AUTOMATIC DIACRITICS RESTORATION FOR ARABIC TEXT

BY

Omar Elsayed Shaaban

A Thesis Presented to the
DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

COMPUTER SCIENCE

MAY, 2013

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN- 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **Omar S. Shaaban** under the direction of his thesis advisor and approved by his thesis committee, has been presented and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN COMPUTER SCIENCE.**

Dr. Husni Al-Muhtaseb
(Advisor)

Dr. Mustafa El-Shafei
(Co-Advisor)

Sayed 8/1/2014

Dr. El-Sayed El-Alfy
(Member)

Dr. Wasfi Al-Khatib
(Member)

Dr. Essam Eid
(Member)

Dr. Adel Ahmad
Department Chairman

Dr. Salam A. Zummo
Dean of Graduate Studies

20/2/14

Date

Dedication

To my beloved family...

ACKNOWLEDGMENTS

I would like to thank my advisor Dr. Husni Al-Muhtaseb who has supervised this research work and guided me throughout, without whom this work would not be the same. I also thank the committee members: Dr. Mustafa Elshafei, Dr. Wasfi Al-Khatib, Dr. El-Sayed El-Alfy, and Dr. Essam Aid for their great support throughout this journey.

Some of the work behind this thesis was the byproduct of two course projects under Dr. Lahouari Gouthi (in the Advanced Artificial Intelligence course) and Dr. Sabri Mahmoud (in the Natural Language Processing course).

In addition, I would like to acknowledge the financial support of the work by the university funded project with numbers RG1104-1 and RG1104-2 for which Dr. Mustafa Elshafei is the Principal Investigator. Many thanks to the people who contributed in developing the corpus: Anas Abu-Dagga, Yousef Elarian, Sadam Al-Azani, and many others.

I would like also to thank Ms. Susan Mohammad from Arabi, an Arabic Software Solutions Company (http://www.arabinlp.com/), who has gracefully given me permission to evaluate the diacritizer developed by the company.

Special thanks for Omar Alzuhaibi who helped in evaluating my developed automatic diacritizer and manual diacritization tool and providing me with valuable feedback.

Finally yet importantly, I would like to thank Tim Buckwalter for his awesome morphological analyzer, and Pierrick Brihaye for his Java port of the former. Also, a lot of open source libraries and tools (kindly check the list in Appendix III) were extensively used throughout this research work, my personal appreciation for the people behind them.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

- **AATD** Arabic Automatic Text Diacritization
- **ASR** Automatic Speech Recognition
- **ATD** Automatic Text Diacritization
- **BAMA** Buckwalter’s Arabic Morphological Analyzer
- **CRF** Conditional Random Fields
- **DER** Diacritic-Error-Rate
- **DL** Diacritization Level
- **DT** Diacritization Tool
- **HMM** Hidden Markov Models
- **LDC** Linguistic Data Consortium
- **MADA** A diacritization toolkit
- **MSA** Modern Standard Arabic
- **NLP** Natural Language Processing
- **OOV** Out-of-vocabulary
- **PM** Peak memory
- **POS** Part-of-Speech
- **POST** Part-of-Speech Tagging
- **SAMA** Standard Arabic Morphological Analyzer
- **SVM** Support Vector Machine
- **TTS** Text-to-Speech
- **WER** Word-Error-Rate
- **WPS** Words Per Seconds

ABSTRACT

Full Name : Omar Elsayed Mohammed Shaaban

Thesis Title : Automatic Diacritics Restoration for Arabic Text

Major Field : Computer Science

Date of Degree : May 2013

Arabic scripts consist of two primary categories: letters and diacritics. The diacritics are often omitted for convenience, as most experienced readers can easily infer the missing diacritics of a word from its context. This, however, poses a challenge to some readers, such as non-native speakers, who may not be able to infer such diacritics easily. In addition, diacritics play an important role in many Arabic Natural Language Processing (ANLP) applications, such as Automatic Speech Recognition (ASR), Automatic Language Translation (ALT), and Text-to-Speech (TTS) converters. Thus, the automatic restoration of missing diacritics is an essential step to achieve acceptable performance. Studies have approached this problem in two ways; either using machine learning (ML) algorithms or using basic rules that were derived from Arabic grammar and orthography. This thesis shows that by combining the two approaches an improved performance can be achieved.

The main contributions of the thesis are: (1) construction of a diacritized corpus, and (2) development of a hybrid diacritizer. In the first contribution, we built a fully diacritized corpus which was collected from different sources, whether already diacritized or not, covering several fields (e.g. news, literature, sports, religion). The developed corpus has more than 28,000,000 words from classical Arabic, and 3,000,000 words from Modern Standard Arabic (MSA). In the thesis, we explain the corpus construction process in details and give in-depth statistics.

The second contribution of the thesis is combining the rule-based approach with the statistical approach for automatic restoration of missing diacritics. Rules were inducted from the corpus such that they have near 100% accuracy. We use a varying number of features in the rules, such as the current letter, previous letters, next letters, stop-words, and so on. Our results show that by using these rules, the performance solidly enhances (with WER=13.8% and DER=3.5%) as compared with the mere statistical approach.

In the statistical approach, we used word-level N-grams, character-level N-grams, and POS-level N-grams that were extracted from the corpus. Then, to select the best diacritization, on each level, we used a greedy algorithm with a good heuristic that ensures optimality time-wise and accuracy-wise. This approach was built upon the results of the aforementioned rules.

ملخص الرسالة

الاسم الكامل: عمر السيد محمد شعبان

عنوان الرسالة: استعادة التشكيل آليًّا للنصوص العربية

التخصص: علوم الحاسب الآلي

تاريخ الدرجة العلمية: مايو 2013

تتكوّن الكتابة العربيّة من أحرف وعلامات للتّشكيل، وهذه الأخيرة عادةً ما تحذف للتّسهيل على الكاتب، لأنّ القارئ العربيّ الخبير يستطيع بسهولة استنتاج تلك العلامات لأيّ كلمة عبر سياق النّصّ. ويستعصي هذا على القارئ المبتدئ الّذي ربّما يجد صعوبةً في استنتاجها. كما تعتبر علامات التّشكيل ذات أهمية بالغة لكثير من تطبيقات الحاسب الآلي اللسانية كالتّعرف الآلي على الكلام، والترجمة الآلية، ونطق النصوص المكتوبة. ولذا فمن المهم أن تُستعاد تلك العلامات عند الشروع في أي من هذه التطبيقات لتحسين أدائها. اتبعت الأبحاث المتعلقة بهذا الشّأن إحدى طريقتين: الأولى هي الطريقة الإحصائية والتي تستخدم في غالبها خوارزميات تعلم الآلة، والثانية طريقة تعتمد على قواعد مشتقة من قواعد النحو والإملاء للغة العربية. سعينا في هذه الرسالة البحثية لاتباع طريقة ثالثة تجمع بين الطريقتين السابقتين، والتي من شأنها تحسين دقة التّشكيل الآلي.

نقدّم في هذه الرسالة البحثية إسهامين رئيسين: الأوّل بناء مكنز مشكل آليًا، والثاني تطوير مشكل آلي هجين يجمع بين الطريقة الإحصائية والقواعد. وقد قمنا ببناء المكنز من مصادر عدة، سواء كانت مشكلة أو غير مشكّلة، مع مراعاة التنوع في مجالات عدة كالأخبار، والرياضة، والأدب، والدين. ويحتوي هذا المكنز على أكثر من 28,000,000 كلمة من الكتب التراثية، وحوالي 3,000,000 كلمة من اللغة العربية الحديثة. ونبين في هذه الرسالة الطريقة المتبعة في بناء المكنز بشكل تفصيلي وكذلك نعرض إحصاءات شتّى مستخرجة منه. ويعتمد الإسهام الثّاني لهذه الرسالة البحثية على دمج الطريقة الإحصائية مع القواعد في نظام هجين للتّشكيل الآلي. وقد استنتجت القواعد من المكنز بحيث تضمن دقة تقترب من 100%. وتتكون كل قاعدة من عدّة خصائص، كالحرف الحالي والأحرف السّابقة واللاحقة والكلمات الوقفية وهلم جرا. وقد أثبتت النتائج المستخلصة أن استخدام هذه القواعد يحسن أداء ودقة التّشكيل بشكل ملحوظ. أما في الطّريقة الإحصائية، فقمنا باستخدام سلاسل الكلمات والأحرف والوسوم المستخرجة من المكنز، ومن ثمّ قمنا باختيار أفضل تشكيل ممكن (لكل مستوى من المستويات الثلاثة) باستخدام خوارزمية بحثية "نهمة" (greedy) وتبنى هذه الطّريقة على نتائج القواعد سالفة الذكر.

CHAPTER 1

INTRODUCTION

Natural Language Processing (NLP) is an important field of both computer science and computational linguistics. Over decades, it has evolved marking great advancement in recent years. The importance of this field has increased due to the ubiquity of the Internet and mobile devices, which have been requiring more and more natural human-machine interactions. Some of NLP applications are intrinsically useful for pure language processing purposes, such as spell and grammatical checking and correction. However, other types of NLP applications are more useful not in themselves but as tools for more complex applications such as Automatic Speech Recognition (ASR), where a person can dictate text or issue commands to a device, and Text-To-Speech (TTS), where the device pronounces text or informs a user about certain situations [1].

In the Arabic language, most of the NLP problems depend heavily on the diacritics, which are often omitted for writer's convenience. For that reason, the automatic restoration of these diacritics is arguably a very important step in any Arabic NLP application, which is the subject of this thesis.

In this chapter, we introduce Arabic orthography in Section 1.1 from the diacritics perspective, problem statement in Section 1.2, applications of automatic diacritization in

Section 1.3, thesis contributions in Section 1.4, and lastly an overview of the thesis in Section 1.5.

1.1 Diacritization in Arabic Orthography

The Arabic alphabet consists of 28 letters; 25 of them represent consonants while the remaining three letters (Alif ا, Waw و, and Ya'a ي) represent the long vowels. These long vowels may also serve as consonants themselves, except Alif [2]. Each consonant may have a diacritization from a set of 14 different diacritical combinations, as shown in Table 1. These diacritical forms can be classified into five categories.

1. The first category represents short vowels, namely Fat-h (◌َ), Damm (◌ُ), and Kasr (◌ِ). For example, the consonant /*b*/ (ب) combined with each short vowel is pronounced /*ba*/, /*bu*/, and /*be*/, respectively.
2. The second category is the syllabification marks, which consist of two diacritics, Sukoon (◌ْ), where the consonant is vowelless, and Shadda or gemination (◌ّ), where the consonant is doubled.
3. The third group is the double case-ending or Tanween, which is a double short vowel (◌ً ◌ٌ ◌ٍ). Tanween is added at the end of a word in order for it to be pronounced with an ending /*an*/ , /*on*/, or /*en*/ sounds, respectively [2].
4. The fourth category is the combination of the first category (short vowels) and Shadda.
5. The fifth category is the combination of the third category (Tanween) and Shadda [3].

The diacritization of a word in Arabic is divided into two parts: the first part is context-insensitive and is affected only by the morphology of the word. The second part is context-sensitive and can be affected by the context of the word in a sentence. The ambiguity in the meaning of a word is controlled by the former while the latter affects the ambiguity of a sentence. In automatic diacritization, statistical methods can be used for the deduction of the first part diacritics while the second part requires the knowledge of the syntactical rules.

Table 1 Arabic Diacritics and Taxonomy

No.	Diacritical Category	Diacritic	Example	Example's Pronunciation
1	Short Vowels	Fat-h (◌َ)	بَ	/ba/
2		Damm (◌ُ)	بِ	/be/
3		Kasr (◌ِ)	بُ	/bu/
4	Syllabification Marks	Sukoon (◌ْ)	بْ	/b/
5		Shadda (◌ّ)	بّ	/bb/
6	Double Short Vowels (Tanween)	Tanween Fat-h (◌ً)	بً	/ban/
7		Tanween Damm (◌ٌ)	بٌ	/bun/
8		Tanween Kasr (◌ٍ)	بٍ	/ben/
9	Shadda + Short Vowel	Shadda + Fat-h (ـَّ)	بَّ	/bba/
10		Shadda + Damm (ـُّ)	بُّ	/bbu/
11		Shadda + Kasr (ـِّ)	بِّ	/bbe/
12	Shadda + Tanween	Shadda + Tanween Fat-h (ـًّ)	بًّ	/bban/
13		Shadda + Tanween Damm (ـٌّ)	بٌّ	/bbun/
14		Shadda + Tanween Kasr (ـٍّ)	بٍّ	/bben/

1.2 Problem Statement

Automatic Text Diacritization (ATD) (aka Automatic Diacritics Restoration) is one of the NLP problems that can be viewed as an independent problem, which has its own applications, or as a complementary one to other more complex problems. This problem is often associated with Semitic languages like Arabic, Hebrew, Amharic and others. However, it is also applicable to other languages such as Latin-based, Greek, and Korean languages [4]. When the Arabic language is considered, which is the focus of this research work, diacritization is often omitted from text leaving the reader with semantic ambiguity. However, fluent readers can deduce this diacritization from the context of the word with the least discomfort. This is not the case though for novice or beginning readers who can find this task quite troublesome. ATD aims to reduce this ambiguity by inferring the word's intended diacritization as closely as possible.

The subject of this research work is to perform ATD for Arabic texts, or Automatic Arabic Text Diacritization (AATD). Specifically, the problem of AATD can be described as the process of restoring missing diacritics from undiacritized or partially diacritized text. Figure 1 (a) shows a sample input and Figure 1 (b) shows the expected output of the AATD process for this input.

خير الناس أنفعهم للناس	خَيْرُ النَّاسِ أَنْفَعُهُمْ لِلنَّاسِ
(a) Input	**(b) Output**

Figure 1: Example on AATD.

1.3 Applications of AATD

AATD, or Automatic Arabic Text Diacritization, can be beneficial both independently and as an input to other Arabic related NLP problems. By applying AATD, Arabic text ambiguity is reduced and the meaning of the text is better understood. Furthermore, having a diacritized text is an essential step in both Arabic ASR (AASR) and Arabic TTS (ATTS). In the former, most of the methods require supervised training based on a diacritized corpus and its corresponding acoustic model. Having to manually diacritize such a corpus can be unnecessarily tiresome and time-consuming step. In the later, the ATTS engine needs the full phonetic transcription of a sentence before pronouncing it, which can only be achieved if the text is fully diacritized.

1.4 Contributions

The main contributions of this thesis are:

1- Construct a diacritized corpus collected from different sources covering a variety of domains.

2- Design a hybrid model for AATD using rules as well as statistics.

3- Compare the performance of the developed model to other tools in the field, objectively.

4- Develop tools and libraries that help in future work of the topic.

Figure 2 shows the work structure behind this thesis. The details of this structure will be discussed in the coming chapters.

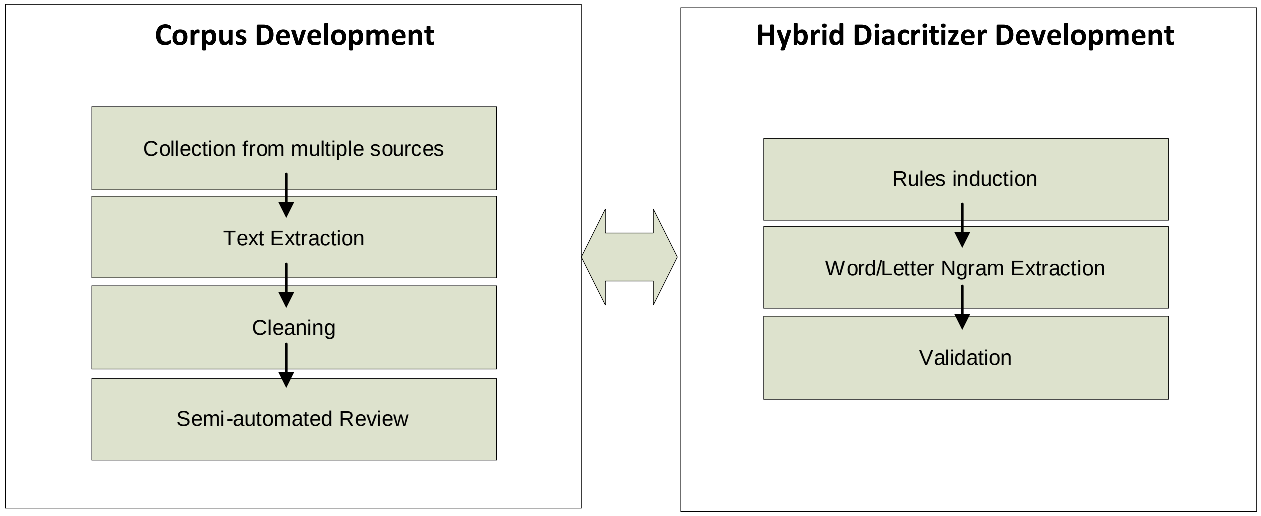

Figure 2: Thesis work structure

1.5 Overview

The rest of this thesis is organized as follows: Chapter 2 gives a survey of prior research in this subject. Chapter 3 describes the corpus building approach while Chapter 4 describes the hybrid approach and the implemented diacritizer. Chapter 5 discuses the evaluation of the diacritizer and compares it with other available ones. Finally, the conclusion and future work are given in Chapter 6.

CHAPTER 2

LITERATURE REVIEW

The problem of automatic diacritization has been explored extensively by researchers in the last two decades, most notably for Arabic. It has been approached as an independent problem, a sub-task of other problems such as Automatic Speech Recognition (ASR), or as a by-product of other Natural Language Processing (NLP) problems such as Part-of-Speech Tagging (POST) [5]. In all three cases, several methodologies have been followed to achieve the desired level of accuracy. In most studies statistical methods have been used. In other studies, a hybrid of two or more different methods were used to maximize the accuracy.

In this survey, we review recent publications regarding this topic for the Arabic language in Section 2.1 and for other languages in Section 2.2. We also present a comparison between various diacritization methods in Section 2.3.

2.1 Arabic Diacritization

Azim [5] added acoustic features to the textual methods as an input to the diacritization problem. The author examined the effect of combining speech with existing text-based models on correcting errors made by the text-based models prediction. The author used Hidden Markov Models (HMM) for speech and Conditional Random Fields (CRF) for text. The author claimed to have achieved better performance than what was achieved using the Morphological Analysis and Disambiguation for Arabic toolkit (MADA). With case-endings, the Diacritic Error Rate (DER) was 1.6% while the Word Error Rate (WER) was 5.2%. Without case-endings, the DER was 1.0% while the WER was 3.0%.

Rashwan et al. [2] introduced a dual-mode stochastic system for Arabic diacritization of raw text. The first mode searches in a dictionary of full-form diacritized words, using A∗ lattice search and long-horizon *N*-gram probability estimation, for the most likely diacritization. When the word or the sequence of words is out-of-vocabulary (OOV), the second mode factorizes each word to all its possible morphological cases and searches the dictionary for each, choosing the best diacritization. The system achieved 3.1% DER and 12.5% WER.

Zitouni and Sarikaya [6] used a maximum entropy (MaxEnt) approach for restoring diacritics. This approach integrates diverse types of information such as lexical, segment-based and part-of-speech tag features. They defined the problem as a classification problem and hence used the MaxEnt classifier. Their conducted experiments on the Linguistic Data Consortium's (LDC) Arabic Treebank Part 3 showed WER 17.3% and 7.2% for case-ending and non-case-ending, respectively and a DER was 5.1% and 2.2%, respectively.

Shaalan et al. [7] used a hybrid approach of an Arabic lexicon and a diacritized corpus. First, the word is searched in the lexicon. If the word has one diacritized form, it is confirmed as the diacritization of the word. However, if the word is not found another look-up is performed with the previous and/or the next word in a bi-gram lexicon. Then, the second stage is to tag the word using the Support Vector Machine-based Part of Speech (SVM-POS) tagger and the diacritized form is then inferred. To determine the case-endings, three features were used: the POS of the word, the chunk position and the sentence position. These features were combined into an SVM model to determine the case-ending of a word. The WER achieved was 17.31% while the DER was 4.41%.

Habash et al. [8] [9] developed the MADA system which uses SVM with POS tagging system. The SVM model was built with the features extracted from Buckwalter Arabic Morphological Analyzer (BAMA). The features include noun case, verb mood, and nunation (Tanween). Their reported WER is 14.9% and DER is 4.9% with case-endings. Their claimed WER is 5.5% and DER is 2.2% without case-endings.

Elshafei et al. [10] used a Hidden Markov Models (HMM) based approach in providing a solution to the problem of automatic diacritization. This approach requires a large corpus of fully diacritized text to extract the features needed. The authors used the holy Qur'an as their training and test corpus. The features used for HMM were the sequence of undiacritized words while the hidden states were the diacritized words. In their testing experiments, the authors found the word error rate to be 4.1% which they improved to about 2.5% using a preprocessing stage and trigrams for selected number of words and articles.

Attia [11] described an Arabic diacritizer (ArabDiac) that was used for automatic Arabic phonetic transcription. This system used a hybrid model in which both statistical data and rules were employed to deduce the most appropriate diacritization of a sentence. The system operated in 4 stages. In the first stage, the plain input text is normalized by converting all numeric and acronyms to their alphabetical forms. In the second stage, a lexical analyzer gets the most likely lexical diacritization, morphemes sequence, and identification of transliterated strings. In the third stage, a POS tagger extracts the POS tags of each word and then a syntactical analyzer infers the correct syntactic diacritization. As for transliterated strings, their diacritization is deduced based on statistical data and phonetic grammar. Finally, in the fourth stage, the phonetic transcription is generated using a phonetic concatenator that takes care of the inter-phonetic effects between adjacent words. According to the author's experimentations, the accuracy in the lexical level was 97% while the accuracy in the syntactical level was 88%.

2.2 Non-Arabic Diacritization

Most of the research in the field of automatic diacritization is Arabic-specific. However, some researchers have studied the same problem on other languages that exhibit similar orthography to Arabic which makes them strongly related to our subject.

Trung et al. [12] have studied the diacritization problem in Vietnamese, a language written in Roman letters along with accents that are usually omitted. The authors approached the problem as a sequential tagging using CRF and SVM where they selected features in two ways: one using letters and the other using syllables. The claimed accuracies were 91% for the former and 93% for the latter (in written language).

Atserias et al. [13] used a bigram model for Spanish to resolve ambiguity in spell-checking where a word may appear more than once in the corrections list but with different accents (or diacritics). They achieved a precision of 85% and a recall of 64%.

