white space is used to separate them. The same word *ktb* can also be a plural noun that can take similar prefixes and suffixes. For many natural language tasks to make sense, some form of morphological analysis, or word segmentation, in which we split the suffixes, prefixes, and clitics from the forms, has to be performed first, but this process introduces other difficulties as well since using word segments, or tokens, instead of whole words, introduces an extra layer of morphological analysis, and the process of morphological analysis itself is not perfect. For example, while the oft-cited word *wbHsnAthm* (وبحسناتهم) (Eng. and+by+their+virtue+s) is not ambiguous by and of itself, we may need to segment it for such tasks as POS tagging, Information Retrieval (IR), collocation extraction as well as many others, but after segmentation, we have a high degree of ambiguity as we have 3 x 2 x 4 x 2 x 6 = 288 possible composite tags for the word (see table 4.1). This means that while segmentation is necessary to overcome the data sparseness problem, it may introduce its own problems as well (see chapter 2 on segmentation). The meanings of these tags, and all the segment tags in the chapter are in appendix 1.

Token	**Possible POS Tags**	**Translation(s)**
w	CONJ, PREP, ABBREV	And, by, W
b	PREP, ABBREV	With, by, B
Hsn	NOUN, ADJ, NOUN_PROP, PV	Beauty, Good, Hassan, improve, be good
At	NSUFF_FEM_PL, ADJ	Coming (adj)
hm	NOUN, POSS_PRON_3MP, IV, IVSUFF_DO:3MP, PRON_3MP, PVSUFF_DO:3MP	Grief, their, to start, them, they

Table 4.1: Possible POS tags for the the segments in the word *wbHsnAthm*

In spite of the perceived ease of the POS tagging task, and the fact that it reaches an accuracy of around 97% for English, there exist external factors that affect the tagging accuracy. Manning and Schütze, (1999: 372) list four factors that can affect tagging accuracy as follows:

- **The amount of training data available**. More data usually lead to better results.
- **The tagset**. The larger the tagset, the more grammatical ambiguity there is in general. When we use fine-grained tags, we introduce subtleties that may affect the tagging quality. For example, the word *to* in English can be a preposition or an infinitive marker. A tagset that has two tags for *to* may lead to incorrect tagging, while a tagset that tags *to* as TO cannot. Dickinson and Jochim (2010), however, note that tagsets with equal numbers of tags have varying accuracies depending on the linguistic quality of the tagset, and one how local it is. For Arabic, the problem of the tagset can be more complicated since the morphological complexity of Arabic led the Penn Arabic Treebank creators to adopt a detailed tagset that takes care of inflections and clitics as well.
- **The difference between the training corpus and dictionary on the one hand and the corpus of application on the other**. The tagger normally performs best when the training and test data are drawn from the same genre and the same time period. Manning and Schütze note that research papers normally use training and test sets from the same genre and time. I also do the same here due to the lack of annotated data that span various genres. This criterion cannot be easily accommodated in full, since the data available for Arabic comes from the Penn Arabic Treebank, which is a homogeneous set, as all the data are newswire text without enough variation.
- **Unknown Words**. The coverage of the dictionary has an effect on accuracy. When there are many out-of-vocabulary words, as in the case of technical domains, the accuracy drops. The percentage of unknown words in the dataset

used in this thesis is 8.55 averaged across five folds. I have divided an English text from the AFP news agency, with the exact same number of words as the data used in this chapter, into five folds and have found that the average percentage of unknown words across the five folds is 3.24. This indicates that Arabic has more than double the number of unknown words as in English, which is not surprising, given the nature of the Arabic orthography (the large number of affixes and clitics).

I have tried to make the results reported in this study as realistic as possible by taking the points pertaining to POS tagging accuracy seriously and trying to accommodate them. For this purpose, in addition to reporting general accuracy, I present accuracy figures for Known vs. Unknown Words, and Ambiguous vs. Unambiguous Words. A peculiarity of Arabic (and other morphologically rich languages), which can affect part of speech tagging accuracy, is the rich morphology of words which may necessitate segmentation or tokenization, since segmentation alleviates data sparseness, but it also introduces errors that percolate to part of speech tagging. The effect of segmentation, and whether POS tagging can be performed without this pre-processing step, is a major concern of this chapter as no real world POS tagging can be performed without considering the nature of the orthography.

The rest of this chapter is divided as follows: section 4.2 introduces the part of speech in the Arabic linguistic tradition, section 4.3 discusses related work, and section 4.4 discusses the current study in terms of data, methods, evaluation, results of using the

Habash and Rambow tagset, and the effect of vocalization and data size on POS tagging accuracy. The chapter ends with a conclusion summarizing the findings of the chapter.

4.2. Parts of Speech in the Arabic Linguistic Tradition

In the Arabic linguistic tradition, a word can belong to one of three parts of speech: NOUN, VERB or PARTICLE. The distinction between nouns and verbs lies in their relationship with the concept of tense. The noun is traditionally defined as an independent unit of which tense is no part, the verb as an independent unit of which tense is part, and a particle as a non-independent unit (Al-Hamalawy 1998: 29). In the sentence *nAm AlTfl fy Albyt* (Eng. slept the-child in the-house), *nAm* indicates a past tense, so it is a verb. *AlbYt* has no indication of tense, so it is a noun, and *fy* does not have an independent meaning, and should thus be treated as a particle. In this categorization, the noun includes adjectives, proper nouns, participles, and pronouns. This tri-partite classification is based on formal as well as distributional criteria. For example, the word *Tyb* (Eng. good) is an adjective that can be used as a noun, and it takes the same clitics and inflections. The pronoun can also take the positions occupied by the noun in a sentence.

Verbs are more restrictive as they include only what we can call verbs proper, while particles include everything that cannot be classified either as a noun or as a verb. This includes all clitics, question words, prepositions, the definite article, as well as others. It was not until the 20th century that Arab linguists proposed to change the traditional classification. Tammam Hassan (1979) proposed a new set of POS tags that comprises seven parts of speech: noun, adjective, verb, pronoun, *khalifa*, verb, and particle. This scheme added the category *adjective* based on distributional and formal criteria, and it seems that this new system allows for a word to be both a noun and an adjective depending on the context. The pronoun category includes personal pronouns, demonstrative pronouns, and the relative pronouns. The *khalifa* category is very similar to exclamation in English as it denotes affective language. It includes verbal nouns, exclamation, and a number of forms whose status between verbalness and nominalness has always been controversial, all of which being fixed expressions. The adverb category includes a limited set of words denoting time and place. The particle is still a cover term for functional words.

With the advent of computational linguistics, several POS tagsets have been proposed for the computational treatment of Arabic. The characteristics of some of these tagsets are detailed below.

4.3. Related Work

Diab et al (2004) use a machine learning approach, Support Vector Machines, to model Arabic part of speech tagging as a classification approach using the Reduced Tagset, which maps Arabic to English-like POS tags. The Arabic Reduced Tagset treats tokens, rather than segments or orthographic units, and does not have tags for inflections (see table 4.2). In a complex form like *wbHsnAthm*, a token is a unit that has a syntactic function, a segment is any unit of the word whether it has a syntactic or a morphological function, and an orthographic unit is the whole word is written in naturally occurring Arabic. For example, *w* is conjunction, so it is a token, *At* is a plural marker and is a segment but not a token since it performs a morphological rather than a syntactic function (see chapter 2 for details). The data used by Diab et al represents the first edition of the Penn Arabic Treebank containing Agence France Presse (AFP) newswire articles ranging over a period of 5 months from July through November of 2000. The corpus contained 734 news articles with 140,000 words (168,000 tokens). The Diab et al feature set includes the focus word in a window of +/-2 words, their POS tags, their type from the set {Alpha, Numeric}, and previous tagging decisions for the words within the context. Diab et al report an accuracy of 95.5% on all tokens drawn from the ATB. The accuracy reported is, however, not representative of real world POS tagging since they use the gold standard tokenization, and since they report accuracy on tokens rather than on whole words as they appear in naturally occurring Arabic.

Tag	**Meaning**	**Tag**	**Meaning**
CC	Conjunction	***VB***	Imperative Verb
CD	Number	***FW***	Foreign word
PRP$	Possessive Pronoun	***VBN***	Passive Verb
RP	Particle	***IN***	Preposition
UH	Interjection	***JJ***	Adjective
DT	Determiner	***VBP***	Imperfective Verb
NN	Singular Noun	***WP***	Interrogative particle
NNS	Plural Noun	***NNP***	Singular Proper Noun
NNPS	Plural Proper Noun	***PRP***	Pronoun
NO_FUNC	Unknown	***RB***	Adverb
WRB	Interrogative adverb	***NNP***	Proper Noun

Table 4.2: The ATB Reduced Tagset

Habash and Rambow (2005) follow Diab et al in using SVM's, and the same data set, for Arabic POS Tagging, but they use a full morphological analyzer, instead of classification, to produce all the possible morphological forms of a certain word. The morphological analyzer makes use of the BAMA morphological analyzer (Buckwalter 2002) to produce all the possible tags of the word. Habash and Rambow then use a classification approach to rank the BAMA produced analyses. They first use their ALMORGEANA morphological analyzer (which makes use of the BAMA morphological analyzer) to produce all possible analyses of the input word, then use a Support Vector Machines classifier trained on the gold standard data of the Penn Arabic Treebank to select the most probable solutions. Habash and Rambow use binary features in the classification determining whether any of the analyses produced by the morphological analyzer has certain POS tags, conjunctions, cliticised pronouns, determiners, gender, person, number, voice, and aspect. They also criticize the Reduced Tagset as unmotivated, since it makes distinctions based on English that may not be relevant for Arabic, but they nonetheless use it for comparison reasons along with a smaller tagset that has only 15 tags. Habash and Rambow report that the use of the morphological analyzer helps achieve better accuracy and report an accuracy of 97.6% using their 15-tag tagset. Like Diab et al (2004), Habash and Rambow use the gold standard tokenization distributed with the ATB, which means that both results are over-optimistic when considering real-world data. As with Diab et al (2004), the evaluation on tokens, rather than full words, is unjustified since it masks the difficulties of tokenization.

Van den Bosch et al (2007) use memory-based learning for both morphological analysis and POS tagging of Arabic. They use ATB1 version 2 (166,000 words) as their dataset. Unlike Habash and Rambow (2005) and Diab et al (2004), they use the full tagset in their approach, but they use tokens instead of whole words (i.e. they use the ATB as is without re-connecting the tokens into words in the POS tagging experiments). Van den Bosch et al. report a total accuracy of 91.5% with 93.3% accuracy on known words and 66.4% accuracy on unknown words. Although I use the same algorithm and perform experiments on the full tagset as well, I differ from van den Bosch et al. in that I do not use gold standard tokenization as I re-attach the words before running the experiments in order to obtain naturally occurring Arabic. This makes the classification problem more difficult, as there are more unique POS tags, and more data sparseness, but this is also how Arabic data is always available. An example of our data is the word *wbdA* (Eng. and he-seemed), which consists of two ATB entries (*w* and *bdA*), and is treated as two words in the approach by van den Bosch et al., but which I treat as one word.

Diab (2007) introduced a new tagset, the ERTS (Extended Reduced Tagset) to be used with Arabic base phrase chunking. The ERTS comprises 75 POS tags and is derived from the full tagset with which the Arabic Treebank is annotated, and it enriches the Reduced Tagset (RTS) with definiteness, gender, and number information. The ERTS explicitly marks gender, number and definiteness on nominals, namely nouns, proper nouns, adjectives and pronouns, but it does not mark person or number for verbs. Diab justifies not including the person feature on verb by stating that "neither person nor mood are explicitly encoded" in naturally occurring Arabic. This is not true since Arabic verbs inflect for person. For example, the Arabic verb *ktb* is *>ktb* for the first person singular but *nktb* for the first person plural, and *yktb* for the third person masculine.

The Columbia Arabic Treebank (Habash et al, 2009) introduced a new system of tagging Arabic based on Arabic linguistic tradition. This tagset has only six tags and it omits many distinctions in the Penn Arabic Treebank. The six tags are (1) VRB which is used for all types of verbs without any distinctions between imperfect verbs and perfect verbs. This category also includes what is termed in Arabic incomplete verbs, which behave syntactically like verbs but do not take the full set of inflections. An example of this is the verb *lys* (Eng. not) which is treated by Arabic linguists as an incomplete verb but which is usually tagged in the Arabic Treebank as a negative particle, (2) VRB-PASS is used for passive-voice verbs, (3) NOM is used for all nominals such as nouns, adjectives, adverbs, pronouns, numbers and interjections. Also, nouns that function as prepositions and all quantifiers are considered NOMs. (4) PROP for proper nouns. (5) PRT: This tag is used for all particles and is a superset that includes many different closed classes such as prepositions, coordinating conjunctions, subordinating conjunctions, conditional conjunctions, verb-like particles, conditional particles, verbal particles, interrogative particles, and the vocative particles. When the definite article *Al* is found in isolation, it is also treated as a particle. (6) PNX is used for all punctuation signs. The tagset differs from conventional Arabic parts of speech in that it includes tags for proper nouns, passive verbs, and punctuation. Up till this point, the new tagset does not seem to have been used in research, but it may be influential in the future due to its simplicity.

4.4. The Current Study

The current study aims to give an outline of the problems involved in Arabic part of speech tagging as well as to introduce novel ways of treating it. It introduces two novel approaches of performing Arabic part of speech tagging, which are suitable for real-world tagging in that they do not assume gold standard segmentation: (a) Segmentation-based tagging using non gold standard segments, and (b) whole word tagging without any form of segmentation, tokenization, or morphological analysis. The study also discusses the effect of vocalization on POS Tagging, and the effect of the amount of information in the POS tagset on tagging accuracy. As outlined in section 4.1 above, the study also examines the effect of ambiguity and unknown words on POS tagging accuracy.

4.4.1. Data, Methods and Evaluation

4.4.1.1. Data

The data used in this POS tagging chapter is based on the same dataset used in the two previous chapters on segmentation and vocalization. I depend on the Penn Arabic Treebank, ATB, specifically on the POS files for extracting the training and test sets.

The ATB has a POS section and a parsed section that also has POS information, but its word segmentation scheme is intended to reflect syntactic rather than POS information as it adopts tokenization. The POS section is both tokenized and segmented with tokens, rather than full words constituting the entries. The POS section is also fully vocalized with both word-internal short vowels and case and mood markings. For the purposes of the experiments presented here, I delete all the vocalizations (a, i, o, u, ~, K, F, N)[15] and their respective tags. Short vowels are assigned POS tags only when they represent case or mood information. For example, the word *ktAb* can be assigned the tags NOUN+NOMINATIVE_CASE when it ends with the short vowel *u* (*ktAbu*). When I delete the final *u* in the word, I also delete the NOMINATIVE_CASE tag associated with it. This guarantees a perfect mapping between segments and tags, besides being more representative of unvocalized data. An example entry from the Arabic Treebank is shown in table 4.3 where the word *AlwlAyAt* (Eng. The-States) receives the annotation Al/DET+wilAy/NOUN+At/NSUFF_FEM_PL+u/CASE_DEF_NOM. For my purposes, I use the devocalized version of the ATB analysis: Al/DET+wlAy/NOUN+At/NSUFF_FEM_PL. This means that the word is analyzed as having the segments *Al*, *wilAy*, *At*, and *u*, with the POS tags: DET, NOUN, NSUFF_FEM_PL, and CASE_DEF_NOM respectively. Table 4.3 provides the analysis for this word as it appears in the ATB. The correct analysis is marked by a star.

```
LOOK-UP WORD: AlwlAyAt
 Comment:
 INDEX: P1W4
* SOLUTION 1: (AlwilAyAtu) [wilAyap_1]
Al/DET+wilAy/NOUN+At/NSUFF_FEM_PL+u/CASE_DEF_NOM
 (GLOSS): the + states/provinces + [fem.pl.] + [def.nom.]
 SOLUTION 2: (AlwilAyAti) [wilAyap_1]
Al/DET+wilAy/NOUN+At/NSUFF_FEM_PL+i/CASE_DEF_ACC
 (GLOSS): the + states/provinces + [fem.pl.] + [def.acc.]
 SOLUTION 3: (AlwilAyAti) [wilAyap_1]
Al/DET+wilAy/NOUN+At/NSUFF_FEM_PL+i/CASE_DEF_GEN
 (GLOSS): the + states/provinces + [fem.pl.] + [def.gen.]
 SOLUTION 4: (AlwilAyAtu) [wilAyap_1]
Al/DET+wilAy/NOUN+At/NSUFF_FEM_PL+u/CASE_DEF_NOM
```

15 *a*, *i*, and *u* are the short vowels, *o* is the sukuun, the case in which no vowel can be heard like in pauses, and *K*, *N*, and *F* are the three forms of nunation, which is usually attached to Arabic indefinite nouns.

```
 (GLOSS): the + States + [fem.pl.] + [def.nom.]
 SOLUTION 5: (AlwilAyAti) [wilAyap_1]
Al/DET+wilAy/NOUN+At/NSUFF_FEM_PL+i/CASE_DEF_ACC
 (GLOSS): the + States + [fem.pl.] + [def.acc.]
 SOLUTION 6: (AlwilAyAti) [wilAyap_1]
Al/DET+wilAy/NOUN+At/NSUFF_FEM_PL+i/CASE_DEF_GEN
 (GLOSS): the + States + [fem.pl.] + [def.gen.]
 SOLUTION 7: (AlwlAyAt) [DEFAULT] AlwlAyAt/NOUN_PROP
 (GLOSS): NOT_IN_LEXICON
 SOLUTION 8: (AlwlAyAt) [DEFAULT] Al/DET+wlAyAt/NOUN_PROP
 (GLOSS): the + NOT_IN_LEXICON
```

Table 4.3: A Sample Entry from the ATB. A word is analyzed by the Buckwalter Arabic Morphological Analyzer before the human annotators choose the correct solution among those returned by BAMA. The solution is marked by a star.

For all the experiments below I use a combination of two sections of the ATB (P1V3 and P3V1) distributed in a 5-fold cross validation setting. The total number of words in this data set is 483,245.

4.4.1.1.1. The Arabic POS Tagsets

While English POS tagging is one of the most successful applications of language processing systems, with an accuracy of around 97% (Daelemans and van den Bosch 2005: 89), Arabic POS tagging is still in the stage of research since Arabic poses different problems than those posed by English. The morphological richness of Arabic lead the Penn Arabic Treebank (ATB) (Maamouri et al. 2004) annotators to create a POS tagset that reflected the Arabic word structure. The ATB tagset is thus a composite tagset that labels the various parts of the word in terms of gender, person, tense, and number. In creating the ATB, a decision had to be made whether to treat the word as a whole or segment it for processing purposes. The composite tagset contains a large number of tags and is more of a morphological analysis than a tagset, and a POS tagger may not be very good for performing morphological analysis. The POS-tagged files are completely segmented, with the word segments along with their respective POS tags, while the parsed section of the ATB adopted a different segmentation scheme where only the segments that have a syntactic function were marked as separate segments. For example, the word *wbHsnAthm* is listed as four tokens and segmented as *w b Hsn+At hm* in the POS section, but is listed as four different entries in the parsed section: *w*, *b*, *HsnAt*, and *hm*. The reason for this is that *w*, *b*, and *hm* serve syntactic function. The entry *HsnAt* is not segmented into two since the plural suffix *At* does not serve a syntactic function. In other words, the POS section adopted segmentation, which also includes tokenization information, while the parsed section utilized only tokenization.

The ATB also has a different POS tagging scheme that fits the syntactic tokenization and is modeled after the English POS tagset. This Reduced Tagset (RTS) is not used in the ATB files but is distributed through a conversion script, and is the one commonly used in previous POS tagging studies. The RTS was meant to make the Arabic POS Tagset similar to the English POS tagset, and to make parsing easier. It includes 24 tags and it masks number, definiteness, and gender information for nouns, verbs, and adjectives, although it maintains the singular/plural distinction in nouns.[16] The rationale behind this masking is that the masked features perform inflectional, rather than derivational, functions and do not thus play a role in parsing (Maamouri et al, 2004). One peculiarity of the RTS is that the definite article, *Al*, is not marked as a separate segment

16 Although the ATB maintains a distinction between singular nouns and plural nouns, it does so only for the regular nouns whose plurals can be formed by suffixes. There is a class of Arabic nouns, which is actually productive, whose plurals are formed through templatic variation (infixation), and do not thus carry a plural marker. These are treated as singular nouns in the ATB, which does not see justified, and which, in turn, justifies the masking of the number inflections in the tagset provided by Habash et al (2005).

and is thus not assigned a POS tag. A word like *AlmSrrywn* (the-Egyptians) is written as is without separating the prefixal *Al* and is tagged as a NOUN. This does not seem to be justified since *Al* is not an inflectional affix. This also leads to data sparseness since the definite and indefinite versions of the same word are treated as two different words. It is not true that the definite article *Al* does not play a syntactic role in Arabic. Definiteness, as an example, usually marks the boundary between predicative and attributive adjectives. *Albld* ***Al****kbyr* means *The big country* while *Albld kbyr* means *The country is big.*

The Arabic Full Tagset comprises 139 segment tags, and each word is tagged with a combination of the following: (1) Lexical Tags, (2) Inflection Tags, (3) Clitic Tags and (4) Other tags. The tags and their descriptions are listed in appendix 1.

Describing the ATB POS Section

The ATB POS section can be viewed either as segments or as whole words. As segments, the ATB (the part used for the experiments described here) contains 19,019 unique segments whose frequencies vary from 114,997 (the definite article *Al*) to 1 (35.6% of the segments). The number of non-unique segments in the ATB is 806,314 including numbers and punctuation marks. These segments are classified into 139 segment tags whose frequencies vary considerably from 166,724 for the tag NOUN to eight different tags whose frequencies are below 5, and one of these eight tags, DE for DEMONSTRATIVE, seems to be an annotation error since there exist detailed

DEMONSTRATIVE tags inflected for person, number and gender, and a generic DE tag does not conform with the ATB design.

The inflection for NUMBER, GENDER, and PERSON affects nouns, adjectives, verbs, adverbs, pronouns, demonstrative, and affixes, and these inflections are well-represented in the ATB. For example, table 4.4 lists the personal pronouns and their tags from the ATB. Person pronouns can be **detached**, when they act as full-fledged words, or can be **attached**, when they act as clitics. The forms are different in each case, but the POS tag is the same. Table 4.4 lists the different inflections of the personal pronouns in Arabic. The numbers denote person, so 1 means first person, 2 second person, and 3 third person. S stands for singular, P for plural, D for dual, M for masculine and F for feminine. PRON_3MP thus stands for *pronoun for the 3rd person masculine plural.*

Detached	**Attached**	**Tag**	**Gloss**
>nA	ny	PRON_1S	I
nHn	nA	PRON_1P	I
>nt	k	PRON_2S	You
>nti	ki	PRON_2FS	You
>ntma	kmA	PRON_2D	You
>ntm	km	PRON_2MP	You
>ntn	kn	PRON_2FP	You
hw	h	PRON_3MS	He
hy	hA	PRON_3FS	She
hmA	hmA	PRON_3D	They
hm	hm	PRON_3MP	They
hn	hn	PRON_3FP	They

Table 4.4: The Arabic personal pronouns inflected for Person, Number, and Gender

The ATB encodes inflections as separate affixes. For example, the taa marbuta (*p*) is the singular feminine marker NSUFF_FEM_SG (Noun Suffix Feminine Singular), and it attaches to both nouns and adjectives. The word *mhndsp* (*mhnds+p*) (Eng. female engineer) thus receives the composite tag NOUN+NSUFF_FEM_SG, and the plural form *mhndsAt (mhnds+At*) is tagged as NOUN+NSUFF_FEM_PL (NOUN+Noun Suffix Feminine Plural).

