PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

MUSTAPHA STAMBOULI UNIVERSITY OF MASCARA

FACULTY OF SCIENCES EXACTES

COMPUTER SCIENCE DEPARTMENT

THESIS

Presented for graduation from

MASTER IN COMPUTER SCIENCES

Through

D I N A R   A B D E L A Z I Z

Thesis title

A R A B I C   W O R D   S E G M E N T A T I O N

Supported in July 2021, in front of the jury

PRESIDENT	Mr R. BACHIR BOUIADJRA	MCA	University of Mustapha Stambouli, Mascara
SUPERVISOR	Mr FEKIR ABDELKADER	MCB	University of Mustapha Stambouli, Mascara
EXAMINER	Mr AHMED BOUDAIEB	MAA	University of Mustapha Stambouli, Mascara
GUEST	Mr SALEM MOHAMMED	MCA	University of Mustapha Stambouli, Mascara

2020-2021

Acknowledgment

الحمد لله في الأولى والآخرة.

I want to thank everything first my god who gave me the patience and the will to finish my job.

I warmly thank my thesis supervisor, ***Dr. Fekir Abdelkader****, for expressing my sincere gratitude for his support, his listening, his remarks which have always enabled me to move forward.*

I greatly thank ***Dr. SALEM Mohammed*** *who has always been there to help me, support me, advise me, encourage me throughout my studies. May he find here the expression of my deep gratitude.*

I warmly thank the jury members, ***Drs. BACHIR BOUIADJRA Rochdi****, and* ***Boudaieb Ahmed*** *who were kind enough to be part of the jury.*

I extend my thanks to colleagues in our computer science department at Mustapha Stambouli University, Mascara.

Dedications

To my dear parents whom I love so much and to whom this work is dedicated

I am very grateful for your encouragement and your boundless support

I hope to remain a subject of pride in your eyes May Allah Almighty keep you.

To all my friends

Thank you for your friendship You were always there to support me, help me and listen to me

Thank you for the good times we had together may our ove stay forever.

Table of contents

List of figures

List of tables

1 General Introduction

i. Study context

We group together under the term automatic natural language processing (NLP), all the research and development aimed at modelling, using machines, the human capacity to produce and assimilate linguistic utterances for communication purposes. Discipline straddling between linguistics and computer science maintaining close links with cognitive sciences and having fields of overlap with Artificial Intelligence [1].

What are the applications used today that implement NLP research? In response to this question, we must focus on machine translation, spelling and grammar correction, writing assistance workshops, speech recognition and synthesis for vocal interfaces, text comprehension, research for information, information extraction, etc. The automation of one of these applications generally requires a step of text analysis which in turn takes place by breaking down this analysis into sub-tasks modelled on the different levels of linguistic analysis, namely the lexical analysis, morphological analysis, syntax analysis, semantic analysis and pragmatic analysis. From a linguistic point of view, these tasks are not independent of each other whose knowledge is interwoven from one level to another [2].

The Arabic language is no exception to this scheme, but it has been studied much less from a computer point of view than English or French. In comparison with these languages, the Arabic language presents singular characteristics, its morphology has always been a challenge for NLP specialists. Thus, its automatic processing must deal with its agglutinating nature, its inflectional richness and its diacritics carrying meaning and certain morphosyntactic features such as declination, mode, case; which leads to a large number of “lexical-morpho-Syntaxic-semantic” ambiguities, thus slowing down its processing. Indeed, the adaptation of existing models which have shown their effectiveness in the treatment of other languages to the Arabic language poses a problem [3].

By using the different language processing tools, the user is sometimes obliged to use several processing tools / levels (morphological, syntactic or semantic). However, these tools are not always encapsulated in homogeneous and interoperable entities. This complexity is due to the diversity of architectures and development languages of these tools. Usually, this problem is circumvented by using dedicated TALN development platforms such as UIMA[1], GATE[2], OpenNLP[3], LinguaStream[4], LingPipe[5], etc. These platforms are more or less mature for Latin languages. Nevertheless, there is still a lot of work to be done for the case of the Arabic language. Hence the need to converge on a development platform dedicated to Arabic naturel language processing standardizing all processing and guaranteeing greater flexibility [4].

ii. Problematic and Contribution

The object of this work is the study of approaches, techniques and tools allowing to develop a system of linguistic segmentation of words in Arabic language, then to offer a help in the activity of part-of-speech tagging. A summary of our problem can be formulated by means of the following questions:

- ❖ How to detect the limits of the units of each word?
- ❖ How to solve the problem of linguistic pre-processing with all the processing difficulties experienced by the Arabic language;

Indeed, word segmentation is a very important task for machine translation, part-of-speech tagging and information retrieval, etc... Segmentation entails breaking words into their constituent stems, affixes and clitics.

In our work, we compare two models for segmenting Arabic word using only several thousand training examples. The two models are from the same approach which is Bi-LSTM RNN coupled with CRF determines where best to segment words, it involves posing the problem as a ranking problem i.e., as a sequence labelling problem.

iii. Organisation of work

The thesis is split into two (02) parts. The first relates a state of the art necessary both for the definition and the contextualization of the elements of the study. The second states and explains our assumptions and contributions.

Part A: State of the art

[1] Apache. (2006) UIMA. [Online]. http://uima.apache.org/
[2] University of Sheffield. (1995) GATE. [Online]. http://gate.ac.uk/
[3] Apache. (2010) OpenNLP. [Online]. http://opennlp.apache.org/
[4] http://linguastream.org
[5] LingPipe (2003). [Online]. http://alias-i.com/lingpipe

This part provides a table of the main works which have contributed to the automatic processing of the Arabic language.

Chapter 1: Arabic Natural Language Processing.

In this chapter, we first present NLP, the different levels of the language, the peculiarities of the Arabic language: its morphology, its syntax as well as its different types of sentences. Then, we give a brief presentation on the obstacles that make its automatic processing a difficult task to master in the face of various problems arising from its agglutinating nature, its inflectional richness, the absence of vowelisation of the majority of written Arabic texts, etc.

Chapter 2: Arabic Word Segmentation.

This chapter is interested in the segmentation of words, even their understandings and more precisely detections of morphemes (clitics, affixes, stem). We start by defining this concept, then we present its typical architecture. Finally, we present the different approaches and works mainly dealing with word segmentation and the part of speech tagging problem.

Part B: Contributions

Part B presents and explains the two different models of our proposed approach

Chapter 3: Bi-LSTM-CRF for Arabic Word Segmentation.

This chapter presents our approach, the application framework of our work, the presentation of the implementation carried out and discussions of the results obtained.

Chapter 1

Arabic-Naturel Language Processing

Contents

2 Arabic Natural Language Processing (A-NLP)

2.1 Introduction

Arabic is a Semitic language now spoken by more than 300 million people, in more than 27 states [5]. A distinction is made between Classical Arabic and Modern & Standard Arabic (MSA). Classical Arabic is the literary form used everywhere for the purposes of writing and printing. It is also the language of religion for Muslims, regardless of their vernacular (the local language spoken within a community). Modern Arabic, derived from classical Arabic, is the language of the press, of political debates, of scientific texts and, increasingly, of secular literary texts. Arabic is a highly structured and derived language where morphology plays a very important role.

In this chapter, we have had recourse to certain fundamental notions that we'll define, which affect not only the general framework but also its realization. So first we present NLP, the different language levels, the Arabic language morphology, its syntax as well as its different types of sentences. Then, we try to give a brief presentation on the problems which make its automatic processing a difficult task to master in the face of the various problems resulting from its agglutinating nature, its inflectional richness, the absence of vowelisation of the majority of written Arabic texts, etc.

2.2 Natural language processing (NLP)

Natural Language Processing is a computerized approach to text analysis based on both a set of theories and technologies, it *encompasses a large set of techniques for the automated generation, manipulation and analysis of natural or human languages. Although most NLP techniques largely inherit linguistics and artificial intelligence, they are also influenced by relatively newer fields such as machine learning, computational statistics, and cognitive science.*

2.2.1 Definition

> *NLP is a range of theory-driven computer techniques for analyzing and representing natural texts at one or more levels of linguistic analysis with the aim of achieving human language processing for a range of tasks or applications* [6].

Through natural language processing, consistency tries to be brought to texts by attaching to the meaning of sentences and formulas. These advances are not only useful for translators or chat bots but also when computers execute oral commands or communicate by voice in order to facilitate communication, for example, for blind people. To be able to summarize long texts, or extract precise information, computers also need to understand the linguistic consistency of texts [7].

2.2.2 Brief History of NLP

In the United States, 1949, Warren Weaver in Memorandum speaks of computer translation and mechanization of the translation problem, tackles the problems of the automation of language processing, proposes to resolve syntactic and semantic ambiguities by using the redundancy of the written language within the framework of information theory.

In the 1960s, appearance of the term Automatic in Computational Linguistics, two currents were born: purely pragmatic research with acceptance of a margin of error that should lead to concrete results within a limited time frame, but whose effective exploitation is essentially linked to the rapid and inexpensive entry of data into the machine; work relating to fundamental research, either with a linguistic tendency or with a mathematizing tendency.

1967, international conference on automatic language processing, organized by CETA (term in French, Centre d'Étude pour la Traduction Automatique) in Grenoble: automatic analysis of natural languages, statistical and semantic analysis of linguistic data, algebraic theory of languages.

In the 1970s, a meeting between language automation and artificial intelligence gave birth to the term NLP. The latter settled in the 1980s and appears to be well established in the 1990s [8].

2.3 Objectives of NLP

The objective of NLP is the design of software capable of automatically processing linguistic data, that means data expressed in a language (so-called "natural"). These linguistic data can be written texts, or written or oral dialogues, or even linguistic units of a size smaller than what are usually called texts, for example: sentences, statements, groups of words or just isolated words.

Depending on the expected objectives of a NLP system, it can be classified into one of the following areas: analysis and generation.

2.3.1 Analysis

The analysis process consists of starting from the structure of surface (phrase or written text) to arrive at the equivalent deep structure. More precisely, the computer must be able to interpret a text written in a natural language in order to obtain a formal representation. To do this, it must be able to analyze language at different levels: morphological, syntactic, semantic, pragmatic, the computer must therefore call on significant linguistic knowledge (grammars, dictionaries, etc.) as well as pragmatic knowledge (immediate context, experience, knowledge of the world, etc.).

2.3.2 Generation

The generation process is the reverse process: it consists of going from the deep structure to the surface structure. More precisely, the computer must be able to switch from the formal representation of an utterance to a formulation in natural language. This implies the same knowledge as for analysis, but with problems specific to the utterance.

2.4 The different levels of language

Linguists distinguish several levels that allow the analysis or generation of a natural language utterance. These levels of knowledge are of course still valid when it comes to automatic language analysis. The main levels are:

- ***Phonology****: It is the study of the organization of sound systematically* [9]*, or how words and sentences are linked to the sounds that make them orally* [10].
- ***Morphology****: It is a study of the construction of words from primitive meaningful units* [11].
- ***Syntaxic****: It refers to the arrangement of words to form clauses and then a correct sentence and finally a text. It is also a question of determining the structural role of the words in the sentence.*
- ***Semantics****: it's interested in the meaning of words and how to combine words into meaningful sentences* [12].
- ***Pragmatic****: We can say that their meaning is how the sentences can be interpreted according to their context of enunciation (interlocutors, previous sentences, common knowledge of the world, etc.)* [13].

2.5 Arabic language

2.5.1 Arabic script

Arabic script is an alphabet with allographic variants, optional zero-width diacritics, and common ligatures. Arabic script is used to write many languages: Arabic, Persian, Kurdish, Urdu, Pashto, etc (see **Figure 2.1**).