Javed et al. [14] studied the diacritization problem for Sindhi, which is spoken in Pakistan and parts of India. In their research, they developed a system based on WordNet structures which stores the semantic relationships between words. This system used three different corpora. The first one, called CRITICAL, was used for ambiguous critical words. The second, so-called HOMONYMY, was used for all homographic words of critical words. The third, called WNL, was used for analogical words. In their testing experiments, they claimed a word error rate of 0.71%, and a diacritic error rate of 3.39%.

Haertel et al. [15] targeted the diacritization of Semitic languages, especially Syriac. The method they used was Conditional Markov Models (CMM) which only required diacritized not-fully tagged corpus. These models were based on features (such as the suffixes and prefixes) extracted from previously diacritized words. The authors claimed a word-error-rate of 15% for Arabic and 10.5% for Syriac.

Raza [16] studied the problem for Urdu. He presented analysis and implementation of a system that performs automatic diacritization for Urdu text. The system was based on a lexicon and a corpus that was manually diacritized and POS tagged. The process of the system is as follows. First, all diacritics are removed from the text prior to its processing. After that, a POS tagger, which was trained using HMM on the corpus of bigrams and tri-grams, is used to identify POS tags for each word. Then, the word and its tags are searched in the lexicon to get a diacritized version of the word. If the word and its tag are not found, the word is sent for rule-based affixation, or else a statistical diacritization module. Overall, the system achieved a maximum of 95% accuracy as per the research experiments.

2.3 Comparison

In the following tables (Table 2 and Table 3), we give a detailed comparison between different research works that address the problem of diacritization. The comparison criteria are as follows: the approach used (whether statistical or rule-based), the corpus (a standard corpus or a custom one), and evaluation metrics (WER with and without case-endings, and similarly DER with and without case endings). These metrics are explained in more details in Section 5.1.

In the table, we can see most researchers have resorted to statistical methods while only a few used rule-based methods (albeit limitedly). Some of these statistical methods are based on machine learning algorithms such as CRF and HMM, while others are based on simple word-level (or letter-level) *N*-grams.

The table also shows how the results differ greatly between the papers in terms of the word-level error rate and the diacritic-level error rate. For example, Azim [5], whose stated results were the best as far as we encountered, has reported a WER of 5.2% as opposed to a DER of 1.6%, with case-endings included. Similarly, Rashwan et al [2] reported a WER of 12.50% and a DER of 3.80%.

Although the problem is the same, it is very hard to compare these papers objectively for two reasons. First, the testing set used in each paper is different than others (with few exceptions which used LDC's Arabic Treebank). Second, although the metrics used are mostly the same, each paper has its own way of computing them. Sometimes these differences are minor and don't matter much and sometimes they are major differences and can greatly affect the performance. We have explained some of these evaluation problems in Section 5.2.

Table 2 Comparison of different approaches followed by other researchers (A)

Paper[1]	Approach	Corpus	$WER_{Without\ CE}$	$WER_{With\ CE}$	$DER_{Without\ CE}$	$DER_{With\ CE}$
Azim 2012 [5]	Statistical	LDC Arabic Treebank	3.0%	5.2%	1.0%	1.6%
Trung 2012 [12]	Statistical	Custom	NA[2]	NA	NA	8.4%
Rashwan 2011 [2]	Statistical	Custom	3.10%	12.50%	1.20%	3.80%
Mahar 2011 [14]	Statistical	Custom	NA	1.13%	NA	3.39%
Haertel 2010 [15]	Statistical	LDC Arabic Treebank	NA	15.02%	NA	5.15%
Alghamdi 2010 [17]	Statistical	KACST	26.03%	46.83%	9.25%	13.83%
Rashwan 2009 [18]	Statistical	Custom	5.70%	21.10%	NA	NA
Raza 2009 [16]	Statistical	Custom	4.80%	NA	NA	NA
Shaalan 2009 [7]	Statistical	LDC Arabic Treebank	33.51%	17.31%	7.99%	4.41%
Mohamed 2009 [19]	Statistical	LDC Arabic Treebank & Custom	5.93%	NA	NA	NA
Zitouni 2009 [20]	Statistical	LDC Arabic Treebank	7.20%	17.30%	2.20%	5.10%

[1] Paper name is abbreviated as the last name of the first author followed by the publishing year.
[2] Not available.

Table 3 Comparison of different approaches followed by other researchers (B)

Paper	Approach	Corpus	$WER_{Without\ CE}$	$WER_{With\ CE}$	$DER_{Without\ CE}$	$DER_{With\ CE}$
Roth 2008 [9]	Statistical	LDC Arabic Treebank	4.60%	13.90%	NA	NA
Schlippe 2008 [21]	Hybrid	LDC Arabic Treebank & AppTek	9.30%	13.80%	3.20%	4.90%
Habash 2007 [8]	Statistical	LDC Arabic Treebank	5.50%	14.90%	2.20%	4.80%
Elshafei 2006 [22]	Statistical	KACST	NA	5.50%	NA	NA
Zitouni 2006 [23]	Statistical	LDC Arabic Treebank	7.90%	18.00%	2.50%	5.50%
Elshafei 2006 [24]	Statistical	Qur'an	NA	NA	NA	4.10%
Sanka 2005 [25]	Statistical	LDC Arabic Treebank	NA	19.73%	NA	NA
Attia 2005 [11]	Statistical (Mainly)	Custom	3.60%	13.50%	NA	NA
Nelken 2005 [26]	Statistical	LDC Arabic Treebank	7.33%	23.61%	6.35%	12.79%

2.4 POS Tagging & Morphological Analysis

Part-of-Speech Tagging (POST) is the process of assigning morpho-syntactic tags to each word in a sentence [27]. The richness and complexity of Arabic can make the needed tag set a very large one. However, researchers often prefer to use small tag sets such as the Buckwalter tag set which has 70 basic tags that can be combined to form 169 tags [27].

POS tagging can be of tremendous value to the diacritization problem. In fact, the best performing automatic diacritizers make use of POS tagging extensively (as in [5] for example as explained before). However, tagging a word does not mean it diacritical form is immediately known since words tend to have multiple diacritical forms that require a subsequent stage of disambiguation.

2.5 Summary

In this chapter, we examined prior research work in the diacritization problem. The problem has been tackled by different approaches that mainly fall under two categories: the statistical approach, and the rule-based approach. Although, many researchers have claimed to achieve remarkable results, it is difficult to verify these claims independently as they usually do not have public implemented systems.

In Chapter 4, we discuss our proposed approach and our methodology to avoid the downsides of existing approaches. But before that we describe in Chapter 3 the development process of the corpus that we will use in our research.

CHAPTER 3

CORPUS CONSTRUCTION

A text corpus is a large and structured set of texts. No matter what statistical method is used, the corpus remains the most essential component that cannot be relinquished [28]. Therefore, one of the primary objectives of this thesis was to build a sizable, diverse and fully-diacritized corpus which sustains an acceptable level of accuracy. In this chapter, we discuss the approach that was followed while developing the corpus. In Section 3.1, we discuss the approach in details, whereas in Section 3.2 we give some insightful analysis and detailed statistics of the corpus. Section 3.3 presents the summary of the chapter.

3.1 Corpus Development Approach & Methodology

The process of building a corpus consists of the four phases demonstrated in Figure 3. The purpose of the first phase was to collect as much text as possible from mostly Internet websites and electronic books. To achieve that, a web crawler was built to download pages and documents from websites. The downloaded files were then saved to the disk in their original format. Table 4 shows a list of 25 sites that were crawled.

In addition to the crawled websites, a special search was made for Arabic documents with the extensions PDF, EPUB, ODF, PPT, PPTX, DOC, and DOCX, which represent the most common document formats. The search was performed on Google Search Engine and produced up to 2,149 documents (although the PDF documents were excluded later on because extracting texts from such files is not always effective, especially for Arabic).

Table 4 List of crawled websites

No.	Website	URL	Last Accessed
1	Aadab Magazine	http://www.adabmag.com/	25 March 2013
2	Adab Encyclopedia of Poetry	http://www.adab.com/	25 March 2013
3	Ahl Al-Lughah Forums	http://www.ahlalloghah.com/	25 March 2013
4	Al-Alukah Network	http://www.alukah.net/	25 March 2013
5	Al-Arabi Magazine	http://www.alarabimag.com/	15 April 2013
6	Al-Arabiya News Network	http://www.alarabiya.net/	25 March 2013
7	Al-Bayan Magazine	http://www.albayan.co.uk/	25 March 2013
8	Al-Bayan Newspaper	http://www.albayan.ae/	25 March 2013
9	Al-Hayat Newspaper	http://www.alhayat.com/	25 March 2013
10	Al-Jazeera News Network	http://www.aljazeera.net/	25 March 2013
11	Al-Maany Dictionary	http://www.almaany.com/	25 March 2013
12	Al-Majalla Magazine	http://www.majalla.com	25 March 2013
13	Al-Masry Al-Yuom Newspaper	http://www.almasryalyoum.com/	25 March 2013
14	Al-Meshkat Islamic Network	http://www.almeshkat.com/	25 March 2013
15	Al-Mujtama'a Magazine	http://www.magmj.com/	25 March 2013
16	Al-Quds Newspaper	http://www.alquds.co.uk/	25 March 2013
17	Al-Sakher Forums	http://www.alsakher.com	25 March 2013
18	Al-Shamela Library	http://www.shamela.ws/	19 March 2013
19	Al-Sharq Al-Awsat Newspaper	http://www.aawsat.com/	25 March 2013
20	ArabDict Dictionary	http://www.arabdict.com/	25 March 2013
21	Dahsha Encyclopedia	http://www.dahsha.com/	25 March 2013
22	Elaph Blog	http://www.elaphblog.com/	25 March 2013
23	Elaph Online Newspaper	http://www.elaph.com/	25 March 2013
24	Saaid Al-Fawaed	http://www.saaid.net/	25 March 2013
25	Sayidaty Magazine	http://www.sayidaty.net/	25 March 2013

In the second phase of the corpus development process, we extracted the texts from the raw files that were crawled in the first phase. To perform the extraction, we built a program based on the Apache Tika [29], which is a Java toolkit that automatically detects and extracts texts from various document formats. After the texts were extracted by Tika, the tool separated Arabic texts from non-Arabic texts using regular expressions. Figure 3 shows the used regular expression to separate Arabic text from non-Arabic text. These extracted texts constituted what we call the General Corpus, which will be discussed in the following subsection.

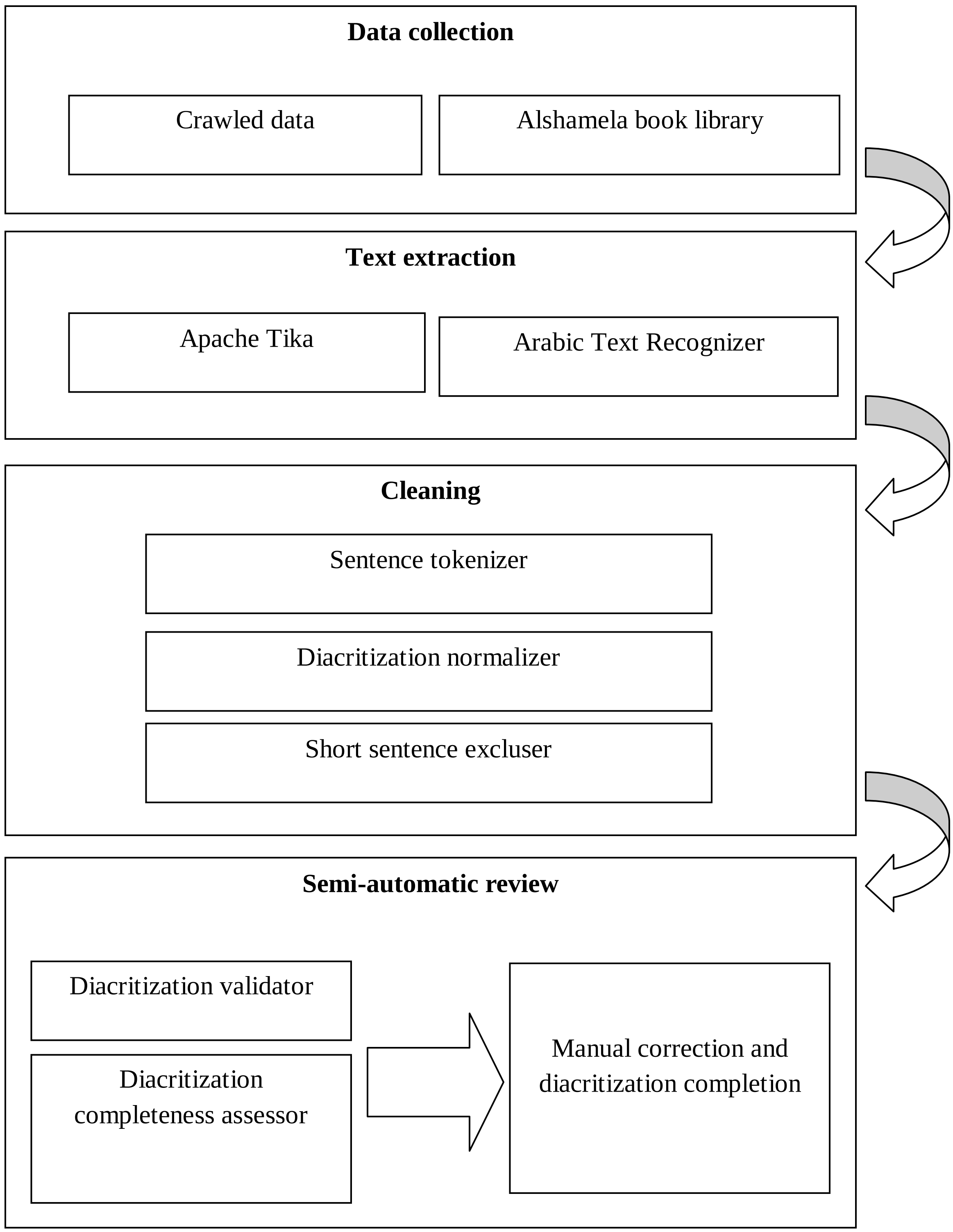

Figure 3: The corpus development process

Using the General Corpus, we separated diacritized texts from non-diacritized ones by computing the diacritization level (see Appendix III for the implementation). Diacritized texts are defined as those texts with diacritization level more than or equal to 90%. In addition, some of the texts that were collected were manually diacritized by a group of volunteers. These texts were also added to crawled texts. The result constituted the Diacritized Corpus. On this corpus we performed cleaning and verification as a third phase.

The third phase, or the cleaning phase, does the following four primary functions in sequence:

1. Sentence tokenization: which divides the extracted texts into sentences, based on a regular expression that is nearly 100% accurate. (See Appendix II)
2. Short sentences exclusion from the corpus: we assumed that any sentence with length less than 100 characters is a short one.
3. Cleaning the diacritization: which essentially means that certain inconsistencies are automatically corrected. One such inconsistency is the position of Tanween diacritics after the Alif letter which are often misplaced at the end of the word. The correct placement of Tanween should be on the letter preceding the Alif not the Alif itself. (See Appendix II for the function and Subsection 3.1.3 for the conventions followed)
4. Removal of repetitions from the corpus disregarding non-Arabic letters: this function ensures that every sentence is different from others. This is essential because many texts in the corpus contain quotes from the holy Qur'an, Hadith or other sources.

The fourth and final phase of the corpus development process involved semi-automatic validation. This validation was performed by a heuristics-based function that determines whether a word is correctly diacritized or not (for this function implementation, refer to Appendix III). When the word is marked as invalid or incompletely diacritized, we manually correct the word.

3.1.1 General Corpus

The General Corpus consists of all the texts extracted from the crawled websites. In this corpus, we ignore the diacritization of the texts, which means that this corpus have a mix of diacritized and undiacritized texts. Table 5 shows some statistics about the General Corpus. These statistics are the number of Arabic letters (6,931,210,613), the number of Arabic words (1,587,511,592), the number of unique Arabic words with diacritics (10,775,960), the number of fully diacritized words (273,558,820), the number of sentences (101,729,156) and the diacritization level (20.577%), which is the ratio of diacritized letters to the total number of letters. To the best of our knowledge, this corpus is the largest and most comprehensive one to date.

Table 5: The General Corpus Statistics

Arabic Letter Count	6,931,210,613
Arabic Word Count	1,587,511,592
Unique Arabic Word Count	10,775,960
Diacritized Word Count	273,558,820
Sentence Count	101,729,156
Diacritization Level	20.577%

3.1.2 Diacritized Corpus

The Diacritized Corpus is the diacritized texts extracted from the general corpus plus the texts that were manually diacritized by our team. Table 6 shows some statistics collected from the Diacritized Corpus. It is important to note here that the computed diacritization level (about 99%) doesn't mean necessarily that the diacritization is incomplete. Rather, it means that the heuristics used to compute the diacritization level is not 100% accurate.

Table 6: The Diacritized Corpus Statistics

Arabic Letter Count	121,777,450
Arabic Word Count	30,169,610
Unique Arabic Word Count	427,436
Sentence Count	710,881
Diacritization Level	99.09%

3.1.3 Rules and conventions

In the Diacritized Corpus, we followed certain rules and conventions to make sure that the diacritization is consistent throughout. Table 7 shows these rules and conventions.

Table 7: Diacritization rules and conventions

1	*Shadda* cannot be used on its own and must be attached with another compatible diacritic.
2	*Shadda* always precedes other diacritics.
3	*Tanween-fath* always precedes the *Alif*.
4	Foreign words always end with *sukoon*.
5	Compound names are diacritized as follows: first part is diacritized as dictated by the context, and the later part is always considered genitive.

3.1.4 Diacritization Assistant

During the course of thesis work, we developed a diacritization assistant which is an editor that helps speed-up manual diacritization. Figure 4 shows a screenshot of the editor. The basic idea of this editor is to reduce the number of mouse/keyboard interactions. In normal text editors, the user needs to navigate between letters in order to add or change the diacritics for a particular letter. Then, the user presses the keys *Shift* and the diacritic simultaneously to add the diacritic.

In this tool, we eliminate the need for manual navigation. We also eliminate the need for the use of the *Shift key*. The user only needs to press the diacritic keys on the keyboard to add the diacritics. Then the editor automatically navigates to the next letter. Since some letters can have multiple diacritics such as *shadda* and *damma*, the editor intelligently waits for another diacritic when the *shadda* key is pressed. If the user needs to delete the previously added diacritic, he/she only needs to press backspace. To navigate between letters or words, he/she can use the *left* and *right* arrows or the mouse wheel.

Another important feature of this tool is the ability to navigate to the next undiacritized, partially diacritized, or invalidly diacritized word just by pressing the keys *Shift* and *Z*. This feature has two advantages: the first is ensuring the complete diacritization of the text, and the second is ensuring the validity of the text. When a word is encountered, the user can easily navigate between all possible diacritizations of the word (which are extracted from the corpus itself) in the order of their probabilities.

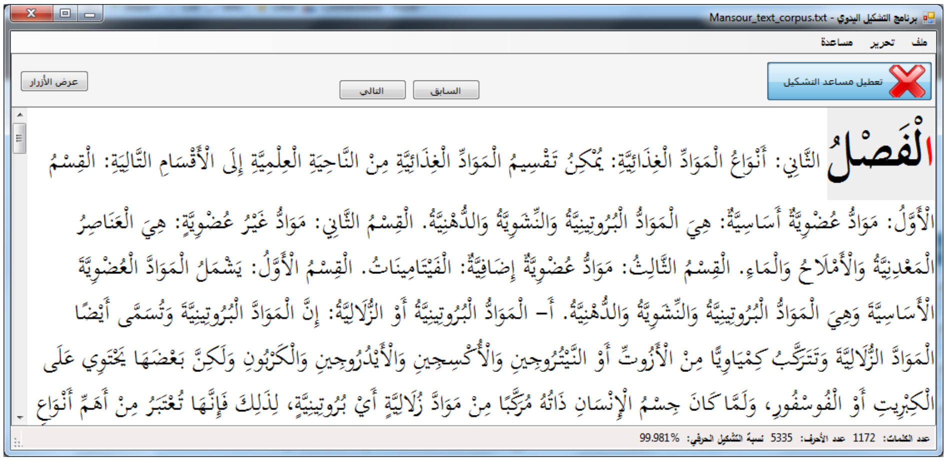

Figure 4: The diacritization assistant

3.2 Diacritized Corpus Analysis

To better understand the corpus, we collected some statistics that relate to the diacritics distribution in the corpus. For example, Table 8 shows the frequency and probability (percentage) of encountering every diacritic in the corpus.

Table 8 Diacritics distribution in the Diacritized Corpus

Letter	Frequency	Percentage
◌ً ◌ّ	78266	0.057%
◌ٌ ◌ّ	104302	0.076%
◌ٍ ◌ّ	149084	0.109%
◌َ ◌ّ	5011889	3.655%
◌ُ ◌ّ	562575	0.410%
◌ِ ◌ّ	860294	0.627%
◌ً	832857	0.607%
◌ٌ	797373	0.581%
◌ٍ	1398457	1.020%
◌َ	67241234	49.032%
◌ُ	16029148	11.688%
◌ِ	22021841	16.058%
◌ْ	22041033	16.072%
◌ّ	9054	0.007%

The table shows that Fat-ha is the most common diacritic followed by Sukoon and Kasra. It also shows that Shadda is almost always associated with short vowels or Tanween and seldom present on its own. This can be considered a tentative measure of how accurate the corpus is diacritized since Shadda should not be used alone.

3.2.1 Letter-Diacritic Matrix

One way to examine the corpus further is to look at the distribution of diacritics on letters. Such distribution is given in the following tables.