POS Ambiguity in the ATB

If words were all unambiguous, then a look-up table would be enough to assign part of speech tags to them, but natural language is rife with ambiguity. What concerns us here is that many words have several possible parts of speech each. The Arabic orthographic form *hm* for example, has 7 tags attested in the ATB as shown in Table 4.5.

Tag	**Frequency**	**Gloss**
POSS_PRON_3MP	1097	Their
PRON_3MP	683	They
IVSUFF_DO:3MP	80	Them
PVSUFF_DO:3MP	52	Them
IV	15	To be about to
NOUN	12	Grief, sadness
PV	1	Was about to

Table 4.5: 7 attested tags for the form *hm* in the Arabic Treebank

While 7 tags for a single word may seem to be a large number, there are actually units that are more ambiguous. If we treat segments as the basis of part of speech tagging, a novel approach that I present in this thesis, we find that out of the 19,019 segment types, there are two segments with 17 POS tags each, 2 segments with 13 POS tags each, 2 segments with 9 POS tags each, 2 segments with 8 POS tags each, 4 segments with 7 POS tags each, 11 segments with 6 POS tags each, 57 segments with 5 POS tags each, 142 segments with 4 POS tags each, 620 segments with 3 POS tags each, 1033 segments (10.68%) with 2 POS tags each, and 16,155 (84.89%) are non-ambiguous in that each one of them has only one attested tag.

If we, however, treat whole words as the basis for POS tagging, which does not seem to have been done so far, we have a different picture. The total number of unique words (word types) is 56,137. The majority of these are unambiguous (92.97%). Only 6.18% of the words are two-way ambiguous, 0.7% are three-way ambiguous, 0.11% are 4-way ambiguous, only 13 words are 5-way ambiguous, 6 words are 6-way ambiguous, and 3 words are 7-way ambiguous. Table 4.6 compares ambiguity figures in whole words and in segments. This indicates that there is more ambiguity involved in tagging segments, but there is no problem of data sparseness. In whole words, the problems are different: while ambiguity is minimal, there is a large number of variations in the word forms, which means that there are many unseen words across the folds (around 8.55% per fold).

	Whole Words	**Segments**
# of types	56137	19019
unambiguous	92.97 %	84.89 %
2-way ambiguous	6.18 %	10.68 %
3-way ambiguous	0.7 %	3.26%
3-way ambiguous	0.11 %	0.75 %
5-way ambiguous	0.02 %	0.3 %
6-way ambiguous	0.01 %	0.06 %
7-way ambiguous	0.005 %	0.02 %

8-way ambiguous	NA	0.01 %
9-way ambiguous	NA	0.01 %
13-way ambiguous	NA	0.01 %
17-way ambiguous	NA	0.01 %

Table 4.6: Ambiguity in both words and segments in the Arabic Treebank

Since the main task of a part of speech tagger is to decide which of the above mentioned tags for the form *hm* is the right one given the context, it is thus useful to have a look at what kind of ambiguity occurs in the Penn Arabic Treebank. Segment-wise, the most common type of ambiguity is that between the tags NOUN and ADJ (in 718 segment types), followed by that between the tags NOUN and NOUN_PROP (389 word types). In whole word tags, table 4.7 lists the most common pairs of ambiguous tags, i.e. pairs of tags that describe the same word.

Frequency	**Tag 1**	**Tag 2**
258	DET+ADJ	DET+NOUN
195	DET+ADJ+NSUFF_FEM_SG	DET+NOUN+NSUFF_FEM_SG
181	NOUN	NOUN_PROP
175	ADJ	NOUN
150	NOUN	PV
147	ADJ+NSUFF_FEM_SG	NOUN+NSUFF_FEM_SG
142	DET+NOUN	DET+NOUN_PROP
97	ADJ+CASE_INDEF_ACC	NOUN+CASE_INDEF_ACC
82	IV3MS+IV	IV3MS+IV_PASS
65	PV+PVSUFF_SUBJ	PV+PVSUFF_SUBJ
57	PV+PVSUFF_SUBJ	PV_PASS+PVSUFF_SUBJ
57	ADJ	NOUN
50	CONJ+NOUN	CONJ+PV

49	DET+ADJ	DET+NOUN_PROP
49	CONJ+DET+ADJ	CONJ+DET+NOUN

Table 4.7: The most common pairs of ambiguous tags

We can see that the tags NOUN, NOUN_PROP and ADJ top the list of ambiguity pairs, either as single tags or in composite tags, and these three tags demand special attention.

Nouns, Adjectives, and Proper Nouns: A Long-Standing Problem[17]

The NOUN/ADJ confusion is as old as the earliest books of Arabic grammar as the Arabic grammarians did not distinguish between these two and treated adjectives as part of nouns. The traditional parts of speech in Arabic grammar books are only three: nouns, verbs and particles, and everything else is subsumed under these. The noun category is characterized by the fact that it accepts prepositions, can receive *nunation*[18], can be prefixed by *Al*, the definite article, and can be included in a vocative construction (Al-Hamalawy, 1998: 29). Adjectives meet all these requirements. Also, nouns and adjectives share the same morphological patterns (*binyanim*), and adjectives can be used as nouns and vice versa. The technical term for the comparative adjective in Arabic is "اسم التفضيل" (Al-Hamalawy, 1998: 161) which translate to "*The Noun of Preference*". Perhaps the first Arabicist to treat adjectives differently from nouns is Tammam Hassan. Hassaan (1979) proposed a 7-tag part of speech system for Arabic, in which he created two new categories for adjectives and pronouns, which traditional grammar treated as nouns.

[17] This problem may not be Arabic-specific. Dickinson and Jochim (2010) report similar confusions in English data.

[18] Nunation is an extra *n* sound at the end of indefinite nouns marked in the orthography by doubling the short vowel diacritic (so, *rajul* (Eng. man) can be either *rajulan*, *rajulun*, or *rajulin*),

Hassaan based his distinction on semantic rather than syntactic or morphological criteria as he formulated the distinctions between nouns and adjectives in that adjectives do not denote an entity while nouns do (p. 95). In reality, however, adjectives are used to name things, and it becomes very hard to tell them apart without studying the distribution. If we take, for example, the Arabic adjective *Tyb* (Eng. good), we can see that it can be used in the structure NOUN+ADJ (*wld tyb*, Eng. boy good), but we can also just say *tzwj tyb wtybp* (Eng. A good man got married to a good woman) where both *tyb* and *tybp* do not qualify nouns and can be substituted by nouns, and thus function as nouns.

Proper nouns in Arabic are usually either nouns or adjectives used as proper nouns and there are no formal distinctions. Capital letters are not used in Arabic, and there is almost no limit on what adjectives or nouns to use as proper nouns. Many of the most common anthroponyms in Arabic, e.g. *Muhammad* (Eng. most praise-worthy), *Ahmad* (Eng. more praise-worthy) and *Khalid* (Eng. immortal) are just adjectives, and are still used as such. Personal names can even be noun phrases such as *Nur Alhuda,* (Eng. the light of guidance) and *Salah Aldin* (Eng. the soundness of religion, Saladin). Even geographical names are usually ordinary nouns. For example, the Arabic word for Cairo, *AlqAhrp*, is a feminine adjective meaning *the compelling*, and it occurs in the original meaning in such expressions as "الظروف القاهرة" (Buckwalter: *AlZrwf AlqAhrp*), (Eng. *compelling circumstances* or *force majeure*.)

The treatment of adjectives and nouns in the Penn Arabic Treebank reflects the fact that annotators may not have found it easy to draw the parting line between adjectives, nouns, and proper nouns. Many adjectives are tagged as nouns, and many nouns are tagged as adjectives. An example of this is the noun *AlmqdsAt* (Eng. sanctities) in the sentence:

نعم نقول لهم هذه **المقدسات** الإسلامية والمسيحية إنما هي في أعناقنا وأرواحنا

Buckwalter: nEm nqwl lhm h*h AlmqdsAt Al<slAmyp wAlmsyHyp <nma hy fy >EnAqnA w>rwAHnA.

Eng. Yes, we say to them that these Islamic and Christian sanctities are worth our necks and souls.

POS Analysis

nEm/INTERJ nqwl/IV1P+IV lhm/PREP+PRON_3MP h`*h/DEM_PRON_F **AlmqdsAt/DET+ADJ+NSUFF_FEM_PL** AlmsyHyp/DET+ADJ+NSUFF_FEM_SG wAl<slAmyp/CONJ+DET+ADJ+NSUFF_FEM_SG <nmA/SUB_CONJ hy/PRON_3FS fy/PREP >EnAqnA/NOUN+POSS_PRON_1P w>rwAHnA/CONJ+NOUN+POSS_PRON_1P

Although the word is a feminine plural noun, preceded by the demonstrative *h`*h*, and qualified by the adjective *AlmsyHyp*, it is incorrectly tagged as an adjective. In another sentence, the word *nbyl* is tagged as an adjective, although it should be a proper noun as it is the first name of the Palestinian minister for international cooperation:

أعلن وزير التعاون الدولي الفلسطيني **نبيل** شعث أن الرئيس الفلسطيني لن يحضر القمة

Buckwalter: >Eln wzyr AltEAwn Aldwly AlflsTyny nbyl $Ev >n Alr}ys AlflsTyny ln yHDr Alqmp.

Eng. The Palestinian minister of international cooperation Nabil Shaath announced that the Palestinian president would not attend the summit.

POS Analysis

<ElAn/NOUN wzyr/NOUN AltEAwn/DET+NOUN Aldwly/DET+ADJ AlflsTyny/DET+ADJ **nbyl/ADJ** $Ev/NOUN_PROP >n/SUB_CONJ Alr}ys/DET+NOUN AlflsTyny/DET+ADJ ln/NEG_PART yHDr/IV3MS+IV Alqmp/DET+NOUN+NSUFF_FEM_SG.

Using distributional criteria for the distinction between these tags may help produce more consistent annotations[19].

4.4.1.2. Methods

Like the segmentation and vocalization experiments in the two previous chapters, I use the memory-based algorithm and the TiMBL implementation for Arabic part of speech tagging. Although TiMBL can be used directly for part of speech tagging, I preferred to use the more convenient Memory-Based Tagger (MBT). MBT was proposed by Daelemans et al (1996). MBT has three modules: (1) A lexicon module that stores all the words in the training set along with all their possible tags, (2) A Known Word Module that selects for each known word what tag it has in the context, and (3) Unknown Word Module which uses the word form to guess its tag. This is necessary since unknown words do not have an ambiguity class (a set of possible POS tags) to select from, so the

[19] It is worth mentioning that the automatic POS tagger tagged both examples correctly: *nbyl* received the tag NOUN_PROP and Al*mqdsAt* the tag DET+NOUN+NSUFF_FEM_PL.

tagger needs to guess the POS tag based on the crude affix information. The Unknown Word Module utilizes features such as word suffixes and prefixes to select a tag for the word. For example, an English word ending in the suffix -*tion* can be assigned the tag NOUN based on the suffix, and a word ending in the suffix -*ed* will receive the tag VBD (Past Verb). An Arabic word ending in -*wA* is most probably a past verb inflected for the 3rd person masculine plural, and this information can be used by MBT if the word is not in the lexicon. One advantage of using MBT is that it takes previous tagging decisions into consideration automatically. Another feature of MBT is that it allows the inclusion of user-defined features and thus allows the inclusion of as many features as appropriate for the task.

For all the experiments in this chapter, the best results were obtained with the Modified Value Difference Metric as a distance metric and with k = 25, the number of nearest neighbors. Larger numbers of nearest neighbours usually work better with MVDM, but with k larger than 25, the accuracy dropped slightly. The MBT features for known words include the two context words to the left along with their disambiguated POS tags, the focus word itself, and one word to the right along with its ambitag (the set of all possible tags a word can take). For unknown words, the features include the first five letters of the word, the last three letters of the word, the possibility of a hyphen in the focus word, the left context tag, the right context ambitag, one word to the left, the focus word itself, one ambitag to the right, and one word to the right. The IB1 algorithm produced the best results with known words, while for unknown words IGTree yielded better results.

Using 5 fold cross validation, I run three basic experiments: (1) POS tagging experiments using gold standard segmentation, (2) POS tagging using the segmentation from chapter 2, and (3) POS tagging using whole words without using any form of segmentation.

- *POS Tagging Using Gold Standard Segmentation*: In this experiment, I assume that the text to be tagged is properly segmented with no segmentation errors of any kind. I take the Penn Arabic Treebank segmentation to be perfect, in spite of the fact that it contains some segmentation errors, which is natural in such a daunting annotation effort. Unlike all the previous work of which I know, and unlike the ATB, I treat connecting conjunctions and pronouns as part of the word and not as separate tokens because they are treated as such in naturally occurring Arabic. A word is considered wrong if any of its segments is wrong. For example, the word *Alktb* (Eng. the book, correct segmentation = *Al+ktb*) is passed to the tagger as two units *Al* and *ktb,* and has to be tagged DET and NOUN respectively to be considered correct. If, however, *Al* is tagged correctly as a DET but *ktb* is tagged as a PV (Perfect Verb), which is a possible tag for the segment, then the whole word is considered wrong.

- *POS Tagging Using Automatic Segmentation*: In this experiment, I do not assume gold standard segmentation. For each one of the five folds, the test set is segmented using the relevant training set, using the memory-based word segmenter developed in chapter 2 of this thesis, and then the same training set is used as the POS tagging training set. For example, the data are divided into 5 sections, each containing 20% of the whole data set, so, when running experiments, I test on section 1 in both segmentation and POS experiments, and train on a combination of sections 2, 3, 4, and 5. This guarantees that there is no overlap between the training set and the test set in either segmentation or POS tagging. The accuracy in this experiment is forecast to be lower than the accuracy in experiment 1, which uses gold standard segmentation, since each segmentation error will definitely lead to at least one POS tagging error. For example the word *Alrjl* (Eng. the man) should be segmented as *Al+rjl*, (DET+NOUN), but if the segmenter does not recognize it as two segments and segments it as *Alrjl*, then this cannot be tagged properly. In this experiment, the segmentation accuracy sets the upper bound for POS tagging: If per word segmentation accuracy is 98%, then POS tagging accuracy cannot go beyond this limit.

- *Whole Word POS Tagging*: In this experiment, words are treated as single units, and no tokenization or segmentation is used. The word *wbHsnAthm* (Eng. and with their virtues) is used as only one unit and its tag (CONJ+PREP+NOUN+NSUFF_FEM_PL+POSS_PRON_3MP) is a composite one that consists of several sub-tags. The tagger's task is to find the whole composite tag that corresponds to the word. This tagging scheme is novel, especially since I re-connect the tokens into words and do not give a word more than one entry as the ATB does. The potential problems with this tagging scheme are: (a) the number of tags is very large as it reaches 993 tags in our data set, (b) many tags occur only once in the training set, and (c) the data is sparse (56,137 word types compared to 19,019 segment types).

To summarize the difference between whole word tagging and segmentation-based tagging:

- With whole word tagging, there is sparseness of data, but less ambiguity.
- With segment-based tagging, data sparseness is not an issue, but there is increased ambiguity.
- If I adopt segment-based tagging, words have to undergo a process of segmentation, which is not perfect.

4.4.1.3. POS Evaluation

In order to make POS evaluation as informative as possible, I not only give general accuracy, but also accuracy on known versus unknown words and on ambiguous versus unambiguous words:

Known Words are those words in the test set that are also in the training set. The accuracy on known words can be a good representative of accuracy when both the test set and the training set belong to the same domain where few words are introduced in the text to be tagged. ***Unknown words are those words in the test set that are not in the training set***. Unknown words can give an indication of how a specific tagging system may perform in the real world in the case when we do not know the nature of the input text. Of course, any new text will have known and unknown words and the ratio between these can be a major determinant of POS tagging quality.

Ambiguous Words are those known words that can have more than one part of speech tag. By providing evaluation for the tagger performance on ambiguous words, we can have some idea of how the tagger will perform on difficult words since any tagger can perform perfectly on unambiguous words. Some part of speech tagging researchers used to give accuracy numbers on ambiguous words only, which was lower than general accuracy (Manning and Schütze, 1999: 371).

Unambiguous words are those known words that have a single POS category each. Unambiguous words can usually be composed of segments that are themselves ambiguous, but the combination of which can only result in one unambiguous word, for example, the word *wbHsnAthm* (Eng. and with their virtues) (CONJ+PREP+NOUN+NSUFF_FEM_PL+POSS_PRON_3MP) as discussed in the introduction to this chapter.

Setting the Baseline

In order to set a baseline, I use the most basic experiment: POS-tagging using whole words and choosing the most frequent tag for each word in the training set. If the word is not in the training set, I assign the tag NOUN, which is the most common open class tag in the training set. Averaged over 5 folds of cross validation, this scheme gives a general accuracy of 77.02%, with an accuracy of 3.67% on unknown words (which have been assigned the NOUN tag). This low accuracy on unknown words can be explained in terms of tag complexity and sparseness: while the tag NOUN is the most common tag across the five folds, nouns also have such tags as DET+NOUN and NOUN+NSUFF_FEM_SG.

Evaluation Metrics

For all the experiments in this chapter, I use the standard accuracy in terms of the number of correctly tagged words divided by the total number of words. While this is straightforward for the whole word experiment, the other two experiments use segments instead of whole words during the tagging process. For both the automatic segmentation experiment and the Gold Standard Segmentation, I re-attach the segments into whole words before running the evaluation in order to avoid the alignment problem resulting from erroneous segmentation. For example, the proper noun *knt* (Eng. Kent) can be segmented as *kn+t (verb+pronoun, Eng. you were).* This leads to a one-segment word in the gold standard to be assigned two segments, and thus two POS tags instead of one.

To give a concrete example of evaluating segmentation-based tagging, the phrase *jrydp AlHyAp AlsEwdyp* is segmented in the gold standard as *jryd+p AlHyAp Al+sEwdy+p*, where the word AlHyAp is only one segment, but the segmenter incorrectly produces the word with three segments: *Al+HyA+p*, and the part of speech tagger assigns it the tags DET+NOUN+NSUFF_FEM_SG, instead of the simple NOUN_PROP in the gold standard. I re-attach the segments in the whole word again to produce tags that can be evaluated against the gold standard as can be seen in table 4.8, where the tagging accuracy for this string of words is 2/3, or 66%.

Word	GS POS Tag	MBT POS Tag
Jrydp AlHyAp	NOUN+NSUFF_FEM_SG NOUN_PROP	NOUN+NSUFF_FEM_SG DET+NOUN+NSUFF_FEM_SG

AlsEwdyp	DET+ADJ+NSUFF_FEM_SG	DET+ADJ+NSUFF_FEM_SG

Table 4.8: Re-combing words for evaluation.

This scheme allows me to both maintain segment boundaries and re-combine words for word-level evaluation. A word is considered correctly tagged if and only if all its segments are correctly tagged. Since segment accuracy is not very meaningful, I will focus on full words in this chapter.

4.4.1.4. Results and Discussion

In this section, I present results of POS tagging experiments on the unvocalized version of the ATB. Most of the experiments below are performed through five fold cross validation, and I will make it clear when any experiment is performed using only one fold.

Table 4.9 below presents the results obtained in a five fold cross validation scheme on known and unknown words with three experiments: **Whole Words, Gold Standard Segments, and Automatic Segmentation.**

Experiment	**Overall Accuracy**	**Known Words**	**Unknown Words**
Whole Words	94.74%	96.62%	74.64%
Gold Standard Segments	94.91%	95.90%	84.25%
Automatic Segmentation	93.47%	95.57%	71.06%

Table 4.9: POS tagging results across 5 folds of cross validation and three experimental settings

When I use whole words in the experiments, i.e. I perform tagging on the orthographic units as they appear in naturally occurring written Arabic without performing any sort of word segmentation, I obtain an overall accuracy of 94.74%, an accuracy of 96.61% on known words and an accuracy of 74.64% on unknown words.

Using gold standard segments, where I take the segmentation scheme provided with the ATB and pass the segments directly to the POS tagger, results in an overall accuracy of 94.91%, with a 95.90% accuracy on known words and an 84.25% accuracy on unknown words.

Automatic segmentation, as explained in chapter 2, using the best settings for segmentation, yields an overall accuracy of 93.47%, with a known word accuracy of 95.57% and an unknown word accuracy of 71.06%.

The results show that gold standard segmentation produces the best results in overall accuracy, but since gold standard segments are not available in real life experiments, I will focus the discussion on the other two settings: Whole Word tagging, and automatic segmentation tagging. While Whole Word tagging produces better overall accuracy than automatic segmentation tagging, we can see from the table that this is basically due to its very high performance on known words. On unknown words, on the other hand, while whole word tagging outperforms automatic segmentation tagging, its accuracy is much worse than that by the gold standard segment tagger. The whole word approach owes its high accuracy to the fact that it does not suffer from problems in an earlier stage of the pipeline as is the case with the segmentation-based tagging. Also, the inclusion of the first and last letters of the word as features helps mitigate the data sparseness problem, and this works positively even with unknown words.

Ambiguous Words

If ambiguity affects the accuracy of POS tagging, then one would expect more ambiguous words to be susceptible to more tagging errors. This is in fact true. Not only is there a difference between ambiguous and unambiguous words in terms of POS tagging accuracy, there is also a difference in terms of the number of tags a word has. The higher the ambiguity, the lower the accuracy. This is true in both the segmentation-based approach and the whole word approach, although the whole word approach is generally better in handling ambiguous words, as is evident from table 4.10. The first column of table 4.10 is the number of tags per word, as attested in the training set, the second column is the accuracy according to the whole word tagger, and the third column is the accuracy of the segmentation-based tagger.

# of possible tags per word	**Whole Word Tagger Accuracy**	**Segmentation-based tagger Accuracy**
1	98.78	98.29
2	92.78	91.98
3	89.71	89.51
4	83.35	83.83
5	84.59	67.75

6	72.08	54.92
7	74.84	56.28

Table 4.10: Ambiguity Effect on POS Tagging in both the Whole Word Approach and the Segmentation-based Approach

While unambiguous words score 98.78% on words in the whole word approach across five folds of cross validation, 2-way ambiguous words score 92.78%, losing 6 percentage points. With 5-way ambiguous words, we start to see a large difference between the segmentation-based approach and the whole word approach in favor of the latter, which scores 84.59% versus 67.75% when I use segmentation. The reason for this is that for a word to be correct, all its constituent segments must be correct, and in the case of poly-ambiguous words, the segments may be ambiguous themselves as well. Figures 4.1 and 4.2 present graphs of ambiguity effect in both word-based and segment-based POS tagging experiments.