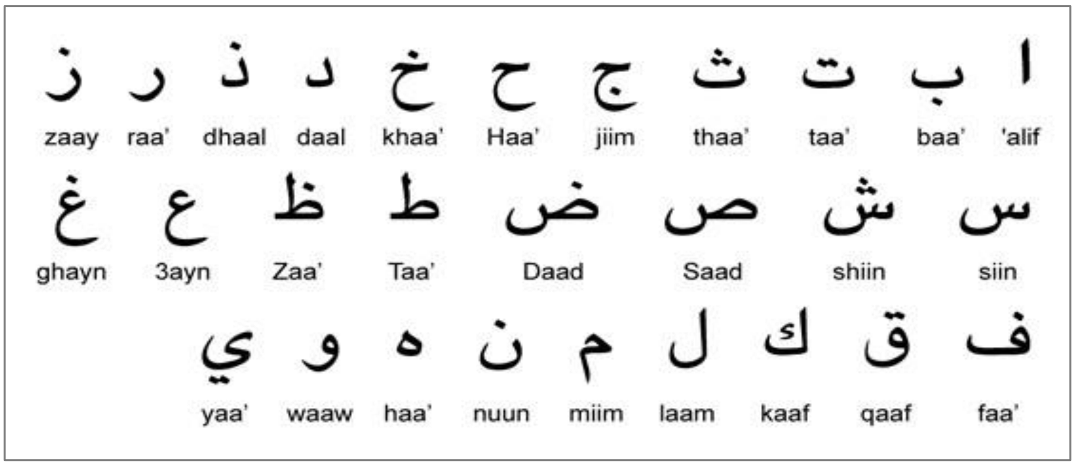

Figure 2.1 *The Arabic alphabet.*

2.5.2 Letter shapes

The morphological representation of Arabic is quite complex due to the morphological variation and the phenomenon of agglutination, the letters change shapes according to their position in the word (isolated, initial, middle and final). **Table 2.1** illustrates an example of the different shapes of the letter « ع ». in different positions:

Table 2.1 *Different shapes of the letter « ع ».*

Positions	***isolated***	***final***	***middle***	***initial***
Shapes	ع	ـع	ـعـ	عـ
Examples	باع	السميع	العزيز	عبد الله

These letters characterized by:

- *No distinction between printing and handwriting.*
- *No capitalization.*
- *From right to left.*
- *Ambiguous forms.*
- *Conjunctive letters.*
- *Letters disconnected.*

2.5.3 Diacritics

For better pronunciation precision, signs have been invented. These are three short vowels and seven orthographic signs that are added to the consonants.

2.5.3.1 The vowels

- *« **Fatha** | ـَ », the accusative sign, it overcomes the consonant and is pronounced like an « **a** » in English.*

- *« **Damma** | ـُ », the lifting sign, it overcomes the consonant and is pronounced like an « **o** », « **u** » in English.*
- *« **Kasra** | ـِ », the preposition sign, it is noted below the consonant and is pronounced like an « **i** » in English.*

2.5.3.2 The seven orthographic signs

- *« **Sukoon** | ـْ », this sign indicates that a consonant is not followed (or silent) by a vowel. It is drawn as a small circle above the letter.*
- *« **Shadda** | ـّ », the shadda is written as one of the movements above the letter, but in the Arabic language it is considered a letter and in the pronunciation of the letter it represents an accentuated pronunciation.*
- *« **Madda** | آ », the madda (extension) is placed on the Alif to indicate that this letter takes the place of two consecutive alifs or that it should not wear the Hamza.*
- *« **Hamzat-wasl** | ٱ », It is the one which is fixed at the initiation and which falls in the case of the vowel, and it is called hamzat-wasl because it leads to the pronunciation of the consonant at the beginning of the speech.*
- *« **Tanwin** », When (the Fatha, the Kasra and the Damma) are doubled, they take on a nasal sound, as if they were followed by « n » and are pronounced respectively:*
  - *« **An** | ـً», for fathatan.*
  - *« **In** | ـٍ», for kasratan.*
  - *« **Un** | ـٌ», for dammatan.*

The **Figure 2.2** illustrates an example of a sentence with most form of diacritics and different shapes of its.

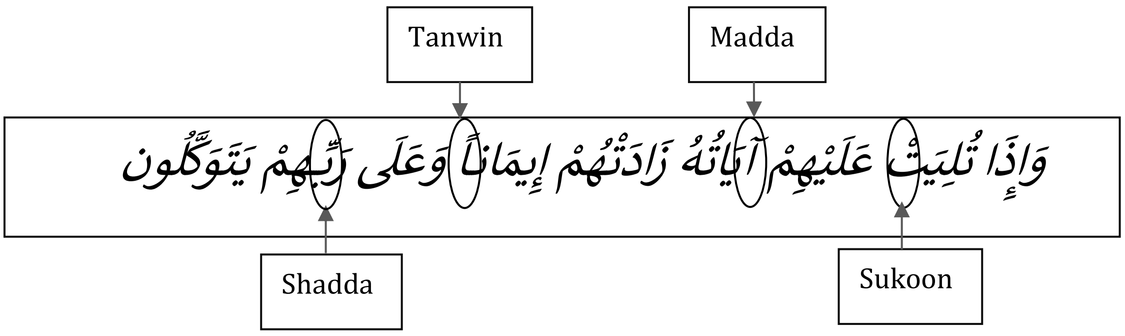
Tanwin
Madda
وَإِذَا تُلِيَتْ عَلَيْهِمْ آيَاتُهُ زَادَتْهُمْ إِيمَانًا وَعَلَى رَبِّهِمْ يَتَوَكَّلُون
Shadda
Sukoon

***Figure 2.2** Examples for diacritics.*

2.6 The grammar of the Arabic language

The traditional grammar of the Arabic language includes two categories of rules, morphology (*الصرف*) and syntax (*النحو*) divided into three areas namely:

- **Derivational morphology**, *with the root-and-pattern model, where the root provides a general abstract meaning and the scheme assigns the grammatical category simultaneously with functional and semantic features.*
- **Inflectional morphology** *which deals with variations of morpho-phonological forms and not variations of meaning.*
- **Syntax** *and syntactic functions such as subject, direct or indirect object, noun complement, etc., in order to determine the correct declension suffixes.*

2.6.1 Morphology

It studies the construction of lexical units and their transformation according to the desired meaning. In MSA, there are three main concepts that describe the majority of nouns and verbs; the root, the pattern and the lemma.

2.6.1.1 Roots

A root (in Arabic, ***الجذر***), for linguists is the basis of all forms of verbs and some Arabic nouns. It is a sequence of consonants - a sequence of three to five – (known as radicals) that can be found in all of the words that are derived from it [14]. It is an important element in derivational languages.

Indeed, each root corresponds a semantic field and using the different schemes, we can generate a family of words belonging to this field. We have an example root (*ع ل م*) used in both Classical and Modern Arabic. This root generates the verb « teach » (*علم*) and the nouns « teaching » (*التعليم*), « teacher » (*المعلم*), etc.

2.6.1.2 Patterns

A pattern (in Arabic, ***الوزن***), is a predefined form that characterizes a class of verbs or nouns: verbal pattern and noun pattern. pattern are models with different structures that are applied to the root to create a word.

Moreover, it's a model consisting of consonants and vowels together with placeholders for a root's radicals.

The **Figure 2.3** illustrates the derivation morphology process, such as the radical is « ك ت ب».

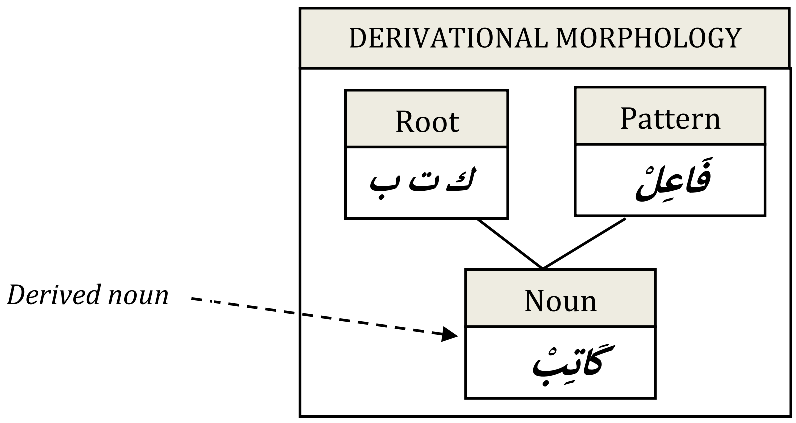

Figure 2.3 *Derived a noun by a root and a pattern.*

2.6.1.3 Lemmas

A lemma is the lexical entry in a lexicon or in a dictionary, it can be analysed as a root inserted in a pattern. It is the intersection between a graphic form and a meaning. So, to deduce the different inflected forms of a verb or a noun, firstly we should be to known the lemme or the root and pattern. Thus, the morphological derivation is described on a morphosemantic basis: from the same root derive different lexical units according to patterns. This inflection forms are known as a lexeme:

- A **lexeme** *is a group of words with the same derivational morphology that differ only by inflection.*
- A **lemma** *(also known as a citation form) is a conventional choice of one word that represents a lexeme.*

In **Table 2.2**, we clearly notice that the lemma differs from the root and is better suited for extracting information from a text.

Table 2.2 *The roots, the lemmas of words.*

Words	الناقة	الحروب	الأعرابي
Roots	نوق	حرب	عرب
Lemmas	ناقة	حرب	أعرابي

2.6.1.4 Inflection and Concatenation

In Arabic, derived words can undergo two changes before appearing in their final structural form, due to inflection and concatenation, and conversely the syntactic analysis, requiring a decomposition as a precondition, to find the morphemes.

- **Inflexion** (*التصريف*), *is the morphological process in which the form of a word is changed by grammatical attributes or a syntactic function.*
- **Concatenation** *is the morphological process in which the shape of a word is changed by attaching prefixes and suffixes.*
- **Segmentation** *is the reverse process of concatenation. That is, the text must be segmented into elementary units, each unit broken down into a stem, suffix and prefix if they exist.*
- **Morphological segments** *are the concatenative morphemes that result from the segmentation. They are stems, prefixes and suffixes.*

An Arabic word is constructed from its root in two stages, derivation and inflection [15]. The first is to apply a precise pattern to the root, which produces the stem. In general, a limited number of these stems can be used [16]. If the stem obtained is usable, affixes (prefix or suffix) can be added to it according to the characteristics of the requested word (time, number, person, mode, etc.). **Figure 2.4** shows this process.

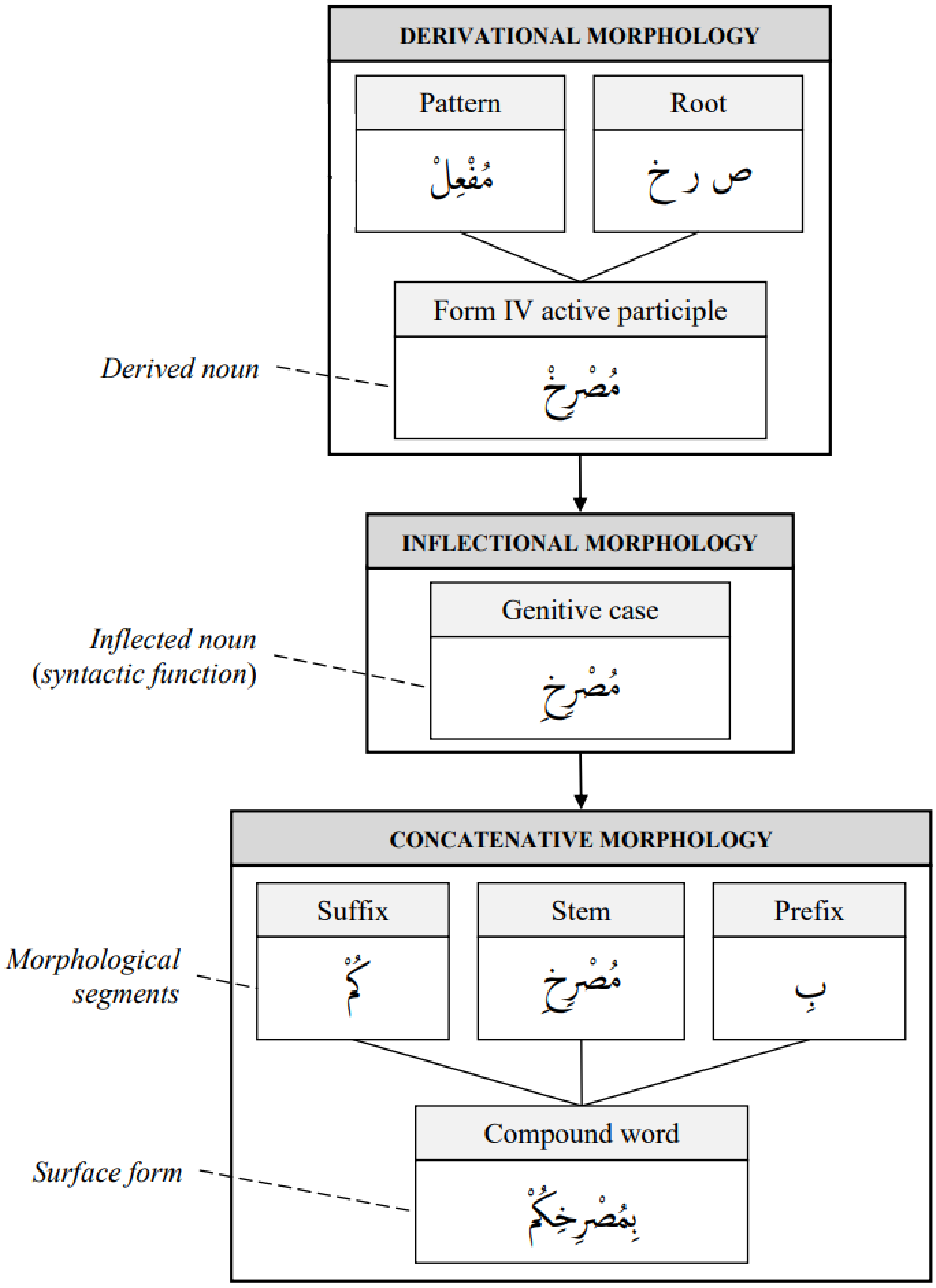

Figure 2.4 *Derivational and inflectional morphology with form and function.*

2.6.2 Part of speech (POS)

In traditional Arabic grammar, parts of speech are organized into a hierarchy made up of three main classes which are further divided into subclasses [17]. The main classes are nouns, verbs and particles.