Table 9 Letter-diacritic distribution (A)

Letter	◌ًّ	◌ٌّ	◌ٍّ	◌َّ	◌ُّ	◌ِّ	◌ً
ء	0.000%	0.000%	0.002%	0.000%	0.000%	0.000%	2.567%
آ	0.000%	0.000%	0.000%	0.223%	0.000%	0.000%	0.000%
أ	0.000%	0.000%	0.000%	0.001%	0.000%	0.000%	0.081%
ؤ	0.000%	0.000%	0.000%	0.013%	0.003%	0.000%	0.229%
إ	0.000%	0.000%	0.000%	0.000%	0.000%	0.000%	0.000%
ئ	0.000%	0.000%	0.000%	0.000%	0.000%	0.004%	24.109%
ا	0.000%	0.000%	0.000%	0.000%	0.000%	0.000%	0.000%
ب	0.023%	0.028%	0.022%	1.592%	0.537%	0.720%	0.360%
ة	0.000%	0.000%	0.000%	0.000%	0.000%	0.000%	14.394%
ت	0.020%	0.016%	0.064%	4.817%	0.181%	0.302%	0.213%
ث	0.009%	0.013%	0.008%	0.603%	0.074%	0.410%	0.237%
ج	0.040%	0.033%	0.158%	2.991%	0.379%	0.634%	0.117%
ح	0.002%	0.002%	0.005%	2.197%	1.456%	0.117%	0.168%
خ	0.002%	0.001%	0.009%	1.122%	0.100%	0.432%	0.182%
د	0.237%	0.139%	0.194%	10.655%	1.464%	1.858%	1.656%
ذ	0.012%	0.006%	0.007%	1.191%	0.182%	0.563%	0.273%
ر	0.147%	0.255%	0.216%	3.581%	0.724%	0.918%	1.091%
ز	0.030%	0.023%	0.093%	2.641%	0.269%	0.525%	0.428%
س	0.020%	0.003%	0.010%	1.015%	0.158%	0.367%	0.381%
ش	0.010%	0.013%	0.013%	0.520%	0.066%	0.271%	0.269%
ص	0.217%	0.127%	0.089%	2.706%	0.899%	0.499%	0.265%
ض	0.010%	0.004%	0.006%	2.005%	0.134%	0.343%	7.014%
ط	0.035%	0.011%	0.032%	1.456%	0.644%	0.566%	0.514%
ظ	0.071%	0.098%	0.076%	0.799%	0.251%	0.915%	0.590%
ع	0.000%	0.000%	0.000%	0.092%	0.005%	0.016%	0.593%
غ	0.000%	0.000%	0.000%	0.111%	0.004%	0.045%	0.063%
ف	0.017%	0.011%	0.028%	0.824%	0.080%	0.422%	0.410%
ق	0.113%	0.213%	0.181%	1.219%	0.704%	0.798%	0.523%
ك	0.014%	0.015%	0.044%	1.602%	0.150%	0.465%	0.174%
ل	0.048%	0.085%	0.142%	5.985%	1.096%	1.433%	0.682%
م	0.027%	0.022%	0.085%	7.034%	0.393%	0.662%	0.473%
ن	0.024%	0.013%	0.026%	14.902%	0.151%	0.423%	0.403%
ه	0.000%	0.000%	0.000%	0.114%	0.030%	0.034%	0.063%
و	0.015%	0.016%	0.019%	1.091%	0.078%	0.264%	0.049%
ى	0.000%	3.390%	9.504%	12.299%	7.665%	4.022%	0.000%
ي	0.378%	0.600%	0.888%	3.803%	0.699%	1.982%	0.116%

Table 10 Letter-diacritic distribution (B)

Letter	◌ٌ	◌ٍ	◌َ	◌ُ	◌ِ	◌ْ	◌ّ
ء	29.542%	47.065%	10.328%	6.056%	4.419%	0.021%	0.000%
آ	0.000%	0.000%	99.497%	0.168%	0.000%	0.000%	0.112%
أ	0.075%	0.035%	91.362%	4.942%	0.017%	3.486%	0.000%
ؤ	0.175%	0.254%	36.695%	18.963%	0.108%	43.559%	0.001%
إ	0.000%	0.041%	0.013%	0.005%	99.939%	0.002%	0.000%
ئ	0.626%	0.486%	22.025%	11.843%	26.838%	14.068%	0.000%
ا	0.000%	0.009%	59.796%	27.870%	11.840%	0.476%	0.009%
ب	0.346%	1.054%	29.672%	6.386%	38.821%	20.436%	0.003%
ة	13.645%	17.577%	27.242%	10.330%	16.789%	0.023%	0.000%
ت	0.167%	0.412%	59.611%	14.562%	9.894%	9.733%	0.009%
ث	0.418%	0.533%	45.753%	29.589%	7.219%	15.133%	0.001%
ج	0.194%	0.797%	45.243%	20.355%	13.427%	15.497%	0.136%
ح	0.218%	0.385%	57.177%	8.449%	9.842%	19.980%	0.003%
خ	0.339%	0.451%	45.616%	12.848%	12.469%	26.430%	0.001%
د	1.912%	4.533%	32.815%	13.142%	18.133%	13.250%	0.011%
ذ	0.029%	0.419%	73.265%	5.636%	11.201%	7.216%	0.001%
ر	1.463%	3.598%	45.546%	12.136%	17.786%	12.526%	0.013%
ز	0.349%	0.771%	52.957%	6.748%	24.269%	10.894%	0.004%
س	0.481%	1.588%	49.070%	12.337%	11.753%	22.816%	0.002%
ش	0.269%	1.027%	59.774%	7.886%	7.681%	22.199%	0.002%
ص	0.358%	0.549%	52.017%	8.280%	15.949%	18.042%	0.003%
ض	1.145%	3.485%	41.743%	12.728%	21.081%	10.304%	0.001%
ط	0.821%	1.062%	52.546%	10.702%	11.565%	20.044%	0.003%
ظ	0.347%	1.126%	51.397%	17.368%	15.123%	11.836%	0.003%
ع	0.478%	0.633%	65.909%	8.264%	8.478%	15.532%	0.000%
غ	0.053%	0.108%	75.659%	6.354%	6.301%	11.303%	0.000%
ف	0.491%	0.693%	54.316%	3.928%	31.093%	7.685%	0.001%
ق	0.482%	0.354%	67.014%	10.412%	9.356%	8.628%	0.003%
ك	0.162%	0.213%	63.138%	18.271%	7.247%	8.503%	0.002%
ل	0.700%	0.878%	50.903%	7.090%	23.212%	7.741%	0.005%
م	0.599%	1.028%	40.081%	13.505%	20.376%	15.708%	0.007%
ن	0.355%	0.596%	21.950%	6.147%	11.017%	43.986%	0.007%
ه	0.092%	0.201%	19.095%	48.828%	28.095%	3.445%	0.000%
و	0.029%	0.044%	75.940%	1.240%	1.493%	19.721%	0.001%
ى	0.234%	0.198%	24.491%	0.072%	1.551%	36.501%	0.072%
ي	0.060%	0.057%	40.750%	14.022%	0.601%	36.026%	0.017%

3.2.2 Letter-Position-Diacritic Matrix

Another way to look at the corpus is by using what we call the “the letter-position-diacritic matrix”, which is a matrix that shows the most frequent diacritics for a particular letter in a particular position. Table 11 below shows the letter matrix for the developed corpus. For readability, we used the following abbreviations for the diacritics.

NI	Nil	**F**	Fat-ha
M	Damma	**K**	Kasra
S	Shadda	**SF**	Shadda+Fat-ha
SM	Shadda+ Damma	**SK**	Shadda+ Kasra
DF	Double Fat-ha	**DM**	Double Damma
DK	Double Kasra	**N**	Sukoun
SDF	Shadda+Double Fat-ha	**SDM**	Shadda+Double Damma
SDK	Shadda+Double Kasra	**Rem.**	Remaining

In the table, the first column represents the letter while the rest of the columns represent the position of the letter in the word. For example, the letter ق in the word قال would have the 1st position, and so on. In each cell, the most probably diacritics are given in a descending order. For example, in the 1st position for the letter ء the most likely diacritic is F (or Fat-ha) with a probability of 17%, then K (or Kasra) with a probability of 14.8%, then M (or Damma) with a probability of 14% and the remaining cases occur 54.3% of the time.

Table 11 Letter-diacritic distribution over letter position (A)

Letter	1st	2nd	3rd	4th	5th	6th
ء	F=16.955 K=14.765 M=14.01 Rem.=54.27	F=48.157 DK=26.718 DM=14.653 Rem.=10.472	K=44.692 F=25.817 M=18.413 Rem.=11.078	K=65.022 M=21.433 F=13.195 Rem.=0.35	K=60.226 M=24.649 F=15.082 Rem.=0.043	F=61.111 K=27.778 M=11.111 Rem.=0
آ	NI=97.197 F=0.839 N=0.29 Rem.=1.674	NI=99.014 Rem.=0.986	NI=98.961 F=1.038 Rem.=0.001	NI=99.523 Rem.=0.477	NI=100 Rem.=0	NI=100 Rem.=0
أ	F=93.753 M=5.977 Rem.=0.27	F=83.181 M=9.453 N=6.289 Rem.=1.077	F=70.347 N=13.515 M=10.662 Rem.=5.476	N=47.632 F=27.033 K=14.035 Rem.=11.3	F=88.421 M=8.421 K=3.158 Rem.=0	F=93.333 N=6.667 Rem.=0
ؤ	F=15.298 M=14.277 K=14.146 Rem.=56.279	N=49.047 M=27.234 F=22.766 Rem.=0.953	N=42.71 M=41.543 F=14.061 Rem.=1.686	M=63.769 K=30.132 F=6.043 Rem.=0.056	M=77.778 N=22.222 Rem.=0	N=50 F=50 Rem.=0
إ	K=52.168 NI=47.198 Rem.=0.634	K=99.447 Rem.=0.553	K=93.454 DK=5.529 NI=0.992 Rem.=0.025	K=96.394 NI=3.606 Rem.=0	K=70.833 NI=29.167 Rem.=0	NI=100 Rem.=0
ئ	K=15.908 M=14.733 F=14.579 Rem.=54.78	K=67.998 DF=18.032 F=9.215 Rem.=4.755	K=87.17 F=6.203 M=4.95 Rem.=1.677	K=87.992 F=6.997 M=4.759 Rem.=0.252	K=87.64 M=7.191 F=4.944 Rem.=0.225	K=100 Rem.=0
ا	NI=99.496 Rem.=0.504	NI=99.962 Rem.=0.038	NI=99.559 Rem.=0.441	NI=99.158 Rem.=0.842	NI=98.922 N=1.073 Rem.=0.005	NI=81.11 N=18.89 Rem.=0
ب	K=48.096 N=27.074 F=22.837 Rem.=1.993	F=52.361 K=14.721 M=12.344 Rem.=20.574	K=38.254 F=29.125 M=18.488 Rem.=14.133	K=50.879 M=24.625 F=22.536 Rem.=1.96	K=78.006 M=13.568 F=8.058 Rem.=0.368	F=41.463 M=19.512 DM=14.634 Rem.=24.391
ة	DM=15.516 K=15.481 DK=14.912 Rem.=54.091	F=22.274 DF=20.659 K=17.504 Rem.=39.563	F=31.197 K=19.951 DM=13.851 Rem.=35.001	K=64.981 M=20.681 F=12.287 Rem.=2.051	K=62.852 M=29.049 F=7.356 Rem.=0.743	K=75.513 M=12.244 F=11.101 Rem.=1.142

Table 12 Letter-diacritic distribution over letter position (B)

Letter	1st	2nd	3rd	4th	5th	6th
ت	F=80.784 M=15.483 K=2.891 Rem.=0.842	F=42.745 K=17.754 SF=17.312 Rem.=22.189	K=32.72 F=26.341 M=20.456 Rem.=20.483	K=53.593 F=17.544 M=15.609 Rem.=13.254	K=78.346 M=20.283 F=0.887 Rem.=0.484	K=71.429 F=16.112 M=11.416 Rem.=1.043
ث	M=52.772 F=41.522 K=3.71 Rem.=1.996	F=61.363 SF=18.288 M=7.154 Rem.=13.195	F=32.394 K=32.187 M=22.192 Rem.=13.227	K=41.806 M=36.707 F=20.506 Rem.=0.981	M=47.68 K=47.531 F=4.64 Rem.=0.149	N=100 Rem.=0
ج	F=73.48 M=14.863 K=10.044 Rem.=1.613	F=47.329 K=20.362 M=16.189 Rem.=16.12	K=33.348 F=29.442 M=17.012 Rem.=20.198	K=57.569 F=21.699 M=18.724 Rem.=2.008	K=65.029 M=20.571 F=12.914 Rem.=1.486	F=33.333 K=33.333 N=25 Rem.=8.334
ح	F=82.174 M=10.476 K=6.62 Rem.=0.73	F=56.464 K=16.838 N=11.439 Rem.=15.259	K=34.377 F=26.585 M=15.6 Rem.=23.438	K=46.837 M=27.429 F=24.68 Rem.=1.054	K=60.956 M=27.238 F=10.873 Rem.=0.933	N=40 M=40 K=20 Rem.=0
خ	F=74.15 M=12.127 K=11.628 Rem.=2.095	F=40.37 N=24.707 M=18.213 Rem.=16.71	M=37.319 K=24.528 F=18.837 Rem.=19.316	K=54.122 M=26.245 F=14.539 Rem.=5.094	K=69.099 M=15.451 F=10.73 Rem.=4.72	N=66.667 K=16.667 SF=8.333 Rem.=8.333
د	F=56.955 M=22.596 K=17.023 Rem.=3.426	F=27.912 K=20.885 N=11.769 Rem.=39.434	K=31.027 F=26.158 M=20.424 Rem.=22.391	K=51.647 M=23.441 F=23.124 Rem.=1.788	K=65.639 M=20.336 F=13.553 Rem.=0.472	K=64.706 M=9.412 F=9.412 Rem.=16.47
ذ	F=77.365 K=12.842 M=8.001 Rem.=1.792	F=42.741 K=40.628 N=5.52 Rem.=11.111	K=22.475 DK=22.185 M=20.043 Rem.=35.297	K=53.338 F=24.105 M=19.277 Rem.=3.28	K=68.293 F=21.138 M=9.756 Rem.=0.813	K=64.706 F=23.529 SK=5.882 Rem.=5.883
ر	F=83.372 K=7.553 M=7.538 Rem.=1.537	F=35.459 K=18.593 SF=12.545 Rem.=33.403	K=35.845 F=29.898 M=16.139 Rem.=18.118	K=50.816 F=24.002 M=23.029 Rem.=2.153	K=75.072 M=15.48 F=9.012 Rem.=0.436	K=86.709 F=6.013 M=3.165 Rem.=4.113
ز	F=68.84 K=14.727 M=11.977 Rem.=4.456	F=33.173 SF=20.629 K=18.143 Rem.=28.055	K=34.025 F=30.309 M=21.315 Rem.=14.351	K=61.181 F=21.277 M=13.091 Rem.=4.451	K=85.556 M=7.778 F=4.444 Rem.=2.222	N=38.462 SM=15.385 K=15.385 Rem.=30.768

Table 13 Letter-diacritic distribution over letter position (C)

Letter	1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}	6^{th}
س	F=68.968 M=20.848 K=8.971 Rem.=1.213	F=30.467 SF=18.918 N=13.704 Rem.=36.911	K=40.398 F=27.331 M=17.021 Rem.=15.25	K=57.072 F=23.551 M=18.038 Rem.=1.339	K=60.691 M=23.026 F=13.898 Rem.=2.385	K=42.857 F=30.612 N=14.286 Rem.=12.245
ش	F=76.889 M=12.74 K=8.594 Rem.=1.777	SF=51.787 F=12.551 N=12.442 Rem.=23.22	K=35.571 F=28.236 N=26.766 Rem.=9.427	K=46.275 F=33.083 M=14.897 Rem.=5.745	K=53.363 M=29.148 F=16.143 Rem.=1.346	F=70 K=10 M=10 Rem.=10
ص	F=88.11 K=6.273 M=4.43 Rem.=1.187	SF=37.469 F=14.631 K=13.373 Rem.=34.527	F=33.083 K=28.397 M=15.237 Rem.=23.283	K=55.963 F=23.827 M=18.2 Rem.=2.01	K=68.317 M=19.802 F=8.911 Rem.=2.97	F=28.571 K=28.571 SF=14.286 Rem.=28.572
ض	F=74.489 K=10.093 M=7.314 Rem.=8.104	F=27.213 K=22.7 M=13.444 Rem.=36.643	K=56.294 F=22.42 M=11.713 Rem.=9.573	K=47.376 F=30.845 M=19.744 Rem.=2.035	K=48.23 F=33.628 M=17.699 Rem.=0.443	M=50 N=33.333 K=16.667 Rem.=0
ط	F=80.459 M=10.524 K=4.564 Rem.=4.453	F=28.272 SF=26.142 K=16.997 Rem.=28.589	F=39.906 K=24.056 M=19.615 Rem.=16.423	K=49.018 N=19.181 F=15.64 Rem.=16.161	K=69.935 M=17.102 F=12.418 Rem.=0.545	K=81.25 F=12.5 M=6.25 Rem.=0
ظ	F=68.383 M=15.159 K=7.17 Rem.=9.288	SF=25.33 F=21.099 M=19.742 Rem.=33.829	K=33.726 F=31.475 M=18.369 Rem.=16.43	K=41.031 F=38.124 M=18.351 Rem.=2.494	K=77.061 M=12.545 F=10.394 Rem.=0	M=50 DM=50 Rem.=0
ع	F=86.741 M=6.614 K=6.413 Rem.=0.232	F=54.615 N=16.546 K=12.231 Rem.=16.608	F=31.198 K=30.588 M=19.287 Rem.=18.927	K=58.551 F=21.842 M=16.914 Rem.=2.693	K=59.124 M=24.447 F=15.86 Rem.=0.569	F=48.276 M=20.69 K=17.241 Rem.=13.793
غ	F=85.332 M=8.158 K=3.422 Rem.=3.088	F=63.81 N=16.562 M=10.811 Rem.=8.817	N=32.244 K=28.38 F=24.68 Rem.=14.696	F=46.981 K=35.22 M=13.815 Rem.=3.984	K=49.254 M=23.881 F=20.896 Rem.=5.969	F=60 M=40 Rem.=0
ف	F=52.888 K=45.824 M=1.049 Rem.=0.239	F=49.028 K=17.948 M=12.139 Rem.=20.885	K=48.17 F=21.926 M=13.55 Rem.=16.354	K=48.507 F=29.412 M=20.543 Rem.=1.538	K=66.012 M=20.438 F=13.26 Rem.=0.29	M=36.667 K=33.333 F=16.667 Rem.=13.333

Table 14 Letter-diacritic distribution over letter position (D)

Letter	1st	2nd	3rd	4th	5th	6th
ق	F=86.758 M=8.458 K=4.367 Rem.=0.417	F=51.54 M=16.866 K=16.553 Rem.=15.041	F=36.756 K=23.34 M=17.824 Rem.=22.08	K=50.267 F=23.339 M=21.042 Rem.=5.352	K=70.399 M=17.382 F=12.013 Rem.=0.206	F=47.368 M=31.579 K=10.526 Rem.=10.527
ك	F=77.301 M=17.737 K=4.229 Rem.=0.733	F=56.616 M=18.649 K=14.872 Rem.=9.863	F=45.497 M=28.176 K=15.774 Rem.=10.553	F=40.13 K=31.58 M=27.678 Rem.=0.612	F=55.658 K=30.591 M=13.253 Rem.=0.498	F=62.963 M=22.222 K=7.407 Rem.=7.408
ل	F=65.178 K=32.947 M=1.026 Rem.=0.849	F=31.005 SF=26.124 N=13.675 Rem.=29.196	K=35.428 F=29.251 M=22.835 Rem.=12.486	K=38.7 F=34.74 M=23.518 Rem.=3.042	K=72.553 M=15.504 F=10.848 Rem.=1.095	M=27.723 F=25.743 N=21.782 Rem.=24.752
م	F=42.724 K=36.417 M=20.064 Rem.=0.795	F=43.018 M=20.7 SF=11.908 Rem.=24.374	N=30.838 F=30.501 K=18.517 Rem.=20.144	N=41.218 F=22.361 K=19.597 Rem.=16.824	N=76.06 K=8.83 M=7.852 Rem.=7.258	N=74.912 F=11.661 K=7.067 Rem.=6.36
ن	F=75.018 M=13.605 K=9.237 Rem.=2.14	SF=30.396 F=30.297 N=22.43 Rem.=16.877	F=51.256 K=26.993 M=10.732 Rem.=11.019	K=45.402 F=41.641 M=7.843 Rem.=5.114	F=69.947 K=28.024 M=1.389 Rem.=0.64	F=76.006 K=22.491 SF=0.621 Rem.=0.882
ه	F=53.246 M=36.428 K=9.076 Rem.=1.25	M=41.649 F=27.544 K=26.849 Rem.=3.958	M=46.022 K=35.122 F=17.003 Rem.=1.853	M=47.063 K=35.281 F=17.102 Rem.=0.554	M=56.046 K=22.078 F=20.044 Rem.=1.832	M=80.182 K=8.884 F=8.428 Rem.=2.506
و	F=98.525 M=1.025 Rem.=0.45	NI=51.748 F=28.501 N=12.036 Rem.=7.715	NI=74.612 F=12.25 N=5.862 Rem.=7.276	NI=94.107 K=2.139 F=1.71 Rem.=2.044	NI=98.154 N=0.929 Rem.=0.917	NI=84.422 F=10.553 N=4.02 Rem.=1.005
ى	NI=97.419 K=0.388 F=0.382 Rem.=1.811	NI=99.729 Rem.=0.271	NI=99.594 Rem.=0.406	NI=99.827 Rem.=0.173	NI=97.736 SM=1.178 SK=0.725 Rem.=0.361	NI=85.714 F=9.524 SK=4.762 Rem.=0
ي	F=71.11 M=28.155 Rem.=0.735	NI=40.915 N=39.099 F=11.319 Rem.=8.667	NI=59.543 N=11.174 F=10.993 Rem.=18.29	NI=50.425 SM=21.8 SK=9.916 Rem.=17.859	SM=47.009 NI=18.07 SK=15.912 Rem.=19.009	SM=53.286 SF=17.57 SK=15.486 Rem.=13.658

Using this matrix, we can extract patterns (or rules) that stem from the systematic nature of Arabic. One such rule would be that when the letter أ is encountered at the 1st position; it's most probably going to have the diacritic Fat-ha (93.8% of the time) and then the diacritic Damma (6% of the time). This means that it is very unlikely that it will have any other diacritic, which can be used to reduce errors produced by statistical methods.

3.3 Summary

In this chapter, we discussed the approach followed in the development of the diacritized corpus (which is used later in our system). The process involved four stages: collecting raw textual materials from different sources, extracting texts from those materials, cleaning the extracted text, and semi-automatic review of the text (with the help of some tools we built such as the diacritization assistant). We also examined the corpus from an analytical point-of-view and looked at the distribution of diacritics on letters and positions within words.

CHAPTER 4

AUTOMATIC DIACRITIZATION APPROACH

The problem of automatic diacritization has been extensively researched using a variety of methodologies (which are mostly statistical), as we conveyed in the literature survey (Chapter 2). Nonetheless, a few researchers have utilized the rule-based methodologies as a primary approach. In our system, we used a hybrid of the two approaches to achieve a maximal performance. In this chapter, we explicate the different methodologies that we use in our system, whether they produce full or partial diacritization. We also explain how the system combines these methods to produce the best possible diacritization.

This chapter is divided into four sections. Section 4.1 explains the statistical methods the system uses, which are made primarily of *N*-grams whether word-grams, POS-grams, or letter-grams. Section 4.2 explains the rule-based methodologies and how rules were inducted and then applied to undiacritized texts. Section 4.3 describes the hybrid approach manifested in the developed system and the way these different methodologies are combined to get the best performance. Finally, Section 4.4 gives the chapter summary.