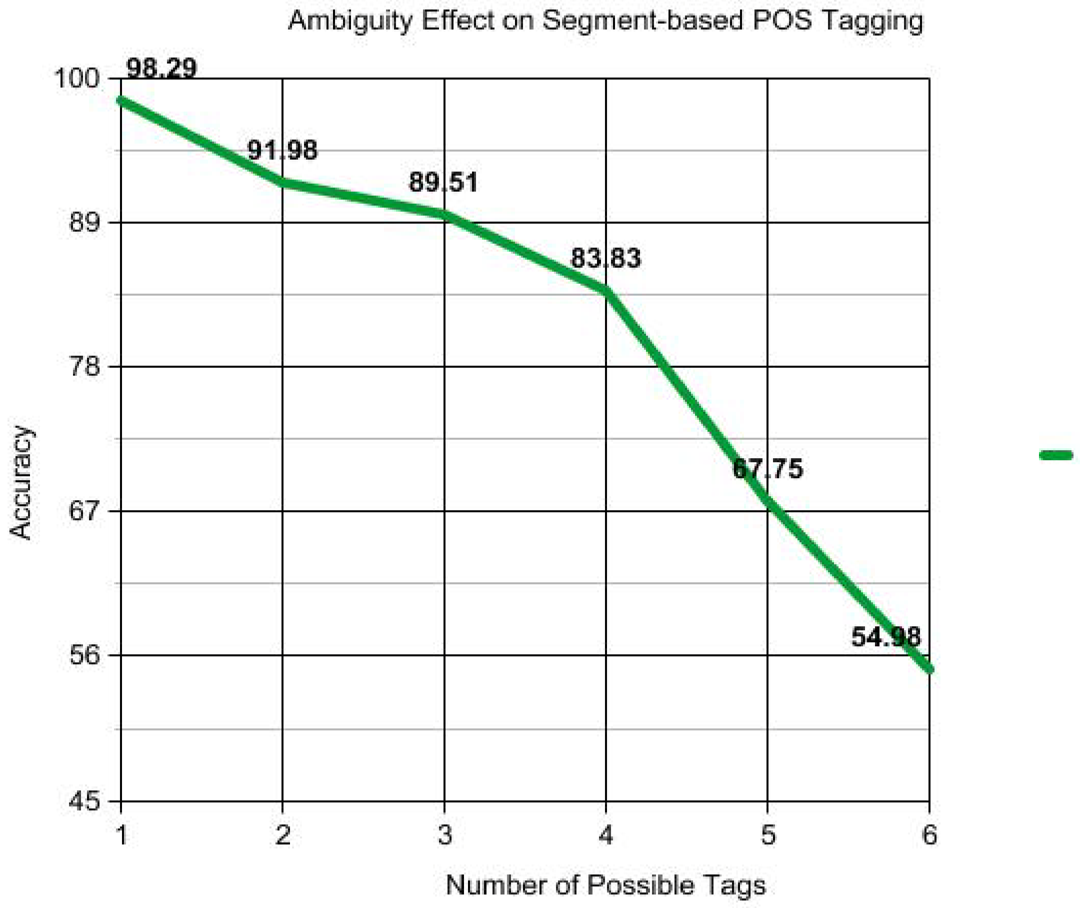

Figure 4.1: Ambiguity effect on segment-based POS tagging

To explore the relationship between ambiguity and accuracy, I use the statistical measure of correlation. Correlation measures the degree to which two variables are related together. There is a near-perfect negative correlation of -0.96 between the number of possible tags per word and the accuracy of the tagger in both tagging schemes, which indicates that ambiguity and accuracy go in opposite directions. While correlation does not necessarily imply causation, it gives an idea of how the two phenomena, ambiguity and accuracy, may be related.

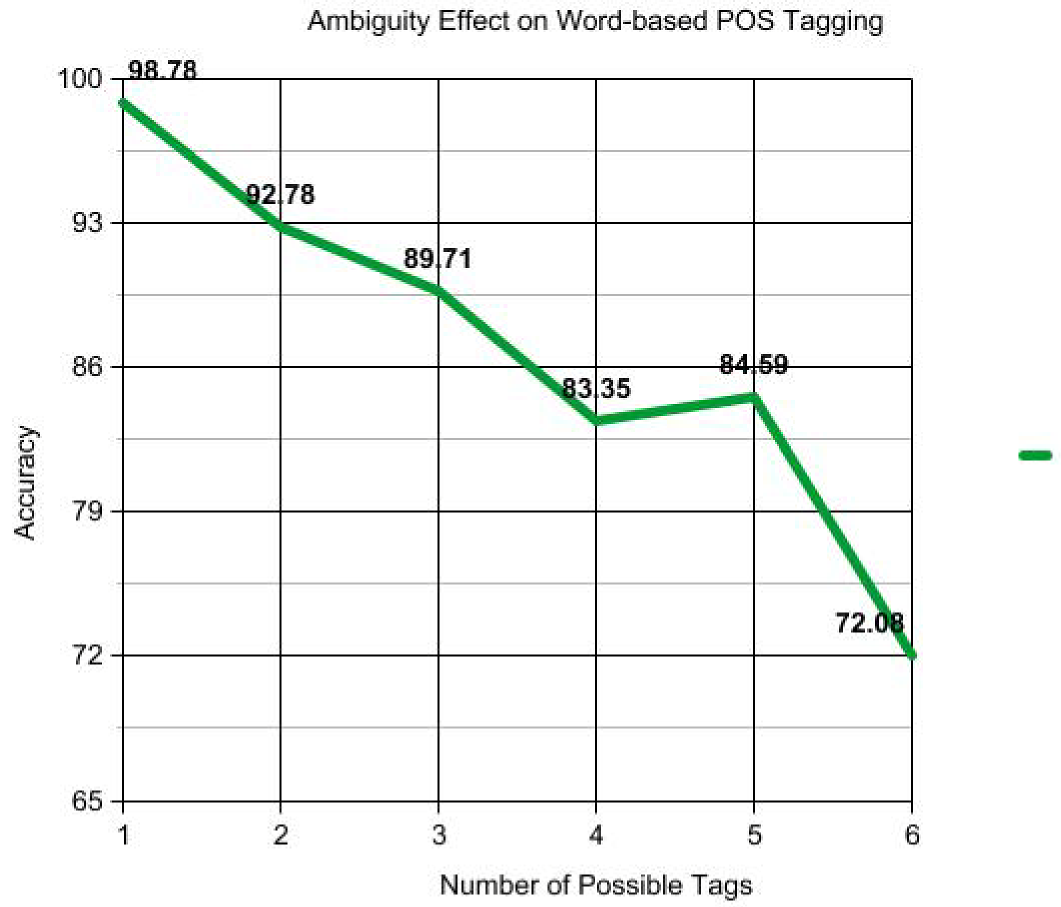

Figure 4.2: Ambiguity effect on word-based POS Tagging

4.4.1.5. Error Analysis

A study of errors in POS tagging requires a confusion matrix. A confusion matrix is a contingency table in which each cell (x, y) “contains the number of times an item with correct classification x was classified by the model as y” (Jurafsky and Martin, 2009: 156). I am focusing here on the most common errors with an attempt to find the reason behind those errors from both a segment-based perspective and whole-word-based perspective.

Error Analysis: The Segment-based Approach.

There are 1056 confusion pairs, only 18 of which occur more than 1% of the time. Table 4.11 displays the 10 most common of these.

OriginalSegmentTag	ConfusedSegmentTag	Percentage
ADJ	NOUN	9.25
NOUN	ADJ	8.92
NOUN_PROP	NOUN	7.90
NOUN	NOUN_PROP	4.65
PV	NOUN	2.61
None	NOUN_PROP	2.37
NOUN_PROP	ADJ	2.25
IV	None	2.16
NOUN	PV	2.00
IV3MP	IV3MS	2.03

Table 4.11 Most Common Errors in the segmentation-based approach

As can be seen from table 4.11, segments whose correct tag is ADJective are incorrectly tagged as NOUNs in 9.25% of all errors, and NOUNs tagged as adjectives constitute 8.92% of all errors. The reason for this is that the ADJ/NOUN distinction is not clear as outlined in section 4.1.1.4 above. This means that 18.17% of all errors can be attributed to the NOUN/ ADJ confusion.

The third and fourth most common confusions are those between nouns and proper nouns, and both together constitute 12.55% of all errors. Also, proper nouns can be derived from adjectives and this is the reason behind the confusion between both. Proper nouns tagged as adjectives constitute 2.25% of all errors.

We can also see that perfect verbs (PV) are incorrectly tagged as nouns in 2.61% of all errors and nouns are tagged as perfect verbs in 2% of all cases. This is a confusion that can potentially be solved through vocalization, as many forms can be either a noun or a perfect verb, but differ in pronunciation. An example of this is the form *Drb*, which can be a noun (Eng. hitting, type) or a verb (Eng. to hit).

An interesting type of error is that resulting from erroneous segmentation, indicated by *None* in Table 4.11. Segmentation errors have a different effect on unknown words than on known words. A closer look at the results for unknown words in segmentation-based tagging shows that 59.68% of the tagging errors are direct results from incorrect segmentation decisions. In comparison, for known words, only 6.24% of the incorrectly tagged words are also ill-segmented. This means that even though the quality of the segmenter is very high, the errors still harm the POS tagging step.

Error Analysis: The Whole Word-based Approach.

The confusion matrix for whole word tagging has 2895 confusion pairs, three times as many as those in the segmentation-based approach, and only 16 of them occur more than 1% each. Table 4.12 lists the 10 most common confusion pairs.

Original Tag	**Confused Tag**	**Percentage**
DET+NOUN_PROP	DET+NOUN	2.85
DET+NOUN	DET+ADJ	2.82
NOUN_PROP	NOUN	2.79
NOUN	NOUN_PROP	2.68
DET+ADJ	DET+NOUN	2.46
ADJ	NOUN	1.93
PV	NOUN	1.87
NOUN	ADJ	1.87
DET+NOUN+NSUFF_FEM_SG	DET+ADJ+NSUFF_FEM_SG	1.69
DET+NOUN	DET+NOUN_PROP	1.69

Table 4.12: Most Common Errors in the Whole Word POS tagging approach

The confusion pairs in Table 4.12 show that the top confusions still involve nouns, adjectives, and proper nouns, albeit in a more fine-grained manner. The confusion between DET+NOUN_PROP and DET+NOUN now tops the list with the low percentage of 2.85%, which reflects the effect of the large number of tags. Second on the list is the confusion between DET+NOUN and DET+ADJ with 2.82%. The only non-nominal confusion pair in this list is the confusion between Perfect Verbs (PV) and nouns, which is also present in the segment confusion (table 4.11).

4.4.1.6. Partially Correct Tags

One advantage of the segmentation-based POS tagging approach is that it is more likely to give partially correct tags, especially when the errors do not have to do with erroneous segmentation. Partially correct tags contain elements of the gold standard tags in the same order of occurrence. For example, if a word whose correct tag is DET+NOUN+NSUFF_FEM_PL is assigned the tag DET+ADJ+NSUFF_FEM_PL, it can be considered partially correct.

It is not unlikely for the Whole Word tagging scheme to give completely unacceptable tags. A word whose correct tag is ADJ+CASE_INDEF_ACC has, for example, been tagged as CONJ+NOUN+CASE_INDEF_ACC, and a word with the tag ADJ has been tagged as NOUN+NSUFF_FEM_SG+POSS_PRON_1S, and it is obvious that these tags, especially the latter one, are completely unrelated.

The errors made by the segmentation-based tagger are not as severe. In fact, if we consider the segment tags within the composite ones, we find that among the errors made by the segmentation-based tagger, 34.92% of the segment tags are correct, compared to 32.12% in the case of Whole Word Tagging. While the difference may not substantial, when a certain application requires more accuracy with the segments, a segmentation-based tagger may be preferred although the Whole Word Tagger generally performs better.

4.4.2. The Habash and Rambow Tagset

Since a smaller tagset is more efficient, i.e. faster to process, than a larger one (Brants, 1995; Elhady and Al-Toby, 2009), it would be desirable to reduce the size of tagsets, especially when the default tagset is very large. Brants (1995) introduced a method of clustering tagsets by which tags that have the same probability distribution are combined as one compound tag given that the same lexical item may not have both tags in the training set. For example, if there is not a lexical item that is ambiguous between the tags VERB and Preposition, then we can have a combined tag, VERB_PREPOSITION for all the verbs and prepositions in the dataset. This combination method is lossless in the sense that the original tags can be recovered after the tagging process by using word/tag lists. Brants (1995) reported a small, possibly insignificant improvement in tagging accuracy. Elhady and El-Toby (2009) tried a number of reduced tagsets in the task of (English) document classification and reported that the most reduced tagset resulted in the same accuracy with improved efficiency. Their most reduced tagset included only four tags: Article, Adjective, Verb and Noun. Another appealing characteristic is that reduced tagsets are less prone to errors by both humans and computers (Dickinson and Jochim, 2010).

The ATB full tagset has many morphological details. One main reason for using the full tagset is that almost any tagset can be derived from it owing to the large amount of information it provides. For example, when we tag the word *llHkwmp* (Eng. for the government) as PREP+DET+NOUN+NSUFF_FEM_SG, we are giving enough information for the characterization of the class of this word, and we can, by reducing the amount of information provided, generate tagsets each focusing on specific aspects. This can help obtain better accuracy as the number of tags may have an effect on POS tagging accuracy. One possible way is to remove inflection tags and focus only on the tokens. The token tags can be as detailed or brief as desired.

The Habash and Rambow tagset comprises 15 tags and handles only tokens without taking inflections into consideration. The 15 tags are listed in table 4.13.

We can see that this tagset is even more reduced than the Reduced Tagset (24 tags) distributed with the Penn Arabic Treebank. The RTS draws distinctions between singular nouns and plural nouns, and between perfective, imperfective, imperative, and passive verbs. While the distinction between singular nouns and plural nouns is a worthy cause, the full tagset does not make the distinction, possibly due to the fact that many of the plurals are broken plurals that cannot be distinguished by means of affixes from singular nouns. For example, the word *>mAkn* (Eng. places), the plural of *mkAn,* is simply tagged as NOUN. The plural form does not have any affixes distinguishing it from singular nouns. This gives justification to the omission of this distinction in the new tagset, but the omission of the distinction between the different types of verbs is harder to justify.

1	**V**	Verb
2	**N**	Noun
3	**PN**	proper noun
4	**A J**	Adjective
5	**A V**	adverb

6	**PRO**	nominal pronoun
7	**P**	preposition or particle
8	**D**	Determiner
9	**C**	Conjunction
10	**NEG**	negative particle
11	**NUM**	Number
12	**AB**	Abbreviation
13	**I J**	Interjection
14	**PX**	Punctuation
15	**X**	Unknown

Table 4.13: The Habash and Rambow Extra-Reduced tagset

In this section, I investigate whether the Habash and Rambow (2005) Extra Reduced Tagset, which is derived from the full tagset through the masking of some information, will yield better results. The Habash and Rambow tagset is not a lossless one as the distinctions removed cannot be recovered. The main question here is: Is it better to (a) derive the Habash and Rambow tagset through a conversion script after performing tagging with the full tagset, or (b) treat the Habash and Rambow tagset as an original tagset?

I will present 3 experiments using the Habash and Rambow tagset to investigate whether reducing the amount of information in the tagset yields better accuracy. These three experiments are as follows:

1. **Derive from Segmentation-based Tagging (Seg-derived)**: In the Seg-derived experiment, I tag the text first using the segmentation-based approach. I then derive the Habash and Rambow tag, and its associated word tokenization from the segmentation-based tagging scheme as outlined in section 2.5.4. For example, the word *wnqAbAt* (Eng. and syndicates) is assigned the segmentation w+nqAb+At and the composite POS tag CONJ+NOUN+NSUFF_FEM_PL. When this word is transformed to the new tagging style, it becomes *w+nqAbAt* (with no marking for the inflection) and the tag CC+N (Conjunction+Noun). This makes it feasible to present and measure word-based accuracy as well as token-based accuracy and tokenization accuracy. If, however, we are not interested in the tokenization process itself, and all we need is obtain tags for whole words, using the Habash and Rambow tagset, we can do so through experiments WW-derived and WWOrig below.

2. **Derive from Whole Word Tagging (WW-derived)**. In this experiment, I tag the text using the whole word approach. I then derive the new tagset from the already tagged whole word files. One downside of this is that the scheme will take care only of the tags, without considering tokenization since the words are not segmented in the first place. The advantage of this approach is that it is simple, since I obtain the new tags as a by-product of another experiment (WW Tagging) without any need of segmentation or morphological analysis.

3. **Original Whole Word Habash Tagging (WWOrig)**. In this tagging scheme, I transform the original Penn Arabic Treebank to the Habash and Rambow tagset,

and then perform the tagging on the new tagset directly using MBT with the Whole Word settings, since we are dealing with whole words rather than segments. This tagset comprises 95 composite tags of which 19 occur fewer than 5 times in the whole corpus. The most frequent tag is the P tag, which stands for particle, and it constitutes 15.05% of all tags, followed by N (13.90%), D+N (12.52%), PX (12.19%), PN (7.40%), and D+JJ (6.89%).

Table 4.14 compares the annotation of a sentence from the ATB using the full tagset and the Habash and Rambow tagset. We can see that all the inflection tags have been removed from nouns and adjectives, the distinction between imperfective, perfective, and passive is no longer available, and that both the preposition and the subjunctive conjunctions are now tagged as P (Particle). One other observation is that the word *t$ryn* (Eng. October), which is usually tagged as a proper noun is tagged as a NOUN. This is a common problem in the ATB annotation. In the same sentence the word *Aktwbr* (Eng. October) is tagged as a Proper Noun.[20]

Word	**Full Tag**	**Habash Tag**
w>$Ar	CONJ+PV	CC+V
byAn	NOUN	N
>Sdrh	PV+PVSUFF_DO:3MS	V+PRO

[20] Each month has two names in Arabic. Which one to use largely depends on the geographical location. In the example cited here, both names of the month of October are used in the same sentence. Both naming schemes suffer from annotation inconsistency.

"	PUNC	PX
mntdY	NOUN	N
>yAm	NOUN	N
mdn	NOUN	N
m$lwlp	ADJ+NSUFF_FEM_SG	JJ
"	PUNC	PX
<lY	PREP	P
>n	SUB_CONJ	P
AldEwp	DET+NOUN+NSUFF_FEM_SG	D+N
>Tlqt	PV_PASS+PVSUFF_SUBJ:3FS	V
"	PUNC	PX
fy	PREP	P
<TAr	NOUN	N
mqATEp	NOUN+NSUFF_FEM_SG	N
Al{ntxAbAt	DET+NOUN+NSUFF_FEM_PL	D+N
Alr}Asyp	DET+ADJ+NSUFF_FEM_SG	D+JJ
fy	PREP	P
22	NUM	NUM
t$ryn	NOUN	N

Al>wl	DET+ADJ	D+JJ
/	PUNC	PX
>ktwbr	NOUN_PROP	PN
"	PUNC	PX
.	PUNC	PX

Table 4.14: A sentence annotated with both the full tagset and the Habash and Rambow tagset

Table 4.15 presents the results of the three experiments above. In experiment Seg-derived, I also present the results for tokenization accuracy, and token tag accuracy. This is not done with the other two experiments since they do not involve tokenization. All the results are averaged across five folds of cross validation[21].

Experiment	**Whole Word Accuracy**	**Token Tagging accuracy**	**Tokenization Accuracy**	**Accuracy on Correctly Tokenized Words**
Seg-derived	94.86%	96.41%	99.36%	96.50%
WW-derived	96.00%	NA	NA	NA
WWOrig	95.89%	NA	NA	NA
Habash and Rambow (2005)	NA	98.1%, 96.5%	NA	NA

Table 4.15: Three Experiments with the Habash and Rambow tagset

[21] The results on segmentation-based tagging are not repeated here since no comparsion with tokenization-based tagging is intended or desired.

As can be seen from Table 4.15, experiment WW-derived yields the best results in terms of whole word accuracy as it reaches 96.00% compared to 95.89% when I run the original tagging experiments (WWOrig), and 94.86% when I derive the tags from segmentation-based tagging. This proves that whole word tagging using the full tagset is still the best scoring experiment, and that there is no need to have two separate taggers for the full tagset and the Habash and Rambow tagset as the latter can be deterministically derived from the former with even higher quality than building a separate tagger for each. It is clear that the tagset with more information yields better results than the tagset with reduced information. The reason for this may be that morphological complexity can better be paired with complex tags. For example, when the tagger is faced with the word *AlHsnAt* (Eng. the alms), for example, a tag like NOUN may not be enough to account for the distribution and surface features while a tag like DET+NOUN+NSUFF_FEM_PL can be a better representative of the word's grammatical category.

The Seg-derived experiment, in spite of its low performance on whole words, as is the case with the original segmentation-based experiment, still has the advantage of the ability to pair tags with their respective tokens, and this is useful in parsing experiments. The experiments also has a high accuracy on tokens as it reaches 96.41%, which is not much worse than the results obtained by Habash and Rambow (98.1%) in spite of the fact that I do not use gold standard tokenization while they do. While the dataset and the split into training and test sets is different, the fact that I use five-fold cross validation and the high results I obtain indicate that tagging without gold standard tokenization is viable.

4.4.3. Vocalization Effect on POS Tagging

Vocalization is the process of restoring short vowels which are usually missing in written Arabic. One of the major advantages of vocalized text is that it is less ambiguous than unvocalized text. For example, while the unvocalized word *ktb* can be a verb, or a noun, when vocalized, the word is unambiguous. When the vocalization is *kutub*, it is a noun, otherwise it is a verb.

This raises the question of whether vocalization can help POS tagging since at least one of the confusion pairs in POS tagging, that between NOUN and PV, should be solvable through vocalization.

In this section, I introduce two sets of experiments to test the effect of vocalization on part of speech tagging: (a) Gold Standard Vocalization effect on POS Tagging, and (b) Automatic vocalization effect on POS Tagging.

- **Gold Standard Vocalization**: The purpose of this experiment is to test the effect of vocalization in principle. By vocalization I mean word-internal vocalization and not case endings since the automatic vocalization experiments have no access to the case system.
- **Automatic Vocalization**: In this experiment, I simulate real world settings and check whether vocalization can help part of speech tagging. I use the best settings that produced the lowest Word Error Rate on vocalization in chapter 3, i.e. characters + stemmed context + previous decisions. The Word Error Rate in this experiment is 6.72%.

Both experiments will be carried out using the best scoring POS tagging settings:

whole words with the relevant settings. The accuracy will be counted across five folds of cross validation.

I have excluded segmentation-based tagging from this experiment for two reasons:

- Whole Word tagging has proved to be a better tagging strategy at all levels: known words, unknown words, and consequently, overall accuracy.
- We have seen in Chapter 3 that automatic vocalization harms segmentation accuracy. While unvocalized segmentation scores an accuracy of 98.34%, vocalized segmentation scores 91.01%, which is much worse than unvocalized segmentation. Since segmentation accuracy sets the ceiling for POS tagging accuracy, POS Tagging accuracy using automatically vocalized segmentation cannot go beyond 91%.

Table 4.16 presents the results of using vocalization for POS tagging using the whole word approach. I do not give results for unknown vs. known words in the automatic vocalization cells since known and unknown words are different from those in the gold standard vocalization, due to the errors introduced through vocalization, and are thus not comparable.

Experiment	**Known**	**Unknown**	**Overall**
GS Vocalization	97.54%	78.08%	95.72%
Automatic Vocalization	NA	NA	92.59%
Unvocalized	96.62%	74.64%	94.74%

Table 4.16: Vocalization Effect on Whole Word POS tagging

When I use gold standard vocalization, I obtain an overall accuracy of 95.72% on all words, with known words scoring 97.54% and unknown words scoring 78.08%. This is obviously better than the results obtained on unvocalized text (94.74%, 96.62%, and 74.64% respectively). This proves that vocalization helps in part of speech tagging for both known words and unknown words. The reason for this is possibly the ambiguity rate in vocalized text is less than the ambiguity rate in the unvocalized text. In an examination of fold 1, I found that the average number of tags per word is 1.04 in the vocalized text versus 1.08 in the unvocalized text, and the number of tags per ambiguous word is 2.08 in the vocalized text versus 2.14 in the unvocalized text. This is only natural since vocalization disentangles some forms that would otherwise be written in the same way.

Using automatic vocalization in part of speech tagging is not expected to yield good results in spite of the fact that vocalization module introduced in chapter 3 yields state of the art accuracy. As we have seen with segmentation, even a small decrease in segmentation quality harms POS tagging, and the same is expected with vocalization here. I will, however, examine the effect of vocalization on part of speech tagging to gain a better insight into the process.