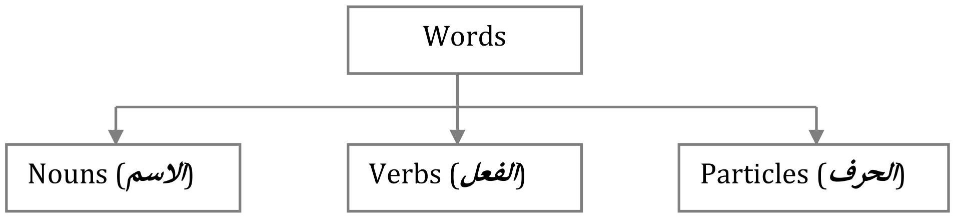

Figure 2.5 *Classification of words.*

2.6.2.1 Nouns

***Al-Farra defined**:*

> *The name is a word that accept Al-Tanween, Al-idafa or the alef and the lām, on the basis of purely formal principles to determine the name [18].*

In addition, a name is a term that indicates a specific thing without associating it with time. Arabic nouns are of two categories, those which are derived from the verbal root and those which are not like proper nouns and common nouns. In the first case, the fact that the noun is derived from a verb, it therefore expresses a certain semantics which could have an influence in the selection of the salient sentences of a text for the summary, regroup:

- *The adjectives: feminine and masculine,*
- *Demerit names, extended names as well as reduced names,*
- *Common names and proper names,*
- *Pronouns and their types (connected and separated),*
- *Relative pronouns and demonstrative pronouns,*
- *Query names, determined and undetermined names,*
- *the names of periphrases,*
- *the nouns of the verb; voice names, etc.*

The declension of names is done according to the following rules:

- ***The feminine singular**: We add the « ة », for example « مؤمن | Believer » becomes « مؤمنة | Believer ».*
- ***The feminine plural**: In the same way, we add for the plural two letters « ات », example « مؤمن | Believer » becomes « مؤمنات | Believers ».*

- ***Masculine plural**: for this declension we add one from these two components « ين | ون » depending on the position of the word in the sentence (subject or additional object), example: « مؤمن | Believer » becomes « مؤمنون / مؤمنين | Believers ».*
- ***The irregular plural**: It follows a variety of complex rules and depends on the name. Example: « طفل | child » becomes « أطفال | children ».*

2.6.2.2 Verbs

2.6.2.2.1 Definition

Semantically, ***Al-Zajjaji*** defined a verb as « a word that represents past, present and future actions », morphologically ***Ibn Hisham*** defined a verb as « a word derived from a root using a well-known verbal model » [17], *moreover, each verb is therefore the root of a family of words. As in English or French, and the word in Arabic is deduced from the root by adding suffixes or prefixes.*

2.6.2.2.2 Conjugation

The conjugation of verbs depends on several factors:

- *Time (past, present, imperative).*
- *The number of the subject (singular, dual, plural).*
- *The subject's gender (male, female).*
- *The person (first, second and third).*
- *The mode (transitive, intransitive).*

2.6.2.2.3 Times and modes

- ***The accomplished (الماضي):** matches the past tense and is distinguished by suffixes.*
- ***Present tense (المضارع):** presents the action in progress, its elements are prefixed, it consists of three types: « Indicative | مَرْفوع », « Subjunctive | مَنْصُوب » , « Jussive | مَجْزوم ».*
- ***Imperative (الامر):** Same value as its French equivalent. Its conjugation is a variant of the present tense.*
- ***Future**: is formed by the addition in a verb in present indicative a prefix « سَ »: « سَيَكْتُبُ | he will write », or we precede it with the particle « سَوْفَ »: « سَوْفَ يَكْتُبُ | he will write ».*
- ***The passive** : differs from active by vocalization. It exists for past and present, but not for the imperative. for example, « يُكْتَبُ | it is written » / « كُتِبَ | it was written ».*

Table 2.3 Tenses and modes of verbs.

Tenses	*Accomplished*	*Present tense*			*Imperative*
		Indicative	*Subjunctive*	*Jussive*	
		it presents the stated action as certain.	*It presents the stated action as potential.*	*It is equivalent in some uses to a hypothetical conditional.*	
Examples	كَتَبَ	يَكْتُبُ	يَكْتُبَ	يَكْتُبْ	أُكْتُبْ

Three verbal nouns are attached to the verb, in addition to the conjugated forms:

- ***Active participle*** *(اِسْمُ ٱلْفاعِلِ): It designates the agent of the action, « كاتِبٌ | writer ».*
- ***Passive participle*** *(اِسْمُ ٱلْمَفْعول): It designates the object of the action, « مَكْتوبٌ | written ».*
- ***Action name****(المَصْدَر): It designates the action entered as a concept, « كِتابَةٌ|writing».*

2.6.2.3 Particles

Grammatical particles are words which have no lexical value when taken separately, but which make it possible to indicate certain grammatical features such as the mode, the case or the tense, when they are associated with other words.

Some linguists define particles as uninflected words, and if we follow this definition, we can say that many western languages like English and German have particles. However, this word is normally used to refer to certain types of words that are found in oriental languages like Chinese, Korean, and Japanese.

In languages which use particles very often, such as Arabic, they can be considered to replace certain prepositions and articles, although they generally play an even more important role. For example, they can also indicate that an entire sentence is interrogative or emphatic.

2.6.3 Morphological features

In addition to part-of-speech tagging, morphological segments are annotated with multiple feature-value pairs encoded as a sequence of feature tags. **Table 2.5** (overleaf) summarizes the feature tags.

2.6.3.1 Prefixes

In the morphological segmentation, words are segmented into morphemes. Prefix features are annotated using the notation ***X:C+*** where ***X*** is the prefixed particle and ***C*** is its POS tag. For instance, ***f:CONJ+*** is used for words with the particle « *ف*» prefixed as a coordinating conjunction « *فاء العاطفة* ».

However, the notation ***X+*** is used for prefixes that belong to only a single POS, such as the prefix feature ***bi+*** for the preposition « bâ' | *حرف جر* » [19].

2.6.3.2 Suffixes

Two suffix features are annotated using the notation ***+X***. The first is the vocative suffix ***+VOC***. This is only used with the word *« Allāh »* to produce the vocative word *« Allāhumma |* اللَّهُمَّ » that occurs several times in the Quran. The second suffix tag is ***+n:EMPH***, used to denote an emphatic suffixed letter « *nūn | نون التوكيد* » [19].

The compound ***PRON:*** tag is used for suffixed pronouns « *ضمير متصل* ». in combination with person, gender and number features. For example, ***PRON:3MS*** represents a suffixed pronoun inflected for the third person masculine singular [19].

2.6.3.3 Classification Features

In addition to the prefix and suffix features, a further three features are used to classify words. The first two specified using Buckwalter transliteration is the ***ROOT:*** and ***LEM:*** indicate roots and lemmas, For example ***LEM:kitaAb*** for the lemma « kitāb | *كِتاب* ».

The last ***SP:*** feature is used to group words with a special syntactic function in traditional grammar. It is used for kāna wa akhwātuhā (كان وأخواتها), kāda wa akhwātuhā (كاد وأخواتها) and inna wa akhwātuhā (إن وأخواتها) [19].

2.6.3.4 POS tagging of words

These tables below show the part-of-speech tags of all categories, and morphological features tags:

Table 2.4 Part-of-speech tags for Classical Arabic [20].

Class	Subclass	Tag	Description	Arabic Term
Nominals	Nouns	N	Noun	اسم
		PN	Proper noun	اسم علم
	Derived nominals	ADJ	Adjective	صفة
		IMPN	Imperative verbal noun	اسم فعل أمر
	Pronouns	PRON	Personal pronoun	ضمير
		DEM	Demonstrative pronoun	اسم اشارة
		REL	Relative pronoun	اسم موصول
	Adverbs	T	Time adverb	ظرف زمان
		LOC	Location adverb	ظرف مكان
Verbs	Verbs	V	Verb	فعل
Particles	Prepositions	P	Preposition	حرف جر
	lām prefixes	EMPH	Emphatic *lām* prefix	لام التوكيد
		IMPV	Imperative *lām* prefix	لام الامر
		PRP	Purpose *lām* prefix	لام التعليل
	Conjunctions	CONJ	Coordinating conjunction	حرف عطف
		SUB	Subordinating conjunction	حرف مصدري
	Other particles	ACC	Accusative particle	حرف نصب
		AMD	Amendment particle	حرف استدراك
		ANS	Answer particle	حرف جواب
		AVR	Aversion particle	حرف ردع
		CAUS	Particle of cause	حرف سببية
		CERT	Particle of certainty	حرف تحقيق
		CIRC	Circumstantial particle	حرف حال
		COM	Comitative particle	واو المعية
		COND	Conditional particle	حرف شرط
		EQ	Equalization particle	حرف تسوية
		EXH	Exhortation particle	حرف تحضيض
		EXL	Explanation particle	حرف تفصيل
		EXP	Exceptive particle	أداة استثناء
		FUT	Future particle	حرف استقبال
		INC	Inceptive particle	حرف ابتداء
		INT	Particle of interpretation	حرف تفسير
		INTG	Interrogative particle	حرف استفهام
		NEG	Negative particle	حرف نفي
		PREV	Preventive particle	حرف كاف
		PRO	Prohibition particle	حرف نهي
		REM	Resumption particle	حرف استئنافية
		RES	Restriction particle	أداة حصر
		RET	Retraction particle	حرف اضراب
		RSLT	Result particle	حرف واقع في جواب الشرط
		SUP	Supplemental particle	حرف زائد
		SUR	Surprise particle	حرف فجاءة
		VOC	Vocative particle	حرف نداء
	Quranic initials	INL	Disconnected letters	حروف مقطعة

Table 2.5 *Morphological feature tags for Classical Arabic [20].*

Type	Category	Tag	Description
Prefixes	Letter *alif* as a prefixed particle	A:INTG+	Interrogative *alif* (همزة استفهام)
		A:EQ+	Equalization *alif* (همزة التسوية)
	Letter *wāw* as a prefixed particle	w:CONJ+	Conjunction *wāw* (الواو عاطفة)
		w:REM+	Resumption *wāw* (الواو استئنافية)
		w:CIRC+	Circumstantial *wāw* (حرف حال)
		w:SUP+	Supplemental *wāw* (الواو زائدة)
		w:P+	Preposition *wāw* (حرف جر)
		w:COM+	Comitative *wāw* (واو المعية)
	Letter *fā'* as a prefixed particle	f:CONJ+	Conjunction *fā'* (الفاء عاطفة)
		f:REM+	Resumption *fā'* (الفاء استئنافية)
		f:SUP+	Supplemental *fā'* (الفاء زائدة)
		f:RSLT+	Result *fā'* (الفاء واقعة في جواب الشرط)
		f:CAUS+	Cause *fā'* (الفاء سببية)
	Letter *lām* as a prefixed particle	l:P+	Preposition *lām* (حرف جر)
		l:EMPH+	Emphasis *lām* (لام التوكيد)
		l:PRP+	Purpose *lām* (لام الامر)
		l:IMPV+	Imperative *lām* (لام التعليل)
	Other prefixes	Al+	Determiner ***al*** (لام التعريف)
		bi+	Preposition *bā'* (حرف جر)
		ka+	Preposition *kāf* (حرف جر)
		ta+	Preposition *tā'* (حرف جر)
		sa+	Future particle *sīn* (حرف استقبال)
		ya+	Vocative particle *yā'* (أداة نداء)
		ha+	Vocative particle *hā'* (أداة نداء)
Core Features	Classification features	POS	Part-of-speech
		LEM:	Lemma
		ROOT:	Root (جذر)
		SP:	Special group (e.g. كان واخواتها)
	Verbal features	Form	I to XII (وزن)
		Aspect	Perfect, imperfect or imperative
		Mood	Indicative, subjunctive or jussive
		Voice	Active (معلوم) or passive (مجهول)
	Nominal features	Derivation	Participle or verbal noun
		State	Definite (معرفة) or indefinite (نكرة)
		Case	Nominative, accusative or genitive
	Phi features	Person	First, second or third (الاسناد)
		Gender	Masculine or feminine (الجنس)
		Number	Singular, dual or plural (العدد)
Suffixes	Suffix features	+VOC	Vocative suffix (used for اَللَّهُمَّ)
		+n:EMPH	Emphasis *nūn* (نون التوكيد)
		PRON:	Pronoun suffix (ضمير متصل)

2.6.4 The syntax

It studies the correct structure of sentences by analyzing the order of lexical units. The sentence definition, which is suitable for English and French, cannot be applied to all sentence models in the Arabic language. The latter has, in addition to the verbal sentence, two sentence schemes in which the central element is not a verb and in which also there can be no explicit mark of temporality. In Arabic, these are the noun phrase and the locative phrase.