4.1 Statistical Approach

The statistical approach is the primary one used for the diacritization problem. Statistical methods have proven to be well-performing in terms of accuracy and speed. But their performance depends heavily on the corpus used to collect statistics and build training models. Thus, it is critical for such methods to work effectively that the corpus used is large enough and accurate enough, which we tried to build as explained in Chapter 3.

In our system, we use basic *N*-grams on three different levels: letter-grams, word-grams, and POS-grams. Letter-grams are better suited for unknown words (which may be because they are not included in the used lexicon or because they constitute foreign proper nouns). In contrast, word-grams are best suited for common phrases or expressions that usually are uniquely diacritized (which is often the case when N is larger than 2).

POS-grams are best-suited for case-endings where they perform very well, especially at simple grammatical rules such as the fact that prepositions are always followed by genitive nouns.

In this section we explain how these *N*-grams were extracted from the training corpus (in Subsection 4.1.1). Thereafter, we explain the diacritization algorithms used for each type of *N*-gram, in Subsection 4.1.2.

4.1.1 *N*-gram Extraction

There are many tools available for *N*-gram extraction on the word and letter levels. In our system, we built a customizable *N*-gram extraction tool that is more applicable to our needs. In this tool, the user can select any of the three *N*-gram types and the required value of N. He/she may also choose whether or not to include non-Arabic words, numbers, or punctuation.

Irrespective of the type of *N*-gram, the basic mechanism is the same. *N*-grams are stored in a hash table (for faster lookup). When an *N*-gram is encountered, the tool checks whether it has been previously seen or not. If seen, the frequency of the *N*-gram is incremented. Otherwise, it is added to the hash table with a frequency of 1. Once the entire corpus is read, the *N*-grams stored in the hash table are sorted by frequency and saved to disk.

In the case of letter-grams and word-grams, the *N*-gram extraction is straightforward, unlike the POS-grams, which require a more complicated way. For such grams, the POS of a word can be found from its diacritics using AraMorph [30]. However, some words can have multiple POS tags for the same diacritical form. In such cases, the tool treats all tags equally and finds all possible *N*-grams combinations. Obviously, this is not the most accurate solution since a word cannot have multiple tags in a certain context. However, the impact of such ambiguity is reduced as the corpus gets larger. (For a complete list of AraMorph's POS tags, please refer to Appendix V)

Table 15 shows the most common POS tags extracted from the corpus and their frequencies. Note that the START and END tags are implicit tags that are used to mark the start and end of a sentence, respectively.

Table 15: Most common POS tags

No.	Tagged word	Frequency
1	START	4920539
2	END	4920539
3	PREP	1969259
4	VERB_PERFECT+PVSUFF_SUBJ:3MS	1735560
5	DET+NOUN+CASE_DEF_GEN	1103516
6	CONJ	969346
7	NOUN+CASE_DEF_GEN	947948
8	PREP+PRON_3MS	884465
9	NOUN+CASE_DEF_NOM	786949
10	CONJ+VERB_PERFECT+PVSUFF_SUBJ:3MS	686729
11	NOUN+CASE_DEF_ACC	613508
12	NOUN+CASE_INDEF_GEN	584100
13	PREP+PRON_1S	569504
14	NOUN	566289
15	NOUN+NSUFF_MASC_SG_ACC_INDEF	529000
16	IV3MS+VERB_IMPERFECT+IVSUFF_MOOD:I	474943
17	NEG_PART	457211
18	DET+NOUN+CASE_DEF_NOM	445821
19	NOUN+CASE_INDEF_NOM	439038
20	RELPRON	409553
21	NOUN+CASE_DEF_GEN+POSS_PRON_3MS	363953
22	ADV	338232
23	NOUN+CASE_DEF_NOM+POSS_PRON_3MS	318228
24	NOUN_PROP+CASE_DEF_NOM	311463
25	ADJ+CASE_INDEF_GEN	304795
26	ADJ+CASE_INDEF_NOM	298107
27	DEM_PRON_MS	277867
28	PREP+NOUN+CASE_DEF_GEN	269387
29	DET+NOUN+NSUFF_FEM_SG+CASE_DEF_GEN	268277
30	NOUN_PROP+CASE_DEF_GEN	268135

Table 16 shows the details of extracted *N*-grams for each type. The maximum N reached is 7 for letter-grams, 5 for word-grams and POS-grams. Beyond these limits, the system crashes because of an out-of-memory exception. In our experimentation platform, the RAM size was 8GB and these limits were the maximum that can be reached for such memory size.

Table 16: Extracted *N*-grams statistics

Type	Min. *N*		Max. *N*		Included Group	
	N	Count	*N*	Count	Non-Arabic	Punctuation
Letter-gram	1	47	7	3,053,221	No	No
Word-gram	1	777,969	5	12,840,859	No	No
POS-gram	1	2,724	5	2,661,370	Yes	Yes

4.1.2 Diacritization Using *N*-grams

Once *N*-grams are extracted, they can be used for diacritization. Finding the best diacritization for a given sentence can be reduced to a graph search problem where nodes represent *N*-grams, and edges represent connectivity between *N*-grams, as depicted by Figure 5. The complexity of a Brute-force search algorithm for such problem would be $O(C^L)$ where C is the number of unigrams, and L is the length of the sentence (letter-wise, word-wise, or POS-wise); assuming of course the worst-case scenario where every unigram is connected with all others (which is not necessarily the case).

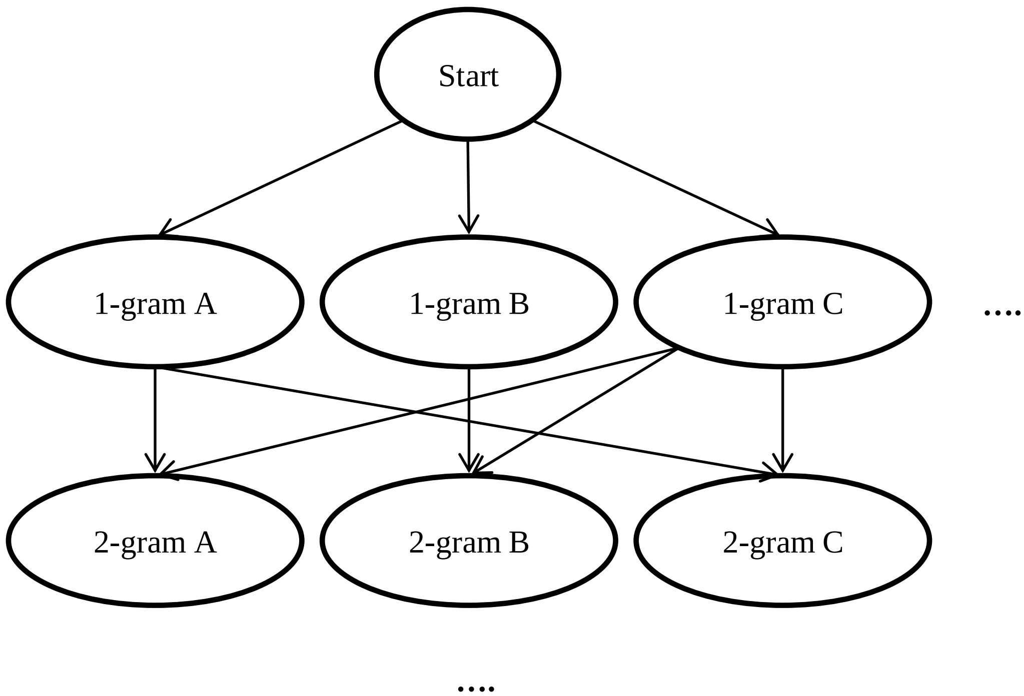

Figure 5 Best *N*-gram Sequence Search Problem

In our system, we chose to use the greedy approach without backtracking; i.e. once a sequence is chosen, it will never be changed. This approach is used for all three types of *N*-grams with slight variations, as we will explain later on.

Letter/Word Grams Search

In the case of letter grams and word grams, the search algorithm is given in Figure 6 and Figure 7. First, the *N*-grams are loaded from the disk with the predefined limits MinN, representing the minimum N to retrieve, and MaxN representing the maximum N to retrieve. Also, the user may choose to limit the *N*-grams by setting the MinFreq which represents the minimum frequency acceptable.

```
Load Letter N-grams from MinN to MaxN, Limited by MinFreq
For each letter in GetLetters(sentence)
   If IsDiacritized(letter) Then Continue
   Set currentN = MaxN
   While currentN >= MinN
      For each possible right N-gram seq_right
         Set currentM = MaxN
         While currentM >= MinN
            For each possible left N-gram seq_left
               If IsCompatible(seq_left, seq_right) Then
                  Set letter.Diacritization =
GetCombinedDiacritiztionForLetter(letter, seq_right,
seq_left)
                  Next letter
               End If
            Loop
         Set currentM = currentM - 1;

      Loop
      Set currentN = currentN - 1;
   Loop
Loop
```

Figure 6 Greedy Letter *N*-gram Diacritizer

After the data is loaded and a sentence diacritization is requested, the algorithm loops over the tokens (letter or words) and it tries to find the maximum-length *N*-gram that includes the token at hand. This *N*-gram must be compatible from right and left in order to be selected. If no such *N*-gram is found, N is decremented by 1 and the search is resumed, until N becomes less than the user-defined MinN. In such a case, the search algorithm terminates for this token and moves to the next one.

Once an *N*-gram is found, the corresponding diacritics are extracted and combined with the current token's diacritics (if they exist).

```
Load Word N-grams from MinN to MaxN, Limited by MinFreq
Set Tokens = Tokenize(sentence)
For each token in Tokens
   If GetDiacritizationLevel(token) >= 0.99 Then Continue
   Set currentN = MaxN
   While currentN >= MinN
      For each possible right N-gram seq_right
         Set currentM = MaxN
         While currentM >= MinN
            For each possible left Word N-gram seq_left
               If IsCompatible(seq_left, seq_right) Then
                  Set token.Diacritization =
GetCombinedDiacritiztionForToken(token, seq_right, seq_left)
                  Next token
               End If
            Loop
         Set currentM = currentM - 1;

      Loop
      Set currentN = currentN - 1;
   Loop
Loop
```

Figure 7 Greedy Word *N*-gram Diacritizer

POS Grams Search

The POS-grams search (explained by Figure 8) is somewhat similar to the word/letter gram search, except that the compatibility condition between the left and the right sequences is relaxed for performance reasons. However, the sequence selected must produce one and only one diacritization. Recalling that a single diacritical form can have multiple POS tags, similarly, a single POS tag can have multiple diacritical forms. In such a case, where there are multiple diacritical forms, the algorithm declares this token as unresolved and moves to the next one.

```
Load POS N-grams from MinN to MaxN, Limited by MinFreq
Set Tokens = Tokenize(sentence)
For each token in Tokens
     Set token.Tags = GetPossibleTags(token)
     If token.Tags.Size == 1 Then Set token.SelectedTag = 1
     Else Set token.SelectedTag = -1
Loop
For each token in Tokens
     If token.SelectedTag != -1 Then Continue
     Set currentN = MaxN
     Set possibleTags = {}
     While currentN >= MinN
          For each possible POS N-gram seq
               add (seq, prob) to possibleTags
          Loop
          Set currentN = currentN - 1;
     Loop
     Set Token.SelectedTag = most probable tag
     Set Token.Diacritization = get case from selected tag
Loop
```

Figure 8 Greedy POS *N*-gram diacritizer

4.2 The Rule-based Approach

The need for rule-based methods to tackle the diacritization problem stems from the fact that Arabic diacritization is systematic by nature. In general, these rules can be on the lexical level, the syntactic level, or the semantic level. However, the complexity of such rules makes them computationally unfeasible. Consequently, most researchers resort to the statistical methods over the rule-based methods (as in [5], [2], and [12]).

In our system, we employ a large number of rules that were inducted automatically from the training set of the corpus. We explain this induction mechanism in Subsection 4.2.1. After, we decide which rules to use, we then apply these rules in a predefined sequence on the validation and test sets using a Hash-table-based search algorithm. The algorithm used to apply the rules is explained in Subsection 4.2.2.

4.2.1 Rule Induction

We divided the rules into three groups; the first one consists of features relating to the current letter such as the letter itself, the previous and the next letters, and the position of the current letter. In this group (Group A), diacritics are assumed to be missing and hence are not part of the feature set. Samples of “Group A” rules are shown in Table 17.

Table 17: A sample of the letter-specific rules (Group A)

Position	PrevLetter	Letter	NextLetter	Diacritic	Hit Rate	Frequency
2	ا	ل	م	◌ْ	99.735	665043
1	N	و	ا	◌َ	99.886	610636
1	N	ق	ا	◌َ	99.85	429363
2	ع	ل	ى	◌َ	99.779	408381
1	N	ك	ا	◌َ	99.88	307931
3	ن	ه	N	◌ُ	99.356	279261
1	N	و	أ	◌َ	99.854	229374
2	ا	ل	ح	◌ْ	99.602	227992
1	N	و	إ	◌َ	99.921	221461
1	N	أ	ي	◌َ	99.917	215167
2	ق	و	ل	◌ْ	99.815	211398
2	ا	ل	ع	◌ْ	99.692	205745
1	N	و	ه	◌َ	99.49	202982
2	ل	ه	N	◌ُ	99.942	194880
2	إ	ل	ى	◌َ	99.825	192718
2	ل	م	N	◌ْ	99.253	191102
2	ل	أ	ن	◌َ	99.877	187423
1	N	و	م	◌َ	99.8	183040
1	N	ف	إ	◌َ	99.957	177549

In the second group (Group B), we added contextual features such as the previous word, and the next word, as shown in

Table 18. It is important to note that Word features are limited to the 1000 most common words only (also extracted from the corpus, see Appendix II). This means that if the word is not a common word, it is replaced with the empty marker “N”.

Table 18: A sample of the letter-word rules (Group B)

Position	PrevWord	PrevLetter	Letter	NextLetter	Diacritic	Hit Rate	Frequency
1	N	N	و	ا	◌َ	99.885	586711
2	N	ا	ل	م	◌ْ	99.727	554168
1	N	N	أ	ن	◌َ	99.055	486933
2	N	ع	ل	ى	◌َ	99.775	387841
1	N	N	ق	ا	◌َ	99.842	381964
3	N	ن	ه	N	◌ُ	99.45	262505
1	N	N	ك	ا	◌َ	99.882	240213
1	N	N	و	أ	◌َ	99.852	225177
1	N	N	و	إ	◌َ	99.92	216608
1	N	N	أ	ي	◌َ	99.924	206619
1	N	N	و	ه	◌َ	99.593	197289
2	N	ا	ل	ح	◌ْ	99.575	193116
2	N	ل	ه	N	◌ُ	99.942	188532
2	N	ق	و	ل	◌ْ	99.806	185933
2	N	إ	ل	ى	◌َ	99.824	185368
2	N	ل	أ	ن	◌َ	99.879	183793
1	N	N	و	م	◌َ	99.797	180071
1	N	N	ف	إ	◌َ	99.956	174126
2	N	ا	ل	ع	◌ْ	99.69	172609

Finally, the third group of rules (Group C) includes diacritics as part of the feature set; excluding the current letter's diacritic which the desired output is. Samples are given in Table 19.

Table 19: A sample of the letter-diacritic rules (Group C)

CurrLetter	Position	PrevLetter1	PrevDiacritic1	Diacritic	Hit Rate	Frequency
ل	2	ع	◌َ	◌َ	93.505	730872
ي	3	ل	◌َ	◌ْ	96.119	424860
و	2	أ	◌َ	◌ْ	93.415	367240
ه	4	ي	◌ْ	◌ِ	93.294	343373
ه	2	ل	◌َ	◌ُ	91.315	228053
و	2	ق	◌َ	◌ْ	98.154	225582
إ	2	و	◌َ	◌ِ	99.953	221418
أ	2	ل	◌ِ	◌َ	98.272	216666
أ	2	و	◌َ	◌َ	91.545	210103
م	2	ل	◌َ	◌ْ	92.556	191317
إ	2	ف	◌َ	◌ِ	99.98	177553
ك	3	ل	◌ِ	◌َ	97.471	175756
ل	2	ذ	◌َ	◌ِ	99.924	175162
ه	3	ن	◌َّ	◌ُ	91.985	156590
ه	4	ل	◌ُ	◌ُ	97.401	151707
و	3	ه	◌ُ	◌َ	95.329	144931
ع	2	ب	◌َ	◌ْ	93.601	141481
ن	2	م	◌َ	◌ْ	91.706	137773
إ	3	ل	◌ْ	◌ِ	99.961	127352
م	2	ث	◌ُ	◌َّ	99.209	126760

4.2.2 Applying the rules

After the rules are decided, they can be applied to any test text according to the algorithm represented by Figure 9. First, the rules are retrieved from text files that were stored in the induction phase. Each text file is a table with the first row representing the features that make up the rules. Following the features are the diacritic corresponding to the rule, the hit rate of this rule in percentage, and the frequency of encountering the rule. The rules are stored in a hash table where the key is the rule while the value is the expected diacritic.

The loading function filters the rules based on the defined limits for both the hit rate and the frequency. If the rules intended are positive rules, then the loading function will also filter rules with empty diacritics. On the other hand, if the rules intended are negative ones, then the loading function will filter rules that do not have empty diacritics.

After the rules are loaded (which usually occur at the initialization phase), the sentence is divided into letter objects each corresponding to an Arabic letter. The object contains meta-data about the letter such as the diacritic of the letter, if any, a link to previous letter, a link to the next letter, and the current word. The letters are stored in an array to be used when searching for applicable rules using the letter meta-data.

```
Load rules into rules, Limited by MinFreq, MinHitRate
Set letters = {}
For each letter in sentence
      Set letter properties (diacritic, prevLetter, nextLetter, etc...)
      Letters += letter
Loop
For each letter in letters
      Get rule from letter.Properties
      If rules contain rule Then
            Set letter.Diacritic = rules[rule]
      End If
Loop
```

Figure 9. Rules Application Algorithm

4.3 The Diacritizer

In this section, we give technical details about the implemented system. In Subsection 4.3.1, we take an overview of the system's architecture. In Subsections 4.3.2, 4.3.3 and 4.3.4, we explain the preprocessing, hybrid diacritization, and the post-processing phases, respectively.

4.3.1 Architecture

The system consists of 5 online components and one offline component, which is the corpus. The main engine, depicted in Figure 10 as the diacritizer, is the basic component which is responsible for all functionality in the preprocessing, diacritization, and post-processing phases. It receives the input text from the user interface, desktop or web, which represents the text to be diacritized. Furthermore, depending on the user options, the diacritizer interacts with the rules and *N*-grams databases. It also interacts with the

Utilities toolbox which provides useful tools such as the tokenizer (for tokenizing Arabic words), the diacritization normalizer (which removes inconsistencies from a given diacritization), and the text replacer (which helps keep the punctuation intact when performing substitution). For tools used in the development of the diacritizer, please refer to Appendix IV.

4.3.2 Preprocessing

When a text is received by the diacritizer, it performs certain preprocessing tasks before it proceeds to the diacritization phase. The diacritization of the text, if any, is normalized such that any undesirable inconsistency is resolved. Then, the text is tokenized into tokens that correspond to words or letters depending on the chosen diacritization method. If needed, the morphological analyzer will produce all the possible morphological analyses for each word to be later used by the POS-grams diacritizer.

When a user inserts the desired text, he/she can also select the diacritization methods and the order in which they are executed. This order is essential because the diacritization can be very different if the order changes. Accordingly, the system will initialize the needed diacritizers with user-defined parameters.

4.3.3 Hybrid Diacritization

Once diacritizers are initialized, the system executes them in order on the given text. The output of each diacritizer is given to the next one. Figure 11 depicts the most recommended sequence of diacritizers. As shown in the figure, we differentiate between the so-called “strict diacritizer” and the “relaxed diacritizer”. The difference between the two is in the strictness of the parameters used. In a strict rule-based diacritizer, for example, the hit rate must be more than or equal to 99.7%. On the other hand, the relaxed rule-based diacritizer has a hit rate of 98% or more.

4.3.4 Post-processing

After the output from the diacritizers is retrieved, final diacritization normalization is performed and the diacritized text is displayed to the user.