Averaged across five folds of whole word experiments, part of speech tagging accuracy with automatic vocalization is 92.59% which is much lower than the results obtained by the unvocalized approach. This indicates that while performing vocalization helps POS tagging in principle, as witnessed by the gold standard vocalization results, automatic vocalization needs to be extremely accurate to produce a similar effect. If we look deeper at the results of this experiment, we find that part of speech tagging accuracy among correctly vocalized words is 96.59%, which indicates that those words are generally easier than the rest of the words. The accuracy drops significantly on incorrectly vocalized words to reach 49.09%. The percentage of words that are both correctly vocalized and correctly tagged is 88.46%.

These results make it clear that incorrect vocalization leads to incorrect part of speech tagging more than 50% of the time.

4.4.4. Training Size Effect on POS Accuracy

In order to examine the effect of the size of the training set on the accuracy of part of speech tagging, I train the MBT tagger using various amounts of data using the same training settings and the same test set in both whole word tagging and segmentation-based tagging. The test set consists of fold one, and it contains 96,649 words. The training set starts with the first 10,000 words of the training set, then 1/10 of the full data size in the fold, and keeps increasing by 1/10 of the data. Table 4.17 presents the various results obtained. Column 1 in Table 4.17 shows the number of words in the training set, column 2 shows the overall accuracy using the Whole Word approach, column 3 shows the accuracy obtained using the segmentation-based tagger, and the last column presents the percentage of unknown words in the test set with respect to the specific training set used. Figure 4.3 is a scatter plot of the relationship between the number of words in the training set and the accuracy on the part of speech tagger using the Whole Word tagger while Figure 4.4 is a scatter plot for the segmentation-based tagger. The percentage of unknown words is presented here since it affects the accuracy of POS tagging, and since it changes according to the size of the training set.

Number of Words	**WW POS Accuracy**	**Segmentation-based POS Accuracy**	**% unknown**
10000	85.63%	87.40%	37.20%
38659	91.19%	91.21%	22.34%
77318	93.09%	92.50%	16.40%
115977	93.67%	93.01%	13.51%
154636	93.94%	93.28%	11.91%
193295	94.16%	93.17%	10.80%
231954	94.28%	93.40%	9.94%
270613	94.36%	93.47%	9.31%
309272	94.51%	93.48%	8.67%
347931	94.99%	93.52%	8.24%
386590	95.06%	93.53%	7.85%

Table 4.17: Training Size Effect on POS Tagging

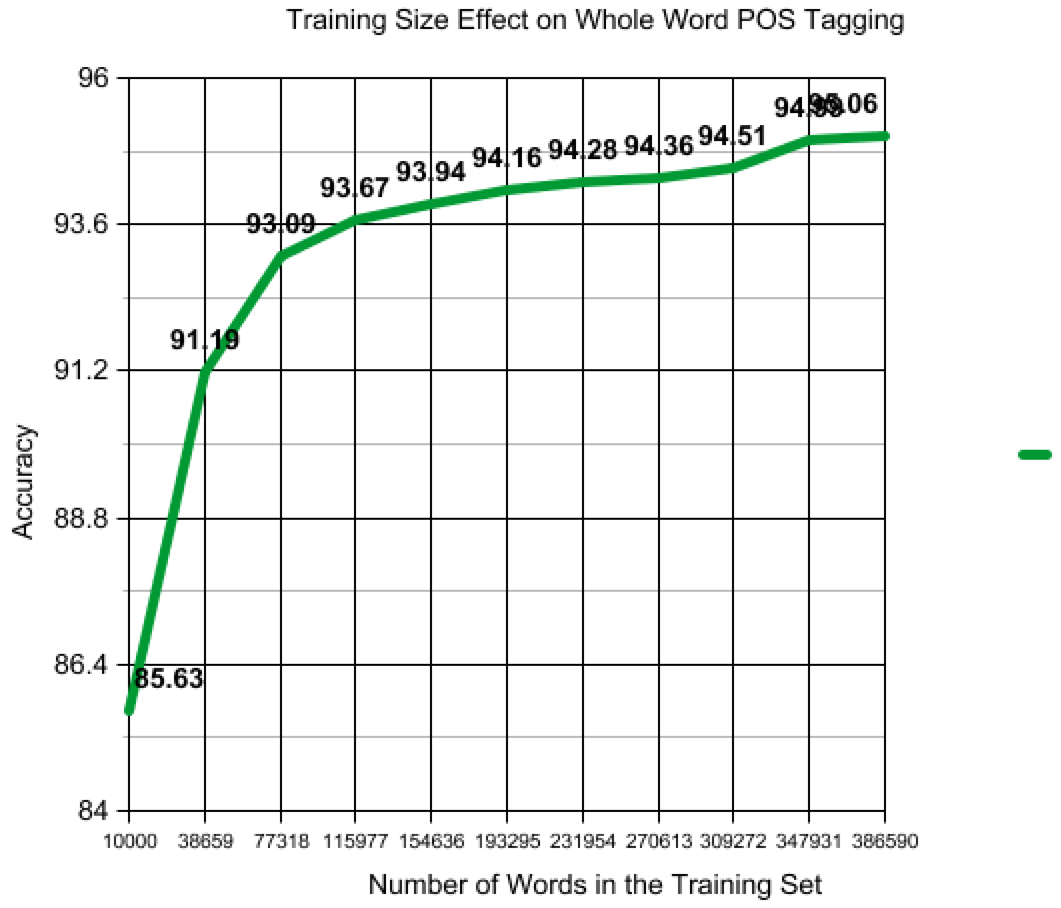

Figure 4.3: Training Size Effect on Whole Word POS Tagging

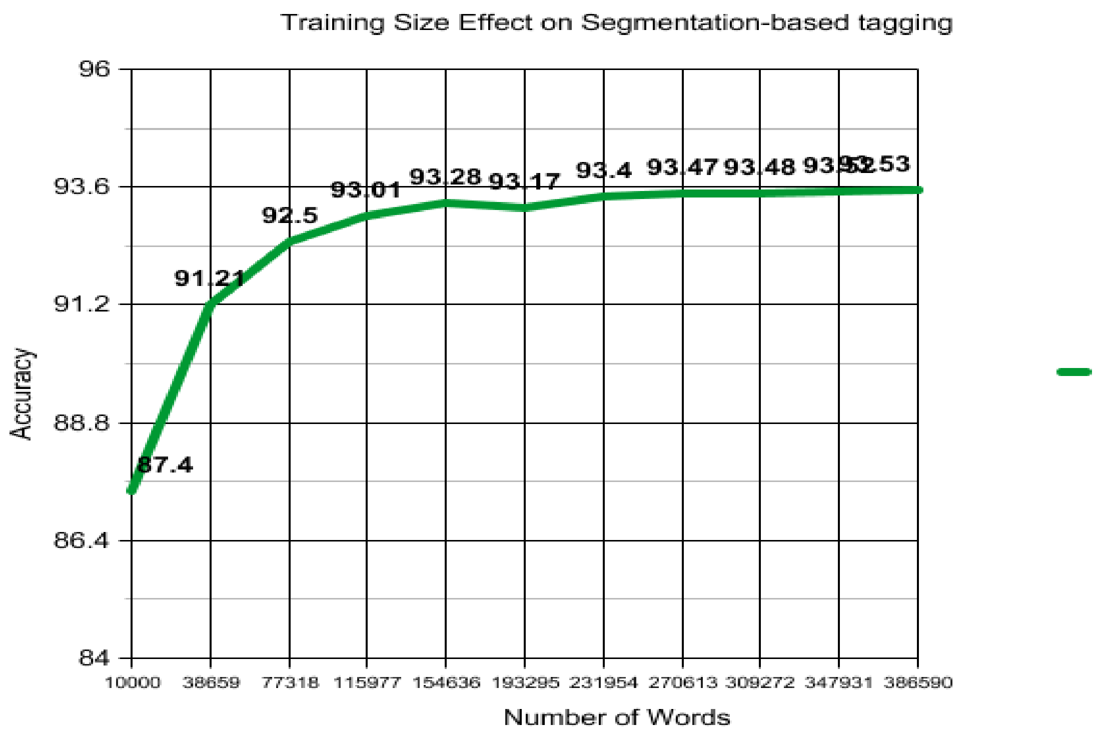

Figure 4.4: Training size effect on segmentation-based POS tagging

We can notice that segmentation-based tagging works better with smaller training sizes. With 10,000 words of training, the segmentation-based approach is almost 2 percentage points better than the whole word approach (87.4% vs. 85.63%), but whole word tagging soon catches up even with only 3/10 of the training size, after which it keeps outperforming the segmentation approach.

While the size of the training data has a discernible effect on the quality of POS tagging in general, examining the percentage of unknown words makes it clear that it is this percentage that has the larger effect. In our experiment here, adding more words led to a reduction in the number of unknown words from 37.20% with 10,000 words of training to 7.85% with the full training set. In fact, there is nearly a perfect inverse correlation of -0.99 between the tagging accuracy and the percentage of unknown words in the test set in the whole word approach, which means that the more unknown words we have the less the accuracy we obtain.

We can also notice that in both the Table 4.17 and Figure 4.3 that the accuracy in the whole word approach is still increasing and has not yet reached a plateau, which suggests that adding more data can achieve better results.

While Figure 4.3 shows that more data can still produce better results for whole word POS tagging, segmentation-based tagging does not promise more improvement as it seems to have reached a plateau. Figure 4.4 is a scatter plot showing the relationship between the training size and the tagging accuracy in the segmentation-based approach. We can see that when we train the tagger on only 10,000 words, the accuracy is 87.40%, and keeps increasing until the data size reaches 3/10 of the full training set where it reaches 93.01% after which the improvement is minimal as the difference between the accuracy using 3/10 of the data and the accuracy using the full training set is merely one half percentage points.

The reason that segmentation-based tagging does not improve significantly after a certain amount of data, and is not expected to improve with even more data than we have is that it is based on segments rather than words, and while the percentage of unknown words with the full training set is 7.85%, the percentage of unknown segments with the same size is 1.43%, and many of these should be erroneously segmented since unknown segments constitute only 1.28% in the comparable gold standard experiment.

Even with the smallest training set, 10,000 words, unknown segments constitute only 8.41% of all segments in the test set. This indicates that whole word tagging still needs more words for better accuracy, while segment-based tagging is saturated.

4.5. Conclusion

I have presented two novel approaches to Arabic part of speech tagging that do not require gold standard tokenization or segmentation: (a) a segmentation-based approach that treats words as composed of segments, and (b) a whole word approach that does not apply any form of pre-processing and treats the word as it occurs in naturally occurring written Arabic.

The segmentation-based approach requires a segmentation module prior to POS tagging, and has proved to be viable in spite of the imperfections resulting from the errors in segmentation. The whole word approach surprisingly performs better than the segmentation-based approach in spite of the difficulty of the task in the former, as it uses a tagset of 993 composite tags compared to 139 simple tags in the latter. The difference can be generally attributed to the errors of segmentation in the segmentation-based approach and/or the low number of unknown words in the whole word approach.

I have used the Habash and Rambow tagset to examine whether a less rich tagset, i.e. a tagset with fewer distinctions and consequently a smaller size, can perform better than a richer tagset. This has turned out not to be true as the full tagset yielded better results. This indicates that smaller tagset size does not necessarily lead to better results.

The training size is also an important factor in POS tagging quality. The more data I feed into the tagger, the higher the accuracy. While size matters considerably for whole word tagging, and adding more data keeps giving better results, this is not the case with segmentation-based tagging. After two tenths of the training data size, the improvement in segmentation-based tagging ceased to be significant, and it reached a plateau early on.

Finally, I have shown that while gold standard vocalization helps part of speech tagging, due to its ability to disambiguate otherwise ambiguous words, automatic vocalization decreases the accuracy of POS tagging owing to the relatively high Word Error Rate in vocalization even in the state of the art vocalization systems.

Chapter 5: Real World Dependency Parsing

5.1. Introduction

In this chapter, I examine the sentence level processing of Arabic after I have examined orthographic enrichment in the first part of the thesis and morphosyntax, represented in part of speech tagging introduced in chapter 4 of this thesis. For this purpose, I have chosen dependency parsing due to its similarity to traditional Arabic grammar, as will be evident below. This similarity makes dependency grammar more able to be appreciated by Arab linguists than constituency grammar, which does not seem to be deeply rooted in the tradition.

In the rest of this chapter, I will introduce dependency grammar and its relationship to traditional Arabic grammar, then introduce dependency parsing. I will then report on the previous studies on Arabic dependency parsing, especially within the CoNLL 2007 shared task (Nivre et al, 2007) and the CoNLL-X shared task on dependency parsing (Buchholz and Marsi, 2006). I will finally present a real-world parsing experiment of Arabic which implements a pipeline of word tokenization, stemming, part of speech tagging and dependency parsing.

5.1.1. Dependency Grammar and Arabic

Dependency parsing is based on dependency syntax whose characteristics can be summarized in the opening chapters of Tesniere (1959), translated by Nivre (2005: 47):

> The sentence is an *organized whole*, the constituent elements of which are words. Every word that belongs to a sentence ceases by itself to be isolated as in the

dictionary. Between the word and its neighbors, the mind perceives *connections*, the totality of which forms the structure of the sentence. These structural connections establish *dependency* relations between the words. Each connection in principle unites a *superior* term and an *inferior* term. The superior term receives the name *governor*. The inferior term receives the name *subordinate*. Thus, in the sentence *Alfred parle* […], *parle* is the governor and *Alfred* the subordinate. (Original Emphasis)

To give an example of dependency grammar, consider the sentence *The man ate the good food*. Rather than think in terms of phrases like in constituent grammar where we see 'the man" as an NP, and the rest of the sentence as a VP, dependency grammar views the sentence structure in terms of bilexical relations between a **head** and a **dependent**. *Man* is the head of *the*, *ate* is the head of both *man* and *food*, and *food* is the head of *good*. Dependency relations may be **typed** or **untyped**. In typed dependency grammars, the relation between the head and the dependent is typed with a grammatical relation. For example, the relation between *ate* and *man* is that of SUBJECT while the relation between *ate* and *food* is that of OBJECT. In untyped dependency parsing, no labels are used. The sentence *The man ate the good food* is represented in the dependency graph in Figure 5.1. In this dependency graph, an arc starts from a dependent and points towards the head, and each arc is labeled with a dependency type. We can also notice that the parts of speech are included in the graph. Parts of speech, as well as other morphological and lexical information, are used by dependency parsers to build the relationships between the lexical items.

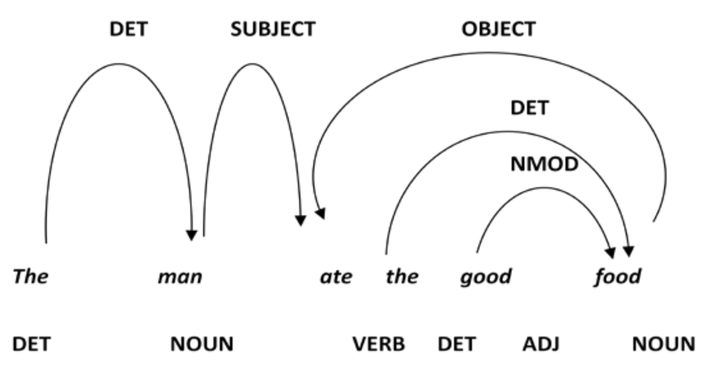

Figure 5.1: a dependency tree

The notion of head and dependent, or governor and subordinate in Tesniere's terms, is very close to Arabic traditional grammar. If we examine the sentence *drb zaydun amran* (Eng. beat Zayd (Nom) Amr (Acc): Zayd beat Amr, where both Zayd and Amr are personal names.), we find that in traditional Arabic grammatical terms, the verb *drb* governs both nouns and assigns the first noun the function agent and the second the function object. Dependency grammar principles are the same principles on which traditional Arabic grammar is based. All grammatical relations in Arabic are binary asymmetrical relations that exist between the tokens of a sentence. Jonathan Owens (1997: 52) writes:

> The largest functional unit in Arabic is the sentence, though a function of special type since its existence is not established by a single substitution class. Nor are the functions which are in a sentence, *jumla*, said to be functions of the sentence. One never finds in Arabic grammar a phrase such as 'agent of the sentence'. Instead, the concept integrating the sub-parts of the sentence, verb, agent, object, etc. is dependency. One item, a verb, for example, as in *darab zaydun amran*, governs another in a particular form, nominative in the case of agent, accusative in the case of object. In general the

Arabic notion of dependency and that defined in certain modern versions e.g. Tesniere (1959) rest on common principles.

This obvious relationship between Arabic and dependency grammar led the creators of the most recent Arabic treebanks to adopt dependency annotation for syntactic analysis. The Columbia Arabic Treebank (Habash and Roth, 2010) and the Quranic Arabic Corpus (Dukes, 2010) are both dependency treebanks.

5.1.2. Dependency Parsing

Dependency parsing makes use of dependency grammar to computationally assign relations within sentences. The CoNLL 2007 shared task (Nivre et al. 2007) introduces the task as follows:

> In dependency-based syntactic parsing, the task is to derive a syntactic structure for an input sentence by identifying the syntactic head of each word in the sentence. This defines a dependency graph, where the nodes are the words of the input sentence and the arcs are the binary relations from head to dependent. Often, but not always, it is assumed that all words accept one syntactic head, which means that the graph will be a tree with the single independent word as the root. In labeled dependency parsing, we additionally require the parser to assign a specific type (or label) to each dependency relation holding between a head word and a dependent word.

According to Covington (2001), (in Nivre (2005: 66)), dependency parsing offers the following advantages:

- Dependency links are close to the semantic relationships needed for the next stage of interpretation; it is not necessary to “read off” head modifier or head compliment relations from a tree that does not show them directly.
- The dependency tree contains one node per word. Because the parser’s job is only to connect existing nodes, not to postulate new ones, the task of parsing is in some sense more straightforward.
- Dependency parsing lends itself to a word at a time operation, i.e., parsing by accepting and attaching words one at the time rather than by waiting for complete phrases. Manning and Schütze (1999: 430) maintain that this word by word relation building is key to resolving most parsing ambiguities. “Because dependency grammars work directly in terms of dependencies between words, disambiguation decisions are being made directly in terms of these word dependencies. There is no need to build a large superstructure (that is, a phrase structure tree) over a sentence and there is no need to make disambiguation decisions high up in the structure well away from the words of the sentence.”
- Dependency parsing allows a more adequate treatment of languages with variable word orders where discontinuous syntactic constructions are more common than in languages like English.

There exist many theories of dependency grammar, all sharing the basic assumption of bilexical relations between words, rather than phrases. Some of these are Word Grammar (Hudson, 1984, 1990), Functional Generative Description (Sgall et al, 1986), and Functional Dependency Grammar (Tapananien and Jarvinen, 1997). In this chapter, I make no attempt to review any of these theories. I will instead give an outline of Inductive Dependency Parsing as implemented in MaltParser (Nivre et al, 2006), the dependency parser I use in the experiments below. Inductive Dependency Parsing is data-driven, and does not require a grammar.

5.1.3. MaltParser

MaltParser, one of the state-of-the-art dependency parsers, was used in the CoNLL shared tasks of 2006 and 2007, and was the second best parser in the former (Buchholz and Marsi, 2006), and the best parser in the latter (Nivre et al, 2007). For these two reasons, its participation in CoNLL shared task and its accuracy, I have decided to use it in the experiments in this chapter.

MaltParser is a data-driven dependency parser. While traditional parsers use hand-crafted grammars to construct parsers, a data-driven parser uses a treebank to learn a grammar, and a machine learner to learn parse actions (see below). MaltParser is an implementation of inductive dependency parsing (Nivre, 2005), where a machine learning approach is used to guide the parser at non-deterministic choice points (Nivre et al, 2006). Inductive dependency parsing is based on three components:

(a) Deterministic parsing algorithms for building dependency graphs. MaltParser implements a number of algorithms including the Nivre algorithm and the Covington algorithm. Nivre's algorithm (Nivre 2003) is a linear-time algorithm limited to projective dependency structures. Covington's algorithm (Covington 2001) is a quadratic-time algorithm for unrestricted dependency structures, i.e. it can handle both projective and non-projective structures. A projective structure is one in which “a word and its descendents form a contiguous substring of the sentence” (McDonald et al, 2005). A sentence like *John saw a dog yesterday that was a Yorkshire Terrier*, shown in the dependency graph in Figure 5.2, has a non-projective structure since the relative clause is separated from the object it modifies by an adverb. Nivre’s algorithm traditionally gave the best results on all languages and data sets (Nivre and Hall (2005)), but in the CoNLL 2007 shared task, an ensemble of parsers with different algorithms, including Nivre’s algorithm, produced the best results (Nivre et al, 2007).

(b) History-based feature models for predicting the next parser action, e.g. assigning heads and dependencies. The feature model implemented in MaltParser considers dependency relations, lexical features, and part of speech information from annotated dependency training data.

(c) Discriminative machine learning to map histories to parser actions. The machine learner plays the role of an oracle in that it uses the features in the feature model to guide the parser on what to do next.

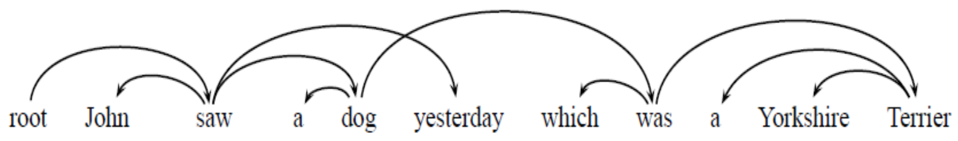

Figure 5.2: A non-projective English sentence (McDonald and Pereira (2005))[22]

[22] The arrows in this graph point from heads to dependents, which is not the notation used in this thesis. I maintained the style used by the authors.

The parser requires two data structures: (a) a stack of partially processed tokens, and (b) a queue of remaining input tokens. MaltParser performs four transitions (parse actions): (a) LEFT-ARC, which makes the top token on the stack a dependent of the next token, (b) RIGHT-ARC, which makes the next token a dependent on the top token, (c) REDUCE, which pops the stack, and (d) SHIFT, which pushes the next token onto the stack. Parsing actions, e.g. shifting, reducing, and building relations, are then built from atomic actions: adding arcs and stack and queue operations. To illustrate how the parser works, I will give an example from Nivre (2006), illustrating how MaltParser handles the sentence:

Economic news had little effect on financial markets.

(1) The parser initializes a stack with a virtual node ROOT and a queue containing the rest of the sentences. The ROOT node is helpful in attaching otherwise unattached tokens.

[ROOT]S [Economic news had little effect on financial markets.]Q

(2) Shift the first word in the queue to the stack

[ROOT Economic]S [news had little effect on financial markets.]Q

(3) Now the first word on the queue is *news*, and there is an NMOD (noun modifier) relation between this word and the word on top of the stack: (*Economic*). *News* is the head of *Economic*, and since the word has found its head, a left arc operation connecting the word is performed by which the head finds its dependent to the left.

(4) The word *Economic* is reduced, which means that the word will not be processed any further.

(5) The word *news* is shifted to the stack, and it is now the active node, the node looking for a head. Through similar operations as above, the node finds its head *had*, and is reduced, and then *had* is shifted to the active node of the stack. These operations are repeated for every word in the queue until a full parse is found.

But how does the parser know that there exists a relationship between the word *Economic* and the word *news*? In order for the parser to decide which action to take next, it has to be guided, and MaltParser is guided by a machine learning algorithm that learns the next move from the features in the training set. The feature set for the MaltParser includes lexical and part of speech features as well as the dependency relations between the heads and the dependents.