In the Arabic language, there are three elementary sentence schemes: the verbal, the nominal and the locative sentence. The organizing core is the verb (a temporal relation) in the first type, a nominal in the second and a locative or a circumstantial (an atemporal relation) in the third [21].

2.6.4.1 The verbal sentences

The scheme of the verbal sentence is as follows (see **Figure 2.6**): the core of this sentence is the verb. It is both specified by the subject and identified by the situation which is the origin of the spatio-temporal coordinates of the action. This situational identification operation gives the verb its marks of tense, aspect and modality, while its subject specification gives it its marks of agreement in person, number and gender.

The verbal sentence makes it possible to express all kinds of actions, whether of the event type or of the state type, punctual or durative. It is an essentially dynamic value structure with al-madi (*الماضي* / the accomplished). With al-mudari (*المضارع* / the unaccomplished), she essentially describes static, permanent situations, which often have a generic value [21].

2.6.4.2 The nominal sentences

The elementary nominal sentence is characterized by two operations carried out on the same entity. It is a nominal entity (the subject | *المبتدأ*), which is identified by the situation and identified by the interlocutors. This entity is specified by another entity (the predicate | *الخبر*) which can be a noun (see **Figure 2.6**), a clause or an adjective or a prepositional phrase. If this second entity (the predicate) is nominal, it must have the same gender and the same number as the predicate. If the predicate is a verbal proposition, or a nominal proposition, it must contain a pronoun co-referring to the subject. If the subject is a prepositional phrase or a circumstantial, it does not contain any element referring to the subject.

The nominal sentence can express a relation between two entities, the subject and the predicate. It is the semantic and syntactic properties of the predicate that determine the interpretation of the nominal sentence as a whole. Usually, this type of sentence describes static situations [21]. The nominal sentence has no equivalent in English or French.

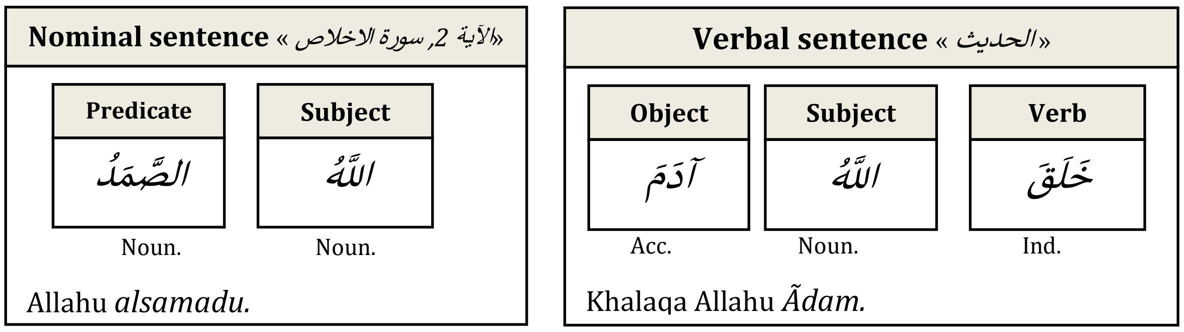

Figure 2.6 Nominal and verbal positions.

2.6.4.3 The locative phrase

The locative phrase is composed of a prepositional group or a circumstantial (of time or place) often followed by a noun in the genitive. It represents the locator group and the localized spotted. This locator group is spotted, and always contains a name that must be spotted by the situation and identified by the interlocutors, i.e., defined.

The semantic characteristic of locative sentences resides in the fact that they only have a syntactic locating relation which goes from the locator to the localized and gives it its case mark. As for the value of this type of sentence, it depends mainly on the value of the locator and in a secondary way on that of the localized. If the locator has a spatial or temporal value, the sentence will be interpreted as a concrete spatial or temporal localization [21].

2.7 Difficulties in automatic processing of Arabic

The Arabic language is considered a difficult language to master in the field of NLP. the latter is always faced with the various problems of the Arabic language stemming from the agglutinating nature of this language, its inflectional richness, the absence of vowels in the majority of written Arabic texts, etc.

In this section, we will try to give a brief overview of those problems, which make automatic processing of the Arabic language a difficult task to master.

2.7.1 Agglutination

Most Arabic words are composed by the agglutination of basic lexical elements (proclitic + base + enclitic). This characteristic can generate ambiguity at the morphological level. Indeed, it is sometimes difficult to distinguish between a proclitic or enclitic and an original character of the word. For example, the character « و » in the word *« وَهْمٌ / Illusion »* is an original character while in the word *« وَهُمْ / And they are »* it is a proclitic.

Unlike Latin languages, in Arabic, articles, certain prepositions, pronouns, etc, stick to the adjectives, nouns, verbs and particles to which they relate. Compared to English, an Arabic word can sometimes correspond to an English sentence. For example, the word « فَسَيَكْفِيكَهُمُ » can be translated as « So that's enough for you against them ».

2.7.2 The absence of vowels

The absence of vowels (non-vowelling) in Arabic texts generates several cases of ambiguities and problems during automatic analysis. Indeed, the grammatical ambiguity increases if the word is not vowelized. This is because a non-vowel word has several possible vowelations, and for each vowel is associated a different list of grammatical categories (see **Table 2.6**)

Table 2.6 *Example of vowelations.*

Words	Vowelized word	In English
يعد	يَعِدُ	To promise
	يَعُدُّ	Count
	يُعِدُّ	To prepare

2.7.3 Text segmentation

Other peculiarities of this language are added and pose a problem during the segmentation of the text in sentences we quote:

- ❖ *The absence of capital letters to mark the start of a new sentence.*
- ❖ *The ambiguity of punctuation marks «**.,** » and certain particles such as « and | و » and « therefore | ف », which may in some cases no longer play as triggers for a new sentence.*

For Example: *In the sentence « تتراوح أسعار البطاطا بالجملة بين 30 د.ج و 45,5 د.ج للكلغ الواحد », neither the point « **.** » nor a comma « **,** », or the conjunction « و »,* can *decide the triggering of the segmentation of the sentence*.

2.7.4 Lemmatization

One of the most important treatments for the Arabic language is the lemmatization of words. The objective of lemmatization is to find the representative form of index of a word from its form represented in the document.

The problem that arises at this level is the lemmatization of broken plurals. How can we recognize that a noun is in its singular form when it can be in the plural broken? And how do you get it to associate its unique shape when there can be several?

In Arabic, out of 10 plural forms, there are on average 3 in the regular plural and 7 in the broken plural [22]. So, we cannot avoid or even get around the problem of these plurals. The broken plural is a non-suffixal form obtained by the combination of a root and a pattern; their lemmatization is a difficult task because this type of plural does not follow obvious grammatical rules [23].

To lemmatize broken plurals, strict application of schemes does not guarantee successful and efficient identification. This app is unable to choose and match any singular form to the broken plural form. Among the 25 possible broken plural schemes, the association is very delicate and the explanation of the choice remains unclear [24]. In fact, the broken plural is one of the most difficult linguistic phenomena to formalize in morphology.

For example, for the word « works | *أعمال*», it matches the broken plural scheme « *أفعال*», and therefore, it is correctly identified as one. Conversely, for the word « prejudice | *أجحاف*» which also matches the broken plural pattern « *أفعال*» when in reality it is a noun in the singular. The similarities that may exist between some patterns of the singular and the broken plural include certain noises by classifying these words as a broken plural [24].

When classifying broken plural patterns, we have noticed that some broken plural patterns may have more than one singular pattern, which causes ambiguity in choosing the right solution. For example, the words « ways | *وسائل* », « messages | *رسائل* » and « clients | *زبائن* » correspond to the same broken plural « *فعائل* » pattern. However, the latter has three different patterns in the singular: « *فعول* | *فعالة* | *فعيلة* » (*example given in **Table 2.7***). The problem we have encountered is that there is no rule or mechanism that we could follow to choose in the singular, the corresponding pattern [25].

Table 2.7 *A broken plural pattern with several singular patterns.*

Plural pattern	Word in broken plural	Singular pattern	Word in singular
فعائل	*وسائل*	*فعيلة*	*وسيلة*
	رسائل	*فعالة*	*رسالة*
	زبائن	*فعول*	*زبون*

2.8 Conclusion

Through this first chapter, we have had presented in the first the Arabic language, the grammar, the basic concepts of NLP, then we explored the problems and difficulties of NLP. In the following chapter, we'll present the Word Segmentation as a problematic, the main approaches and the related works for it, and part-of-speech tagging problem.

Chapter 2

Arabic Word Segmentation

Content

3 Arabic Word Segmentation

3.1 Introduction

Today, information overload has increasingly become a challenge that information systems must cope with. Consequently, it would be interesting to set up tools to automate the processing of languages linked to the search for information, to facilitate access to it, to reduce information overload, etc.

Until then, the IT market has tried to respond to this problem by developing specific tools dealing with the different languages of the world such as: search engines, Question / Answer systems, extraction systems. information, morphological and syntactic analyzers, etc.

From all tools of NLP, in this chapter we'll present a necessary part in NLP i.e., Arabic word segmentation, so in the first we'll define the segmentation, types of segmentation, Arabic word form, and some rules for segmentation. Then, we'll mention all approaches used in Arabic word segmentation and POS tagging problem followed by the field Artificial Neural Networks.

3.2 Segmentation

3.2.1 Definition

Before analyzing segmentation morphology in texts written mainly in Arabic, we must first define the term segmentation (*التجزئة*) to be able to understand the use of segmentation in this context, from the ***Cambridge*** dictionary, the segmentation is:

- *The division of something into smaller parts.*
- *Division of a word into its constituent letters.*
- *Embryonic division of the body of annelids and arthropods into a series of segments, or multicellular rings, first similar, then differentiated* [26].

3.2.2 Types

There are several levels of analysis for signs of interest that make it possible to identify the different elements constituting the text and to define its boundaries. We can wind up at the graphic word level, at the level of lexical units or go beyond these to arrive at the basic units (the morphemes).

Depending on the aim of the analysis to be undertaken: lexical, morphological, syntactic or thematic, we can generally find four types of segmentation:

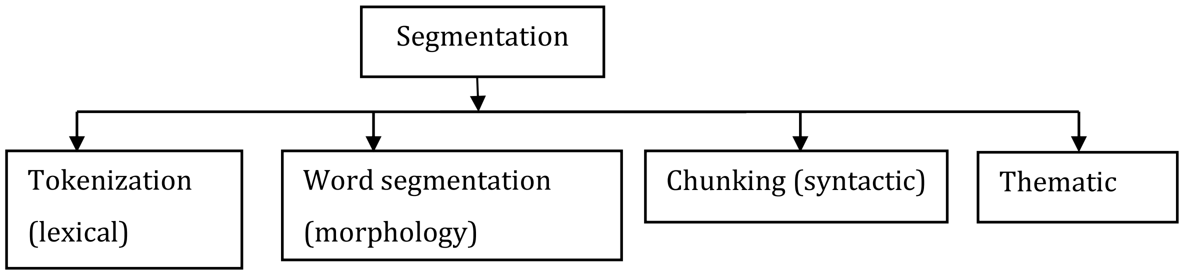

Figure 3.1 Types of segmentation.