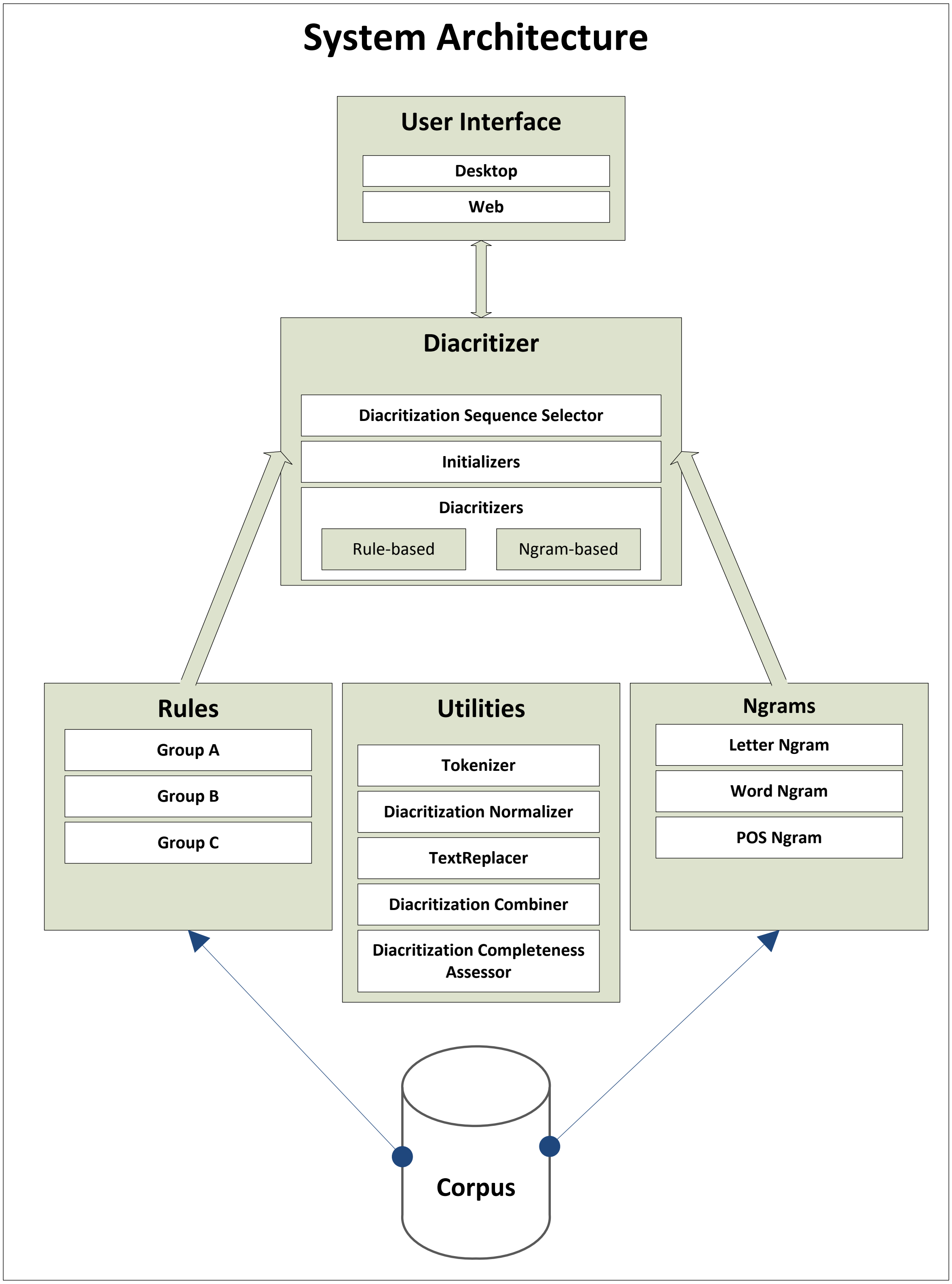

Figure 10: The system architecture

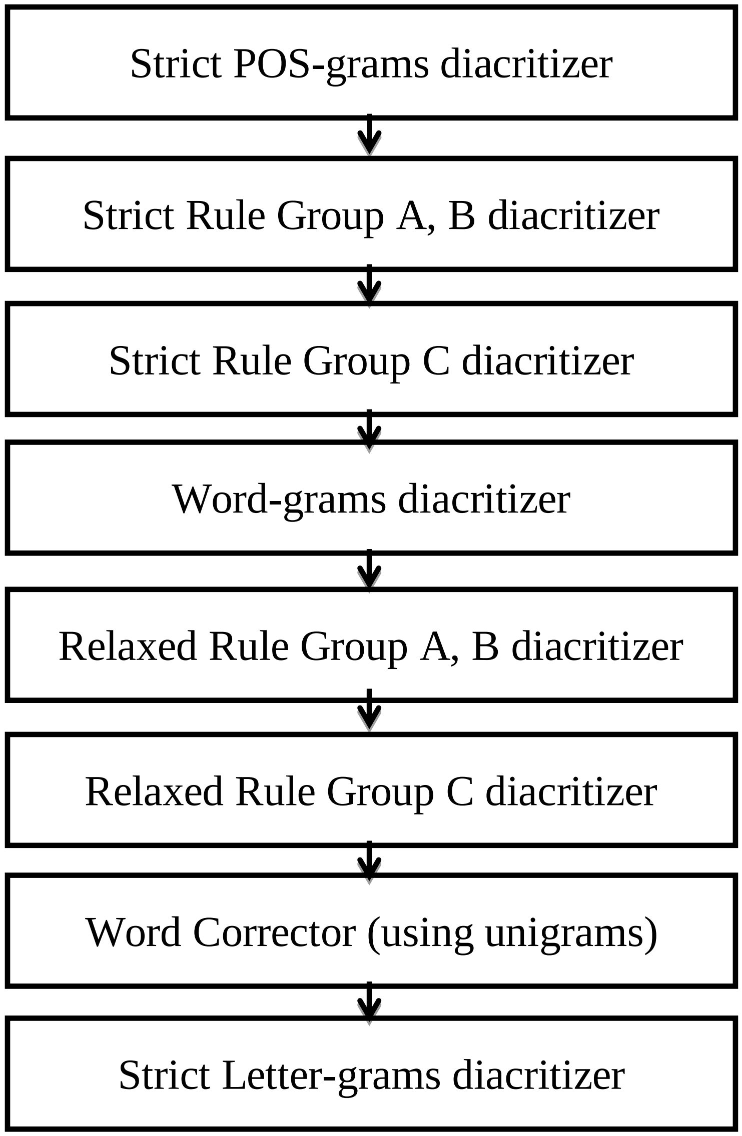

Figure 11: Recommended sequence of diacritizers

4.4 Summary

In this chapter, we explained our hybrid approach which combines statistical methods and mined rules. The first component uses *N*-grams extracted the diacritized corpus on three different levels: word-level, letter-level, and POS-level. The component uses those grams in a greedy way to find an optimal diacritization for a given sentence.

The mined rules in the second component are extracted from the corpus such that their hit rates are as close to 100% as possible. We grouped those rules into three groups: group A uses only features extracted from the current word such as the current letter, the preceding letter, and the succeeding letter. The second group, group B, adds previous words and next words as features in addition to those in group A. The third group, group C, takes the same features as group A and B but with the inclusion of known diacritics. Those rules from all groups are then applied to the sentence at hand.

In the next chapter, we evaluate each component independently as well as in complement to the other.

CHAPTER 5

DIACRITIZER EVALUATION

The evaluation phase of any implemented system is perhaps the most important as it defines the boundary between success and failure. In this chapter, we discuss our evaluation approach by defining the performance metrics that were used. Section 5.1 gives brief description of each metric and examples to demonstrate how they are computed. In Section 5.2, we explain the evaluation methodology for our system compared with other similar systems. Finally, Section 5.3 discusses the results of the diacritizer and compares it with five others.

5.1 Performance Metrics

There are many metrics to measure the performance of any automatic diacritizer. In broad terms, these metrics fall under two main categories: the correctness measuring metrics (such as the error rate in the produced output), and the usability measuring metrics (such as the number of words processed per second or the memory utilized). Most researchers focus primarily on the first category of metrics. Notwithstanding, the second category is as important as the first one from the end-user perspective, who could be novice Arabic learners who want to improve their Arabic skills or other researchers in the Arabic computing field. In either case, it would be extremely troublesome if the system at hand could not produce the diacritized text in convenient time.

There are five evaluation metrics that will be utilized in our system evaluation, three of which fall under the correctness metrics while the other two fall under the usability metrics. These metrics are Word-Error-Rate, Diacritic-Error-Rate, Diacritization-Level, Words-per-Second, and Peak-Memory. Following is a brief description of each metric.

5.1.1 Word-Error-Rate (WER)

Word-Error-Rate is the ratio of erroneous words to the total number of Arabic words (including or excluding case-endings) [23], as denoted by Equation 1.

$$WER = \frac{number\ of\ incorrectly\ diacritized\ words}{total\ number\ of\ words}$$

Equation 1 Word-error rate (WER)

It is important to notice that words included in both the nominator and the denominator must be Arabic words. This means that no non-Arabic words are taken into consideration. Examples of non-Arabic words are English words, numbers, and punctuations. The reason for this is obvious: why include non-Arabic words or numbers in the calculation when they do not need diacritization to begin with.

An example of applying computing this metric is given in Figure 12, with case-endings (CE) ($WER_{with\ CE}$) and without case-endings ($WER_{without\ CE}$).

Original text	لكل مجتهد نصيب
Automatically diacritized text	لِكُلِّ مِجْتَهِدٍ نَصِيبٍ
Correctly diacritized text	لِكُلِّ مُجْتَهِدٍ نَصِيبٌ
$WER_{with\ CE} = \frac{2}{3} = 0.67$ $WER_{without\ CE} = \frac{1}{3} = 0.33$	

Figure 12 Example of calculating WER metric.

5.1.2 Diacritic-Error-Rate (DER)

Diacritic-Error-Rate is the ratio of erroneous diacritics to the total number of Arabic letters (including or excluding case-endings) [23], as denoted by Equation 1.

$$DER = \frac{number\ of\ incorrectly\ diacritized\ letters}{total\ number\ of\ Arabic\ letters}$$

Equation 2 Diacritic-error rate (DER)

In this metric also, only Arabic letters are included in the computation. An example of applying this formula is given in Figure 13.

Original text	لكل مجتهد نصيب
Automatically diacritized text	لِكُلِّ مِجْتَهِدٍ نَصِيبٍ
Correctly diacritized text	لِكُلِّ مُجْتَهِدٍ نَصِيبٌ
$DER_{with\ CE} = \frac{2}{12} = 0.17$ $DER_{without\ CE} = \frac{1}{9} = 0.11$	

Figure 13 Example of calculating DER metric.

Another thing to note here is that, when case-endings are not considered, the denominator changes to exclude the letters that represent case-endings. This is tricky because the position of the case-endings is not always known. Some researchers assumed that case-endings are always the last letter of a word, which is obviously not true. This assumption potentially produces unreliable metric values, whether in favor of the system or against it. So, by not using this assumption, the case-endings in the test data must be manually marked, which we followed as explained in the results and discussion chapter.

Table 20: The effect of the last-letter assumption

Case Letter	**Last Letter**	$\boldsymbol{DER_{with\ CE}}$	$\boldsymbol{DER_{without\ CE}}$	$\boldsymbol{WER_{with\ CE}}$	$\boldsymbol{WER_{without\ CE}}$
Correct	Correct	No effect	No effect	No effect	No effect
Incorrect	Incorrect	No effect	No effect	No effect	No effect
Correct	Incorrect	No effect	Smaller	No effect	Potentially smaller[1]
Incorrect	Correct	No effect	Larger	No effect	Potentially larger[2]

Table 20 examines the effect of the last-letter assumption on the calculation of metrics. As the table shows, there are 4 possibilities for the correctness of diacritics that are put on the case letter and the last letter (of a particular word). The first 2 possibilities, when both letters are diacritized either correctly or incorrectly, have no effect on any of the metrics. However, when the case letter is correctly diacritized while the last letter is not (3rd possibility), the $\boldsymbol{DER_{without\ CE}}$ will necessarily become smaller (since the number of errors is reduced by 1 when it shouldn't) and $\boldsymbol{WER_{without\ CE}}$ may potentially become smaller as well (since the word may happen to contain other incorrect diacritics). Same logic applies to the 4th possibility where the case letter is incorrectly diacritized while the last letter is correctly diacrtitized.

[1] The reason it is "potentially smaller" is that it depends on other diacritics. If any other diacritic is incorrect, the WER will not be affected.

[2] The reason it is "potentially larger" is that it depends on other diacritics. If any other diacritic is incorrect, the WER will not be affected.

5.1.3 Diacritization-Level (DL)

We introduced Diacritization-Level metric in this thesis, which measures how much of the text is diacritized. The reason for its introduction is the fact that some of the methods produce partial diacritization, albeit these methods are not used on their own but with other complementing methods. This metric is denoted by Equation 3.

$$DL = \frac{number\ of\ diacritized\ letters\ (implcitily\ or\ explcitily)}{total\ number\ of\ Arabic\ letters}$$

Equation 3 Diacritization level (DL)

An example of applying this formula is given in Figure 14.

Original text	لكل مجتهد نصيب
Diacritized text	لِكُلِّ مُجْتَهِدٍ نَصِيبٌ
$DL_{with\ CE} = \frac{11+1}{12} = 100\%$ $DL_{without\ CE} = \frac{8+1}{9} = 100\%$	

Figure 14 Example of calculating DL metric.

In the numerator, there is no distinction between explicitly diacritized letters and implicitly diacritized ones. This means that some letters are left undiacritized (even in the most complete diacritization) because of one of three reasons.

1. The first reason is that they may be silent letters (not pronounced) such as *the plural suffix* as in جَاؤُوا.
2. The second reason is that their diacritics can easily be inferred from the context such as *the definitive article* (ال) as in الرِّجَالُ.
3. The third reason is that they may represent long vowels as in the previous two examples.

Please note that there is no direct way to count the number of implicit diacritics automatically except using heuristics.

5.1.4 Words-per-Second (WPS)

Words-per-Second metric falls under the usability measuring metrics. It measures the number of diacritized words in a second. This metric is denoted by the following formula:

$$WPS = \frac{number\ of\ diacritized\ words}{time\ taken\ is\ seconds}$$

So, if the input to the system was 100 words and the system outputted the diacritized words in 1 second, then WPS will be 100 words/second. However, if the system took 0.5 seconds for the same input, the WPS will then be 200 words/second, which is clearly much better.

5.1.5 Peak-Memory (PM)

Peal- Memory is the largest size of RAM used at any point during the processing of the input, measured in bytes. So, the smaller the PM used by a system, the better.

5.2 Evaluation Methodology

To evaluate our system, we made a clear distinction between the validation dataset that is used in the training phase to measure the performance and the test dataset that is used in the final evaluation. This distinction is important because often the text used in training phase comes from the same sources as in the training dataset, so the data would be biased towards that particular domain. Choosing the testing dataset from different sources than the validation texts is particularly important when using POS tags as an aiding factor to the diacritization process. This is because the training and testing datasets come from a pre-tagged corpus, which is not realistic scenario where real-world texts do not come pre-tagged.

Having this distinction in mind, we selected a number of texts from various sources and carefully diacritized them. Although these texts essentially represent MSA, they include various types of texts, such as poetry, literature, and religion. Finally, for evaluation, we gave the system the validation texts (undiacritized), and measured the evaluation metrics for the produced texts. Table 21 shows some statistics about the validation and test datasets. For sample sentences from the test set, please see Appendix I.

Table 21: Statistics about the Validation & Test sets

	Validation Set	**Test Set**
Arabic Letter Count	71933	77732
Arabic Word Count	15072	16242
Unique Arabic Word Count	7244	7640
Sentence Count	482	495
Diacritization Level	99.265%	99.260%

5.3 Results & Discussion

To evaluate the proposed system (denoted DT henceforth[1]) we compared it with four other diacritizers, namely Arabi NLP [31], Mishkal, [32], AraDiac [2], and Sakhr [33]. Those companies have gracefully agreed to help us evaluate their diacritization systems, which represent the state-of-the-art in the field.

The test was performed offline for all systems and hence speed and memory metrics could not be collected. In Figure 15, we show the DER (with case-endings) for each system. Our system, DT, performs better (at 3.511%) than Arabi (at 6.557%) and Mishkal (at 11.663%) but worse than Sakhr's (at 2.905%). The results also show the WER (with case-endings) which is 13.792%, 25.768%, 39.985%, and 10.941%, respectively. Finally, the diacritization level (DL) is computed for each. While DT achieves the third highest DL (81.672%), it is still less than the ground truth (99.260%). The other three achieved 72.455% and 39.985%, respectively.

[1] Diacritization Tool

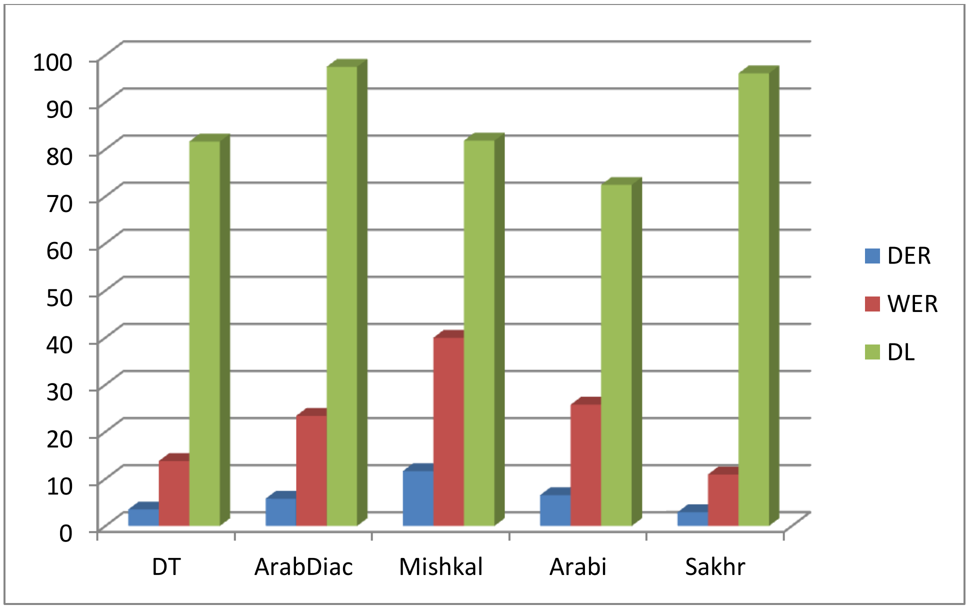

Figure 15: Comparison of diacritizers in terms of DER, WER, and DL.

During our experiments, we noticed that the performance of the diacritizer relies heavily on the order of the diacritization method used. To assess the performance gain or lack thereof of each method we calculated the same metrics after each finishes, as displayed by Table 22.

Table 22 shows how each method used affects the overall performance of the diacritization process. We can see from the table that rules play an important role in both increasing the diacritization level and reducing the error rate (whether DER or WER). Also, the word corrector, which is a unigram-based module to assess which diacritical form is closest to a given word based on the current diacritization, improves the accuracy by reducing DER and WER by almost 1%.

Table 22: Accumulative performance of each method (POS first)

Order	Method	DL	DER	WER
1	Strict POS-grams	6.29%	0.355%	1.404%
2	Strict Group A-B rules	34.054%	0.481%	1.989%
3	Strict Group C rules	34.403%	0.482%	1.995%
4	Relaxed Group A-B rules	53.491%	0.803%	3.362%
5	Relaxed Group C rules	54.262%	0.828%	3.442%
6	Word-grams	65.784%	1.863%	7.703%
7	Relaxed Group A-B rules	80.649%	3.627%	14.605%
8	Word Corrector	81.691%	3.471%	13.663%

Similarly, Table 23 and Table 24 show the same results but with different ordering. Table 23 shows the results when rules are used first while Table 24 shows the results when word-grams are used first.

Table 23: Accumulative performance of each method (Rules first)

Order	Method	DL	DER	WER
1	Strict Group A-B rules	34.769%	0.241%	1.041%
2	Strict Group C rules	37.763%	0.361%	1.601%
3	Strict POS-grams	40.345%	0.657%	2.802%
4	Relaxed Group A-B rules	45.227%	0.836%	3.516%
5	Relaxed Group C rules	46.734%	0.928%	3.867%
6	Word-grams	62.758%	1.99%	8.288%
7	Relaxed Group A-B rules	65.324%	2.163%	8.934%
8	Letter-grams	97.458%	11.375%	38.323%
9	Word corrector	98.825%	9.124%	33.662%

Table 24: Accumulative performance of each method (Word-grams first)

Order	Method	DL	DER	WER
1	Word-grams	31.974%	1.286%	5.338%
2	Strict Group A-B rules	54.997%	1.476%	6.188%
3	Strict Group C rules	57.266%	1.575%	6.65%
4	Relaxed Group A-B rules	61.073%	1.739%	7.278%
5	Relaxed Group C rules	62.092%	1.816%	7.586%
6	Strict POS-grams	63.005%	1.936%	8.041%
7	Relaxed Group A-B rules	65.584%	2.107%	8.682%
8	Letter-grams	97.448%	11.302%	38.138%
9	Word Corrector	98.806%	9.071%	33.52%

5.4 Summary

In this chapter, we discussed the followed approach to evaluate our system. This approach consists of dividing the corpus into 3 datasets: training, validation, and testing. The first two are used during the development of the diacritizer while the later dataset is used for evaluation. We compared our diacritizer (denoted as DT) with 4 other existing systems, namely Arabi NLP [31], Mishkal, [32], AraDiac [2], and Sakhr [33]. The performance of our system shows clear edge over 3 of the 4 diacritizer when all major metrics are considered (WER, DER, and DL). Nonetheless, we show that our approach can be easily enhanced to achieve even higher accuracy and performance.

CHAPTER 6

CONCLUSION & FUTURE WORK

6.1 Conclusion

Researchers have long explored the problem of automatic diacritics restoration. The focus was primarily on statistical methods, which have proven successful to a reasonable extent. However, this accuracy, as shown by this thesis, can be further improved using mined rules that were extracted from and validated against a diacritized corpus. This approach has not been explored in the literature to the best of our knowledge.

In this thesis, we made two main contributions. The first was to build and develop a relatively large fully-diacritized corpus. The corpus was collected from various sources that constitute the major domains of MSA, which guarantees some level of balance. This corpus can be used as a benchmark by future researchers as we plan to make it publicly available.

The second contribution of the thesis was developing a diacritization system that uses a hybrid of the statistical approach and a newly introduced rule-based approach. The statistical approach used basic N-grams on three levels: letter-level, word-level, and POS tag-level. On the other hand, rules were mined and extracted from the developed corpus. These rules use lexical features such as the previous letter, and contextual features, such as the previous word.

The evaluation method of the system involved creating a separate test dataset (about 15,000 words) that were not used during the development of the system. Besides the conventional metrics used in the literature (i.e. the error rates WER and DER), we also introduced a new metric we call the diacritization level (DL) that measures how complete the diacritization of a sentence is. The system achieved a DER of 3.511% and a WER of 13.792% with a diacritization level of 81.672%.

6.2 Future work

In this research work, we showed how using the hybrid model can improve the results of automatic diacritization. However, there is still a large room of improvement to enhance the results even further. For example, the morpho-syntactic analyzer can be improved in both quality of results and breadth of coverage. This can affect the results positively in a great way since it will also affect the POS tagger which is an essential component of the system.

Another future work would be the further development of the corpus. Although the corpus we created was large and diverse, it still needed more review. Also, the MSA texts in the corpus need to be increased since the majority of the corpus is collected from classical Arabic texts.

The diacritized corpus can be developed in two ways. One way is by performing more validation to the texts to ensure a higher accuracy. Another way is by adding texts collected from the General Corpus after diacritizing them manually (or semi-automatically). A good way to doing this would be to extract the most common trigrams and diacritize them. Due to the gigantic size of the General Corpus, we're not able to extract these trigrams because of memory limitations. However, this problem can be resolved by collecting partial trigrams and combining them later on.