As for the learning algorithm, MaltParser traditionally used memory-based learning and Support Vector Machines for classification, but the newest version, 1.3.1., supports only Support Vector Machines in the implementation of the LIBSVM library (Chang and Lin, 2001).

MaltParser is first run in a learning mode where it takes as input a dependency training set (in the CoNLL format) and induces a classifier for predicting parser actions based on the parsing algorithm, the feature model, and the learning algorithm. The parsing model created during the learning phase is then used to parse new sentences, provided that they are in the same format as the learning set.

5.1.2. Related Studies

The Conference on Computational Natural Language Learning has a shared task each year in which the shared task organizers provide data, split into training and test sets, as well as evaluation metrics for a certain language task. Teams compete to obtain the best results on that task. The importance of these shared tasks is that they provide data in areas that would lack it otherwise. For example, the shared tasks of 2006 (Bulchhoz and Marsi, 2006) and 2007 (Nivre et al, 2007) were dedicated to dependency parsing, and this resulted in the availability of data for many languages. The bulk of literature on Arabic Dependency Parsing stems from the two CoNLL shared tasks of 2006 and 2007.

In CoNLL-X (Buchholz and Marsi, 2006), data was provided for 13 languages, and the Arabic data was extracted from the Prague Arabic Dependency Treebank (Haijc et al 2004, Smrz et al 2002). The training data included 54,000 tokens and the test data included 4990 scoring tokens. Scoring tokens do not include punctuation, which was excluded from the evaluation in this shared task. The average number of tokens per parsing unit was 37.2, and a parsing unit was sometimes more than one sentence. 17.3% of the tokens in the test were not in the training set, i.e. out of vocabulary tokens.

The official scoring metric for the CoNLL-X shared task was the standard Labeled Attachment Score, the percentage of tokens that had the correct head and the correct dependency label. The average Labeled Attachment Score on Arabic across all results presented by the 19 participating teams was 59.9% with a standard deviation of 6.5. The best results were obtained by McDonald et al (2006) with a score of 66.9% followed by Nivre et al (2006) with 66.7%. McDonald et al used a two-stage parser that first performs unlabeled dependency parsing then adds the syntactic labels in a second stage using a sequence classifier. While McDonald et al obtain a Labeled Accuracy Score of 80.3 on average over all languages, their Arabic results are more than 13 percentage points below their average. Arabic was considered the second most difficult dataset, after Turkish, due to the small size, and the unusually long parsing units. For example, Chang et al (2006) observe that their Unlabeled Attachment Score for Arabic is lower than that of the other languages, which they attribute to the sentence length in Arabic since they use a pipeline in their algorithm, and such an approach requires more predictions to complete a long sentence. In a pipeline approach, any error in an initial step will propagate to the next step, and the longer the sentence the higher the chance that an error will occur.

In 2007, The Prague Arabic Dependency Treebank was used again in the multilingual track of the CoNLL 2007 shared task on dependency parsing (Nivre et al, 2007), but the training data more than doubled in size as the number of tokens in the training set increased from 54,000 tokens to 112,000 tokens and the test set increased in size from 4990 tokens to 5124 tokens. In addition, the morphological annotation was made more informative. While the data size problem of 2006 was alleviated, the problem of sentence size, the number of tokens per sentence, remained unsolved. With 2900 sentences in the Arabic data set, the average number of tokens per sentence is 38.3 in the training data, while the test data showed longer sentence sizes at 39.1 tokens per sentence. Arabic was among the most difficult languages (Arabic, Basque, and Greek), which are all characterized by a high degree of inflection combined with a relatively free word order. It is worth noting that in the evaluation scheme in 2007, punctuation was included in the evaluation and was no longer considered as non-scoring unit, unlike in the 2006 shared task.

The best results on Arabic in the CoNLL 2007 shared task were obtained by Hall et al (2007) as they obtained a Labeled Attachment Score of 76.52%, 9.6 percentage points above the highest score of the 2006 shared task. Hall et al used an ensemble system, based on the MaltParser dependency parser that extrapolates from a single MaltParser system. Hall et al developed their system in two stages: (1) a Single Malt system, and (2) a Blended Malt system. The settings with the Single MaltParser led to a Labeled Accuracy Score of 74.75% on Arabic, and this parser was used as the basis for the Blended Malt stage. The blended experiment uses six different parsers for the language, each of which is a variation of the Single MaltParser. The best results, a LAS of 76.52%, used a combination produced from weighting the results of each of the six parsers.

In terms of the parsing algorithm, Hall et al used the default Nivre algorithm with the arc-eager algorithm, "in the sense that right dependents are attached to their heads as soon as possible". The stack was initialized with an artificial ROOT node (with token id 0). This enables arcs originating from the root to be added explicitly during parsing. Pseudo-projectivization, in which the training data is projectivized and information about these transformations is encoded in extended arc labels to support deprojectivization of the parser output, was not found to be helpful for Arabic. Hall et al attribute this to the fact that non-projective sentences constitute only about 10% of the data.

The feature model for Arabic included the standard features found in the training set: FORM, LEMMA, CPOSTAG (coarse-grained POS tags), POSTAG, FEATS (linguistic features) as well as the dependency relations. The feature model includes:

- **POS features:** POS of the token on the stack, the token on the input (queue), the next token on the input, the next+1 token on the input, the previous token on top of the stack, the left dependent of the partially built dependency structure, the right dependent of the partially built dependency structure.
- **CPOS features:** CPOS of the token on top of the stack, the token on top of the input, the right dependent of the partially built dependency structure.
- **Dependency** relation of the token on top of the stack
- **Linguistic Features** of the token on top of the stack, and the token on top of the input
- **Lemma** of the token on top of the stack, and the token on top of the input
- **LEX** features determine which words (or rather tokens) are included in the feature set. The LEX features include the token on top of the stack, token on the input, next token on the input, two previous token on the stack, and the token -1 position in the original input string.

Hall et al used Support Vector Machines as the learning algorithm. In order to reduce training times, Hall et al split the training data into smaller sets and trained separate multi-class classifiers for each set, using the POSTAG as the defining feature for the split.

5.3. The Current Study

The current study aims at measuring the effect of orthographic enrichment, as presented in the previous chapters, on the quality of dependency parsing of Arabic. To this end, I implement a pipeline that assumes only that I have sentences, and does not assume any gold standard tokenization, lemmatization, part of speech tags, or linguistic features.

The shared tasks of 2006 and 2007 used gold standard components in all fields, which is not realistic for Arabic, or for any other language. For Arabic and other morphologically rich languages, it may be more unrealistic than it is for English, for example, since the CoNLL 2007 Arabic dataset has tokens, rather than orthographic units, as entries. Orthographic units are white-space delimited forms. A single OU may have more than one syntactically functional token. For example, the OU *bh*A* (Eng. with this) comprises the preposition *b* and the demonstrative *h*A*, which means that tokenization has to be performed first; a step that may not be necessary in English.

5.3.1. Data

The data used for the current study is the same data set used for the CoNLL (2007) shared task, with the same division into training set, and test set. This design helps in comparing results in a way that enables us to measure the effect of automatic pre-processing on parsing accuracy.

The data is in the CoNLL column format. In this format, each token is represented through columns each of which has some specific information. The first column is the ID, the second the token, the third the lemma, the fourth the coarse-grained POS tag, the fifth the POS tag, and the sixth column is a list of linguistic features. The last two columns of the vector include the head of the token and the dependency relation between the token and its head. Table 5.1 shows the Arabic sentence:

تعداد سكان 22 دولة عربية سيرتفع إلي 654 مليون نسمة في منتصف القرن.

(Buckwalter: *tEdAd skAn 22 dwlp Erbyp syrtfE <lY 654 nsmp fy mntSf Alqrn*)

(Eng. The population of 22 Arab states will rise to 654 million people in the middle of the century)

The word *taEodAdu*, for example, is the first word of the sentence, and thus has the ID number 1. The second column presents the word itself, and the third the lemma. The token has the coarse POS tag N and the fine grained POS tag N-. The linguistic features in column 6 indicate that the token is in Case 1, which is the nominative case, and that the word is definite by virtue of its being in a construct state (Defin=R). The HEAD column lists token number 7 as the head of this token, and the last column specifies that the relation between this token and its head is that of SUBJECT, i.e. the word is the subject of the verb with ID number 7.

ID	**FORM**	**LEMMA**	**CPOS**	**POS**	**FEAT**	**HEAD**	**DEPREL**
1	taEodAdu	taEodAd_1	N	N-	Case=1|Defin=R	7	Sb
2	suk~Ani	sAkin	N	N-	Case=2|Defin=R	2	Atr
3	22	[DEFAULT]	Q	Q-	_	2	Atr
4	dawolapF	Dawolap_1	N	N-	Gender=F|Number=S|Case=4|D	3	Atr

					efin=I		
5	Earabiy~apF	Earabiy~_1	A	A-	Gender=F|Number=S|Case=4|Defin=I	4	Atr
6	sa	sa_FUT	F	F-	_	7	AuxM
7	yarotafiEu	AirotafaE_1	V	VI	Mood=I|Voice=A|Person=3|Gender=M|Number=S	0	Pred
8	<ilaY	<ilaY_1	P	P-		7	AuxP
9	654	[DEFAULT]	Q	Q-	_	8	Adv
10	miloyuwna	miloyuwn_1	N	N-	Case=4|Defin=R	9	Atr
11	nasamapK	nasamap_1	N	N-	Gender=F|Number=S|Case=2|Defin=I	10	Atr
12	fiy_1	fiy	P	P-	_	7	AuxP
13	munotaSafi	munotaSaf_1	N	N-	Case=2|Defin=R 12	12	Adv
14	Alqaroni	qaron_1	N	N-	Case=2|Defin=D	13	Atr

Table 5.1: The first sentence of the Arabic training set

These column elements are detailed as follows:

1. **ID**: Token counter, starting at 1 for each new sentence. The training set contains 2913 sentences. The longest sentence in the Arabic training data comprises 396 tokens. This rather long sentence contains a quotation that can easily be divided into several sentences. The average number of tokens per sentence is 38.34. The test set contains 132 sentences, with the longest sentence being 156 tokens.

2. **FORM**: Word form or punctuation symbol. The word forms in the Arabic data are tokens rather than words. The differences is that Arabic words can have more than one token each since conjunctions, prepositions, pronouns, as well as other clitics that have a syntactic function constitute part of the orthographic unit, a white-space delimited unit, and these have to be split off before any syntactic analysis can be performed. An example from the test set is the three tokens *w*, *b*, and *Alnsbp* which constitute a single written form *wbAlnsbp* (Eng. and with regard) in naturally occurring Arabic. This means that the lexical entries in the Arabic data are idealized, and that automatic tokenization has to be performed if dependency parsing is to target real-world data.

3. **LEMMA**: Lemma or stem of word form, or an underscore if not available. In the Arabic data, the lemmas rather than the stems are used. The difference is that lemmatization produces the base form of the word. For example, an Arabic lemma is always the singular masculine form of the word, while the stem is the word without any prefixes or suffixes. Lemmatization is harder than stemming since lemmatization involves working with both inflectional and templatic/derivational morphology such as finding that *>mAkn* is the plural of *mkAn*, which cannot be discovered through stemming.

4. **CPOSTAG**: Coarse-grained part-of-speech tag. The CoNLL 2007 CPOS tagset is in table 5.2.

Tag	**Meaning**
N	Noun
A	Adjective
Z	Proper Noun
D	Adverb. This includes place and time words.
V	verb

C	Different forms of conjunctions including conditional and subjunctive particles
F	Negation and Exception Particles. This also includes emphatic particles.
Q	Numbers
P	Preposition
S	Pronouns, Demonstratives, Relative Pronouns
Y	Abbreviation

Table 5.2: Arabic coarse-grained POS tags

5. **POSTAG**: Fine-grained part-of-speech tag, or identical to the coarse-grained part-of-speech tag if not available. The Arabic fine grained POS tagset details the V (verb) into three separate tags for Perfect, Imperfect, and Imperative Verbs. It also includes separate tags for Demonstrative and Relative pronouns.

6. **FEATS**: Unordered set of syntactic and/or morphological features, separated by a vertical bar (|), or an underscore if not available. The features in the CoNLL 2007 Arabic dataset represent case, mood, definiteness, voice, number, gender and person. Table 5.3 lists the features used in the Arabic dataset.

Feature	**Meaning**	**Example**
Mood=S	Subjunctive Mood	
Mood=I	Indicative Mood	
Mood=J	Jussive Mood	
Mood=D	Imperative Mood	
Defin=D	Definite Noun/Adjective	
Defin=I	Indefinite Noun/Adjective	
Defin=C	Construction s in which the two nouns/adjectives are definite	الطويلة الأمد
Defin=R	Definite by virtue of idafa	
Voice=P	Passive Voice	
Voice=A	Active Voice	
Number=P	Plural Noun	
Number=S	Singular Noun	
Number=D	Dual Noun	
Gender=M	Masculine Gender	

Gender=F	Feminine Gender
Case=1	Nominative Case
Case=2	Genitive Case
Case=4	Accusative Case
Person=1	First Person
Person=2	Second Person
Person=3	Third Person
_	No feature available

Table 5.3: The linguistic features in the Arabic Dataset

7. **HEAD**: Head of the current token, which is either a value of ID or zero (0). Depending on the original treebank annotation, there may be multiple tokens with HEAD=0, which means that the tokens have no explicit head and are headed by a virtual ROOT token. This is usually the case with sentence-final punctuation as well as main verbs.

8. **DEPREL**: Dependency relation to the HEAD. Table 5.4 gives the dependency relations and their basic definitions[23].

Relation	**Description**
Pred	Predicate, a node not depending on another node.
Sb	Subject
Obj	Object
Adv	Adverbial
Atv	Complement (so-called determining) technically hung on a non-verb.

[23] The PADT manual does not include descriptions of these relations. The definitions are taken from the Prague Czech Treebank and there may be some slight differences. A complete description is available at http://ufal.mff.cuni.cz/pdt2.0/doc/manuals/en/a-layer/html/ch03.html

	element
AtvV	Complement (so-called determining) hung on a verb, no 2nd gov. node
Atr	Attribute
Pnom	Nominal predicate, or nom. part of predicate with copula *be*
AuxV	Auxiliary vb. *be*
Coord	Coord. node
Apos	Apposition (main node)
AuxT	Reflex. tantum
AuxR	Ref., neither Obj nor AuxT, Pass. refl.
AuxP	Primary prepos., parts of a secondary p.
AuxC	Conjunction (subord.)
AuxO	Redundant or emotional item, 'coreferential' pronoun
AuxZ	Emphasizing word
AuxX	Comma (not serving as a coordinating conj.)
AuxG	Other graphic symbols, not terminal
AuxY	Adverbs, particles not classed elsewhere
AuxS	Root of the tree (#)
AuxK	Terminal punctuation of a sentence
ExD	A technical value for a deleted item; also for the main element of a sentence without predicate (Externally-Dependent)
AtrAtr	An attribute of any of several preceding (syntactic) nouns
AtrAdv	Structural ambiguity between adverbial and adnominal (hung on a

	name/noun) dependency without a semantic difference
AdvAtr	Same as above with reverse preference
AtrObj	Structural ambiguity between object and adnominal dependency without a semantic difference
ObjAtr	Same as above with reverse preference

Table 5.4: Dependency relations

5.3.2. Methods

Since this study aims at performing real world dependency parsing of Arabic, I need to minimize the assumptions in the previous studies, i.e. gold standard tokenization, POS tags, lemmas, and linguistic features. The study does thus implement a pipeline approach that takes as input a string of words, and produces as output a dependency parsed text similar to that produced by the CoNLL (2007) experiments. The pipeline is implemented as follows:

(1) The text is passed to the word segmenter which produces the orthographic units in which the different segments of the orthographic unit are marked by a + sign. I use the memory-based segmenter I developed in chapter 2 of this thesis as it is capable of producing segmentation with very high accuracy.

(2) A rule-based converter converts the segmented orthographic units into tokens as outlined in the section on tokenization in chapter 2 (Section 2.4.2.2.4). This has a general accuracy of 99.3%.

(3) A rule-based stemmer takes the tokens as input and strips all the inflectional affixes to produce the stems.

(4) The tokens are passed to the part of speech tagger. I use the tokenization-based Memory-based Tagger, MBT, (Daelemans et al: 1996), with the same settings used in chapter 4, albeit with different datasets.

(5) The dependency parser (MaltParser 1.3.1) takes all the information above and produces the data with head and dependency annotations.

Although the purpose of this experiment is to perform dependency parsing of Arabic without any assumptions, one assumption I cannot avoid is that the input text should be divided into sentences. For this purpose, I use the gold standard division of text into sentences without trying to detect the sentence boundaries myself, although this would be necessary in actual real-world use of dependency parsing. The reason for this is that it is not clear how sentence boundaries are marked in the data as there are sentences whose length exceeds 300 tokens. If I detected the boundaries automatically, then I would face the problem of aligning my sentences with those of the test set for evaluation.

In the parsing experiments below, I will use the dependency parser MaltParser (Nivre et al., 2006). I will use Single MaltParser, as used by Hall et al (2007), with the same settings for Arabic that were used in the CoNLL 2007 shared task on the same data to be as close as possible to the original results in order to be able to compare the effect of non gold standard elements in the parsing process[24]. The settings include Nivre's

[24] Although MaltParser 0.4 was used in both the shared tasks of 2006 and 2006, I use MaltParser 1.3.1 upon the recommendation of Joakim Nivre (personal communication). MaltParser 0.4 is not supported any more,

algorithm in the arc-eager mode, with stack initialization with an artificial root node, with Support Vector Machines as a learning algorithm. The features include lexical, part of speech and dependency information as outlined in section 5.2 above.

Since each one of the pipeline components above is a complete process in and of itself, I will give detailed information about each one below:

5.3.2.1. Tokenization

Given a string of naturally occurring Arabic, the natural first step is to perform tokenization, a process in which I split off those parts of the orthographic form that have syntactic functions since some of these will be heads and/or dependents, and dependency relations will depend on them. For example, the orthographic unit *lmSr* (Eng. for Egypt) has a head (the preposition *l*) and a dependent (the noun *mSr*) without any white space between them. Tokenization separates these two tokens. I will use the tokenizer that I developed in the context of word segmentation in chapter 2. I will use the same training data that were used in the tokenization section. I made sure that there are no overlapping sentences between the dependency task test set and the tokenizer training set. This is in principle possible since the Penn Arabic Treebank and Prague Arabic Dependency Treebank both depend on raw data from the Arabic Gigaword corpus (Graff, 2007).

Since the Prague Arabic treebank is tokenized by default, I had to reconnect the various tokens of each orthographic form in order to obtain naturally occurring Arabic in

and Nivre kindly provided me with the Arabic-specific settings for MaltParser 1.3.1. I am deeply grateful for Nivre for his helpfulness.

order to simulate real world processing of Arabic. The re-connection depended on a list of clitics, their POS tags and linguistic features. For example, the token *hum* can either be a clitic (Eng. them, their) or an independent personal pronoun (Eng. they). When it is a clitic it has to be part of the orthographic unit, but in the CoNLL data it has its own line of features without any indication of its status as a clitic. Given the following two lines within a sentence:

```
6   >n~a   >n~a_1   C   C-   _
7   hum    hum-1    S   S-   Person=3|Gender=M|Number=P|Case=4
```

we can see that one of the features has Case=4, which means that it cannot be an independent pronoun as only nominative pronouns (Case=1) can be standalone tokens. The pronoun thus has to be cliticized to the previous unit. The reconnection results in the orthographic unit >*an~ahum* instead of the two *tokens* >*an~a* and *hum*.

The tokenization step is potentially problematic since tokenization can produce more or fewer tokens than there are in the test set, in which case the assignment of dependencies and heads can suffer severely. For example, the word *bmhmp* (Eng. with a task), has two tokens: the preposition *b* and the noun *mhmp*, in which case the preposition is the head of the noun, but if the word is tokenized incorrectly as *bmhmp,* this will lead to a problem in part of speech tagging, and also in the assignment of heads and dependency relations, and this can affect the whole sentence. Another associated problem is that of text alignment, which leads to problems in the evaluation as we have seen in the part of speech tagging chapter (Section 4.4.1.3).

5.3.2.2. Stemming

In the dependency parsing experiments within the CoNLL 2007 shared task, the lemmas are used as features in the vector, but since Arabic lemmatization is an arduous process, and the process of developing an Arabic lemmatizer is beyond the scope of this thesis, I will use the stems instead. I will use a stemmer based on the segmentation scheme I developed in chapter 2 (Section 2.4.2.2).

Manning et al (2009: 32) differentiate between stemming and lemmatization in the context of information retrieval.

> *Stemming* usually refers to a crude heuristic process that chops off the ends of words in the hope of achieving this goal correctly most of the time, and often includes the removal of derivational affixes. *Lemmatization* usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the *lemma*. If confronted with the token *saw*, stemming may return just *s*, whereas lemmatization would attempt to return either *see* or *saw* depending on whether the use of the token was as a verb or a noun. The two may also differ in that stemming most commonly collapses derivationally related words, whereas lemmatization commonly only collapses the different inflectional forms of a lemma.

While the distinction may not be the same for Arabic, in which derivation is carried out through templatic morphology, lemmatization, which returns the citation word, poses many challenges of which the following two are only examples:

- The broken plural: In order to obtain the lemma, one needs to obtain the singular forms of nouns and adjectives. A large portion of Arabic nouns and adjective have broken plurals which are not formed through affixation, but through changes in the internal morphological structure, i.e. templatic morphology. For example, the plural of *kitAb* is *kutub*, and the plural of *Tifol* is *>TofAl*. Most of these forms are ambiguous. For example, the form *ktb* can be a plural noun, a transitive verb, a ditransitive verb, or a passive verb, with the lemma being different among nouns and verbs. A less likely solution of the word is Preposition + Verb, which can be used in giving examples[25].
- Word-internal vowel change. When one of the root consonants of the word is from the set (>, y, w), these undergo many changes in inflections. For example, the past tense verb *wqf* (Eng. he stood) changes to *yqf* (Eng. he stands) in the present tense. With the complex nature of the orthography, it is usually hard to see which letters are original, and this requires analysis at the root level.

For these reasons I use the stems rather than the lemmas. Deriving a stemmer from a segmenter is a trivial matter since it only involves removing the inflectional affixes and keeping the stem. The function stem is defined as follows:

(a) List all the inflectional affixes: *Al*, *t*, *y*, *>*, *n*, *A*, *wA*, *An*, *yn*, *l*, *At*.

(b) Examine each token. If the token is made up of a single segment, return that segment.

[25] It is rare for prepositions to precede verbs, but in Arabic grammar books, one encounters these when the authors enumerate examples. An English example is "We use *to* with *listen*", where *listen* is a verb preceded by a preposition.

(c) If the token is made of more than one segment, return only that segment which is not in the affix list.

For example, given a token like *AlmhndsAt* (Eng. the-female-engineers), the segmenter returns *Al+mhnds+At*, and since *Al* and *At* are in the affix table, they are simply removed, and I obtain the stem *mhnds*.

5.3.2.3. Part of speech tagging

I use the dependency training set as a training set for building a part of speech tagger by which to tag the test set. I use this training set rather than the ones I have used so far in the part of speech tagging chapter (chapter 4) experiments because the tagset for the dependency parsing is different from the part of speech tags in the Penn Arabic Treebank. I will, however, test the effect of the Penn tagset on the dependency parsing experiment as well. I use the same settings for part of speech tagging used in chapter 4. I use the memory-based tagger (MBT) as a part of speech tagger. The MBT features for known words include the two context words to the left along with their disambiguated POS tags, the focus word itself, and one word to the right along with its ambitag (the set of all possible tags it can take). For unknown words, the features include the first five letters and the last three letters of the word, the presence of a hyphen in the focus word, the left context tag, the right context ambitag, one word to the left, the focus word itself, one ambitag to the right, and one word to the right.