3.2.2.1 Tokenization

It's the segmentation of a text into lexical segments (tokens) or words. This type of segmentation is also called itemisation.

3.2.2.2 Morphological segmentation

It seeks to isolate the different constituents of lexical items into distinct, smaller units, which are morphemes.

3.2.2.3 Chunking

It consists to isolate the different constituents of the text into independent units, superior to words, such as clauses, phrases, etc.

3.2.2.4 Thematic segmentation

It consists to subdivide a text into a set of thematically coherent segments based on the lexical and semantic relationships between them.

3.3 Sentence and Word segmentation

3.3.1 Sentence Segmentation

This is an important first step in textual content processing. The segmentation of a text into sentences is generally based on punctuation [27]. In the Arabic language, estimating the sentence boundary is a relatively simple task, approximately same as in English. The average number of words per sentence is higher than the average of the English word, which will not affect on the segmentation process but on the parsing process.

Phrase boundaries and sentence boundaries can be estimated based on Arabic punctuation marks which are {=,][, -, "", ؛ ,. ,: ,... ,؟! ،} [28].

3.3.2 Word Segmentation

The word segmentation is a conventional trouble in NLP. It focuses on segmenting certain morphemes, that is, affixes and clitics, from the beginning and end of words and stems. Arabic word segmentation is essential for various natural language processing and text mining tasks, such as machine translation, text classification, and parsing [29].

3.4 Arabic word form

The Arabic word has special case where the letters are attached together with high possibility of including two categories of Part-Of-Speech or more. It leads to problems in segmentation and stemming stages in NLP application like in Tagger. For instance, the word «and by the Monotheism | *وبالتوحيد*», in classical definition of a word, it is a one term but, as we can see, it has four POSs.

Arabic clitics attach themselves to the inflected base word in a strict order which can be represented as follows classes:

$$W \equiv [BASE + Affixes + Clitics] \equiv [lemma + morphological\ features + Clitics]$$

$$W \equiv [Stem + affixes + Clitics] \equiv [Inflected\ word + Clitics]$$

Some researchers did not distinguish between the affixes and Clitics which take the Arabic word in general as (prefixes + stem + suffixes) [28].

3.5 Arabic word clitics

A clitic is a linguistic unit that is pronounced and written whose status lies in between that of an affix or a word. The phonological behaviour of the first: they tend to be short and unaccented but their syntactic behaviour is more like words, often appearing as pronouns, articles, conjunction, or verbs [27]. It's a morpheme that has the syntactic traits of a word but suggests evidence of being phonologically bound to another word [30].

Clitics can be proclitics that precede the word (such as a prefix) or enclitics that follow the word (such as a suffix).

3.5.1 Proclitics

It can be prefixes of a verb, a noun or a pronoun, the **Figure 3.2** list just about all known combinations of proclitic verbs and nouns. Each level can be one or zero occurs except that the last level must exist (noun, verb).

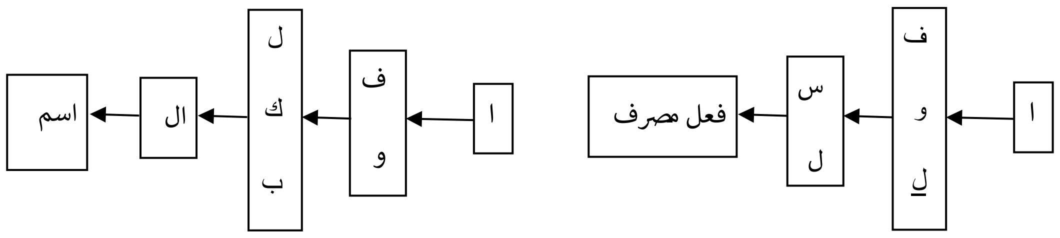

Figure 3.2 Verb and noun proclitics.

3.5.2 Enclitics

Enclitics can be after the verb or the noun. The Enclitic « *نا* » is ambiguous and has two possible roles (either a clitic suffix or an inflection suffix). for example, the word « *قتلنا* » *can* be *«we killed »* or *« he killed us »* which is affix in the first context and enclitic in the second context. All enclitics are pronouns and therefore the pronouns themselves do not have enclitics. **Figure 3.3** shows all the common enclitics for nouns and verbs with their order [28].

(فعل) [ني][ه, هما, هم, ها, هن, ك, كما, كم, كن, نا]

(اسم) [ه, هما, هم, ها, هن, ك, كما, كم, كن, نا, ي]

Figure 3.3 Verb and noun enclitics.

3.6 Related works

Arabic word segmentation is typically considered as a part of the POS tagging problem. Two approaches have been conducted in the field of Arabic POS tagging research to manage Arabic word segmentation: rule-based and statistical.

3.6.1 Rules-based approaches

For the rule-based approach, different methods have been used, we'll mention some of them.

3.6.1.1 Lexicon approach (morphological rules)

Khoja [31], *Zribi et al.* [32], *Alqrainy et al.* [33], *Al-Taani and Salah* [34], and *Hadni et al.* [35] used a predefined set of morphological rules and lexicon technique to find affixes and clitics. These current systems were built to segment/tag unvocalized Arabic text using a lexicon or dictionary that was segmented/tagged manually and used as a training corpus containing all possible tags (lexical information) for each word.

For instance, *Alqrainy* [33] generated a lexicon of patterns which are associated with the last diacritical mark and generated automatically by combining a single lexicon of all prefixes[6], all forms and all suffixes, where the tag is associated with each morpheme in each single lexicon [33].

Two reasons for choosing a partially-vocalized[7] Arabic text were mentioned by *Alqrainy*:

- *To investigate the importance of the last diacritical mark in reducing the lexical ambiguity and assigning the correct POS tag to the words regardless of the context in most cases.*
- *To explore the possibility of assigning the POS tag based on the pattern of the word instead of the word itself.*

For testing word matched if its correct pattern, they developed an algorithm Pattern-matching with five steps (see more in [33]).

The dataset is manually tagged for comparison with the results of the algorithm. The algorithm correctly tagged 91% of the data set words. Most of the errors came from proper and Arabised nouns, which did not have a pattern to follow and tagged incorrectly.

In Addition, *Al-Taani* [34] use a tagging system allows labelling the words in a non-vocalized Arabic text to their tags. It is constituted of three main phases: the lexicon analyzer, the morphological analyzer, and the syntax analyzer.

The first level contains all Arabic particles including prepositions, adverbs, conjunctions, interrogative Particles, exceptions, and interjections. The second level uses morphological information such as the patterns of the word and its affixes to presume the class of the words. The last level consists of two stages; the first stage depends on specific keywords that inform us the tag of the successive word, and the second stage is the reversed parsing technique [34].

The proposed system divides the input text into separate words, then enter each word into the first level, if it exists return the corresponding TAG, if not; the system transfers the word to the second level. After processing the word; if it matches return the presumed TAG, if not; the system transfers it to the final level. After testing words positions, if the TAG is found then the TAG is returned, otherwise there is no presumption about the corresponding TAG or the TAG is UNKNOWN [34], **Figure 3.4** shows the architecture of this system.

[6] including all valid concatenations.
[7] assigned only to the last letter of each word in the data set.

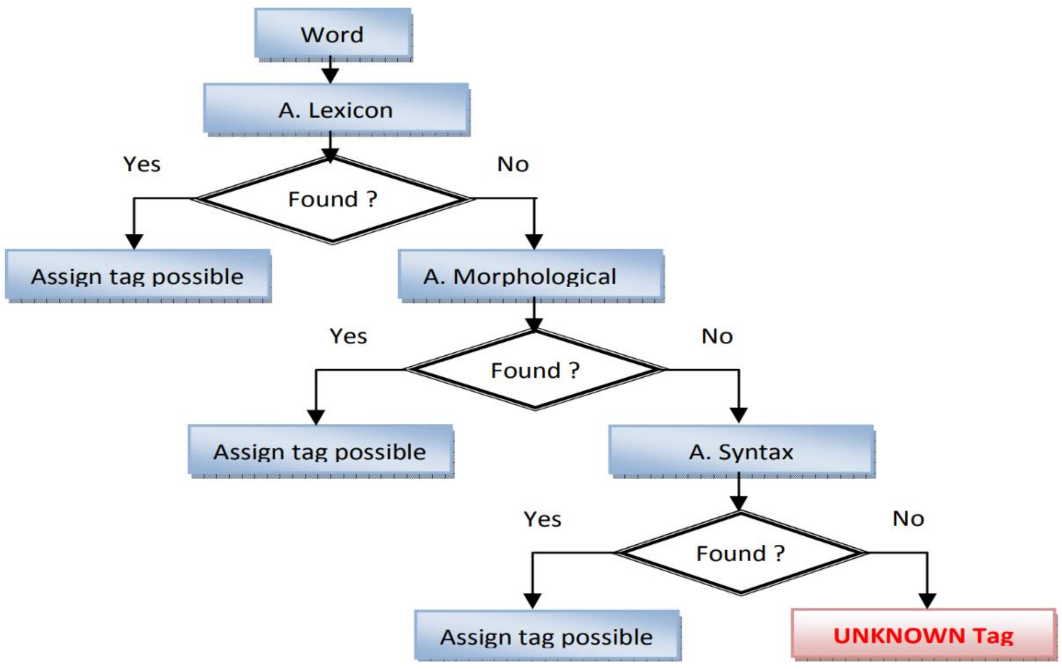

Figure 3.4 *Architecture of the Rule-Based Arabic POS Tagger [34].*

In following of this section, we present the HMM (Hidden Markov Model) model it was integrated in TAANI'S method for POS tagging Arabic text by *Hadni et al* [35] (see **Figure 3.5**).

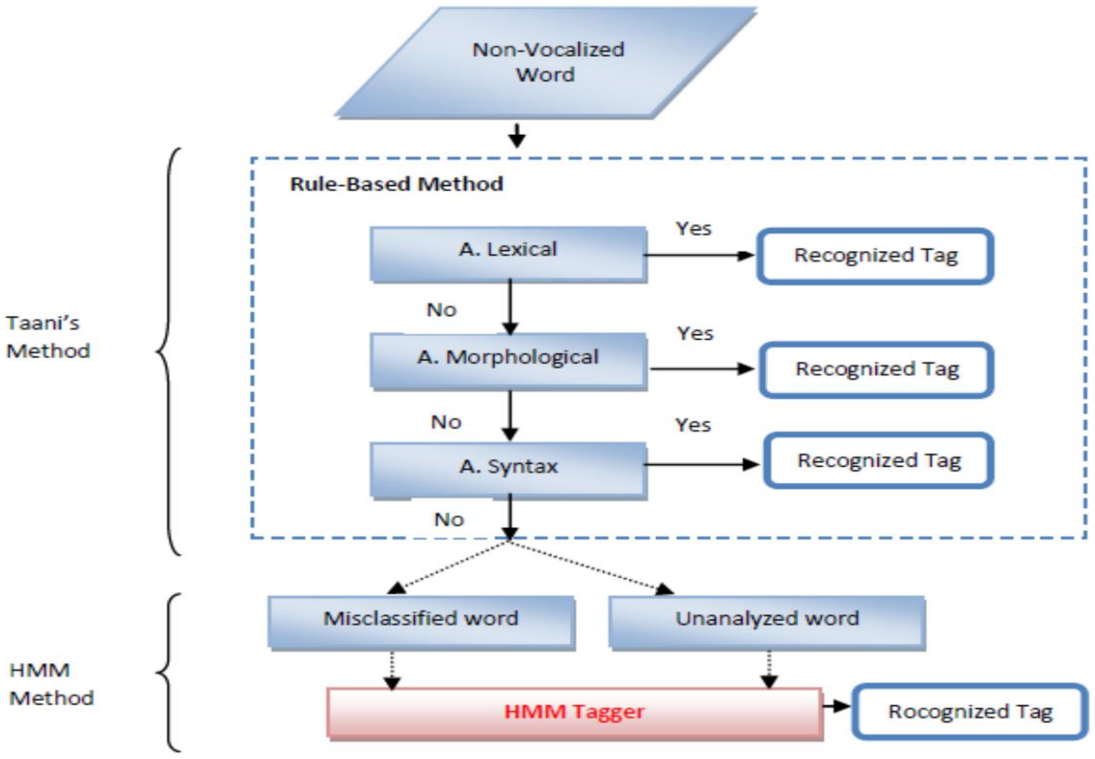

Figure 3.5 *Flowchart of the HADNI'S Arabic POS Tagger [35].*

3.6.1.2 Other rule-based approaches

Other rule-based methods have been used, such as Transformation-Based Learning [36], regular expressions [37], and the first solution provided by the morphological analyzer [38].