Appendix I

Sample sentences from the test set

1	وَكَانَتِ الْمُدَّةُ الَّتِي قَضَاهَا فِي الْكَتَاتِيبِ وَجِيزَةً لَمْ يُتِمَّ حِفْظَ الْقُرْآنِ خِلَالَهَا؛ إِذْ كَانَ دَائِمَ التَّبَرُّمِ مِنْ نِظَامِ (الْكُتَّابِ)، وَلَمْ يُطِقْ أَنْ يَسْتَمِرَّ فِيهِ، فَالْتَحَقَ بِالْمَدْرَسَةِ الْإِعْدَادِيَّةِ رُغْمَ مُعَارَضَةِ وَالِدِهِ الَّذِي كَانَ يَحْرِصُ عَلَى أَنْ يُحَفِّظَهُ الْقُرْآنَ، وَلَمْ يُوَافِقْ عَلَى الْتِحَاقِهِ بِالْمَدْرَسَةِ إِلَّا بَعْدَ أَنْ تَعَهَّدَ لَهُ (حَسَنٌ) بِأَنْ يُتِمَّ حِفْظَ الْقُرْآنِ فِي مَنْزِلِهِ.
2	اسْتُخْدِمَ فِي هَذِهِ الدِّرَاسَةِ الْمَنْهَجُ الْوَصْفِيُّ الْوَثَائِقِيُّ وَالْمَنْهَجُ الْوَصْفِيُّ التَّحْلِيلِيُّ الْقَائِمُ عَلَى وَصْفِ الْوَاقِعِ وَمُعْطَيَاتِهِ مِنْ خِلَالِ الْمَصَادِرِ الْأَوَّلِيَّةِ لِلْبَحْثِ، وَمُرَاجَعَةِ الدِّرَاسَاتِ وَالْبُحُوثِ وَالْمَصَادِرِ الْأُخْرَى الَّتِي لَهَا صِلَةٌ بِمَوْضُوعِ الْبَحْثِ.
3	آفَةُ الْمُلُوكِ سُوءُ السِّيرَةِ وَآفَةُ الْوُزَرَاءِ خُبْثُ السَّرِيرَةِ وَآفَةُ الْجُنْدِ مُخَالَفَةُ الْقَادَةِ وَآفَةُ الرَّعِيَّةِ مُخَالَفَةُ السَّادَةِ وَآفَةُ الرُّؤَسَاءِ ضَعْفُ السِّيَاسَةِ وَآفَةُ الْعُلَمَاءِ حُبُّ الرِّيَاسَةِ وَآفَةُ الْقُضَاةِ شِدَّةُ الطَّمَعِ وَآفَةُ الْعُدُولِ قِلَّةُ الْوَرَعِ وَآفَةُ الْقَوِيِّ اسْتِضْعَافُ الْخَصْمِ وَآفَةُ الْجَرِيءِ إِضَاعَةُ الْحَزْمِ وَآفَةُ الْمُنْعِمِ قُبْحُ الْمَنِّ وَآفَةُ الْمُذْنِبِ حُسْنُ الظَّنِّ وَالْخِلَافَةُ لَا يُصْلِحُهَا إِلَّا التَّقْوَى وَالرَّعِيَّةُ لَا يُصْلِحُهَا إِلَّا الْعَدْلُ، فَمَنْ جَارَتْ قَضِيَّتُهُ ضَاعَتْ رَعِيَّتُهُ وَمَنْ ضَعُفَتْ سِيَاسَتُهُ بَطَلَتْ رِيَاسَتُهُ.
4	وَيُحَدَّدُ فِي ضَوْءِ ذَلِكَ الْمِيزَانِيَّةُ التَّقْدِيرِيَّةُ الَّتِي يَجِبُ أَنْ تُقَارَنَ بِالْمَصْرُوفَاتِ الْفِعْلِيَّةِ.
5	فَمَثَلًا الْبَيْضُ يُوضَعُ فِي أَقْفَاصٍ أَوْ صَنَادِيقَ خَاصَّةٍ وَكَرْتُونَاتٍ وَالسَّوَائِلُ كَالْخَلِّ أَوِ الْمَشْرُوبَاتِ فِي بَرَامِيلَ أَوْ زُجَاجَاتٍ وَالْفَاكِهَةُ وَالْخَضْرَاوَاتُ تَجْرِي تَعْبِئَتُهَا فِي صَنَادِيقَ ذَاتِ مُوَاصَفَاتٍ خَاصَّةٍ وَالْأَغْذِيَةُ الْمَحْفُوظَةُ فِي الْعُلَبِ الصَّفِيحِ ثُمَّ تَجْرِي تَعْبِئَتُهَا فِي صَنَادِيقَ مِنَ الْكَرْتُونِ الْمُقَوَّى.
6	مِنْ هُنَا كَانَتْ تَسْمِيَةُ هَذِهِ الِاسْتَرَاتِيجِيَّةِ اسْتِنَادًا لِمَفَاهِيمِ عِلْمِ الْإِدَارَةِ بِبَرْنَامَجِ إِدَارَةِ الْأُصُولِ، أَيْ تِلْكَ الْإِدَارَةُ الْفَاعِلَةُ الَّتِي تُعَظِّمُ الِاسْتِفَادَةَ الْقُصْوَى مِنْ هَذِهِ الْأُصُولِ بِمَا يَعُودُ بِالنَّفْعِ عَلَى الْمُجْتَمَعِ الَّذِي يُعَدُّ هُوَ الْمَالِكَ الْحَقِيقِيَّ وَالنِّهَائِيَّ لِهَذِهِ الْأُصُولِ الْعَامَّةِ.
7	الثَّقَافَةُ الْوَهْرَانِيَّةُ صَنَعَتْ لِلْمَدِينَةِ سُمْعَةً إِقْلِيمِيَّةً وَعَرَبِيَّةً وَحَتَّى عَالَمِيَّةً.
8	الْحَضَارَةُ كَمَا أَسْلَفْنَا قَصْرٌ عَلَى الْإِنْسَانِ دُونَ غَيْرِهِ، فَهِيَ نَتَاجُ الْعَقْلِ الْبَشَرِيِّ وَالْإِرْثِ الِاجْتِمَاعِيِّ لِتُرَاثٍ مُتَطَوِّرٍ، وَهِيَ فِي الْوَاقِعِ سِلْسِلَةٌ مُتَّصِلَةُ الْحَلَقَاتِ، وَهَذَا مَا يُؤَكِّدُهُ السِّجِلُّ الْأَثَرِيُّ بِنَاءً عَلَى مَا وُجِدَ مِنْ أَدَوَاتٍ صُوَانِيَّةٍ مِنْ صُنْعِ الْإِنْسَانِ دُفِنَتْ مَعَ مُخَلَّفَاتِهِ، فَتِلْكَ الْأَدَوَاتُ فِي صِيَاغَتِهَا وَتَشْكِيلِهَا تَدُلُّ عَلَى تَطَوُّرٍ صَاعِدٍ نَحْوَ التَّرَقِّي فِي كَافَّةِ جِهَاتِ الْعَالَمِ الْمَأْهُولِ آنَذَاكَ.
9	مِنَ الْعَادَاتِ غَيْرِ الْمُسْتَحْسَنَةِ طِبِّيًّا عَادَةُ التَّقْبِيلِ فِي الْفَمِ، وَهِيَ أَكْثَرُ اسْتِهْجَانًا عِنْدَمَا نَسْتَعْمِلُهَا مَعَ أَطْفَالِنَا؛ لِأَنَّهُمْ فِي سِنٍّ تَصْعُبُ فِيهِ الْمُقَاوَمَةُ لِلْأَمْرَاضِ وَيُصْبِحُ مِنَ السَّهْلِ أَنْ نَنْقُلَ إِلَيْهِمُ الْأَمْرَاضَ عَنْ طَرِيقِ هَذِهِ الْعَادَةِ السَّخِيفَةِ.
10	بِحُلُولِ رَبِيعِ عَامِ 1915، كَانَ الْجَيْشُ فِي تَرَاجُعٍ مُسْتَمِرٍّ، وَالَّذِي لَمْ يَكُنْ دَائِمًا بِشَكْلٍ مُنَظَّمٍ ؛ بَلْ كَانَ مِنَ الْمَأْلُوفِ انْتِشَارُ النَّهْبِ وَالْفَوْضَى أَثْنَاءَ الِانْسِحَابِ.

Appendix II

Top 1000 words in the diacritized corpus

Serial	Word	Frequency
1	فِي	544469
2	مِنْ	500427
3	عَلَى	408261
4	قَالَ	355779
5	أَوْ	324747
6	عَنْ	322820
7	لَا	258999
8	عَلَيْهِ	252319
9	أَنْ	222651
10	بْنِ	210091
11	مَا	208894
12	لَهُ	194883
13	كَانَ	194810
14	لَمْ	191050
15	ذَلِكَ	174553
16	بْنُ	173596
17	أَيْ	150013
18	أَنَّ	141848
19	بِهِ	137844
20	قَوْلُهُ	130618
21	أَنَّهُ	128419
22	اللَّهُ	127698
23	ثُمَّ	126754
24	وَلَا	125047
25	وَإِنْ	120923
26	مَنْ	111560
27	اللَّهِ	111463

Serial	Word	Frequency
28	وَهُوَ	110878
29	فِيهِ	109582
30	إلَى	102243
31	أَبِي	93919
32	صَلَّى	92536
33	هَذَا	92188
34	إِلَى	90363
35	وَسَلَّمَ	84544
36	لِأَنَّ	83527
37	فَإِنْ	83017
38	لِأَنَّهُ	82024
39	وَلَوْ	77006
40	إنْ	76051
41	إذَا	75965
42	مِنَ	75350
43	مِنْهُ	72692
44	لَوْ	72176
45	هُوَ	71490
46	وَقَالَ	69263
47	كَمَا	68321
48	فَقَالَ	68282
49	حَدَّثَنَا	66551
50	حَتَّى	64002
51	بَعْدَ	63675
52	فَلَا	63077
53	ابْنُ	62650
54	عَنْهُ	61855

Serial	Word	Frequency
55	أَبُو	59160
56	مَعَ	56391
57	وَقَدْ	55052
58	وَلَمْ	52217
59	عَنِ	51877
60	عَبْدِ	51209
61	إلَّا	50504
62	قَدْ	48681
63	ابْنِ	46464
64	قَبْلَ	45126
65	تَعَالَى	42901
66	عِنْدَ	42875
67	بَيْنَ	40607
68	الَّذِي	40388
69	وَكَانَ	37047
70	هَذِهِ	36385
71	لَيْسَ	36378
72	وَفِي	35399
73	عُمَرَ	35150
74	إِذَا	34323
75	غَيْرِ	34307
76	فَهُوَ	33428
77	فِيهَا	33080
78	بِهَا	33063
79	يَكُونَ	33016
80	بَلْ	32579
81	كَانَتْ	32529

Serial	Word	Frequency
82	يَكُنْ	31187
83	إلَخْ	31050
84	فَإِنَّهُ	30403
85	إِلَّا	29744
86	بْنَ	29738
87	وَمَا	28206
88	كُلِّ	27649
89	وَهَذَا	26858
90	يَجُوزُ	26474
91	أَيْضًا	26412
92	مِنْهُمْ	26365
93	يَقُولُ	26349
94	وَمَنْ	26254
95	قَوْلِهِ	26093
96	بِمَا	26042
97	وَأَمَّا	25831
98	عَبْدُ	25826
99	أَهْلِ	25721
100	لَهُمْ	25660
101	إلَيْهِ	25127
102	فِيمَا	24930
103	يَكُونُ	24900
104	وَقِيلَ	24860
105	فَإِذَا	24153
106	وَاحِدٍ	23873
107	فَإِنَّ	23546
108	إلَيْهِ	23291
109	غَيْرُ	22641
110	مِنْهَا	22523
111	رَوَاهُ	22483
112	مُحَمَّدُ	22338

Serial	Word	Frequency
113	فَلَمْ	22200
114	وَهِيَ	22187
115	بِذَلِكَ	22152
116	إنْ	22130
117	غَيْرِهِ	22038
118	أَخْبَرَنَا	22035
119	مَاتَ	21842
120	إنَّ	21607
121	بِأَنْ	21356
122	وَإِلَّا	21069
123	وَ	19994
124	مَالِكٍ	19780
125	يَحْيَى	19778
126	مِمَّا	19775
127	لَهَا	19687
128	وَإِذَا	19356
129	عَلَيْهِمْ	19228
130	رَسُولُ	19205
131	أَحْمَدُ	19142
132	وَقَوْلُهُ	19059
133	بِخِلَافِ	18883
134	أَبِيهِ	18877
135	يَا	18684
136	وَلَيْسَ	18637
137	مَعَهُ	18589
138	لِمَا	18538
139	فَلَمَّا	18056
140	نَفْسِهِ	18038
141	شَيْئًا	17954
142	وَكَذَا	17840
143	قِيلَ	17771

Serial	Word	Frequency
144	غَيْرَ	17725
145	حَيْثُ	17595
146	رَضِيَ	17553
147	عَلَيْهَا	17536
148	الْمَالِ	17534
149	فَقَدْ	17370
150	مُحَمَّدٍ	17333
151	دُونَ	17333
152	يَوْمَ	17299
153	شَيْءٍ	17246
154	وَإِنَّمَا	17053
155	قُلْتُ	17048
156	بِغَيْرِ	16972
157	عَبَّاسٍ	16963
158	كَذَلِكَ	16940
159	مَالِكٌ	16727
160	وَكَذَلِكَ	16636
161	وَمِنْ	16632
162	الَّتِي	15961
163	شَيْءٌ	15895
164	قَوْلُ	15875
165	مُحَمَّدِ	15871
166	وَعَنْ	15705
167	كُلُّ	15652
168	اه	15628
169	مُوسَى	15576
170	رَجُلٌ	15468
171	كَذَا	15432
172	وَعَلَى	15309
173	بَكْرٍ	15010
174	وَذَلِكَ	14936

Serial	Word	Frequency
175	رَسُولَ	14674
176	هِيَ	14466
177	تَقَدَّمَ	14330
178	رَجُلٍ	14223
179	الرَّحْمَنِ	14207
180	مِنْهُمَا	14172
181	يُقَالُ	14033
182	شَاءَ	13982
183	حَدِيثِ	13923
184	أَمْ	13901
185	فَلَوْ	13601
186	عِنْدَهُ	13575
187	هَلْ	13431
188	نَعَمْ	13416
189	الَّذِينَ	13383
190	الْمُسْلِمِينَ	13265
191	النَّبِيِّ	13235
192	سَمِعْتُ	13228
193	أَعْلَمُ	13202
194	أَحَدُهُمَا	13191
195	عَلِيٍّ	13104
196	أَحْمَدَ	13102
197	بِأَنَّ	13027
198	يَجِبُ	13017
199	سَوَاءٌ	12984
200	إذْ	12960
201	إبْرَاهِيمَ	12950
202	قَالَهُ	12858
203	إنَّمَا	12851
204	وَابْنُ	12817
205	كُلَّ	12706

Serial	Word	Frequency
206	جَازَ	12703
207	بِقَوْلِهِ	12593
208	بِلَا	12531
209	لِمَنْ	12508
210	سَعِيدٍ	12484
211	قَوْلِ	12475
212	يَصِحُّ	12312
213	آخَرَ	12309
214	النَّاسِ	12019
215	هُنَا	11995
216	أَرَادَ	11976
217	حَنِيفَةَ	11966
218	وَفِيهِ	11846
219	أَمَّا	11824
220	لَمَّا	11760
221	اللَّهَ	11623
222	يَعْنِي	11611
223	خَرَجَ	11611
224	بَيْنَهُمَا	11518
225	بَنِي	11506
226	عُمَرُ	11391
227	أَبَا	11354
228	أَنَّهَا	11354
229	اللهُ	11342
230	هُرَيْرَةَ	11234
231	انْتَهَى	11231
232	أَوْلَى	11091
233	الْقَاضِي	11028
234	قَالُوا	11024
235	إسْحَاقَ	10988
236	فَكَانَ	10793

Serial	Word	Frequency
237	وَلَهُ	10789
238	رَسُولِ	10777
239	كَانُوا	10754
240	عَشَرَ	10700
241	مَرَّ	10652
242	الْأَوَّلِ	10611
243	وَبَيْنَ	10534
244	إنَّ	10533
245	وَأَبُو	10519
246	جَاءَ	10493
247	لَكِنْ	10414
248	لِي	10399
249	الْإِسْلَامِ	10391
250	الْحَدِيثِ	10355
251	بَعْدَهُ	10351
252	الْإِمَامُ	10312
253	الْقَاسِمِ	10297
254	مَالِهِ	10269
255	الْإِمَامِ	10258
256	رَوَى	10250
257	بَعْضُهُمْ	10243
258	ذَكَرَ	10232
259	حِينَ	10200
260	مِمَّنْ	10120
261	الْعَبْدِ	10104
262	ذَكَرَهُ	10103
263	بِأَنَّهُ	10067
264	لِأَنَّهَا	10014
265	أَحَدٌ	9962
266	مَكَّةَ	9775
267	مُسْلِمٍ	9749

Serial	Word	Frequency
268	بَعْضُ	9688
269	الثَّانِي	9687
270	الْأَرْضِ	9622
271	حَدَّثَنِي	9424
272	رَجُلًا	9411
273	دَخَلَ	9391
274	عَلَيَّ	9358
275	زَيْدٍ	9335
276	رَحِمَهُ	9290
277	تِلْكَ	9281
278	فَلَهُ	9252
279	بَعْضِ	9244
280	الْحَسَنِ	9092
281	وَعَلَيْهِ	9071
282	فَقَطْ	9051
283	سَلَمَةَ	9019
284	عُثْمَانَ	8899
285	قَالَتْ	8894
286	وَاحِدٌ	8874
287	أُخْرَى	8868
288	النَّبِيَّ	8697
289	مَعْنَى	8684
290	عَائِشَةَ	8665
291	وَقَعَ	8641
292	حَلَفَ	8613
293	فَمَا	8605
294	مِثْلَ	8603
295	لِلَّهِ	8541
296	أَكْثَرَ	8497
297	فَيَكُونُ	8496
298	يُوسُفَ	8481

Serial	Word	Frequency
299	إِنَّهُ	8480
300	أَخَذَ	8351
301	مِثْلُ	8348
302	عَنْهَا	8346
303	ص	8325
304	مُسْلِمٌ	8285
305	وَجَبَ	8231
306	غَيْرُهُ	8226
307	فَلَيْسَ	8218
308	الْعِلْمِ	8213
309	بَقِيَ	8200
310	يَزِيدَ	8164
311	أَنَا	8163
312	فَهَلْ	8125
313	أَقَرَّ	8052
314	الْأُولَى	8023
315	طَرِيقِ	7980
316	وَهُمْ	7957
317	أَنَّهُمْ	7917
318	صَحَّ	7899
319	أَهْلُ	7886
320	وَلَكِنْ	7872
321	قَبْلَهُ	7864
322	أَسْلَمَ	7819
323	مُطْلَقًا	7796
324	ابْنَ	7777
325	بِهِمْ	7735
326	إِنَّمَا	7704
327	لَكَ	7696
328	سُفْيَانَ	7694
329	وَرَوَاهُ	7672

Serial	Word	Frequency
330	رَجَعَ	7606
331	الرَّزَّاقِ	7592
332	سَنَةَ	7561
333	يَدُلُّ	7516
334	عَمْرٍو	7418
335	فَمَنْ	7411
336	يَلْزَمُهُ	7394
337	يَدِهِ	7362
338	إِذْ	7361
339	ثَبَتَ	7360
340	الْآخَرِ	7339
341	سُلَيْمَانَ	7292
342	وَغَيْرِهِ	7287
343	دَاوُدَ	7177
344	بِمَعْنَى	7153
345	الْأَوَّلُ	7153
346	الصَّلَاةِ	7131
347	الْمُؤْمِنِينَ	7092
348	دِرْهَمٍ	7087
349	الشَّافِعِيُّ	7079
350	وَأَنَّهُ	7044
351	صَارَ	7040
352	سَنَةً	7019
353	لِكُلِّ	6995
354	عَلِمَ	6978
355	طَالِقٌ	6968
356	صَحِيحٌ	6968
357	وَأَنَّ	6944
358	رَمَضَانَ	6937
359	بِشَيْءٍ	6925
360	هَؤُلَاءِ	6910

Serial	Word	Frequency
361	وَبِهِ	6903
362	فَهِيَ	6892
363	صَاحِبُ	6878
364	بُدَّ	6855
365	النَّبِيُّ	6831
366	يُرِيدُ	6806
367	إنَّهُ	6756
368	وَأَنْ	6735
369	فَصْلٌ	6728
370	السَّلَامُ	6675
371	وَاحِدَةٍ	6654
372	النَّاسُ	6652
373	سَمِعَ	6644
374	رُوِيَ	6625
375	ذُكِرَ	6619
376	الدِّينِ	6608
377	وَابْنِ	6599
378	وَرَوَى	6595
379	كَلَامِ	6590
380	عِيسَى	6585
381	وَغَيْرُهُ	6578
382	عَنْهُمْ	6571
383	الْمَلِكِ	6539
384	وَاحِدَةً	6523
385	سَعِيدِ	6482
386	الْمَسْجِدِ	6476
387	رِوَايَةِ	6461
388	لَزِمَهُ	6437
389	مَسْعُودٍ	6434
390	الْمَاءِ	6412
391	قُلْنَا	6389

Serial	Word	Frequency
392	وَذَكَرَ	6381
393	بِنَفْسِهِ	6378
394	عَدَمِ	6372
395	بَابِ	6372
396	لَكُمْ	6345
397	أَنَسٍ	6325
398	ش	6316
399	مَالِ	6314
400	وَاَللَّهُ	6305
401	الْخَطَّابِ	6300
402	لِعَدَمِ	6281
403	دِينَارٍ	6280
404	حِينَئِذٍ	6264
405	يَدَيْهِ	6263
406	قَوْلَهُ	6235
407	الْعَزِيزِ	6229
408	حَدِيثُ	6225
409	كَيْفَ	6225
410	وَلِأَنَّ	6217
411	الْمَدِينَةِ	6217
412	أَوْصَى	6208
413	يَوْمٍ	6190
414	يَجُزْ	6178
415	لِذَلِكَ	6159
416	فَعَلَ	6133
417	الْعَبْدُ	6130
418	لِنَفْسِهِ	6098
419	وَاللَّهُ	6074
420	تَكُونَ	6060
421	يُمْكِنُ	6039
422	رَأَى	6032

Serial	Word	Frequency
423	يَقُولَ	6031
424	هُمْ	6026
425	عَمْرٍو	5975
426	سَعْدٍ	5964
427	أَنْتَ	5937
428	جَمِيعِ	5916
429	بَيْنَهُمْ	5868
430	فَعَلَيْهِ	5856
431	زَادَ	5847
432	جَعْفَرٍ	5840
433	الْكِتَابِ	5828
434	غَيْرَهُ	5826
435	مُعَاوِيَةَ	5817
436	نَفْسَهُ	5814
437	سُئِلَ	5800
438	بِهَذَا	5791
439	فَأَمَّا	5780
440	عَلِيٍّ	5768
441	حَقٌّ	5725
442	فُلَانٍ	5721
443	الرَّجُلُ	5714
444	الْمُشْتَرِي	5710
445	يَلْزَمُ	5703
446	فَعَلَى	5695
447	يَوْمًا	5678
448	إِسْمَاعِيلَ	5674
449	يَرْجِعُ	5654
450	مَثَلًا	5648
451	الْمَعْنَى	5630
452	وَالثَّانِي	5630
453	أَيَّامٍ	5617