5.3.2.4. Linguistic Features

The column containing the linguistic features in the real world dependency experiment will have to remain vacant due to the fact that it is hard to produce these features automatically given only naturally occurring text.

The gender feature, for example, has two values: Feminine and Masculine, but gender is grammatical in Arabic, and the rules, if they exist at all, are complex. The following notes about the feminine in Arabic (Haywood and Nahmad, 1965: 365-371) illustrate the problem:

- The taa marbuta (Buckwalter *p*) is the most common feminine marker. It has to be noted though that many masculine nouns end in the *taa marbuta*. Examples of these are *xlyfp* (Eng. successor), *Hmzp* (Proper Noun).
- Many collective nouns, whether feminine or masculine, can be modified by a feminine adjective, for example, *nHl kvyrp* (Eng. many sects)
- Some forms, which cannot, for obvious reasons, be masculine, retain the masculine form. Examples of these include *Amr>p HAml* (Eng. A pregnant woman). Since no man can be pregnant, the adjective does not need to have the feminine form.
- Many forms can be feminine or masculine for no obvious reason. For example, in both *Al$ms* (Eng. the sun) and *Alqmr* (Eng. the moon), there is no feminine marker, but the sun is feminine while the moon is masculine.

For these reasons, identifying gender is not a trivial task, and in real world dependency parsing, this is a task of its own.

The identification of the verb voice is no less trivial than identifying gender. Aside from the wider context, passive and active verbs look exactly the same in written Arabic, with the only difference between them being vocalization. The active verb starts with a *fatha* while the passive verb starts with a *damma*, but this was a major confusion in the vocalization experiments (see section 3.3), and since the major experiments in this chapter do not make use of vocalization, the distinction cannot be available. While POS tagging, in its capacity as a disambiguator, can potentially be used to identify the verb voice feature, the performance of the POS tagger introduced in chapter 4 was not impressive as the general accuracy on passive verbs was merely 60.43%.

The number feature has three possible values: (a) Singular, (b) Dual, and (c) Plural. While the CONLL 2007 Arabic dataset marks all these features, in reality it is very hard to distinguish between these. While the plural is sometimes marked through suffixes, many nouns cannot be pluralized this way, and a form called the broken plural has to be used (Haywood and Nahmad, 1965: 40-59). The broken plural is not a regular form, and thus has to be memorized. Another issue with the number system is that plural nouns denoting irrational beings have to be modified by singular feminine adjectives, and not plural ones.

5.3.3. Evaluation

The official evaluation metric in the CoNLL 2007 shared task on dependency parsing was the **labeled attachment score** (LAS), i.e., the percentage of tokens for which a system has predicted the correct HEAD and DEPREL, but results reported also included **unlabeled attachment score** (UAS), i.e., the percentage of tokens with correct HEAD, and the **label accuracy** (LA), i.e., the percentage of tokens with correct DEPREL. In this chapter, I will use the official evaluation software for the CoNLL 2007 shared task, eval07.pl, and will report the three measures reported in the literature.

One problem involved in using the official evaluation script, eval07.pl, is that it assumes perfect tokenization, which is not the case in the experiments presented here. To overcome this problem, I have re-aligned the non-aligning sentences resulting from wrong tokenization. If a word was under-tokenized, i.e. if two tokens for example were tokenized as one, I repeated the incorrectly tokenized token. If, however, a word was over-tokenized, I deleted the line including the second token in the parser's output.

5.3.4. Real World Dependency Parsing Experiment

In this section, I present an experiment in which I perform real-world parsing. In real-world parsing I assume that all I have is a string of text divided into sentences. In order for this string of text to be parsed as in the CoNLL shared tasks of 2006 and 2007, I need to provide tokenization, parts of speech, lemmas or stems, and linguistic features.

One major difference between the parsing experiments which were performed in the 2007 shared task and the ones performed here is vocalization. The data set which was used in the shared task was completely vocalized with both word-internal short vowels and case markings. Since vocalization in such a perfect form is almost impossible to produce automatically, I have decided to primarily use unvocalized data instead. I have removed the word internal short vowels as well as the case markings from both the training set and the test set. This has the advantage of representing naturally occurring Arabic more closely, and the disadvantage of losing information that is only available through vocalization. I will, however, report on the effect of vocalization on dependency parsing in the discussion (section 5.3.7).

To give an estimate of the effects vocalization has on dependency parsing, I have replicated the original task with the vocalized data, and then re-run the experiment with the unvocalized version. Table 5.5 presents the results:

	Vocalized	**Unvocalized**
Labeled Attachment Score	74.77%	74.16%
Unlabeled Attachment Score	84.09%	83.53%
Label Accuracy Score	85.68%	85.44%

Table 5.5: Vocalized versus unvocalized dependency parsing

The results of the experiment indicate that vocalization has a positive effect on the quality of the parsing output. Labeled attachment score drops from 74.77% on the vocalized data to 74.16% on unvocalized data. Unlabeled attachment score drops from 84.09% to 83.53% and labeled accuracy score from 85.68% to 85.44%. The difference is minimal, and is expected to be even smaller with automatic vocalization. This is the reason I primarily use unvocalized data.

It is well known, as we have seen in the previous chapters, that vocalization in many cases equals disambiguation, and removing the vowels adds more ambiguity, which negatively affects the results. In the training set under discussion, the type / token ratio is 1:7.64 in the unvocalized version compared to 1:5.30 in the vocalized one. Each unvocalized token has 1.44 vocalizations on average. Most of these differ, however, only in case endings. Case endings disambiguate syntactic functions and they may help in parsing.

The study will henceforth use the unvocalized results as the gold standard results, and any comparison will be made in reference to these.

To give a concrete example of how the data will differ in our experiments from those in the CoNLL 2007 shared task, tables 5.6 and 5.7 present the same sentence in the two settings. In the first table, all the linguistic features are present, unlike in the second table in which not all the features are present, and those that are present are automatically generated features. For example, the second word in the sentence, *skAn*, is a broken plural and has the lemma *sAkn*, but since I do not use lemmatization, I use the stem *skAn* instead, which, in this case, is the same as the original token.

ID	FORM	LEMMA	CPOS	POS	FEAT	HEAD	DEPREL
1	taEodAdu	taEodAd_1	N	N-	Case=1|Defin=R	7	Sb
2	suk~Ani	sAkin	N	N-	Case=2|Defin=R	2	Atr
3	22	[DEFAULT]	Q	Q-	_	2	Atr
4	dawolapF	Dawolap_1	N	N-	Gender=F|Number=S|Case=4|Defin=I	3	Atr
5	Earabiy~apF	Earabiy~_1	A	A-	Gender=F|Number=S|Case=4|Defin=I	4	Atr
6	sa	sa_FUT	F	F-	_	7	AuxM
7	yarotafiEu	AirotafaE_1	V	VI	Mood=I|Voice=A|Person=3|Gender=M|Number=S	0	Pred
8	<ilaY	<ilaY_1	P	P-		7	AuxP
9	654	[DEFAULT]	Q	Q-	_	8	Adv
10	miloyuwna	miloyuwn_1	N	N-	Case=4|Defin=R	9	Atr
11	nasamapK	nasamap_1	N	N-	Gender=F|Number=S|Case=2|Defin=I	10	Atr
12	fiy_1	fiy	P	P-	_	7	AuxP
13	munotaSafi	munotaSaf_1	N	N-	Case=2|Defin=R 12	12	Adv
14	Alqaroni	qaron_1 N	N	N-	Case=2|Defin=D	13	Atr

Table 5.6: A sentence in the original format

ID	FORM	LEMMA	CPOS	POS	FEAT	HEAD	DEPREL
1	tEdAd	tEdAd	N	N	_	7	Sb
2	skAn	skAn	N	N	_	2	Atr
3	22	22	Q	Q	_	2	Atr
4	dwlp	dwl	N	N	_	3	Atr
5	Erbyp	Erby	A	A		4	Atr
6	s	s	F	F	_	7	AuxM
7	yrtfE	rtfE	V	V	_	0	Pred
8	<lY	<lY	P	P	_	7	AuxP
9	654	654	Q	Q	_	8	Adv
10	mlywn	mlywn	N	N	_	9	Atr
11	nsmp	nsm	N	N	_	10	Atr
12	fy	fy	P	P	_	7	AuxP
13	mntSf	mntSf	N	N	_	12	Adv
14	Alqrn	qrn	N	N	_	13	Atr

Table 5.7: The same sentence from Table 5.6 with the features used in this study.

5.3.4.1. Results and discussion

5.3.4.1.1. Tokenization

Tokenization proved to be easy: I obtain an accuracy of 99.34%. Out of the 4550 words which the test set comprises, there are only 30 errors affecting 21 out of the 132 sentences in the test set. 17 of the errors can be characterized as over-segmentation while the other 13 are under-segmentation. The case of equal segmentation does not occur in this experiment. 13 of the over-segmentation cases are different tokens of the word type *blywn* (Eng. billion) as the initial *b* in the words was treated as a preposition while it is an original part of the word.

A closer examination of the errors in the tokenization process reveals that most of the words which are incorrectly tokenized do not occur in the training set, or occur there only in the form produced by the tokenizer. For example, the word *blywn* does not occur in the training set, but the form *b+lywn+p* occurs in the training set, and this is the reason the word is tokenized erroneously.

Another example is the word *bAsm*, which is ambiguous between a one-token word *bAsm* (Eng. smiling), and a two-token word, *b+Asm* (Eng. in the name of). Although the word should be tokenized as *b+Asm*, the word occurs in the training set as *bAsm*, which is a personal name.

In fact, only five words in the 30 mis-tokenized words are available in the training set, which means that the tokenizer has a very high accuracy on known words. There are yet two examples that are worthy of discussion. The first one involves suboptimal orthography. The word *r>smAl* (Eng. capital in the financial sense) is in the training set but is nonetheless incorrectly tokenized in our experiments because it is written as *brAsmAl* (with the preposition *b*) but with an *alif* instead of the *hamza*. The word was thus not tokenized correctly. The other example involves an error in the tokenization in the Prague Arabic Dependency Treebank. The word *>wjh* (Eng. I give/address) has been tokenized in the Prague Arabic dependency treebank as >wj+h (Eng. its utmost/prime), which is not the correct tokenization in this context as the *h* is part of the word and is not a different token. The classifier did nonetheless tokenize it correctly but it was counted as wrong in the evaluation since it does not agree with the PADT gold standard.

I have to note that the results of tokenization reported here are very close to the results of tokenization reported in the segmentation chapter. While tokenization experiments using training and testing on the Penn Arabic Treebank yields an accuracy of 99.56%, training on the Penn Arabic Treebank and testing on the test set of the CoNLL 2007 shared task data set yields an accuracy of 99.34%. This is good performance given that the percentage of unknown words in the latter experiment is 12.59% compared to 8.55% in the former experiment. This means that our tokenizer is capable of handling texts that have a fair amount of unknown words.

5.3.4.1.2 Stemming

Since stemming involves removing all the inflectional prefixes and suffixes from the words, and since inflectional affixes are not demarcated in the PADT data set used in the CoNLL shared tasks, there is no way to know the exact accuracy of the stemming process in that specific experiment, but since stemming is a by-product of segmentation, and since segmentation in general reaches an accuracy in excess of 98%, stemming should be trusted as an accurate process.

5.3.4.1.3 Part of speech tagging

Before discussing part of speech tagging in the real-world experiment, I need to consider the performance of the tagger on gold standard data (i.e. data with gold standard tokenization). For the gold standard data I train a tagger on the CoNLL 2007 training data and tagset and test on the test set. The experiment results are shown in Table 5.8. The experiment yields an accuracy of 96.39% on all tokens. Known tokens reach an accuracy of 97.49% while unknown tokens reach an accuracy of 81.48%. These numbers constitute the ceiling for accuracy since the real-world experiment makes use of automatic tokenization, which definitely leads to lower numbers.

Unknown	**Known**	**Total**
81.48%	97.49%	96.39%

Table 5.8: Part of speech tagging on gold standard tokenization

I cannot compare the performance of this POS tagger with the tagger which was developed in chapter 4, since both the tagset and the dataset are different.

When I run the experiment using automatic tokenization I obtain an accuracy of 95.70% which is less than 1% lower than the gold standard accuracy. This indicates that part of speech tagging has been affected by tokenization quality. The drop in quality in part of speech tagging is almost identical to the drop in quality in tokenization.

Apart from tokenization, the confusion between nouns, adjectives, and proper nouns that characterized the part of speech tagging experiment using the Penn Arabic

Treebank (chapter 4) persists also in the experiments reported here which use the Prague Arabic Dependency Treebank. The tag NOUN was correctly tagged as NOUN in 95.21% of the cases and as an adjective in 1.9% of all cases.

While some of the errors made by the part of speech tagger are due to the fact that nouns, adjectives, and proper nouns cannot be distinguished by any formal features as explained in the part of speech tagging chapter (section 4.4.1.1.1), a large number of the nominal class annotation can hardly be justified. For example, the expression الاتحاد الأوروبي (Eng. the European Union) is annotated once in the training data as proper noun and adjective, and another time as a noun and adjective. A similar confusion holds for the names of the months and the weekdays, which are sometimes tagged as NOUN and sometimes as proper nouns. Some other examples of the data are the following:

- In *dwl >wrwbyp w >mrykyp* (Eng. European and American States), the token *>mrykyp* (Eng. American) is tagged as a noun.
- In *AlEqwbAt AltjAryp w AlmAlyp* (Eng. commercial and financial sanctions), the token *mAlyp* (Eng. financial) is tagged as a noun.
- In *AlslTp AlErAqyp Alm&qtp* (Eng. interim Iraqi authority), the token *Alm&qtp* (Eng. interim) is tagged as a noun.

Proper nouns are the lowest in accuracy (83.3%) among all tags since many proper nouns received the ADJECTIVE and NOUN tags for the reasons outlined above.

5.3.4.1.4. Dependency parsing

Now that I have stems, tokens, and part of speech tags, I can proceed with the parsing experiment.

The parsing experiment is the final step and the ultimate goal of the preprocessing modules I have introduced so far. In order to prepare the training data, I have replaced the lemmas in the training set with the stems since I do not have access to lemmas in real-world experiments, and I have decided to use stems instead of lemmas in the test set. While this introduces an automatic element in the training set, it guarantees the similarity between the features in the training set and those in the test set.

In order to discover whether the fine-grained POS tagset is necessary, I have run two parsing experiments using gold standard parts of speech with stems instead of lemmas, but without any of the linguistic features included in the gold standard: the first experiment has the two distinct part of speech tags and the other one has only the coarse-grained part of speech tags. Table 5.9 outlines the results.

	LAS	**UAS**	**Label accuracy**
CPOS+POS	72.54%	82.92%	84.04%
CPOS	73.11%	83.31%	84.39%
CoNLL 2007	74.75%	84.21%	85.73%

Table 5.9: 1 POS tagset versus 2 POS tagsets

As can be seen from table 5.9, using two part of speech tagsets harms the performance of the dependency parser. While the one-tag dependency parser obtains a Labeled Accuracy Score of 73.11%, the number goes down to 72.54% when I used the fine-grained part of speech set. In unlabeled attachment score, the one tag parser achieves an accuracy of 83.31% compared to 82.92% on two tag parser. The same is also true for Label Accuracy Score as the numbers go down from 84.39% when using only one tagset compared to 84.04% when using two tagsets. This means that the fine-grained tagset is not needed to perform real world parsing. I have thus decided to use the coarse-grained tagset in the two positions of the part of speech tags. We can also see that this setting produces results that are 1.64% lower than those of the Single MaltParser results reported in the CoNLL 2007 shared task in terms of Labeled Accuracy Score. The difference can be attributed to the lack of linguistic features, vocalization, and the use of stems instead of lemmas. The LAS of 73.11% now constitutes the upper bound for real world experiments where also parts of speech and tokens have to be obtained automatically (since vocalization has been removed, linguistic features have been removed, and lemmas have been replaced with automatic stems). It should be noted that my experiments, with the complete set of gold standard features, achieve higher results than those reported in the CoNLL 2007 shared task: a LAS of 74.77 (here) versus a LAS of 74.75 (CoNLL, 2007). This may be attributed to the change of the parser since I use the 1.3.1 version whereas the parser used in the 2007 shared task was the 0.4 version.

Using the settings above, I have run an experiment to parse the test set, which is now automatic in terms of tokenization, lemmatization, and part of speech tags, and in the absence of the linguistic features that enrich the gold standard training and test sets. Table 5.10 presents the results of this experiment.

	Completely Automatic	Gold Standard
Labeled Attachment Score	63.10%	73.11%
Unlabeled Attachment Score	72.19%	83.31%
Label Accuracy	82.61%	84.39%

Table 5.10: Automatic dependency parsing experiment

The LAS drops more than 10 percentage points from 73.11 to 63.10. This considerable drop in accuracy is expected since there is a mismatch in the tokenization which leads to mismatch in the sentences. The 30 errors in tokenization affect 21 sentences out of a total of 129 in the test set. When I evaluate the dependency parsing output on the correctly tokenized sentences only, I obtain much better results (as shown in Table 5.11). Labeled Attachment Score on correctly tokenized sentences is 71.56%, Unlabeled Attachment Score 81.91%, and Label Accuracy Score is 83.22%. This indicates that no good quality parsing can be obtained if there are problems in the tokenization. A drop of a half percent in the quality of tokenization causes a drop of ten percentage points in the quality of parsing, whereas automatic POS tags and stemming, and the lack of linguistic features do not cause the same negative effect.

	Correctly-tokenized Sentences	Incorrectly-Tokenized Sentences
Labeled Attachment Score	71.56%	33.60%
Unlabeled Attachment Score	81.91%	38.32%
Label Accuracy Score	83.22%	80.49%

Table 5.11: Dependency parsing Evaluation on Correctly vs. Incorrectly Tokenized Sentences

While correctly tokenized sentences yield results that are not extremely different from those using gold standard information, and the drop in accuracy in them can be attributed to the differences introduced through stemming and automatic parts of speech as well as the absence of the linguistic features, incorrectly tokenized sentences show a completely different picture as the Labeled Attachment Score now plummets to 33.6%, which is 37.96 percentage points below that on correctly tokenized sentences. The Unlabeled Attachment Score also drops from 81.91% in correctly tokenized sentences to 38.32% on incorrectly tokenized sentences with a difference of 43.59 percentage points.

5.3.5. Error Analysis

Considering the total number of errors, out of the 5124 tokens in the test set, there are 1425 head errors (28%), and 891 dependency errors (17%). In addition, there are 8% of the tokens in which both the dependency and the head are incorrectly assigned by the parser. The POS tag with the largest percentage of head errors is the Adverb (D) with an error rate of 57%, followed by Preposition (P) at 34%, and Conjunctions at 34%. The preposition and conjunction errors are common among all experiments: those with gold standard and those with automatic information. These results also show that assigning the correct head is more difficult than assigning the correct dependency. This is reasonable since some tokens will have specific dependency types. Also, while there are a limited number of dependency relations, the number of potential heads is much larger.

If we look at the lexicon and examine the tokens in which most errors occur, we can see one conjunction and five prepositions. The conjunction *w* (Eng. and) tops the list, followed by the preposition *l* (Eng. for, to), followed by the preposition *fy* (Eng. in), then the preposition *b* (Eng. with), then the preposition *ElY* (Eng. on), and finally the preposition *mn* (Eng. from, of).

I conclude this section by examining a very short sentence in which we can see the effect of tokenization on dependency parsing. Table 5.12 is a sentence that has an instance of incorrect tokenization.

Arabic	المساعدات الأمريكية الاستثنائية لمصر بليون دولار حتي آذار
English	The American exceptional aid to Egypt is a billion dollars until March.
Buckwalter (Gold Standard Tokenization)	AlmsAEdAt Al>mrykyp AlAstvnA}yp l mSr **blywn** dwlAr HtY |*Ar
Buckwalter (Automatic Tokenization)	AlmsAEdAt Al>mrykyp AlAstvnA}yp l mSr **b lywn** dwlAr HtY |*Ar

Table 5.12: A sentence showing the effect of tokenization

The sentence has 8 words (orthographic units) one of which comprises two tokens. The word *lmSr* comprises a preposition *l*, and the proper noun *mSr* (Eng. Egypt). The tokenizer succeeds in splitting the word into two tokens, but it fails on the word *blywn* (Eng. billion) and splits it into two tokens *b* and *lywn*. The word is ambiguous between *blywn* (Eng. one billion) and *b+lywn* (Eng. in the city of Lyon), and since the second solution is much more frequent in the training set, it is the one incorrectly selected by the tokenizer.

This tokenization decision leads to an ill-alignment between the gold standard sentence and the automatic one as the gold standard has 8 tokens while the automatically produced one has 9. This thus affects the POS tagging decisions as *blywn*, which in the gold standard is a NOUN, has been now tagged as b/PREPOSITION and lywn/PROPER_NOUN. This has also affects the assignment of heads and dependency relations, as can be seen in figures 5.3 and 5.4. While *blywn* is a predicate dependent on the root of the sentence, it has been annotated as two tokens: b is a preposition dependent on the subject, and *lywn* is an attribute dependent on *b*.

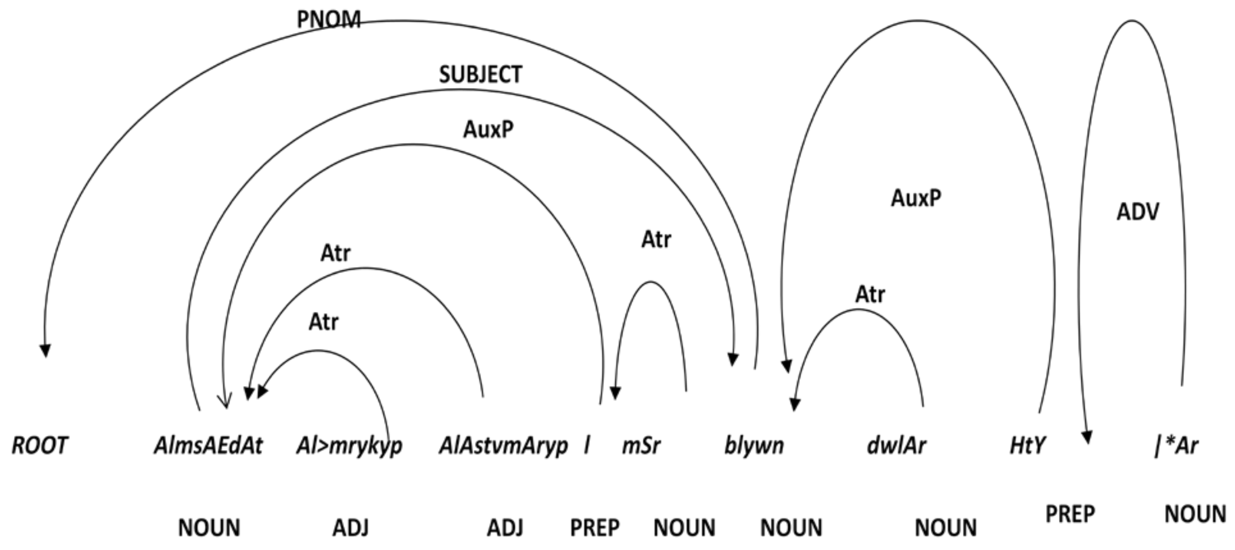

Figure 5.3: A sentence in the gold standard annotation

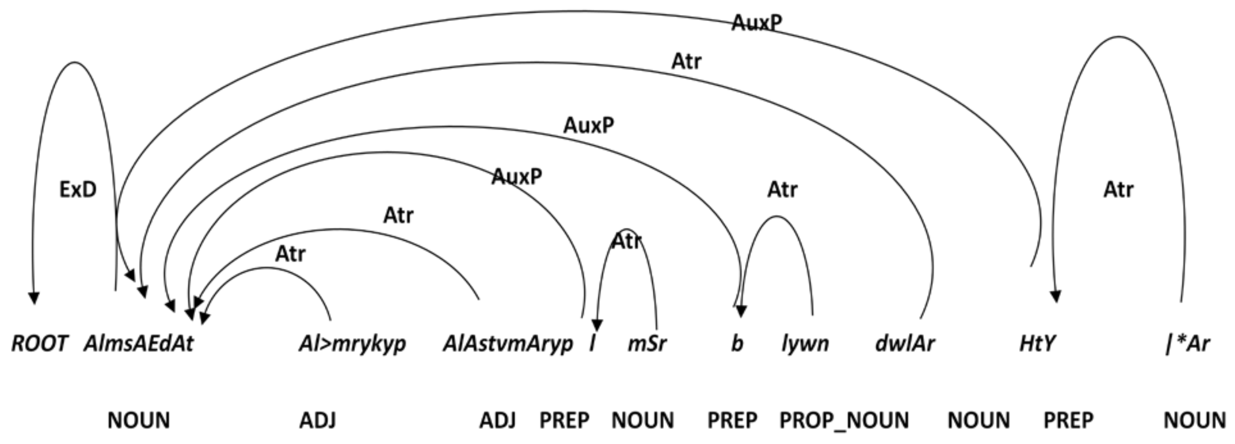

Figure 5.4: The same sentence in Figure 5.3 as produced by the parser, showing the effect of wrong tokenization

When the tokenization is correct, even with the absence of gold standard features and vocalization, the errors are not as many as can be seen when there is incorrect tokenization, as in Figure 5.5. One difference is that the head of the first token, *AlmsAEdAt*, is the root rather than the predicate, *blywn*, as in the gold standard annotation.