Transformation-Based Learning (TBL), was introduced by *Eric Brill* in 1994 and achieved an accuracy of 97.2% in same corpus outperforming HMM tagger. It is an error-driven approach to induce the retagging rules from a training corpus [36].

In the first phase of algorithm, the system builds a lexicon (contain each word with all possible tags and the frequency of each tag) then it maintains two corpuses: gold-standard corpus (contains word/tag), training corpus (contains only words), next it constructs an initial state tagging by assigns the most frequent tag to each word in the training corpus [36].

After that it compares initially annotated corpus it with gold-standard corpus to detect the error class, after applies a set of templates to correct it and chooses the rule with highest gain (is a number of corrections in the training corpus), store this rule in a list after being applied to the training corpus. Then, it calculates the largest error class from the updated training corpus again and so on until no correction could be made [36].

In the tagging phase, the raw text will use the lexicon to tag each word with its most common tag then the learned rules list is applied [36].

3.6.2 Statistical

For the statistical approach, two strategies have been used for Arabic word segmentation. The first strategy separates the segmentation process from the POS tagging process. In this method, segmentation is considered as a classification problem and machine learning (ML) strategies are applied to train the selected classifier.

While the other strategy, the process of selecting the correct segmentation is conducted within the process of selecting the correct morphology analysis, which contains the segmentation information and POS tagging decision.

3.6.2.1 ML algorithms

Different ML algorithms have been applied, such as SVM [39, 40, 41], conditional random fields (CRF) [42], k-nearest neighbour algorithm [43, 44], and maximum likelihood model [45].

MADAMIRA 1.0 [39], follows the same general design as MADA [46, 47], with some additional components inspired by AMIRA [39]. It is constituted of seven main phases (see **Figure 3.6**).

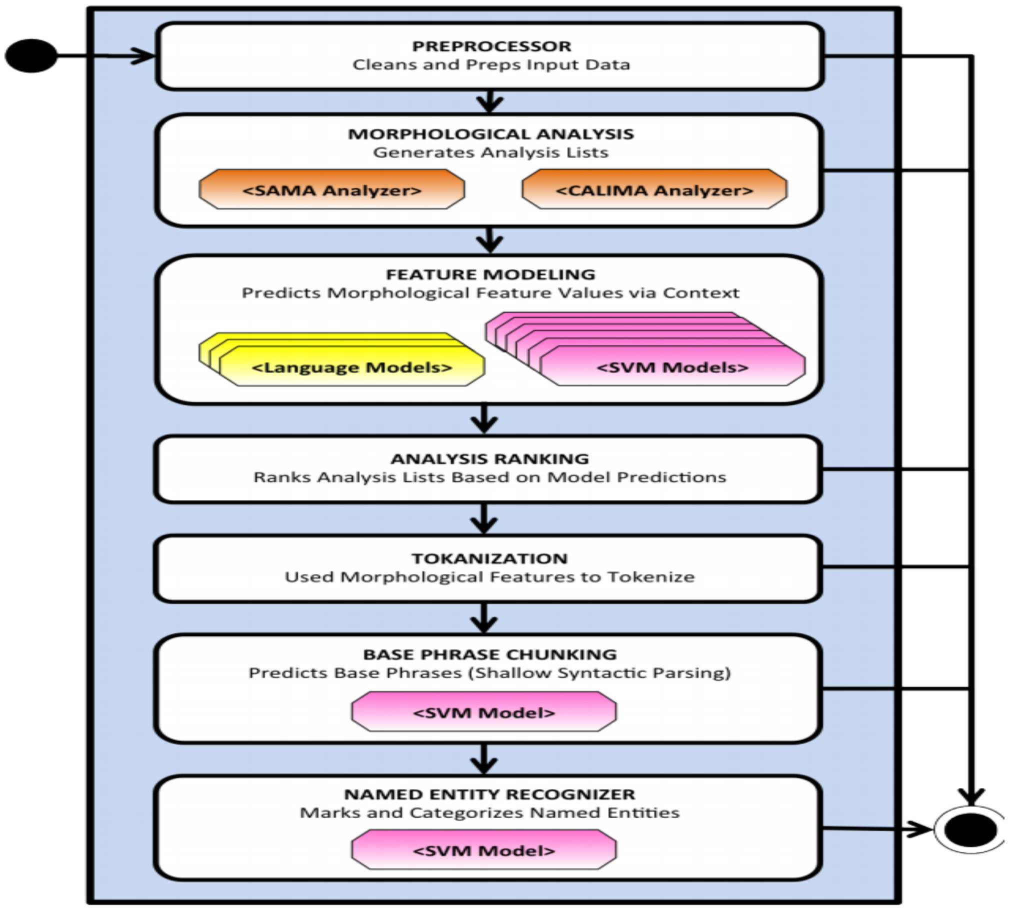
PREPROCESSOR
Cleans and Preps Input Data
MORPHOLOGICAL ANALYSIS
Generates Analysis Lists
<SAMA Analyzer>
<CALIMA Analyzer>
FEATURE MODELING
Predicts Morphological Feature Values via Context
<Language Models>
<SVM Models>
ANALYSIS RANKING
Ranks Analysis Lists Based on Model Predictions
TOKENIZATION
Used Morphological Features to Tokenize
BASE PHRASE CHUNKING
Predicts Base Phrases (Shallow Syntactic Parsing)
<SVM Model>
NAMED ENTITY RECOGNIZER
Marks and Categorizes Named Entities
<SVM Model>

Figure 3.6 *The MADAMIRA Architecture [39].*

Firstly, a text entered in the Pre-processor phase, where cleaned and converted to Buckwalter format. Then it's passed into the Morphological Analysis phase to develop a list of all possible analyses (independent of context) for each word [39].

The Features Modelling phase applies SVM and Language Models on the text and analyses list to derive predictions for the word's morphological features. SVMs are used for closed-class features, while the Language Models predict open-class features, such as lemma and diacritics forms. After that for each word's analysis list is scored by an Analysis Ranking phase on how well each analysis agrees with the model predictions and then sorts the analyses list based on that score [39].

The top-scoring analysis of each word can then be passed to the Tokenization phase to generate a customized tokenization (or several) for the word, according to the schemes requested by the user. The chosen analyses and tokenizations can then be used by the Base-Phase-Chunking phase to divide the input text into chunks (using another SVM model). Similarly, the Named Entity Recognizer phase uses a SVM to mark and categorize named entities within the text. When all the requested components have finished, the results are returned to the user. Users can request specifically what information they would like to receive; in addition to tokenization, base phrase chunks and named entities, the diacritic forms, lemmas, glosses, morphological features, parts-of-speech, and stems are all directly provided by the chosen analysis [39].

In addition, *Muhammad et al.* [43] use a Memory-Based Learning (MBL) classifier. MBL is « a lazy learning method that does not abstract rules from the data, but rather keeps all training data. In training, the learner stores the training instances without abstraction. Given a new instance, the classifier finds the k-nearest neighbours in the training set and chooses their most frequent class for the new instance » [43].

The classification in MBL is performed by the k-NN algorithm that searches for the k-nearest neighbours according to the ***Δ(X, Y)***[8] function. The majority class of the k-nearest neighbours then determines the class of the new case. With symbolic feature values, distance ties can occur when two nearest neighbours mismatch with the test instance on the same feature value, while all three instances have different values. In the k-NN implementation used by [44], equidistant neighbours are taken as belonging to the same k, so this implementation is effectively a k-nearest distance classifier. This implies that when k = 1, more than one nearest neighbour may be found, all at the same distance to the test instance [44].

3.6.2.2 Other strategy

Habash et al. [48] and *Roth et al.* [49] used the SVM algorithm to choose the best solution of BAMA, and *Freihat et al.* [50] used a maximum entropy POS tagger.

Some studies have focused only on Arabic word segmentation such as a Bi-LSTM- based taggers used by *Zalmout et al.* [51] for the morphological feature tagging tasks, with various embedding levels and morphological features. Also, it's used by *Eldesouki* [52] coupled with CRF for a DA segmenters to perform sequence labelling over the characters in words.

All the aforementioned studies have either relied on the Buckwalter morphological analyzer or dictionaries with rules to solve the issue of segmentation with rewriting. While *Almuhareb et al.* [29] propose an Arabic word segmentation technique based on a Bi-LSTM approach and Word Embedding, this work addresses the two tasks of word segmentation only and word segmentation for nine cases of the rewrite. Word segmentation with a rewrite concerns inferring letters that are dropped or changed when the main word unit is attached to another unit, and it writes these letters back when the two units are separated as a result of segmentation.

[8] A distance function Δ (X, Y) between the new instance X and the memory instance Y.

3.7 Artificial neural networks (ANNs)

It's a system whose design was originally schematically inspired by the functioning of biological neurons, and which subsequently approached statistical methods.

Neural networks are generally optimized by probabilistic-type learning methods, in particular Bayesian. They are placed in two parts:

- *One in the family of statistical applications, which they enrich with a set of paradigms making it possible to create rapid classifications (Kohonen networks in particular),*
- *And the other in the family of artificial intelligence methods to which they provide a perceptual mechanism independent of the implementer's own ideas, and input information to formal logical reasoning (Deep Learning).*

3.7.1 Network Model

3.7.1.1 Network structure

A neural network is generally made up of a succession of layers, each of which takes its inputs from the outputs of the previous one. Each layer (i) is composed of Ni neurons, taking their inputs from the Ni-1 neurons of the previous layer (see **Figure 3.7**).

Each synapse has a synaptic weight associated with it, so the Ni-1s are multiplied by this weight, then added by neurons of level i, which is equivalent to multiplying the input vector by a transformation matrix. Putting the different layers of a neural network one behind the other would amount to cascading several transformation matrices and could be reduced to a single matrix, product of the others, if there were not at each layer, the output function which introduces a non-linearity at each step. This shows the importance of the judicious choice of a good output function: a neural network whose outputs would be linear would have no interest [53].

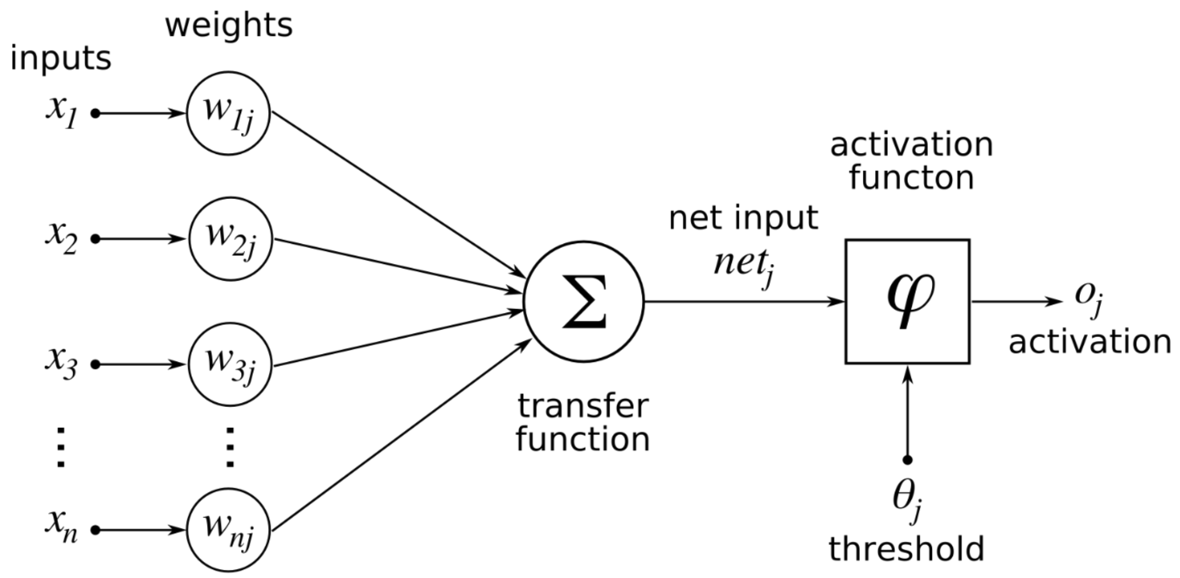

Figure 3.7 *ANNs Structure [53].*

Note: *'' The neuron calculates the **sum** of its inputs **x**, weighted by the weights **w**, then this value passes through the activation function **phi** to produce its output **o** ''.*

Beyond this simple structure, the neural network can also contain loops which radically change its possibilities but also its complexity. Just as loops can transform combinatorial logic into sequential logic, loops in a neural network transform a simple input recognition device into a complex machine capable of all kinds of behaviours.