Serial	Word	Frequency
454	أَتَى	5613
455	الْمَوْلَى	5589
456	قَتَادَةَ	5556
457	بِمَنْزِلَةِ	5545
458	فَهَذَا	5517
459	بِيَدِهِ	5513
460	أَكْثَرُ	5499
461	شَيْءَ	5495
462	آخَرُ	5483
463	فِيهِمْ	5478
464	لِقَوْلِهِ	5461
465	يَأْتِي	5443
466	وَيَجُوزُ	5441
467	عَرَفَةَ	5435
468	شَهِدَ	5409
469	ثَلَاثًا	5399
470	حَدِيثٌ	5395
471	الْقُرْآنِ	5376
472	عَتَقَ	5371
473	بِفَتْحِ	5366
474	الْحَجِّ	5352
475	الْبَيْعِ	5351
476	فَقُلْتُ	5319
477	أَوِ	5318
478	ادَّعَى	5317
479	مَعْنَاهُ	5315
480	قُتِلَ	5313
481	فِيمَنْ	5312
482	يَوْمِ	5307
483	رَأَيْتُ	5292
484	إِنِّي	5284

Serial	Word	Frequency
485	الْمَيِّتِ	5269
486	الْحَدِيثَ	5264
487	حُرٌّ	5263
488	الْمُصَنِّفِ	5251
489	بَاعَ	5241
490	قُلْت	5229
491	تَكُونُ	5225
492	وَلِأَنَّهُ	5193
493	الْبُخَارِيُّ	5190
494	وَأَبِي	5174
495	الصَّلَاةُ	5169
496	بَيْنَهُ	5141
497	أَشْهُرٍ	5132
498	تَحْتَ	5105
499	الْحُكْمِ	5105
500	مَالٍ	5098
501	مِلْكِهِ	5096
502	ثَنَا	5080
503	أَخْرَجَهُ	5066
504	مُحَمَّدٌ	5062
505	عَلِيٌّ	5047
506	أَشَارَ	5043
507	جَمِيعًا	5037
508	الْحَقِّ	5034
509	بَأْسَ	5028
510	ذَهَبَ	5022
511	وَنَحْوِهِ	5018
512	كِتَابِ	4998
513	كَأَنَّ	4978
514	وَكَانَتْ	4977
515	أَحَدٍ	4968

Serial	Word	Frequency
516	إِبْرَاهِيمُ	4961
517	بِنْتُ	4938
518	مَرَّةً	4937
519	يَنْبَغِي	4924
520	قَامَ	4922
521	لَنَا	4912
522	الْحَسَنُ	4903
523	يُونُسَ	4903
524	الْقِيَامَةِ	4898
525	شَرْحِ	4891
526	نَحْوِ	4879
527	أُمِّ	4878
528	فَفِي	4870
529	تَكُنْ	4864
530	الْحَارِثِ	4858
531	وَكُلُّ	4836
532	اشْتَرَى	4815
533	وَعِشْرِينَ	4808
534	وَرُوِيَ	4804
535	كُلُّهُ	4803
536	بَعْضَ	4796
537	عَلِيٌّ	4793
538	عَدَمُ	4789
539	ظَاهِرٌ	4788
540	إِلَيْهِمْ	4787
541	عَشْرَةَ	4765
542	الدَّيْنِ	4749
543	لَهُمَا	4749
544	وَيُقَالُ	4748
545	وَهَلْ	4743
546	لِغَيْرِهِ	4739

Serial	Word	Frequency
547	أَيُّوبَ	4730
548	مَالٌ	4720
549	نَافِعٍ	4720
550	سُفْيَانُ	4719
551	يَزِيدُ	4716
552	جَعَلَ	4713
553	عَمَّا	4705
554	رَبِّ	4700
555	الْأَصْلِ	4698
556	عَبْدَ	4693
557	قَتَلَ	4681
558	تَرَكَ	4681
559	أَحَدِهِمَا	4664
560	حُكْمُ	4663
561	يَقَعُ	4650
562	الْعَقْدِ	4647
563	وَحْدَهُ	4638
564	عَبْدًا	4638
565	كَثِيرٍ	4637
566	الْمَوْتِ	4632
567	عَلَيْكُمْ	4627
568	عَزَّ	4620
569	بَعْضٍ	4619
570	أَصْحَابِ	4618
571	وَعَنْهُ	4614
572	وَجْهٍ	4613
573	الْحَالِ	4611
574	أَلَا	4597
575	أَمَرَ	4592
576	نَوَى	4588
577	بِحَيْثُ	4566

Serial	Word	Frequency
578	الْمُدَوَّنَةِ	4563
579	بَيْتِ	4561
580	الْحَرْبِ	4534
581	دَاوُد	4522
582	أَهْلَ	4520
583	ثَابِتٍ	4510
584	وَيَكُونُ	4505
585	الْحَاكِمُ	4501
586	صَالِحٍ	4499
587	ضَعِيفٌ	4499
588	الصَّحِيحِ	4496
589	وَجَلَّ	4492
590	خِلَافًا	4490
591	حَقِّهِ	4486
592	لِأَنَّهُمْ	4462
593	يُقَالَ	4455
594	الْمُرَادُ	4455
595	فِيهِمَا	4453
596	الْجُمُعَةِ	4441
597	جَائِزٌ	4428
598	سُبْحَانَهُ	4423
599	الرَّجُلِ	4414
600	خَالِدٍ	4413
601	الْوَلَدِ	4411
602	فَقِيلَ	4409
603	أَحَدُ	4408
604	عُرْوَةَ	4389
605	جُرَيْجٍ	4380
606	سِنِينَ	4380
607	فَرْعٌ	4373
608	عَلَيْكَ	4366

Serial	Word	Frequency
609	حَالَ	4364
610	بَكْرٍ	4361
611	زَيْدٍ	4355
612	صَاحِبِهِ	4339
613	الْمُصَنِّفُ	4334
614	جِهَةِ	4332
615	ظَهَرَ	4318
616	نَظَرٌ	4318
617	أَنْتِ	4317
618	سَبَقَ	4316
619	الْعَيْنِ	4289
620	مَوْتِهِ	4288
621	لِغَيْرِ	4287
622	نَصَّ	4257
623	وَاحِدَةٌ	4246
624	وَلِهَذَا	4246
625	نَحْوَ	4241
626	كُنْتُ	4219
627	جَابِرٍ	4217
628	الْأَمْرِ	4214
629	يَصِحَّ	4207
630	الْمَسْأَلَةِ	4203
631	الْمَذْهَبِ	4203
632	طَالِبٍ	4200
633	أَفْضَلُ	4196
634	اللهِ	4195
635	بَلَغَ	4172
636	الْبَابِ	4162
637	الْآيَةِ	4150
638	جَمَاعَةٌ	4144
639	وَاحِدًا	4135

Serial	Word	Frequency
640	يَعْلَمُ	4127
641	كَوْنِهِ	4123
642	دَيْنٌ	4113
643	ثَلَاثَةَ	4113
644	الْكَلَامِ	4105
645	الْوَقْتِ	4093
646	قَدِمَ	4088
647	وَمِنْهُ	4086
648	وَفِيهَا	4085
649	يَدِ	4064
650	الْقَوْلِ	4063
651	وَاللَّهِ	4061
652	حَصَلَ	4057
653	يَخْرُجُ	4053
654	النَّاسَ	4050
655	أَعْتَقَ	4039
656	وَهَذِهِ	4038
657	كُلِّ	4032
658	الْوَقْفِ	4022
659	يَحِلُّ	4015
660	م	4015
661	فَكَيْفَ	4005
662	ذِي	4005
663	الْعَرَبِ	3999
664	عَادَ	3998
665	أَحَدِ	3996
666	فَيَقُولُ	3991
667	مَوْلَى	3985
668	الْعُلَمَاءِ	3969
669	تَرَى	3959
670	وَإِنَّ	3947

Serial	Word	Frequency
671	يَقْتَضِي	3946
672	وَالْمُرَادُ	3930
673	يَثْبُتُ	3921
674	عَبْدٍ	3921
675	يَقُولُونَ	3917
676	خَيْرٌ	3912
677	مَسْأَلَةٌ	3907
678	هُنَاكَ	3904
679	أَوَّلًا	3898
680	أَحَدًا	3898
681	مِنْكُمْ	3896
682	وُجِدَ	3880
683	ثَلَاثَةٌ	3879
684	الصَّحَابَةِ	3876
685	دَارِ	3860
686	وَهْبٍ	3850
687	كَثِيرًا	3835
688	كَأَنَّهُ	3832
689	الشَّافِعِيِّ	3830
690	بِضَمٍّ	3826
691	شُعْبَةَ	3826
692	مَالًا	3806
693	حَقٌّ	3803
694	ضَمِنَ	3802
695	هَكَذَا	3799
696	كَقَوْلِهِ	3796
697	أَخَذَهُ	3784
698	تُوُفِّيَ	3769
699	مَنْصُورٍ	3763
700	بَعْدُ	3758
701	صَاحِبِ	3757

Serial	Word	Frequency
702	كَثِيرٌ	3753
703	فَقَالَتْ	3747
704	ثَلَاثَةِ	3743
705	يَمْلِكُ	3743
706	بِقَدْرِ	3738
707	وَأَنَا	3730
708	هِشَامٍ	3727
709	الْأَبِ	3725
710	سَيِّدِهِ	3720
711	لَكَانَ	3718
712	وَلَدِهِ	3711
713	دَرَاهِمَ	3711
714	سَنَةٍ	3708
715	حَسَنٌ	3707
716	ظَاهِرُ	3704
717	ثَلَاثَةٌ	3698
718	لَيْسَتْ	3693
719	نِصْفُ	3686
720	يَحْتَاجُ	3681
721	يَرَى	3672
722	مِثْلِهِ	3671
723	يُشْتَرَطُ	3665
724	إِلَيَّ	3662
725	عَطَاءٍ	3661
726	عُبَيْدِ	3658
727	عُثْمَانُ	3657
728	الزُّهْرِيِّ	3656
729	مِثْلَهُ	3655
730	جُبَيْرٍ	3652
731	آدَمَ	3649
732	رِوَايَةٍ	3648

Serial	Word	Frequency
733	فَمَاتَ	3646
734	سَعِيدُ	3634
735	حُكْمِ	3632
736	آخِرِ	3620
737	كَتَبَ	3620
738	تَصِحُّ	3616
739	إِمَّا	3615
740	أَقَلَّ	3615
741	عُيَيْنَةَ	3614
742	يَدْخُلُ	3614
743	الْآخَرُ	3613
744	أَهْلِهِ	3612
745	تَجِبُ	3612
746	وَلَمَّا	3611
747	مُتَّفَقٌ	3595
748	صَرَّحَ	3593
749	حَالِ	3589
750	عَلَيْهِمَا	3583
751	عَامِرٍ	3583
752	فُلَانٌ	3579
753	فَكَذَلِكَ	3576
754	لَا	3575
755	سَقَطَ	3574
756	يَمِينِهِ	3573
757	مَعَهُمْ	3569
758	كَلَامِهِ	3550
759	تَنْبِيهٌ	3549
760	مَاجَهْ	3545
761	وَخَرَجَ	3540
762	وَمِنْهُمْ	3538
763	رَأْسِهِ	3533

Serial	Word	Frequency
764	الْمَالَ	3526
765	وَلَدٍ	3523
766	الْجَنَّةِ	3522
767	الْيَمِينِ	3519
768	ذَكَرْنَا	3518
769	أَصْحَابِهِ	3514
770	يَضْمَنُ	3513
771	بِعَيْنِهِ	3512
772	عِنْدَهُمْ	3497
773	فَصَارَ	3494
774	مِثْلُهُ	3490
775	وَجْهَانِ	3489
776	وَالسَّلَامُ	3486
777	وَمِنْهَا	3481
778	الزُّبَيْرِ	3478
779	يَبْقَى	3473
780	كُلٌّ	3472
781	بِنْتِ	3466
782	السَّنَةِ	3459
783	وَغَيْرِهِمْ	3453
784	وَجَدَ	3435
785	رَكْعَتَيْنِ	3435
786	رَأْسَهُ	3427
787	قَوْمٌ	3426
788	يَظْهَرُ	3425
789	أُمُّ	3425
790	قَضَى	3424
791	مَتَى	3423
792	عِنْدِي	3413
793	عُبَيْدٍ	3413
794	يُصَلِّي	3409

Serial	Word	Frequency
795	بِسَبَبِ	3406
796	الْوَجْهِ	3404
797	بِمَكَّةَ	3403
798	ر	3402
799	الْكِتَابَةِ	3402
800	بَاطِلٌ	3397
801	وَعِنْدَ	3396
802	أَوَّلِ	3394
803	إِلَيْهَا	3394
804	الْمَسْأَلَةُ	3390
805	الْأُخْرَى	3385
806	نَزَلَ	3377
807	الْأَصَحِّ	3376
808	قَطَعَ	3373
809	شَيْخُنَا	3361
810	يَدَهُ	3358
811	الْحَاكِمِ	3353
812	نِصْفَ	3351
813	قِيمَتِهِ	3347
814	حَاتِمٍ	3346
815	دَلِيلٌ	3341
816	قِيمَتُهُ	3340
817	يُكْرَهُ	3336
818	الْحُكْمُ	3334
819	بَابُ	3330
820	أَوَّلَ	3329
821	الثُّلُثِ	3327
822	يَحْصُلُ	3321
823	حَبِيبٍ	3318
824	إِلَيْهَا	3308
825	الْقَضَاءِ	3307

Serial	Word	Frequency
826	لِأَجْلِ	3302
827	أَخِيهِ	3296
828	الْعِتْقِ	3295
829	شُعْبَةُ	3294
830	بِكَسْرِ	3289
831	الْعَمَلِ	3284
832	فَخَرَجَ	3272
833	الْأَمْرُ	3262
834	بِنَاءً	3258
835	لِئَلَّا	3257
836	بِدُونِ	3255
837	الْيَوْمَ	3255
838	وَعَبْدُ	3251
839	صَلَاةِ	3241
840	لَقَدْ	3241
841	بِهِمَا	3238
842	قَتَلَهُ	3233
843	أَخْبَرَنِي	3231
844	أَرْبَعِينَ	3223
845	أَقَامَ	3219
846	الْبَيْتِ	3217
847	شَيْبَةَ	3216
848	يَعْلَمْ	3213
849	التِّرْمِذِيُّ	3208
850	فَقَالُوا	3204
851	فَوْقَ	3203
852	مَوْضِعٍ	3202
853	بِكُلِّ	3200
854	ثَمَّ	3198
855	أَرَأَيْتَ	3192
856	الْحَدِيثُ	3183

Serial	Word	Frequency
857	الْبَائِعِ	3181
858	جَمِيعَ	3181
859	بَيْعُ	3177
860	دَفَعَ	3177
861	مِثْلِ	3174
862	يَمْنَعُ	3173
863	يُوجِبُ	3173
864	حَرَامٌ	3170
865	عِنْدَنَا	3164
866	الْمِلْكِ	3163
867	الْمِثْلِ	3161
868	مَعْمَرٍ	3161
869	الْوَرَثَةِ	3158
870	غَيْرِهَا	3146
871	فَإِنَّهَا	3139
872	كُلِّهِ	3135
873	يَوْمَئِذٍ	3135
874	وَأَخَذَ	3134
875	الْعَبْدَ	3133
876	إِسْمَاعِيلُ	3122
877	الْأُمِّ	3118
878	الدُّنْيَا	3114
879	طَلْحَةَ	3108
880	الْمَتْنِ	3104
881	يُقْبَلُ	3097
882	عَنْهُمَا	3097
883	صِحَّةِ	3096
884	جَوَازِ	3093
885	رَبِيعَةَ	3087
886	يَحْرُمُ	3087
887	أَحْرَمَ	3077

Serial	Word	Frequency
888	أُمِّهِ	3075
889	بَاعَهُ	3072
890	الْمُرَادَ	3071
891	الْمَذْكُورِ	3070
892	مَالَ	3069
893	الْمُسْلِمُونَ	3057
894	فَفِيهِ	3050
895	الْأَوْلَى	3049
896	أَوَّلُ	3046
897	سُلَيْمَانُ	3045
898	هَارُونَ	3038
899	الْمُسْلِمِ	3038
900	أَقُولُ	3037
901	قُلْ	3034
902	مَعًا	3019
903	سِيرِينَ	3017
904	الْوَلِيدِ	3012
905	وَقَفَ	3006
906	الْمَالُ	3002
907	حَنْبَلٍ	3001
908	عُلِمَ	2999
909	الْوَاحِدِ	2995
910	يَقُومُ	2994
911	وَجَعَلَ	2988
912	الْحُسَيْنِ	2985
913	وُجُوبِ	2985
914	شَكَّ	2985
915	أُمَّ	2978
916	الْبَحْرِ	2978
917	مِصْرَ	2975
918	النَّارِ	2974

Serial	Word	Frequency
919	أَذًى	2973
920	الشَّيْخُ	2972
921	الْحَكَمِ	2969
922	مُعَيَّنٍ	2968
923	قَيْسٍ	2967
924	وَقْتَ	2966
925	يُؤْخَذُ	2963
926	وَجْهِ	2959
927	لَكِنَّ	2957
928	جَمْعُ	2955
929	مَضَى	2952
930	لِرَجُلٍ	2950
931	عِمْرَانَ	2948
932	يَأْخُذَ	2941
933	الْمُسَيِّبِ	2935
934	وَالْمَعْنَى	2933
935	وَتَعَالَى	2933
936	قَصَدَ	2932
937	الثَّمَنِ	2929
938	زَيْدٌ	2924
939	نَحْوُ	2922
940	رُشْدٍ	2920
941	قَوْلَ	2919
942	وَقْتِ	2919
943	بَيْعِ	2916
944	الْبَلَدِ	2916
945	الْمَرْأَةِ	2905
946	قَرَأَ	2904

Serial	Word	Frequency
947	شِهَابٍ	2901
948	قُرَيْشٍ	2901
949	ثَلَاثَ	2900
950	اللَّهُمَّ	2900
951	الْمَاءَ	2894
952	الْوَصِيَّةِ	2893
953	الْمُدَّعَى	2893
954	أَصْلًا	2892
955	مِلْكِ	2890
956	إِسْرَائِيلَ	2886
957	زِيَادٍ	2885
958	شَرَطَ	2881
959	وَتَقَدَّمَ	2881
960	اثْنَيْنِ	2879
961	عَجَزَ	2875
962	بِشَرْطِ	2870
963	أَلْفَ	2870
964	وَلِذَلِكَ	2868
965	وَسَوَاءٌ	2867
966	سِوَى	2866
967	يَقُلْ	2863
968	لِكَوْنِهِ	2859
969	سَبِيلِ	2858
970	امْرَأَةً	2854
971	الْمُشْرِكِينَ	2840
972	بِدَلِيلِ	2836
973	الْبَيْعُ	2834
974	بَيْعُهُ	2834

Serial	Word	Frequency
975	الشَّامِ	2833
976	إِلَيْكَ	2832
977	الْحَافِظُ	2829
978	يَأْخُذُ	2828
979	فَذَكَرَ	2826
980	مَرْفُوعًا	2823
981	حِبَّانَ	2809
982	يَتَعَلَّقُ	2808
983	حَالٍ	2803
984	الْمُكَاتَبِ	2784
985	الصَّلَاةَ	2784
986	قَوْمٍ	2782
987	حَمَّادُ	2776
988	وَإِلَى	2774
989	عَمْدًا	2766
990	امْرَأَةٍ	2757
991	الثَّالِثُ	2757
992	وَاجِبٌ	2757
993	وَيُحْتَمَلُ	2756
994	بِإِذْنِ	2756
995	عِشْرِينَ	2751
996	مَلَكَ	2750
997	وَكِيعٌ	2744
998	الْقَوْلُ	2740
999	صَلَاتُهُ	2734
1000	بَعْدَهَا	2734

Appendix III

Primary Functions

- **IsValidDiacritization:**

A function used extensively to validate the diacritization of a given text using basic heuristics. One such heuristic is when two (or more) diacritics are incompatible (e.g. Sukoon with Shadda).

```
public static bool IsValidDiacritization(string word)
     {

         if (Regex.Match(word, "[ًٌٍَُِ]{2,}", RegexOptions.None).Success
             || Regex.Match(word, "\b[ًٌٍَُِّْ]", RegexOptions.None).Success
             || Regex.Match(word, "[ًٌٍَُِ]{3,}", RegexOptions.None).Success
             || Regex.Match(word, "ّ{2,}", RegexOptions.None).Success
         )
         return false;

         Match m = Regex.Match(word, "[ًٌٍَُِ][ء-ي]", RegexOptions.None);
         if (m.Success)
         {
         int endIndex = m.Index + m.Length;

if (!(m.Value == "اً" || m.Value == "ىً" && (endIndex >= word.Length ||
 !IsArabicLetter(word[endIndex]))) && !(m.Value.EndsWith("و") && (endIn
dex >= word.Length || !IsArabicLetter(word[endIndex]))))
             return false;
         }

         return true;
     }
```

- **CleanDiacritics:**

A function used to clean diacritics (such as misplaced Shadda) from a text. This is important when we compare the output of our system with other ones.

```
public static string CleanDiacritics(string text)
    {
        return rx_repeated_diac.Replace(
            text.Replace("اً", "ًا")
            .Replace("ىً", "ًى")
            .Replace("أَْ", "اَ")
            .Replace("َّ", "َّ")
            .Replace("ًّ", "ًّ")
            .Replace("ُّ", "ُّ")
            .Replace("ٌّ", "ٌّ")
            .Replace("ِّ", "ِّ")
            .Replace("ٍّ", "ٍّ")
            .Replace("يِْ", "يُ")
            .Replace("وُْ", "وُ")
            .Replace("\r", "")
            .Replace("_", ""), "$1");
       }
```