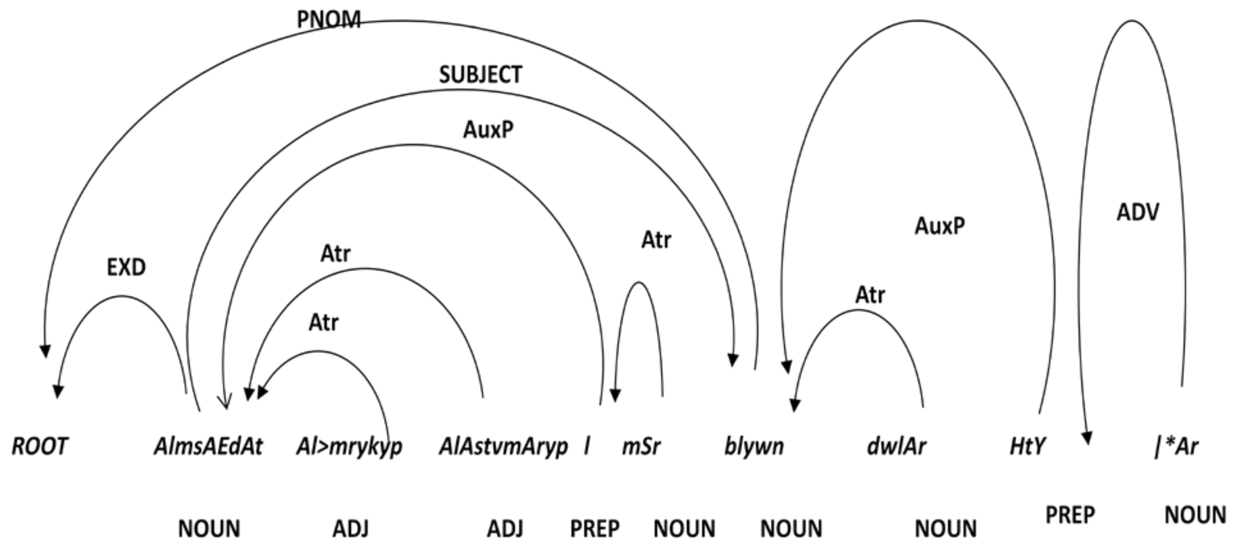

Figure 5.5: Parser output when the tokenization is correct

5.3.6. Using the Penn Tags

So far, I have used only the POS tags of the PADT, and have not discussed the possibility of using the Penn Arabic Treebank tagset for which a tagger was developed in chapter 4 of this thesis. We have seen in chapter 4 that using the Penn full tagset with more information gives better results than using either the segment tags or the Habash and Rambow tagset. The wealth of information in the Penn tagset may also have positive effect on dependency parsing. Table 5.13 shows the effect of using the Penn tagset with the gold standard full-featured dataset in three different experiments as compared with the PADT tagset:

(1) The original Unvocalized Experiment with the full set of features and gold standard components. The Penn tagset is not used in this experiment, and it is provided for reference purposes only.

(2) Unvocalized experiment with Penn tags as CPOS tags. In this experiment, the Penn tagset is used instead of the coarse grained POS tagset, while the fine-grained pos tagset remains unchanged.

(3) Using Penn tags as fine grained POS tags, while the CPOS tags remain unchanged.

(4) Using the Penn POS tags in both positions.

In the four experiments, the only features that change are the POS and CPOS features.

Experiment	**LAS**	**UAS**
Unvocalized Original	74.16%	83.53%
Using Penn Tags as CPOS tags	74.12%	83.43%
Using Penn tags as POS	72.40%	81.79%
Using Penn tags in both positions	69.63%	79.33%

Table 5.13: Using the full tagset with the PADT dataset

As can be seen from Table 5.13, in all three cases the Penn tagset produces lower results than the PADT tagset. The reason for this may be that the tagset is automatic in both cases, and the perfect accuracy of the PADT helps the classifier embedded in the MaltParser parser to choose the correct label and head. The results also show that when we use the Penn tagset as the CPOS tagset, the results are almost no different from the gold standard PADT tagset (74.12% vs. 74.16%). The fact that the Penn tagset does not harm the results encourages the inclusion of the Penn tags as CPOS tags in the automatic experiments that have been used throughout this chapter. The worst results are those obtained by using the Penn tags in both positions (POS and CPOS).

Using the Penn tagset with the reduced experiments, those without the linguistic features, gives a different picture from that in the full standard experiments, as detailed in table 5.14.

Experiment	**LAS**	**UAS**
Reduced with both PADT tags	72.54%	82.92%
Reduced with Penn tags as CPOS	73.09%	83.16%
Reduced with Penn tags as CPOS and automatic tokenization	63.11%	72.38%

Table 5.14: Including the Penn full tagset in the reduced experiments

While the Penn tagset does not help improve parsing accuracy with the full-featured parsing experiments, it helps with the reduced experiments. While the experiment without the Penn tags score an LAS of 72.54%, replacing the CPOS tags in this experiment with the Penn tagset raises the accuracy to 73.09%, with an increase of 0.55%. This may be due to the fact that the full tagset gives more information that helps the parser. The increase is not as noticeable in the automatic tokenization experiment where the accuracy minimally changes from 63.10% to 63.11%.

5.3.7. Effect of Vocalization

I have stated in the methodology section that I use unvocalized data since naturally occurring Arabic is hardly vocalized. While this is a reasonable approach, it is worth checking the effect of vocalization on dependency parsing. Table 5.15 presents the results of vocalization effect in three experiments: (a) All the gold standard features with vocalization. This is the experiment reported in the literature on Arabic dependency parsing in CoNLL (2007), (b) All the gold standard features without the vocalization, (c) All gold standard features except for vocalization which is automatic, and (d) the automatic experiment with automatic vocalization. The vocalizer in experiments in the latter 2 experiments, c and d, is trained on the PADT training section using the same settings in chapter 3, and has an accuracy of 93.8% on the PADT test set.

	Experiment	**LAS**	**UAS**
A	Fully Gold Standard Vocalized	74.77%	84.09%
B	Fully Gold Standard Unvocalized	74.16%	83.53%
C	Full-featured with automatic vocalization	74.43%	83.88%
D	Completely automatic (with automatic vocalization)	63.11%	72.19%
E	Completely automatic without vocalization	63.11%	72.38%

Table 5.15: Vocalization Effect on Dependency Parsing

As can be seen from Table 5.15, gold standard vocalization with gold standard features produces the best results (LAS: 74.77%) followed by the same settings, but with automatic vocalization with a LAS of 74.43%, then unvocalized gold standard with a LAS of 74.16%. The fact that even automatic vocalization produces better results than unvocalized text given the same conditions, in spite of a token error rate of 6.2%, may be attributed to the ability of vocalization to disambiguate text even when it is not perfect. We can also notice that the LAS for the Automatic experiment is the same whether or not vocalization is used. This indicates that vocalization, in spite of its imperfections, does not harm performance, although it also does not help the parser. Tokenization sets a ceiling for parsing accuracy.

5.4. Conclusion

I have presented an experiment in real world dependency parsing of Arabic using the same data, algorithm and settings used in the CoNLL (2007) shared task on dependency parsing. The real world experiment included performing tokenization, stemming, and part of speech tagging of the data before it was passed to MaltParser, a data-driven dependency parser.

Tokenization was performed using the memory-based segmenter/tokenizer developed in chapter 2 of the thesis, and it reached an accuracy of 99.34% on the CoNLL 2007 test set. Stemming was based on the same segmenter, but no direct evaluation can be performed due to the absence of stem markers in the PADT data. I performed stemming rather than lemmatization due to the many problems and difficulties involved in obtaining the lemmas like, for example, obtaining the singular form of the broken plural.

Part of speech tagging scored 96.39% on all tokens on gold standard tokenization, but the accuracy dropped to 95.70% when I used automatic tokenization. I also found that using the coarse grained POS tagset (CPOS) alone yielded better results than using it in combination with the fine-grained POS tagset (POS).

The tokens, stems, and CPOS tags were then fed into the dependency parser, but the linguistic features were not since it was not feasible to obtain these automatically. The parser yielded a Labeled Accuracy Score of 63.10%, more than 10% below the accuracy obtained on when all the components are gold standard. The main reason behind the accuracy drop is the tokenization module, since tokenization is responsible for creating the nodes that carry syntactic functions. Since this process was not perfect, many nodes were wrong, and the right heads were missing. When I evaluated the parser on those sentences that were correctly tokenized, I obtained a Labeled Accuracy Score of 71.56%. On incorrectly tokenized sentences, however, the LAS score drops to 33.60%.

I have also tried to incorporate the full tagset of the Penn Arabic Treebank, and found that the ATB full tagset improves parsing results minimally in the automatic experiments, but that, in the gold standard experiments with the complete set of gold standard features, the results are worse.

Vocalization leads to slightly better results compared with unvocalized data even though vocalization is not perfect. The reason for this is that vocalization helps disambiguate the lexical items. In the real world experiment, the effect of vocalization was neutralized by the effect of tokenization, and the inclusion of vocalization did not improve the results.

These results show that tokenization, a byproduct of segmentation, is the major hindrance to obtaining high quality parsing in Arabic. Arabic computational linguistics should thus focus on ways to perfect segmentation, or try to find ways to parsing without having to perform tokenization.

Chapter 6: Conclusions and Future Work

6.1. Conclusions

In order to list the conclusions of this thesis, I shall repeat the questions raised in the introduction, and answer them in light of the results obtained from individual chapters. I will then follow the questions with a general conclusion.

(1) How can we perform orthographic unit segmentation of Arabic, a morphologically rich language, that is suitable for grammatical analysis?

I have presented segmentation, in which each segment of the word, whether inflectional or clitical, is marked with a boundary, and I have derived tokenization, in which only syntactically functional segments are marked. Each of these two processes has it own use. While both can be used for part of speech tagging, syntactic analysis requires tokenization. While part of speech tagging can be performed without any pre-processing, parsing requires the splitting of tokens since syntactic relations hold between tokens rather segments or orthographic units. Segmentation was cast as a classification problem using supervised machine learning, and I obtained results on segmentation of 98.23% average across five folds. Tokenization scored an accuracy of 99.36%.

(2) How can we perform vocalization in a Semitic language whose orthography is impoverished? What are the best settings? How much context do we need? Does word segmentation help diacritic restoration?

I have treated vocalization as classification in the same way as I did segmentation, using the same machine learning approach. The best settings involved the characters of the word itself, the stemmed context and the previous decisions. I have found that there is a considerable difference between the quality of vocalization on known versus unknown words. I have also found that segmentation and vocalization are separate processes, and that automatic vocalization and automatic segmentation do not help each other.

(3) Can Arabic part of speech tagging be performed without using gold standard word segmentation? Does segmentation help POS tagging? If yes, which type of segmentation? Does vocalization, whether gold standard or automatic, help POS tagging?

I have performed part of speech tagging with gold standard segmentation, automatic segmentation, and whole words without any pre-processing. I have found that while gold standard segmentation yields the best results in general, one does not need to perform segmentation to obtain high quality POS tagging as whole word tagging outperforms that using automatic segmentation (94.74% vs. 93.74%). One also does not need a separate process of tokenization to perform token-based tagging as tokens and token tags can be derived, through rules, from segmentation-based tagging.

(4) What is the effect of orthographic enrichment, or the lack thereof, on Arabic dependency parsing? What is the effect of using automatic enrichment and POS tags on the parsing task?

I have, for the first time, used a real-world pipeline to perform dependency parsing of Arabic. While gold standard information yields results that are comparable to those on other languages, real world Arabic dependency parsing suffers from the need to perform automatic tokenization. While tokenization itself is very accurate, any error in tokenization leads to multiple errors in parsing. Obtaining perfect tokenization is the real challenge to Arabic dependency parsing. Vocalization, on the other hand, helps dependency parsing when it is gold standard. Automatic vocalization helps dependency parsing only when the tokenization and the POS tags are gold standard. When using a completely automatic pipeline, automatic vocalization neither improves nor hurts the performance of the parsing process.

The overall conclusion of this thesis is that orthographic enrichment is useful in and of itself, but not necessarily for grammatical analaysis. Segmentation can be used in lexical acquisition, but it has not proved very useful for POS tagging as Whole Word tagging performs better. Vocalization can be used in Text to Speech systems, but its contribution to POS tagging is negative, and its contribution to parsing is minimal, if not non-existent. Tokenization is necessary for parsing, as grammatical relations hold between tokens, but it is also the main source of errors in the dependency parsing experiments introduced in this thesis.

6.2. Accomplishments

This thesis has contributed to Arabic computational linguistics. The following accomplishments are worth highlighting:

- The segmenter and tokenizer introduced in chapter 2 are the most accurate tools published so far, with segmentation scoring an average accuracy of 98.23% and tokenization averaging an accuracy of 99.36% on all words.

- The vocalize introduced in chapter 3 performs within the range of published results, although it uses only lexical input and does not require any morphosyntactic pre-processing,
- The segmentation-based POS tagger introduced in chapter 4 is the first one to use automatic segmentation. The same holds true for the tokenization-based one. All the previous works on Arabic POS tagging used gold standard tokenization.
- This thesis introduced the first word-based POS tagger of Arabic. I have proved that Arabic POS tagging can be performed without any sort of pre-processing, and with accuracies very close to those reported using gold standard tokenization and segmentation.
- The thesis has introduced the first real-world experiment in Arabic dependency parsing. The experiment involved automatic tokenization, stemming, POS tagging, and vocalization.

6.3. Future Directions

Future research should address the problems left unsolved in this thesis. One problem that has not been addressed is how to handle non-standard orthography which is common in both formal and informal written Arabic. Research in this area should handle the restoration of standard orthography and / or treating suboptimal orthography as is. A special problem here is word boundary detection.

Another problem left unsolved is that of diglossia. Any comprehensive computational treatment of Arabic should aim at analyzing any raw text regardless of the dialect in which it is written. This requires tools that handle those variant forms of the language simultaneously since the different dialects co-exist and can hardly be separated. It is worth mentioning that Chiang et al (2005) worked on parsing the Levantine dialect, but their work does not seem to have been followed up.

The effect of genre on the quality of the tools developed here is a worthy cause. The data used throughout this thesis is only newswire, and there will definitely be accuracy loss when we try to use them to process other genres, e.g. literary or religious works. The main obstacle to this project is the unavailability of annotated data in other domains than newswire.

Another direction is the interaction between the different levels of grammatical analysis. While the thesis investigated the interaction between segmentation and vocalization, and to a lesser extent POS tagsest on parsing, the impact of parsing on POS tagging has not been discussed.

Arabic NLP is still exploring the fundamental questions like POS tagging and parsing while in a lanaguage like English higher levels of analysis and generation are being, or have been, explored, but unfortunately, it is those fundamental questions that are most problematic in Arabic. It is my hope that NLP practioners will overcome these obstacles in order to handle higher levels of semantic analysis.

References

Abboud, Peter F. and Mccarus, Ernest N. (1999). *Elementary Modern Standard Arabic 1. Pronunciation and Writing*. Cambridge University Press.

AbdulJaleel, Nasreen and Larkey, Leah S. (2003). *Statistical Transliteration for English-Arabic Cross Language Information Retrieval*, CIKM 2003: Proceedings of the Twelfth International Conference on Information and Knowledge Management, New Orleans, LA.

Abu-Rabia, Salim (1995). *Learning to read in Arabic: Reading, syntactic, orthographic and working memory skills in normally achieving and poor Arabic readers*, Reading Psychology: An International Quarterly 16: 351–394.

Abu-Rabia, Salim (1996). *The role of vowels and context in reading of highly skilled native Arabic readers*, Journal of Psycholinguistic Research 25: 629–641.

Abu-Rabia, Salim (1997). *Reading in Arabic orthography: The effect of vowels and context on reading accuracy of poor and skilled native Arabic readers in reading paragraphs, sentences, and isolated words,* Journal of Psycholinguistic Research 26: 465–482.

Abu-Rabia, Salim (1998). *Reading Arabic texts: Effects of text type, reader type and vowelization*, Reading and Writing: An Interdisciplinary Journal 10: 105–119.

Abu-Rabia, Salim (1999). *The effect of Arabic vowels on the reading comprehension of secondand sixth-grade native Arab children*, Journal of Psycholinguistic Research 28: 93–101.

Abu-Rabia, Salim (2001). *The role of vowels in reading Semitic scripts: Data from Arabic and Hebrew*. Reading and Writing: An Interdisciplinary Journal 14: 39–59, 2001.

Abu-Rabia, Salim & Siegel, Linda (1995). *Different orthographies, different context effects: The effects of Arabic sentence context in skilled and poor readers*, Reading Psychology: An International Quarterly 16: 1–19.

Al-Hamalawy, Ahmad (1998). *شذا العرف في فن الصرف* (A summary of Arabic morphology). Dar Al-Bayruti Publishing, Damascus.

Al-Najjar, Ashwaq (2006). *دلالة اللواصق التصريفية في اللغة الغربية (The semantics of inflectional affixes in the Arabic language)*. Djlp Publishing, Amman, Jordan.

Anis, Ibrahim (1958). *رأي في الإعراب الحركات* (The vowel-based *I'rab* re-considered). In mjlp Allgp AlErbyp. vol. 10, 55-56.

Bellamy, James. *The Arabic Alphabet.* In Senner, Wayne (ed.) (1989). *The Origins of Writing*. University of Nebraska Press, Lincoln and London.

Brants, Thorsten (1995). *Tagset Reduction without Information Loss*. In Proceddings of ACL-95, Cambride, MA.

Buchholz, Sabine and Marsi, Erwin (2006). *CoNLL-X shared task on multilingual dependency parsing*. In Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL), pages 149–164.

Buckwalter, Tim (2002). *Arabic morphological analyzer version 1.0*. Linguistic Data Consortium. LDC Catalogue Number: LDC2002L49.

Chang, Chih-Chung and Lin, Chih-Jen, 2001. *LIBSVM: A Library for Support Vector Machines*. Software available at http://www.csie.ntu.edu.tw/cjlin/libsvm.

Chang, Ming-Wei; Do, Quang and Roth, Dan (2006). *A Pipeline Model for Bottom-up Dependency Parsing*. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL)*.

Chiang, David; Diab, Mona; Habash, Nizar; Rambow, Owen and Shareef, Safiullah (2005). *Parsing Arabic Dialects*. Final Report, 2005 JHU Summer Workshop.

Covington, M. A. (2001). A fundamental algorithm for dependency parsing. Proceedings of the 39th Annual ACM Southeast Conference, pp. 95-102, as cited in Nivre, Joakim (2005). Inductive Dependency Parsing of Natural Language Text, an Unpublished Doctoral Thesis, Vaxio University.

Crystal, David (2008). A Dictionary of Linguistics and Phonetics. 6th edition, Wiley-Blackwell.

Cuvalay-Haak, Martine (1997). *The Verb in Literary and Colloquial Arabic*. Functional Grammar Series 19. Mouton de Gruyter.

Daelemans, Walter; van den Bosch, Antal and Zavrel, Jakub (1999). *Forgetting exceptions is harmful in language learning*. Machine Learning, 34:11–43. Special Issue on Natural Language Learning.

Daelemans, Walter; Zavrel, Jakub; van der Sloot, Ko and van den Bosch, Antal (2007). *TiMBL: Tilburg memory based learner – version 6.1 – reference guide*. Technical Report ILK 07-07, Induction of Linguistic Knowledge, Computational Linguistics, Tilburg University.

Darwish, Kareem (2002). Building a Shallow Arabic Morphological Analyzer in One Day. *Proceedings of the Workshop on Computational Approaches to Semitic Languages*, pages 47−54.

Debili, Fahti; Achour, Hadhebi and Emna Souissi (2002). *l'etiquetage grammatical a la voyellation automatique de l'arabe*. Technical report, Correspondances de l'Institut de Recherche sur le Maghreb Contemporain.

Diab, Mona (2007). *Towards an optimal POS tag set for Arabic processing*. In Proceedings of the International Conference on Recent Advances in Natural Language Processing, RANLP 2007, pages 157–161, Borovets, Bulgaria.

Diab, Mona; Hacioglu, Kadri and Jurafsky, Daniel (2004). *Automatic Tagging of Arabic Text: From Raw Text to Base Phrase Chunks*. In *Proceedings of Human Language Technology-North American Association for Computational Linguistics (HLT-NAACL), 2004.*

Dickinson, Markus and Jochim, Charles (2010). Evaluating Distributional Properties of Tagsets. *Proceedings of the 7th Language Resources and Evaluation Conference (LREC 2010).* Valletta, Malta.

Drozdik, Ladislav (2001). *Modern Written Arabic*. Publishing House of the Slovak Academy of Sciences.

Elhady, Mohamed and Al-Toby, Amjad (2009). *Part of Speech (POS) Tag Sets Reduction and Analysis Using Rough Set techniques*. Proceddings of the 12th International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing, Delhi, india.

Faber, Alice. *Genetic Subgrouping of the Semitic Languages*. In Hetzron, Robert (ed.) (1997). *The Semitic Languages*. Routledge, London.

Ferguson, Charles (1959). *Diglossia*. Word 15: 325-40. In Warshaugh, Ronald (2006). *An Introduction to Sociolinguistics*. Fifth edition, Blackwell Publishing.

Gal, Ya'akov (2002). *An HMM approach to vowel restoration in Arabic and Hebrew*. In Proceedings of the ACL-02 Workshop on Computational Approaches to Semitic Languages, Philadelphia, PA.

Graff, David (2007). Arabic Gigaword Third Edition. Linguistic Data Consortium. Philadelphia, USA.

Habash, Nizar and Rambow, Owen (2005). *Arabic Tokenization, Morphological Analysis, and Part-of-Speech Tagging in One Fell Swoop*. In Proceedings of the Conference of American Association for Computational Linguistics (ACL'05).