3.7.1.2 Combination function

Consider any neuron, it receives a number of values from upstream neurons via its synaptic connections, and it produces a certain value using a combination function. This function can therefore be formalized as being a vector-to-scalar function [53], in particular:

- *Multi-layer perceptron (MLP) type networks compute a linear combination of inputs, i.e., the combination function returns the dot product between the vector of inputs and the vector of synaptic weights.*
- *RBF (radial basis function) networks calculate the distance between the inputs, i.e., the combination function returns the Euclidean norm of the vector resulting from the vector difference between the input vectors.*

3.7.1.3 Activation function

The activation function (or thresholding function, or even transfer function) is used to introduce a non-linearity into the functioning of the neuron.

Threshold functions generally have three intervals:

- *below the threshold, the neuron is inactive (often in this case, its output is 0 / -1);*
- *around the threshold, a transition phase;*
- *above the threshold, the neuron is active (often in this case, its output is 1).*

There are many activation functions, for instance:

- *The sigmoid function.*
- *The hyperbolic tangent function.*
- *The function of Heaviside.*

3.7.1.4 Information dissemination

Once this calculation is done, the neuron propagates its new internal state on its axon. In a simple model, the neural function is simply a thresholding function: it is equal to 1 if the weighted sum exceeds a certain threshold; 0 otherwise.

In a richer model, the neuron works with real numbers (often in the range [0.1] / [-1.1]). We say that the neural network goes from one state to another when all its neurons recalculate their internal state in parallel, according to their inputs [53].

3.7.2 Deep Learning

It's a set of machine learning methods attempting to model with a high level of data abstraction through articulated architectures of different nonlinear transformations. These techniques have enabled significant and rapid progress in the fields of sound or visual signal analysis and in particular facial recognition, voice recognition, computer vision and automated language processing. In the 2000s, this progress prompted significant private, academic and public investment, particularly from GAFAM (Google, Apple, Facebook, Amazon, Microsoft).

Deep learning techniques constitute a class of machine learning algorithms that:

- *Use different layers of nonlinear processing unit for feature extraction and transformation, each layer takes as input the output of the previous one, algorithms can be supervised or unsupervised, and their applications include pattern recognition and statistical classification.*
- *Operate with learning at multiple levels of detail or data representation; through the different layers, we move from low-level parameters to higher-level parameters, where the different levels correspond to different levels of data abstraction.*

3.8 Conclusion

Through this chapter, we have tried to present briefly the field of word segmentation, Arabic word forms, its features, then we made an overview of the existing approaches and methods, followed by a definition the field of our approach used, that is Artificial Neural Networks.

The following part will be devoted to the design of the word segmentation system, this part will be marked by a comparison between our model proposed and the old model, and a comparison with the state-of-the-art Arabic word segmenters. Our system will be based on a statistical approach, specifically Bi-LSTM-CRF approach.

Chapter 3

Bi-LSTM-CRF for Arabic Word Segmentation

Content

4 Bi-LSTM-CRF for Arabic Word Segmentation

4.1 Introduction

Word segmentation is indeed a very good solution, it is used to identify the limits of words in the sentence, i.e., the different morphemes. In order to POS tagging, we need to a good segmenter, for group the correct units of words, then the system can will predict the correct POS tag of each unit.

In our work, we use a deep learning approach based on a recurrent neural network (RNN), specifically bi-directional long short-term memory (Bi-LSTM) coupled with CRF, which performs sequence-to-sequence mapping to guess word segmentation, and to solve the problem of Arabic word segmentation with rewriting, we use dictionaries with rules.

4.2 Bi-LSTM-CRF

4.2.1 LSTM

Long Short-Term Memory (LSTM) is a recurrent neural network (RNN) architecture used in the field of deep learning. Unlike forward propagating neural networks, the LSTM has feedback connections.

LSTMs allow RNNs to remember their inputs over a long period of time. This is because LSTMs hold their information in a memory, which is very similar to computer memory because the LSTM can read, write, and delete information from its memory [54].

This memory can be seen as a gated cell, where gated means that the cell decides to store or delete information (for example whether it opens the doors or not), depending on the importance it assigns to the information. The attribution of importance is done through weights, which are also learned by the algorithm. It just means that it learns over time which information is important and which is not [54].

In an LSTM you have three gates: input, forget and output gate (see **Figure 4.1**). These gates determine whether or not to let a new entry enter (input gate, i_t), delete the information because it is not important (forget gate, f_t) or let it influence the exit at the current time step (output gate, o_t).

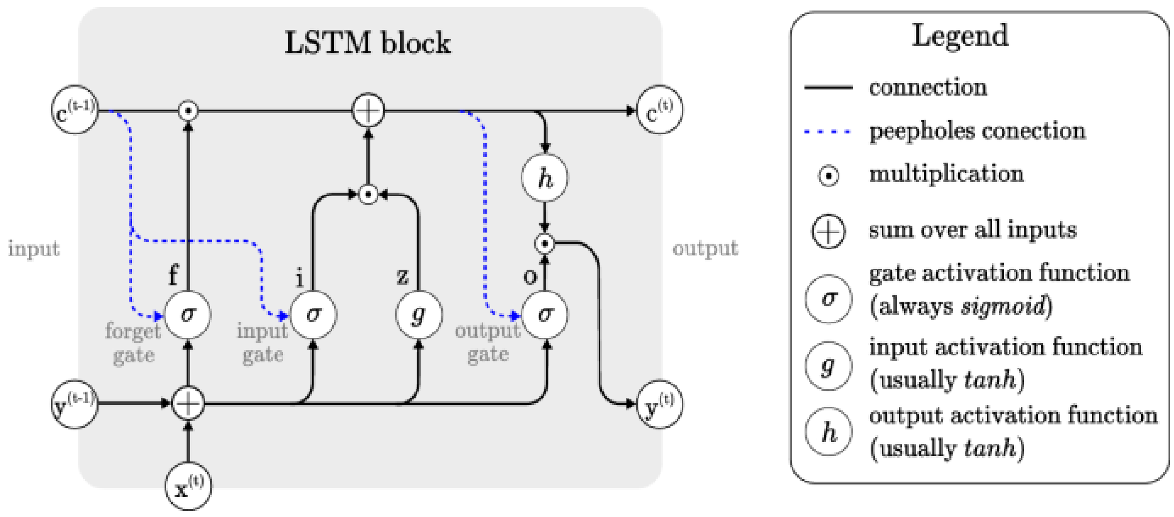

Figure 4.1 LSTM architecture [55].

Given an input sequence $x = (x_1, ..., x_n)$, an RNN computes the output vector y_t of each word x_t by iterating the following equations from $t = 1$ *to* n:

$$h_t = f(W_{xh} x_t + W_{hh} h_{t-1} + b_h) \quad (4.1)$$

$$y_t = W_{hy} h_t + b_y \quad (4.2)$$

Where h_t is the hidden states vector, W denotes weight matrix, b denotes bias vector and f is the activation function of the hidden layer.

The output of the LSTM hidden layer ht given input xt is computed via the following intermediate calculations [56]:

$$i_t = \sigma(W_{xi} x_t + W_{hi} h_{t-1} + W_{ci} c_{t-1} + b_i) \quad (4.3)$$

$$f_t = \sigma(W_{xf} x_t + W_{hf} h_{t-1} + W_{cf} c_{t-1} + b_f) \quad (4.4)$$

$$c_t = f_t c_{t-1} + i_t \tanh(W_{xc} x_t + W_{hc} h_{t-1} + b_c) \quad (4.5)$$

$$ot = \sigma(W_{xo} x_t + W_{ho} h_{t-1} + W_{co} c_t + b_o) \quad (4.6)$$

$$h_t = o_t \tanh(c_t) \quad (4.7)$$

4.2.2 Bi-LSTM

The Bi-LSTM network [57] is an extension of a single LSTM network. In BI-LSTM architecture, each hidden layer of the forward LSTM sequence is stacked with a sequence of backward LSTM cells. As illustrated in **Figure 4.2**, they compute the forward hidden sequence $\overrightarrow{h}$, the backward hidden sequence $\overleftarrow{h}$ and the output sequence y by iterating the backward layer from $t = T$ *to* 1:

$$\overrightarrow{h_t} = \sigma(W_{x\overrightarrow{h}}\, x_t + W_{\overrightarrow{h}\overrightarrow{h}}\, h_{t-1}) + b_{\overrightarrow{h}} \qquad (4.8)$$

$$\overleftarrow{h_t} = \sigma(W_{x\overleftarrow{h}}\, x_t + W_{\overleftarrow{h}\overleftarrow{h}}\, h_{t-1}) + b_{\overleftarrow{h}} \qquad (4.9)$$

$$y_t = W_{\overrightarrow{h}y}\,\overrightarrow{h_t} + W_{\overleftarrow{h}y}\,\overleftarrow{h_t} + b_y \qquad (4.10)$$

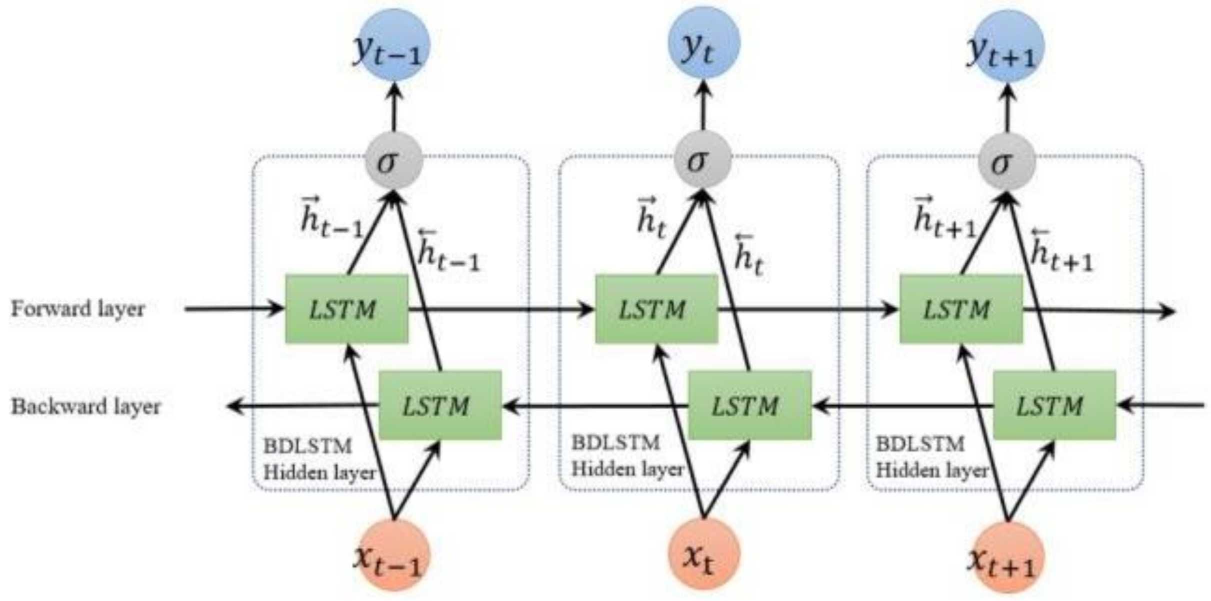

Figure 4.2 *Bi-LSTM architecture [58].*

4.2.3 CRF

Conditional random fields (CRFs) are a class of statistical models used in pattern recognition and more generally in statistical learning. CRFs allow to take into account the interaction of variables «neighbours», they are often used for sequential data (NLP, biological sequences, computer vision, etc.).