- **GetDiacritizedLetters**

Returns the estimated number of diacritized letters (accounting for implicit diacritics). This function is primary used to calculate the diacritization level (which is simply the number of diacritized letters, returned by this function, over the total number of Arabic letters.

```
public static int GetDiacritizedLetters(string word)
        {
            int diacritized_letters = 0;
            bool letter_started = false;

            int i = 0;
            Character[] Characters = Character.ParseWord(word);
            for (i = 0; i < Characters.Length; i++)
            {
                if (!Characters[i].isDiacritic)
                {
                    if (letter_started)
                    {
                        if (
                                (i - 2 >= 0 && Characters[i -
 1].c == 'ا' && Characters[i - 2].c == 'َ') ||
                                (i - 2 >= 0 && Characters[i -
 1].c == 'و' && Characters[i - 2].c == 'ُ') ||
                                (i - 2 >= 0 && Characters[i -
 1].c == 'ي' && Characters[i - 2].c == 'ِ')
                            )
                            diacritized_letters++;
                    }

                    if (Characters[i].c == 'إ' || Characters[i].c == 'آ' || Cha
racters[i].c == 'ى')
                        diacritized_letters++;

                    letter_started = true;
                }
                else
                {
                    if (letter_started && Characters[i].c != 'ّ' && !(Characte
rs[i - 1].c == 'إ' || Characters[i - 1].c == 'آ' || Characters[i -
 1].c == 'ى'))
                    {
                        diacritized_letters++;
                    }

                    if (Characters[i].c != 'ّ')
                    {
                        letter_started = false;
                    }
                }
            }
```

```
            if (
                    (i - 2 >= 0) && (
                        (Characters[i - 1].c == 'ا' && Characters[i -
 2].c == 'َ') ||
                        (Characters[i - 1].c == 'و' && Characters[i -
 2].c == 'ُ') ||
                        (Characters[i - 1].c == 'ي' && Characters[i -
 2].c == 'ِ') ||
                        (Characters[i - 1].c == 'ا' && Characters[i -
 2].c == 'ً') ||
                        (Characters[i - 1].c == 'ى' && Characters[i -
 2].c == 'ً')
                    )
                )
                diacritized_letters++;

            if ((word.StartsWith("الْ") || word.StartsWith("بِالْ") || word.StartsW
ith("لِل") || word.StartsWith("بِال") || word.StartsWith("الَّ")) && word.Length >= 5
) diacritized_letters++;
            else if (word.StartsWith("الإ")) diacritized_letters += 2;
            else if (word.Length > 3 && word.StartsWith("ال") && !Character.Par
se(word[2]).isDiacritic && word[3] == 'ّ') diacritized_letters += 2;
            else if (word.Length > 5 && word.StartsWith("وَال") && !Character.Pa
rse(word[4]).isDiacritic && word[5] == 'ّ') diacritized_letters += 1;
            else if (word.Length > 5 && word.StartsWith("بِال") && !Character.Pa
rse(word[4]).isDiacritic && word[5] == 'ّ') diacritized_letters += 2;
            else if (word.Length > 5 && word.StartsWith("فَال") && !Character.Pa
rse(word[4]).isDiacritic && word[5] == 'ّ') diacritized_letters += 1;
            else if (word.Length > 5 && word.StartsWith("كَال") && !Character.Pa
rse(word[4]).isDiacritic && word[5] == 'ّ') diacritized_letters += 1;

            if (word.Length > 4 && word.EndsWith("وا")) diacritized_letters +=
word[word.Length - 3] != 'ُ' ? 2 : 1;
            else if (word.Length > 4 && word.EndsWith("وْا")) diacritized_letter
s += 1;


            return diacritized_letters;
        }
```

- **GetSentences**

This function separates the text into a sentence array using Regular Expressions. The performance is questionable but it serves its intended purpose.

```
public static string[] GetSentences(string text)
    {
        List<string> sentences = new List<string>();
        //Regex rx = new Regex(@"(\S.+?[.!?(\r\n)])(?=\s+|$)");

    Regex rx = new Regex(@"[^.!?\s][^.!?]*(?:[.!?](?!['""]?\s|$)[^.!?]*
)*[.!?]?['""]?(?=\s|$)");
        foreach (Match match in rx.Matches(text))
        {
        string sentence = match.Value.Replace("\r\n", " ");
        sentence = sentence.Replace("\n", " ").Trim();
        for (int i = 10; i > 2; i--)
            sentence.Replace(new string(' ', i), " ");
        if (!string.IsNullOrEmpty(sentence))
            sentences.Add(sentence);
        }

        if (sentences.Count == 0) sentences.Add(text);

        return sentences.ToArray();
    }
```

Appendix IV

Tools used

AraMorph

AraMorph is a morphological analyzer which was ported to Java from the Perl version developed by Tim Buckwalter on behalf of the Linguistic Data Consortium (LDC).

Usage: used in various stages of this research work including rule extraction and corpus development and the diacritization.

Website: http://www.nongnu.org/aramorph/

Apache Tika

The Apache Tika™ toolkit detects and extracts metadata and structured text content from various documents using existing parser libraries.

Usage: used to extract text from crawled web documents (HTML and other formats).

Website: http://tika.apache.org/

Alkhalil Morpho Sys

Alkhalil Morpho Sys is a morphological analyzer. For a given word, it identifies all possible solutions with their morphosyntactic features:

Usage: used in various stages of this research work including rule extraction and corpus development.

Website: http://sourceforge.net/projects/alkhalil/

IKVM.NET

IKVM.NET is an implementation of Java for Mono and the Microsoft .NET Framework. It includes the following components:

- A Java Virtual Machine implemented in .NET
- A .NET implementation of the Java class libraries
- Tools that enable Java and .NET interoperability

Usage: converting AraMorph.NET and Akhalil to .NET.

Website: http://www.ikvm.net/

Appendix V

AraMorph Tagset

Prefixes

Tag	Description
CONJ	Conjunction
EMPHATIC_PARTICLE	Emphatic particle
FUNC_WORD	Function word
FUT_PART	Future particle
INTERJ	Interjection
INTERROG_PART	Interrogative particle
IV1S	Imperfective 1st person singular
IV2MS	Imperfective 2nd person masculine singular
IV2FS	Imperfective 2nd person feminine singular
IV3MS	Imperfective 3rd person masculine singular
IV3FS	Imperfective 3rd person feminine singular
IV2D	Imperfective 2nd person dual
IV2FD	Imperfective 2nd person feminine dual
IV3MD	Imperfective 3rd person masculine dual
IV3FD	Imperfective 3rd person feminine dual
IV1P	Imperfective 1st person plural
IV2MP	Imperfective 2nd person masculine plural
IV2FP	Imperfective 2nd person feminine plural
IV3MP	Imperfective 3rd person masculine plural
IV3FP	Imperfective 3rd person feminine plural
NEG_PART	Negative particle
PREP	Preposition
RESULT_CLAUSE_PARTICLE	Result clause particle

Stems

Category	Description
ABBREV	Abbreviation
ADJ	Adjective
ADV	Adverb
DEM_PRON_F	Feminine demonstrative pronoun
DEM_PRON_FS	Feminine singular demonstrative pronoun
DEM_PRON_FD	Dual demonstrative pronoun
DEM_PRON_MS	Masculine singular demonstrative pronoun
DEM_PRON_MD	Masculine dual demonstrative pronoun
DEM_PRON_MP	Masculine plural demonstrative pronoun
DET	Determinative
INTERROG	Interrogative particle
NO_STEM	No stem for the word
NOUN	Noun
NOUN_PROP	Proper noun
NUMERIC_COMMA	Decimal separator
PART	Particle
PRON_1S	Personal pronoun : 1st person singular
PRON_2MS	Personal pronoun : 2nd person masculine singular
PRON_2FS	Personal pronoun : 2nd person feminine singular
PRON_3MS	Personal pronoun : 3rd person masculine singular
PRON_3FS	Personal pronoun : 3rd person feminine singular
PRON_2D	Personal pronoun : 2nd person common dual
PRON_3D	Personal pronoun : 3rd person common dual
PRON_1P	Personal pronoun : 1st person plural
PRON_2MP	Personal pronoun : 2nd person masculine plural
PRON_2FP	Personal pronoun : 2nd person feminine plural
PRON_3MP	Personal pronoun : 3rd person masculine plural
PRON_3FP	Personal pronoun : 3rd person feminine plural
REL_PRON	Relative pronoun
VERB_IMPERATIVE	Imperative verb
VERB_IMPERFECT	imperfective verb
VERB_PERFECT	Perfective verb

Suffixes

Category	Description
CASE_INDEF_NOM	Indefinite, nominative
CASE_INDEF_ACC	Indefinite, accusative
CASE_INDEF_ACCGEN	Indefinite, accusative/genitive
CASE_INDEF_GEN	Indefinite, genitive
CASE_DEF_NOM	Definite, nominative
CASE_DEF_ACC	Definite, accusative
CASE_DEF_ACCGEN	Definite, accusative/genitive
CASE_DEF_GEN	Definite, genitive
NSUFF_MASC_SG_ACC_INDEF	Nominal suffix : masculine singular, accusative, indefinite
NSUFF_FEM_SG	Nominal suffix : feminine singular
NSUFF_MASC_DU_NOM	Nominal suffix : dual masculine, nominative
NSUFF_MASC_DU_NOM_POSS	Nominal suffix : dual masculine, nominative, construct state
NSUFF_MASC_DU_ACCGEN	Nominal suffix : dual masculine, accusative/genitive
NSUFF_MASC_DU_ACCGEN_POSS	Nominal suffix : dual masculine, accusative/genitive, construct state
NSUFF_FEM_DU_NOM	Nominal suffix : dual feminine, nominative
NSUFF_FEM_DU_NOM_POSS	Nominal suffix : dual feminine, nominative, construct state
NSUFF_FEM_DU_ACCGEN	Nominal suffix : dual feminine, accusative/genitive
NSUFF_FEM_DU_ACCGEN_POSS	Nominal suffix : dual feminine, nominative, construct state
NSUFF_MASC_PL_NOM	Nominal suffix : masculine plural, nominative
NSUFF_MASC_PL_NOM_POSS	Nominal suffix : masculine plural, nominative, construct state
NSUFF_MASC_PL_ACCGEN	Nominal suffix : masculine plural, accusative/genitive
NSUFF_MASC_PL_ACCGEN_POSS	Nominal suffix : masculine plural, accusative/genitive, construct state
NSUFF_FEM_PL	Nominal suffix : feminine plural
POSS_PRON_1S	Personnal suffix : 1st person singular
POSS_PRON_2MS	Personnal suffix : 2nd person masculine singular
POSS_PRON_2FS	Personnal suffix : 2nd person feminine singular
POSS_PRON_3MS	Personnal suffix : 3rd person masculine singular
POSS_PRON_3FS	Personnal suffix : 3rd person feminine singular
POSS_PRON_2D	Personnal suffix : 2nd person common dual
POSS_PRON_3D	Personnal suffix : 3rd person common dual
POSS_PRON_1P	Personnal suffix : 1st person plural
POSS_PRON_2MP	Personnal suffix : 2ème person masculine plural
POSS_PRON_2FP	Personnal suffix : 2ème person feminine plural
POSS_PRON_3MP	Personnal suffix : 3ème person masculine plural

Category	Description
POSS_PRON_3FP	Personnal suffix : 3ème person feminine plural
IVSUFF_DO:1S	Imperfective verb direct object : 1st person singular
IVSUFF_DO:2MS	Imperfective verb direct object : 2nd person masculine singular
IVSUFF_DO:2FS	Imperfective verb direct object : 2nd person feminine singular
IVSUFF_DO:3MS	Imperfective verb direct object : 3rd person masculine singular
IVSUFF_DO:3FS	Imperfective verb direct object : 3rd person feminine singular
IVSUFF_DO:2D	Imperfective verb direct object : 2nd person common dual
IVSUFF_DO:3D	Imperfective verb direct object : 3rd person common dual
IVSUFF_DO:1P	Imperfective verb direct object : 1st person plural
IVSUFF_DO:2MP	Imperfective verb direct object : 2nd person masculine plural
IVSUFF_DO:2FP	Imperfective verb direct object : 2nd person feminine plural
IVSUFF_DO:3MP	Imperfective verb direct object : 3rd person masculine plural
IVSUFF_DO:3FP	Imperfective verb direct object : 3rd person feminine plural
IVSUFF_MOOD:I	Imperfective verb : indicative mode
IVSUFF_SUBJ:2FS_MOOD:I	Imperfective verb : subject marker, 2nd person feminine singular, indicative mode
IVSUFF_SUBJ:D_MOOD:I	Imperfective verb : subject marker, dual, indicative mode
IVSUFF_SUBJ:3D_MOOD:I	Imperfective verb : subject marker, 3rd person common dual, indicative mode
IVSUFF_SUBJ:MP_MOOD:I	Imperfective verb : subject marker, masculine plural, indicative mode
IVSUFF_MOOD:S	Imperfective verb : subjunctive/jussive mode
IVSUFF_SUBJ:2FS_MOOD:SJ	Imperfective verb : subject marker, 2nd person feminine singular, subjunctive/jussive mode
IVSUFF_SUBJ:D_MOOD:SJ	Imperfective verb : subject marker, dual, subjunctive/jussive mode
IVSUFF_SUBJ:MP_MOOD:SJ	Imperfective verb : subject marker, masculine plural, subjunctive/jussive mode
IVSUFF_SUBJ:3MP_MOOD:SJ	Imperfective verb : subject marker, 3rd person du masculine plural, subjunctive/jussive mode
IVSUFF_SUBJ:FP	Imperfective verb : subject marker, feminine plural
PVSUFF_DO:1S	Perfective verb direct object : 1st person singular
PVSUFF_DO:2MS	Perfective verb direct object : 2nd person masculine singular
PVSUFF_DO:2FS	Perfective verb direct object : 2nd person feminine singular
PVSUFF_DO:3MS	Perfective verb direct object : 3rd personmasculine singular
PVSUFF_DO:3FS	Perfective verb direct object : 3rd person feminine singular
PVSUFF_DO:2D	Perfective verb direct object : 2nd person common dual
PVSUFF_DO:3D	Perfective verb direct object : 3rd person common dual
PVSUFF_DO:1P	Perfective verb direct object : 1st person plural
PVSUFF_DO:2MP	Perfective verb direct object : 2nd person masculine plural
PVSUFF_DO:2FP	Perfective verb direct object : 2nd person feminine plural

Category	Description
PVSUFF_DO:3MP	Perfective verb direct object : 3rd person masculine plural
PVSUFF_DO:3FP	Perfective verb direct object : 3rd person feminine plural
PVSUFF_SUBJ:1S	Perfective verb subject : 1st person singular
PVSUFF_SUBJ:2MS	Perfective verb subject : 2nd person masculine singular
PVSUFF_SUBJ:2FS	Perfective verb subject : 2nd person feminine singular
PVSUFF_SUBJ:3MS	Perfective verb subject : 3rd person masculine singular
PVSUFF_SUBJ:3FS	Perfective verb subject : 3rd person feminine singular
PVSUFF_SUBJ:2MD	Perfective verb subject : 2nd person dual masculine
PVSUFF_SUBJ:2FD	Perfective verb subject : 2nd person dual feminine
PVSUFF_SUBJ:3MD	Perfective verb subject : 3rd person dual masculine
PVSUFF_SUBJ:3FD	Perfective verb subject : 3rd person dual feminine
PVSUFF_SUBJ:1P	Perfective verb subject : 1st person plural
PVSUFF_SUBJ:2MP	Perfective verb subject : 2nd person masculine plural
PVSUFF_SUBJ:2FP	Perfective verb subject : 2nd person feminine plural
PVSUFF_SUBJ:3MP	Perfective verb subject : 3rd person masculine plural
PVSUFF_SUBJ:3FP	Perfective verb subject : 3rd person feminine plural
CVSUFF_DO:1S	Imperative verb direct object : 1st person singular
CVSUFF_DO:3MS	Imperative verb direct object : 3rd person masculine singular
CVSUFF_DO:3FS	Imperative verb direct object : 3rd person feminine singular
CVSUFF_DO:3D	Imperative verb direct object : 3rd person common dual
CVSUFF_DO:1P	Imperative verb direct object : 1st person plural
CVSUFF_DO:3MP	Imperative verb direct object : 3rd person masculine plural
CVSUFF_DO:3FP	Imperative verb direct object : 3rd person feminine plural
CVSUFF_SUBJ:2MS	Imperative verb subject : 2nd person masculine singular
CVSUFF_SUBJ:2FS	Imperative verb subject : 2nd person feminine singular
CVSUFF_SUBJ:2MP	Imperative verb subject : 2nd person masculine plural

References

[1] Wikipedia, "Natural language processing --- Wikipedia{,} The Free Encyclopedia," 2013. [Online]. Available: http://en.wikipedia.org/w/index.php?title=Natural_language_processing&oldid=554418471.

[2] M. Rashwan, M. Al-Badrashiny, M. Attia, S. Abdou and A. Rafea, "A Stochastic Arabic Diacritizer Based on a Hybrid of Factorized and Unfactorized Textual Features," *Audio, Speech, and Language Processing, IEEE Transactions on,* vol. 19, no. 1, pp. 166-175, jan. 2011.

[3] Wikipedia, "Arabic Diacritics," [Online]. Available: https://en.wikipedia.org/wiki/Arabic_diacritics.

[4] Wikipedia, "Diacritic --- Wikipedia, The Free Encyclopedia," 16 January 2012. [Online]. Available: http://en.wikipedia.org/wiki/Diacritic. [Accessed 1 February 2012].

[5] A. S. Azim, "Combining Speech with Textual Methods for Arabic Diacritization," 2012.

[6] I. Zitouni and R. Sarikaya, "Arabic Diacritic Restoration Approach Based on Maximum Entropy Models," *Computer Speech & Language,* vol. 23, no. 3, pp. 257-276, 2009.

[7] K. Shaalan, H. M. Abo Bakr and I. Ziedan, "A Hybrid Approach for Building Arabic Diacritizer," in *Proceedings of the EACL 2009 Workshop on Computational Approaches to Semitic Languages*, Athens, Greece, 2009.

[8] N. Habash and O. Rambow, "Arabic Diacritization through Full Morphological Tagging," in *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, Rochester, New York, 2007.

[9] R. Roth, O. Rambow, N. Habash, M. Diab and C. Rudin, "Arabic Morphological Tagging, Diacritization, and Lemmatization Using Lexeme Models and Feature Ranking," in *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, Stroudsburg, PA, USA, 2008.

[10] M. Elshafei, H. Al-Muhtaseb and M. Alghamdi, "Statistical Methods for Automatic Diacritization of Arabic Text," in *Proceedings of the Saudi 18th National Computer Conference (NCC18)*, 2006.

[11] M. Attia, "Theory and Implementation of a Large-Scale Arabic Phonetic Transcriptor, and Applications," PhD. Dissertation, RDI, Cairo, Egypt, 2005.

[12] N. M. Trung, N. Q. Nhan and N. H. Phuong, "Vietnamese Diacritics Restoration as Sequential Tagging," in *Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF), 2012 IEEE RIVF International Conference on*, 2012.

[13] J. Atserias, M. Fuentes, R. Nazar and I. Renau, "Spell Checking in Spanish: The Case of Diacritic Accents," in *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12),* Istanbul, Turkey, 2012.

[14] J. Mahar, G. Memon and H. Shaikh, "Sindhi Diacritics Restoration by Letter Level Learning Approach," *Sindh University Research Journal (Science Series),* vol. 43(2), p. 119–126, 2011.

[15] R. A. Haertel, P. McClanahan and E. K. Ringger, "Automatic Diacritization for Low-resource Languages Using a Hybrid Word and Consonant CMM," in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Stroudsburg, PA, USA, 2010.

[16] A. Raza and S. Hussain, "Automatic Diacritization for Urdu," in *Proceedings of the Conference on Language and Technology 2010 (CLT10)*, 2010.

[17] M. Alghamdi, Z. Muzaffar and H. Alhakami, "Automatic Restoration of Arabic Diacritics: A Simple, Purely Statistical Approach," *The Arabian Journal for Science and Engineering,* vol. 35, 2010.

[18] M. Rashwan, M. Al-Badrashiny, M. Attia and S. Abdou, "A Hybrid System for Automatic Arabic Diacritization," in *Proceedings of the Second International Conference on Arabic Language Resources and Tools*, 2009.

[19] E. Mohamed and S. Kübler, "Diacritization for Real-World Arabic Texts," 2009.

[20] I. Zitouni and R. Sarikaya, "Arabic Diacritic Restoration Approach Based on Maximum Entropy Models," *Computer Speech & Language*, vol. 23, no. 3, pp. 257-276, 2009.

[21] T. Schlippe, T. Nguyen and S. Vogel, "Diacritization as a Machine Translation Problem and as a Sequence Labeling Problem," in *Proceedings of the Eighth Conference of the Association for Machine Translation in the Americas*, 2008.

[22] M. Elshafei, H. Al-Muhtaseb and M. Al-Ghamdi, "Machine Generation of Arabic

Diacritical Marks," in *Proceedings of the International Conference on Machine Learning; Model Technologies & Applications*, 2006.

[23] I. Zitouni, J. S. Sorensen and R. Sarikaya, "Maximum Entropy Based Restoration of Arabic Diacritics," Stroudsburg, PA, USA, 2006.

[24] M. Elshafei, H. Al-Muhtaseb and M. Al-Ghamdi, "Statistical Methods for Automatic Diacritization of Arabic Text," 2006.

[25] S. Ananthakrishnan, S. Narayanan and S. Bangalore, "Automatic Diacritization of Arabic Transcripts for Automatic Speech Recognition," 2005.

[26] R. Nelken and S. M. Shieber, "Arabic Diacritization Using Weighted Finite-State Transducers," Stroudsburg, PA, USA, 2005.

[27] N. Habash, Introduction to Arabic Natural Language Processing, Morgan & Claypool publishers, 2010.

[28] Wikipedia, "Text corpus," [Online]. Available: http://en.wikipedia.org/wiki/Text_corpus.

[29] Apache, "Apache Tika," [Online]. Available: http://tika.apache.org/.

[30] P. Brihaye, "AraMorph Project Page," [Online]. Available: http://www.nongnu.org/aramorph/english/.

[31] "Arabi NLP Website," [Online]. Available: http://www.arabinlp.com/.

[32] "Mishkal Diacritization Tool," [Online]. Available: http://sourceforge.net/projects/mishkal/.

[33] "Sakhr Software," [Online]. Available: http:/www.sakhr.com.

[34] M. A. Badrashiny, "Automatic Diacritizer for Arabic Texts," 2009.

[35] O. R. a. R. R. Nizar Habash, "MADA+TOKAN: A Toolkit for Arabic Tokenization, Diacritization, Morphological Disambiguation, POS Tagging, Stemming and Lemmatization," in *Proceedings of the Second International Conference on Arabic Language Resources and Tools*, Cairo, Egypt, 2009.

Vitae

Name	:	Omar Elsayed Mohammed Shaaban
Nationality	:	Egyptian
Date of Birth	:	9/22/1986
Email	:	omar.s.shaaban@gmail.com
Address	:	Khobar – Saudi Arabia
Academic Background	:	

Earned B.Sc. in Software Engineering from KFUPM in 2010. Published three papers on free-riding in P2P networks, Cloud computing, and Arabic automatic diacritization. Participated in a TICET 2012 workshop for Arabic technologies and IECH 2012 higher education conference.