Habash, Nizar, Ryan Gabbard, Owen Rambow, Seth Kulick and Mitch Marcus (2007). *Determining Case in Arabic: Learning Complex Linguistic Behavior Requires Complex Linguistic Features*. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Prague, Czech Republic.

Habash, Nizar; Soudi, Abdelhadi and Buckwalter, Tim. *On Arabic Transliteration*. Book Chapter. In *Arabic Computational Morphology: Knowledge-based and Empirical Methods*. Editors Antal van den Bosch and Abdelhadi Soudi. 2007.

Habash, Nizar; Faraj, Reem and Roth, Ryan (2009). *Syntactic Annotation in the Columbia Arabic Treebank*. In Proceedings of the 2nd International Conference on Arabic Language Resources and Tools (MEDAR), Cairo, Egypt.

Hajič, Jan; Smrž, Otakar; Zemánek, Petr; Šnaidauf, Jan, and Beška, Emanuel (2004). *Prague Arabic Dependency Treebank: Development in Data and Tools*. In *Proceedings of the EMLAR International Conference on Arabic Language Resources and Tools*, pages 110-117, Cairo, Egypt, September 2004.

Hall, Johan; Nilsson, Jens; Nivre, Joakim; Eryigit, Gülsen; Megyesi, Beáta; Nilsson, Mattias and Saers, Markus (2007). Single Malt or Blended? A Study in Multilingual Parser Optimization. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, 933-939.

Hassan, Tammam (1979). *اللغة العربية معناها ومبناها (The Arabic language, its forms and functions)*. The Egyptian Public Book Authority.

Haywood, JA and Nahmad, HM. (1965). *A new Arabic grammar of the written language*. Lund Humphries London.

Holes, Clive (2004). *Modern Arabic: Structures, Functions, and Varieties*, Georgetown Classics in Arabic Language and Linguistics. Georgetown University Press.

Hudson, Richard A. (1984). Word Grammar. Blackwell, as cited in Nivre, Joakim (2005). Inductive Dependency Parsing of Natural Language Text.

Ibn Manzur, Jamulddin (1968). *Lisan Al-Arab*. Sadir Publishing House, Beirut.

Jurafsky, Daniel, and Martin, James H. (2009). *Speech and Language Processing: An Introduction to Natural Language Processing, Speech Recognition, and Computational Linguistics*. 2nd edition. Prentice-Hall.

Kashani, Mehdi; Popwich, Fred and Sadat, Fatiha (2006). *Automatic Transliteration of Proper Nouns from Arabic to English*. The Challenge of Arabic for NLP/MT. International conference at the British Computer Society, London, 23 October 2006; pp.76-83.

Kaye, Alan S. and Rosenhaus, Judith. *Arabic Dialects and Maltese*. In Hetzron, Robert (ed.) (1997). *The Semitic Languages*. Routledge, London. pp. 263-311

Kirchhoff, Katrin; Bilmes, Jeff; Henderson, John; Schwartz, Richard; Noamany, Mohamed; Schone, Pat; Ji, Gang ; Das, Sourin; Egan, Melissa ; He, Feng; Vergyri, Dimitra; Liu, Daben and Duta, Nicolae (2002). *Novel speech recognition models for Arabic* - final report of the JHU summer workshop. Technical report, Johns Hopkins University.

Kübler, Sandra; McDonald, Ryan and Nivre, Joakim (2009). *Dependency Parsing*. Morgan Claypool.

Lee, Young-Suk; Papineni, Kishore; Roukos, Salim; Emam, Osama and Hassan, Hany (2003). *Language Model Based Arabic Word Segmentation*. Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, July 2003, pp. 399-406.

Lipinsky, Edward (2001). *Semitic Languages: Outline of a Comparative Grammar*. 2nd Edition. Orientalia Lovanienisa Analecta, 80.

Maamouri, Mohamed and Bies, Ann (2004) *Developing an Arabic Treebank: Methods, Guidelines, Procedures, and Tools.* In Proceedings of the Workshop on Computational Approaches to Arabic Script-based Languages, COLING 2004, Geneva, August 28, 2004.

Manning, Christopher D.; Raghavan, Prabhakar and Schütze, Hinrich (2009). *Introduction to Information Retrieval*, Cambridge University Press. Online edition available at http://nlp.stanford.edu/IR-book/pdf/irbookonlinereading.pdf

Manning, Christopher D. and Schütze, Hinrich (1999). *Foundations of Statistical Natural Language Processing*. MIT Press.

Matthews, Peter (2007). *The Concise Oxford Dictionary of Linguistics*. Second Edition. Oxford University Press.

McDonald, Ryan; Lerman, Kevin and Pereira, Fernando (2006). *Multilingual dependency analysis with a two-stage discriminative parser*. CoNLLX shared task on multilingual dependency parsing. In Proceedings of the 10th Conference on Computational Natural Language Learning

McDonald, Ryan; Pereira, Fernando; Ribarov, Kiril and Hajič , Jan (2005). *Non-projective Dependency Parsing using Spanning Tree Algorithms*. Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP), pages 523–530, Vancouver, October 2005.

Nelken, Rani and Shieber, Stuart (2005). *Arabic vocalization using weighed finite-state transducers*. In Proceedings of the ACL Workshop on Computational Approaches to Semitic Language, Ann Arbor, MI.

Nivre, Joakim (2003). *An Efficient Algorithm for Projective Dependency Parsing*. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT 03)*, Nancy, France, 23-25 April 2003, pp. 149-160.

Nivre, Joakim; Hall, Jonathan; Nilsson, Jens; Eryigit, Gülsen and Marinov, Svetsolav (2006). Labeled Pseudo-Projective Dependency Parsing with Support Vector Machines. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL)*

Nivre, Joakim; Hall Johan, and Nilsson, Jens (2006). *Maltparser: A data-driven parser-generator for dependency parsing.* In Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC), pages 2216–2219.

Nivre, Joakim; Hall, Johan; Kübler, Sandra; McDonald, Ryan; Nilsson, Jens; Riedel, Sebastian, and Yuret, Deniz. (2007). *The CoNLL 2007 shared task on dependency parsing*. In Proceedings of the CoNLL Shared Task of EMNLP-CoNLL 2007, pages 915–932

Nivre, Joakim (2005). Inductive Dependency Parsing of Natural Language Text, an Unpublished Doctoral Thesis, Växjö University.

Rogers, Henry (2004). *Writing Systems: A Linguistic Approach* (Blackwell Textbooks in Linguistics), Wiley-Blackwell

Sgall, P.; Hajicova, E. and Panavova, J. (1986). The Meaning of The Sentence in Its Pragmatic Aspect. Reidel, as cited in Nivre, Joakim (2005). Inductive Dependency Parsing of Natural Language Text, an Unpublished Doctoral Thesis, Vaxio University.

Smrž, Otakar; Šnaidauf, Jan and Zemánek, Petr (2002). *Prague Dependency Treebank for Arabic: Multi-Level Annotation of Arabic Corpus*. In *Proceedings of the International Symposium on Processing of Arabic*, pages 147-155, Manouba, Tunisia, April 2002.

Sproat, Richard; Tao, Tao and Zhai, ChengXiang (2006). *Named Entity Transliteration with Comparable Corpora*. COLINGACL, Sydney, Australia.

Stalls, Bonnie G. and Knight, Kevin (1998). *Translating Names and Technical Terms in Arabic Text*. In Proceedings of the COLING/ACL Workshop on Computation Approaches to Semitic Languages.

Tapanainen, P. and Jarvinen, T. (1997). A non-projective dependency parser. Proceedings of the 5th Conference on Applied Natural Language Processing, pp. 64-71, as cited in

Nivre, Joakim (2005). Inductive Dependency Parsing of Natural Language Text, an Unpublished Doctoral Thesis, Växjö University.

Watson, Janet C. E. (2002). *The Phonology and Morphology of Arabic*, New York: Oxford University Press

Wunsch, Holger; Kübler, Sandra and Cantrell, Rachael (2009). *Instance Sampling Methods for Pronoun Resolution*. Proceedings of RANLP 2009, Borovets, Bulgaria.

Zitouni, Imed; Sorensen, Jeffrey; and Sarikaya, Ruhi (2006). *Maximum entropy based restoration of Arabic diacritics*. In Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, COLING-ACL-2006, Sydney, Australia.

Appendix: The Arabic Treebank Full Tagset

Tag	Explanation
ABBREV	Abbreviation
ADJ	Adjective
ADV	Adverb
CASE_INDEF_ACC	Accusative Indefinite Case
CONJ	Conjunction
CV	Imperative Verb
CVSUFF_DO:1P	Imperative Verb 1st Person Plural Object Pronoun
CVSUFF_DO:3MS	Imperative Verb 3rd Person Singular Masculine Object Pronoun
CVSUFF_SUBJ:2FS	Imperative Verb 2nd Person Feminine Singular Pronoun
CVSUFF_SUBJ:2MP	Imperative Verb 2nd Person Plural Pronoun
CVSUFF_SUBJ:2MS	Imperative Verb 1st Person Plural Subject Pronoun
DE	Demonstrative
DEM_PRON_F	Demonstrative Singular Feminine
DEM_PRON_FD	Demonstrative Feminine Dual
DEM_PRON_FS	Demonstrative Singular Feminine
DEM_PRON_MD	Demonstrative Masculine Dual
DEM_PRON_MP	Demonstrative Masculine Plural
DEM_PRON_MS	Demonstrative Masculine Singular
DET	Determiner
EMPH_PART	Emphatic Particle
EXCEPT_PART	Exception Particle
FOCUS_PART	Focus Particle (like: As for...)
FUT	Future Prefix
FUT_PART	Future Particle
INTERJ	Interjection
INTERROG_PART	Interrogative Particle
IV	Imperfect Verb
IV1P	Imperfect Verb Plural 1st Person subject prefix
IV1S	Imperfect Verb Singular 1st Person subject prefix for present tense
IV2D	Imperfect Verb Dual subject prefix for present tense

Tag	Explanation
IV2FP	Imperfect Verb Plural Feminine subject prefix for present tense
IV2FS	Imperfect Verb Singular Feminine 2nd Person subject prefix
IV2MP	Imperfect Verb Plural Masculine 2nd Person subject prefix
IV2MS	Imperfect Verb Singular Masculine 2nd Person subject prefix for
IV3FD	Imperfect Verb Dual Feminine 3rd Person subject prefix
IV3FP	Imperfect Verb Plural Feminine 3rd Person subject prefix
IV3FS	Imperfect Verb Singular Feminine 3rd Person subject prefix
IV3MD	Imperfect Verb Dual Masculine 3rd Person subject prefix
IV3MP	Imperfect Verb Plural Masculine 3rd Person subject prefix
IV3MS	Imperfect Verb Singular Masculine 3rd Person subject prefix
IVSUFF_DO:1P	Imperfect Verb Object pronoun 1st person plural
IVSUFF_DO:1S	Imperfect Verb Object pronoun 1st person singular
IVSUFF_DO:2MP	Imperfect Verb Object pronoun 2nd person masculine plural
IVSUFF_DO:2MS	Imperfect Verb Object pronoun 2nd person masculine singular
IVSUFF_DO:3D	Imperfect Verb Object pronoun 2nd person dual
IVSUFF_DO:3FS	Imperfect Verb Object pronoun 3rd person feminine singular
IVSUFF_DO:3MP	Imperfect Verb Object pronoun 3rd person masculine plural
IVSUFF_DO:3MS	Imperfect Verb Object pronoun 3rd person masculine singular
IVSUFF_SUBJ:2FS_MOOD:I	Imperfect Verb Suffix Subject Feminine Singular in the indicative mood
IVSUFF_SUBJ:2FS_MOOD:SJ	Imperfect Verb Suffix Subject Feminine Singular in the subjunctive mood
IVSUFF_SUBJ:D_MOOD:I	Imperfect Verb Suffix Subject Dual in the indicative mood
IVSUFF_SUBJ:D_MOOD:SJ	Imperfect Verb Suffix Subject Dual in the

Tag	Explanation
	subjunctive mood
IVSUFF_SUBJ:FP	Imperfect Verb Suffix Subject Feminine Plural
IVSUFF_SUBJ:MP_MOOD:I	Imperfect Verb Suffix Subject Masculine Plural in the indicative mood
IVSUFF_SUBJ:MP_MOOD:SJ	Imperfect Verb Suffix Subject Masculine Plural in the subjunctive mood
IV_PASS	Passive Imperfect Verb
JUS	Jussive mood particle
LATIN	Foreign Word
NEG_PART	Negative Particle
NOUN	Noun
NOUN_PROP	Proper Noun
NSUFF_FEM_DU_ACC	Noun Suffix Feminine Dual marker
NSUFF_FEM_DU_ACC_POSS	Noun Suffix Feminine Dual marker(indicating Idafah)
NSUFF_FEM_DU_GEN	Noun Suffix Feminine Dual marker
NSUFF_FEM_DU_GEN_POSS	Noun Suffix Feminine Dual marker(General)
NSUFF_FEM_DU_NOM	Noun Suffix Feminine Dual marker (Nominative)
NSUFF_FEM_DU_NOM_POSS	Noun Suffix Feminine Dual marker(indicating Idafah)
NSUFF_FEM_PL	Noun Suffix Feminine Plural Marker
NSUFF_FEM_SG	Noun Suffix Feminine Singular Marker
NSUFF_MASC_DU_ACC	Noun Suffix Masculine Dual Marker
NSUFF_MASC_DU_ACC_POSS	Noun Suffix Masculine Dual Marker(indicating Idafah)
NSUFF_MASC_DU_GEN	Noun Suffix Masculine Dual Marker
NSUFF_MASC_DU_GEN_POSS	Noun Suffix Masculine Dual Marker(indicating Idafah)
NSUFF_MASC_DU_NOM	Noun Suffix Masculine Dual Marker
NSUFF_MASC_DU_NOM_POSS	Noun Suffix Masculine Dual Marker(indicating Idafah)
NSUFF_MASC_PL_ACC	Noun Suffix Masculine Plural Marker
NSUFF_MASC_PL_ACC_POSS	Noun Suffix Masculine Plural Marker(indicating Idafah)
NSUFF_MASC_PL_GEN	Noun Suffix Masculine Plural Marker
NSUFF_MASC_PL_GEN_POSS	Noun Suffix Masculine Plural Marker(indicating Idafah)
NSUFF_MASC_PL_NOM	Noun Suffix Masculine Plural Marker

Tag	Explanation
NSUFF_MASC_PL_NOM_POSS	Noun Suffix Masculine Plural Marker(indicating Idafah)
NUM	Number
NUMERIC_COMMA	Numeric Comma
PART	Particle
POSS_PRON_1P	Possessive Pronoun 1st Person Plural
POSS_PRON_1S	Possessive Pronoun 1st Person Singular
POSS_PRON_2FP	Possessive Pronoun 2nd Person Feminine Plural
POSS_PRON_2FS	Possessive Pronoun 2nd Person Feminine Singular
POSS_PRON_2MP	Possessive Pronoun 2nd Person Masculine Plural
POSS_PRON_2MS	Possessive Pronoun 2nd Person Masculine Singular
POSS_PRON_3D	Possessive Pronoun 3rd Person Dual
POSS_PRON_3FP	Possessive Pronoun 3rd Person Feminine Plural
POSS_PRON_3FS	Possessive Pronoun 3rd Person Feminine Singular
POSS_PRON_3MP	Possessive Pronoun 3rd Person Masculine Plural
POSS_PRON_3MS	Possessive Pronoun 3rd Person Masculine Singular
PREP	Preposition
PRON_1P	1st Person Plural Pronoun
PRON_1S	1st Person Singular
PRON_2D	2nd Person Dual Pronoun
PRON_2FP	2nd Person Feminine Plural Pronoun
PRON_2FS	2nd Person Feminine Singular Pronoun
PRON_2MP	2nd Person Masculine Plural Pronoun
PRON_2MS	2nd Person Feminine Singular Pronoun
PRON_3D	3rd Person Dual Pronoun
PRON_3FP	3rd Person Feminine Plural Pronoun
PRON_3FS	3rd Person Feminine Singular Pronoun
PRON_3MP	3rd Person Masculine Plural Pronoun
PRON_3MS	3rd Person Masculine Singular Pronoun
PUNC	Punctuation Mark
PV	Perfect Verb
PVSUFF_DO:1P	Perfect Verb Suffix 1st Person Plural Object
PVSUFF_DO:1S	Perfect Verb Suffix 1st Person Singular Object
PVSUFF_DO:2MP	Perfect Verb Suffix 2nd Person Masculine Plural Object
PVSUFF_DO:2MS	Perfect Verb Suffix 2nd Person Masculine Singular Object

Tag	Explanation
PVSUFF_DO:3D	Perfect Verb Suffix 3rd Person Masculine Dual Object
PVSUFF_DO:3FS	Perfect Verb Suffix 3rd Person Feminine Singular Object
PVSUFF_DO:3MP	Perfect Verb Suffix 3rd Person Masculine Plural Object
PVSUFF_DO:3MS	Perfect Verb Suffix 3rd Person Masculine Singular Object
PVSUFF_SUBJ:1P	Perfect Verb Suffix 1st Person Plural Subject
PVSUFF_SUBJ:1S	Perfect Verb Suffix 1st Person Singular Subject
PVSUFF_SUBJ:2FS	Perfect Verb Suffix 2nd Person Feminine Singular Subject
PVSUFF_SUBJ:2MP	Perfect Verb Suffix 2nd Person Masculine Plural Subject
PVSUFF_SUBJ:2MS	Perfect Verb Suffix 2nd Person Masculine Singular Subject
PVSUFF_SUBJ:3FD	Perfect Verb Suffix 3rd Person Feminine Dual Subject
PVSUFF_SUBJ:3FP	Perfect Verb Suffix 3rd Person Feminine Plural Subject
PVSUFF_SUBJ:3FS	Perfect Verb Suffix 3rd Person Feminine Singular Subject
PVSUFF_SUBJ:3MD	Perfect Verb Suffix 3rd Person Masculine Dual Subject
PVSUFF_SUBJ:3MP	Perfect Verb Suffix 3rd Person Masculine Plural Subject
PVSUFF_SUBJ:3MS	Perfect Verb Suffix 3rd Person Feminine Singular Subject
PV_PASS	Passive Perfect Verb
RC_PART	Relative Clause Particle
REL_ADV	Relative Adverb
REL_PRON	Relative Pronoun
SUB	Subordinating Particle
SUB_CONJ	Subordinating Conjunction
VERB_PART	Verb Particle
VERB_PERFECT	Perfect Verb

Emad Mohamed
Resume

Name: Emad Soliman Ali Mohamed
Email: emohamed AT umail.iu.edu, emadnawfal AT gmail.com

Education

2010 Indiana University PhD in Linguistics

I work with professor Sandra Kuebler on Computational Linguistics. My main focus is on using machine learning techniques in investigating the characteristics of Arabic from a computational perspective with the purpose of developing materials for Arabic language research and teaching. Arabic is a morphologically complex language, and its grammatical analysis, including Part of Speech Tagging and parsing are much more demanding than that of English. While all the previous research on Arabic assumes that the input to the grammatical analysis is word-segmented, I have succeeded in creating a tagger without such an unrealistic assumption. I am currently working on the parser.

2009. MA in Computational Linguistics, Department of Linguistics, Indiana University, USA
I obtained an MA on the way to the PhD.

2007 PhD Student , Department of Linguistics, Indiana University Bloomington

2000–2002 Minia University Minia, Egypt

MA in linguistics with a thesis titled "A Linguistic Investigation of some Aspects of Translation in the Arabic Version of the Newsweek”.

The thesis discussed translation equivalence problems from a cultural/linguistic perspective when translating an International magazine into Arabic, a language with a cultural, religious, and political heritage that may be at odds with the Source Language.

2000 California State University, LA

Diploma in Language Testing. I studied test building theory and practice with the purpose of building high quality tests that measure students understanding and performance in learning a foreign language.

1993 - 1998 AlAzhar University Cairo, Egypt
BA in English and Translation, I received my BA from AlAzhar University with Excellent with Honor Degree (Summa cum Laude). My undergraduate education was focused on translation and interpretation from English into Arabic and vice versa.

Publications

- Emad Mohamed and Sandra Kuebler (2010). Is Arabic Part of Speech Tagging feasible without word segmentation. Proceedings of NACCL (2010), Los Angeles, USA. To appear.
- Emad Mohamed and Sandra Kuebler (2010). Arabic Part of Speech Tagging. Proceedings of LREC (2010), Malta. To appear.
- Emad Mohamed and Yasser Chuttur. *Automatic Orthographic Enrichment for Arabic Language Teaching*. Calico 27th Annual Conference, Amherst College, Boston, 2010. To appear.
- Emad Mohamed; Sandra Kübler, (2009). *Diacritization of Real World Arabic*. Proceedings of RANLP 2009, Borovets, Bulgaria
- Kübler, Sandra; Emad Mohamed (2008) *Memory-Based Vocalization of Arabic*. Proceedings of the LREC Workshop on HLT and NLP within the Arabic World, Marrakesh, Morocco, May 2008.

Invited Talks

Language Technology for Arabic Language Teaching and Research, Department of Near Eastern Languages and Cultures, Indiana University. (Spring, 2009)

Computer Knowledge:

I use Python most of the time, but I also used Java and Prolog in the past (It's been ages since I used Java though). I'm a Unix user, but can also invest some time in Windows and MSOffice products if need be.

I'm good at database design concepts and, to some extent, implementation using MySQL (although I do not usually do so). Currently, I'm learning R, and I believe it is very useful for language research

Computer Programs I wrote:

Memory-based Diacritizer: Most text in Semitic and Farsi languages lacks the vowels, and this leads to ambiguity. This programs takes raw (Arabic) text and inserts the diacritics.

Arabic Word Segmenter: The segmenter is very useful given the nature of Arabic word structure. Given a word like “wllmhndsAt”, the program analyses the word into w+l+Al+mhndsp+At. This is then passed to the Arabic Part of Speech Tagger. The Segmenter has an accuracy rate of over 99%.

Rule-based Phrase Detector. The Phrase detector takes tagged text, and delimits phrases like Noun Phrases, and Verb Phrases. Works only for Arabic as it depends on regular expressions.

Arabic Smart Concordancer: A concordancing program that takes care of the complex nature of Arabic morphology. You do not need to use the root or the exact match for this program. You just give it a word in any form, and it will display all the possible forms of the word in context. This is a necessity for compiling vocabularies, and Arabic language teaching.

Arabic Part of Speech Tagger: The Tagger takes a string of words and assigns them parts of speech. For example, the word *wllmhndsAt* is assigned the tags: conjunction+preposition+definite_article+noun+feminine_plural. The POS Tagger is necessary for linguistic analysis, Computer-Assisted Language Learning, and virtually any processing of natural language.

Swahili Word Segmenter: This program uses supervised machine learning to segment Swahili words. Swahili is an agglutinative language whose words are made up of several segments. Segmentation is the primary step for any further processing.

Experience
My work experience is in teaching and translation

2000– 2006 The American University in Cairo, Cairo, Egypt
Part-time EFL Instructor
- Taught English to beginner and advanced students. I also taught TOEFL preparation classes.
- Appeared three times on the teacher honor roll

2000– 2006 Suez Canal University, Suez, Egypt
Assistant lecturer in linguistics
I assisted the professors in teaching phonetics and phonology, syntax, and Arabic <> English translation.

1999–2003 Free-lance translator and interpreter
- Translated American books on management and psychology for Jarir bookstore, Saudi Arabia.
- Worked as a conference interpreter for the League of Arab State

Languages
Arabic: mother tongue
English: near-native fluency
French: Read only
Swahili: understand structure in general.