CRFs are an example of an undirected probabilistic network. This is a model type version discriminating Markov random fields generally presented as discriminative models, i.e., we seek to model the conditional probability ***p (X | Y)***, ***X*** being the observations and ***Y*** the variables to be estimated, instead of ***p (X, Y)***. A CRF is defined by Lafferty, et al. [59] as following:

Let $G = (V, E)$ to be a graph. Y is indexed by the nodes (v for vertices) of G: $Y = (Y_v)_{v \in V}$

Then (X, Y) is a conditional random field if the random variables Y_v conditioned to X obey the Markov property: $p\,(Y_v \mid X, Y_w, w \neq v) = p\,(Y_v \mid X, Y_w, w \sim v)$, where $w \sim v$ means that w and v are neighbours in G. For example, X might range over natural language sentences and Y range over POS tagging's of those sentences, with alphabet $\mathcal{Y}$ the set of possible POS tags [59].

During the training phase of the Bi-LSTM networks, the resulting probability distributions for different time steps are independent from each other. To overcome the independence assumptions imposed by the Bi-LSTM and to exploit these kind of labelling constraints in our

Arabic segmentation system, we model label sequence logic jointly using Conditional Random Fields (CRF) [59].

4.3 Our system

4.3.1 RNN model

In our approach we consider the segmentation as a sequence labelling problem at character level. For labelling we use two methods, the first is inspired by *Eldesouki et al.* [52], each char is labelled with one of five labels {B, M, E, S, WB} that define the segmentation decision boundaries of a word, where Beginning, Middle, End of a multi-character segment, Single character segment, and Word Boundary respectively. While the second method we removed WB from the sequence, to increase the probability of rest labels. The **Figure 4.3** show the RNN model.

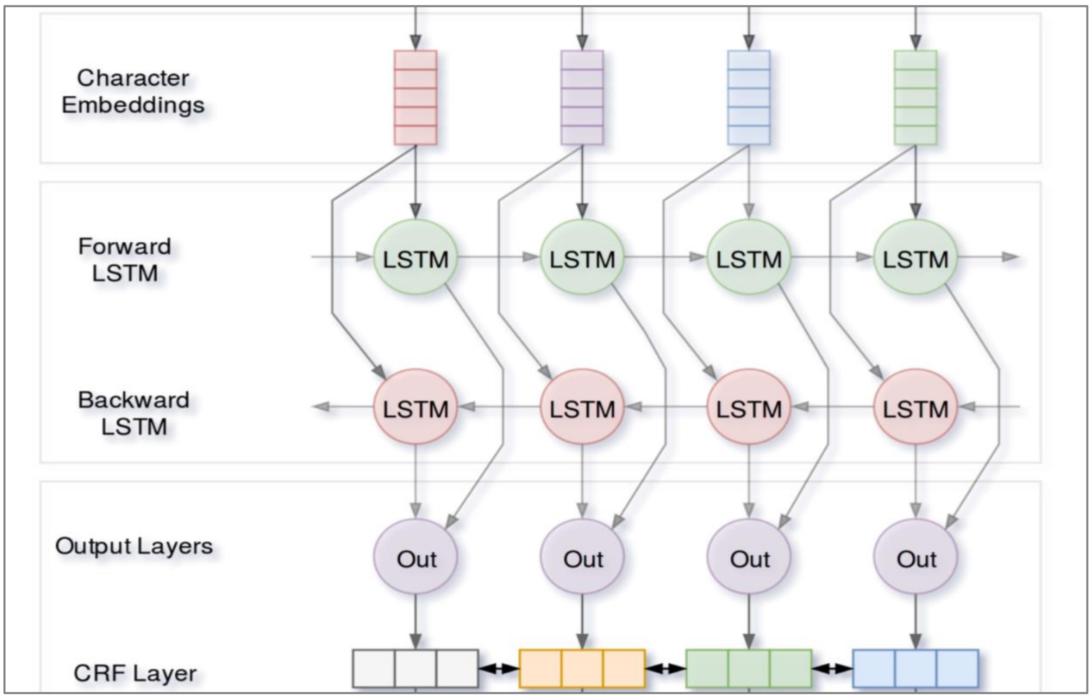

Figure 4.3 *Architecture of BI-LSTM-CRF model [52].*

In addition, on the first method the segmentation is based on the weight of word in each sentence, because the WB is considered a char from it, but in the second method it's based on the weight of word in all words of dataset. Then we'll compare between the both results (see section 4.4.5).

The model takes each word as its current input and predict the correct sequence labelling for segmentation. Our system consists of three main layers:

4.3.1.1 Input layer

Each input word is represented as a dense vector, containing character embeddings, we use Word2Vec method of Word-Embedding to encoding each input. a look-up table is initialized with randomly uniform sampled embeddings mapping each character in the input to d-dimensional vector, the **Table 4.1** shows an example of a lock-up table, and the **Figure 4.4** illustrates an example of an input representation.

Table 4.1 A lock-up table instance.

Char	Code	Char	Code
ب	0	أ	*7*
ق	1	*EMP*	*8*
ح	2	*UNK*	*9*
ت	3	…	…
ج	4		
م	5		
د	6		

أحمدY [أ, ح, م, د, Y] ➔ [7, 2, 5, 6, 9]

Figure 4.4 Example of input representation from a lock-up table.

4.3.1.2 Hidden layer

After encoding the input word, the following layer is a Bi-LSTM system, which maps character representations to hidden sequences, to obtain a fixed-dimensional representations for each character.

4.3.1.3 Output layer

At the output layer, a CRF is applied over the hidden representation of the Bi-LSTM to obtain the probability distribution over all the labels.

4.3.2 Optimisation

To make sure that our system learns significant representations, we add a dropout layer to mitigate overfitting, the basic idea behind dropout involves randomly omitting a certain percentage of the neurons in each hidden layer for each presentation of the samples during training. This encourages every neuron to depend less on the other neurons to learn the correct segmentation decision boundaries.

We apply dropout masks to the character embedding layer before inputting to the Bi-LSTM and to its output vector. with a fixed rate of 0.5 is decreases overfitting and improves the overall performance of our system.

Table 4.2 shows the main configuration of the best settings that were used for the two experiments in this thesis.

Table 4.2 *Basic network configuration for the both methods.*

Settings	Method 1 (5 label)	Method 2 (4 label)
N° hidden layer (dropout)	2	2
Dropout value	0.5	0.5
Batch size	64	64
Epochs	(50,150)	(50,150)
Steps per epoch	(50, None)	(50, None)
Embedding name	Char_emb	Word_emb & Char_emb
LSTM dim	--	--
Embedding dim	200	200
Input length	--	--
Optimiser	RMSprop	RMSprop
Learning rate	--	--
Early stopping	10	10
Model loss	crf.sparse_loss	crf.sparse_loss
Metrics	sparse_categorical_accuracy	sparse_categorical_accuracy

The empty cases, it's will be selected at later (see section 4.4.4).

4.3.3 Word and char embedding

4.3.3.1 Word-Embedding

It's a class approaches for representing words and documents using a dense vector representation. It is an improvement over more the traditional bag-of-word model encoding schemes where large sparse vectors were used to represent each word or to score each word within a vector to represent an entire vocabulary. These representations were sparse because the vocabularies were vast and a given word or document would be represented by a large vector comprised mostly of zero values.

Instead, in an embedding, words are represented by dense vectors where a vector represents the projection of the word into a continuous vector space. The position of a word within the vector space is learned from text and is based on the words that surround the word when it is used. The position of a word in the learned vector space is referred to as its embedding [60].

Two popular examples of methods of learning word embeddings from text include:

- *Word2Vec.*
- *GloVe.*

In addition to these carefully designed methods, a word embedding can be learned as part of a deep learning model. This can be a slower approach, but tailors the model to a specific training dataset.

4.3.3.2 Char-Embedding

Beside word embeddings more and more approaches use char-based embeddings. This kind of embeddings has been found useful for morphologically rich languages and to handle the out-of-vocabulary (OOV) problem for tasks, e.g., in part-of-speech (POS) tagging, language modelling, dependency parsing or named entity recognition. Zhang, Zhao, and, LeCun presented one of the first approaches to sentiment analysis with char embedding using convolution networks [60].

4.3.4 Keras libraries

Keras runs on top of open-source machine libraries like TensorFlow, Theano or Cognitive Toolkit (CNTK). Theano is a python library used for fast numerical computation tasks. TensorFlow is the most famous symbolic math library used for creating neural networks and deep learning models. TensorFlow is very flexible and the primary benefit is distributed computing. CNTK is deep learning framework developed by Microsoft. It uses libraries such as Python, C#, C++ or standalone machine learning toolkits. Theano and TensorFlow are very powerful libraries but difficult to understand for creating neural networks.

Keras is based on minimal structure that provides a clean and easy way to create deep learning models based on TensorFlow or Theano. Keras is designed to quickly define deep learning models. Well, Keras is an optimal choice for deep learning applications.

It's provides a complete framework to create any type of neural networks. Keras is innovative as well as very easy to learn. It supports simple neural network to very large and complex neural network model [61].

4.3.4.1 Architecture of Keras

Keras API can be divided into three main categories (see **Figure 4.5**):

- *Model.*
- *Layer.*
- *Core Modules.*

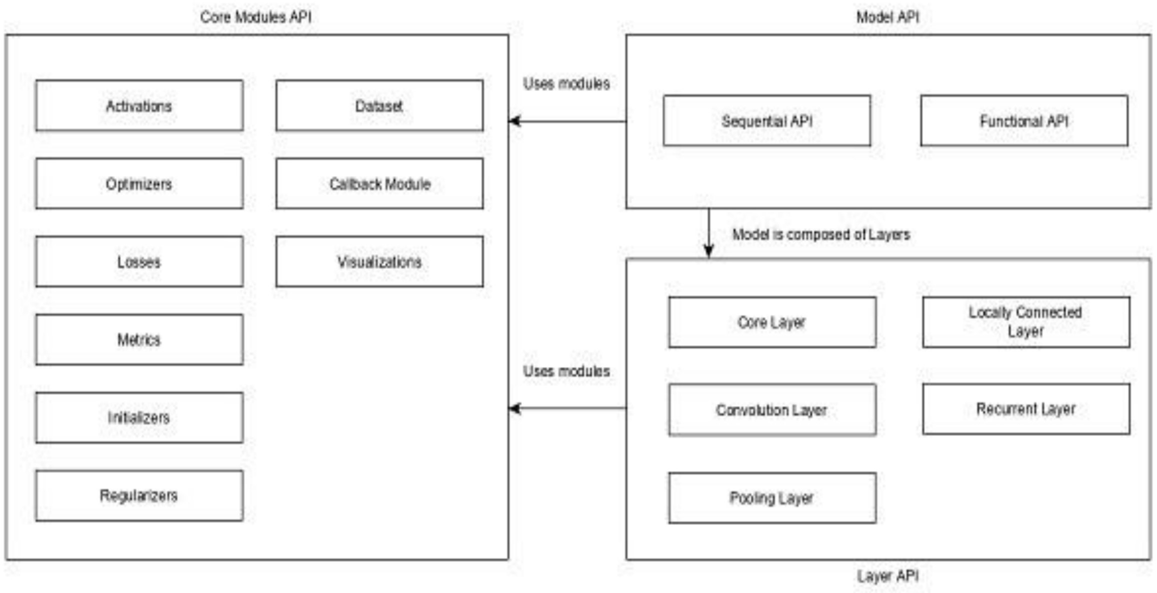

Figure 4.5 *Keras architecture [61].*

As learned earlier, Keras model represents the actual neural network model. It's provides a two mode to create the model, simple and easy to use Sequential API as well as more flexible and advanced Functional API.

4.3.4.1.1 Sequential

The core idea of ***Sequential API*** is simply arranging the Keras layers in a sequential order and so, it is called Sequential API. Most of the ANN also has layers in sequential order and the data flows from one layer to another layer in the given order until the data finally reaches the output layer.

4.3.4.1.2 Functional

Sequential API is used to create models layer-by-layer. ***Functional API*** is an alternative approach of creating more complex models. Functional model, you can define multiple input or output that share layers. First, we create an instance for model and connecting to the layers to access input and output to the model.

4.3.4.2 Keras Bi-LSTM model

The idea of Bidirectional RNNs is straightforward. It involves duplicating the first recurrent layer in the network so that there are now two layers side-by-side, then providing the input sequence as-is as input to the first layer and providing a reversed copy of the input sequence to the second. i.e., split the state neurons of a regular RNN in a part that is responsible for the positive time direction (forward states) and a part for the negative time direction (backward states).

The use of bidirectional LSTMs may not make sense for all sequence prediction problems, but can offer some benefit in terms of better results to those domains where it is appropriate.

> *We have found that bidirectional networks are significantly more effective than unidirectional ones…* [62]

Bi-LSTMs are supported in Keras via the ***Bidirectional*** layer wrapper. This wrapper takes a recurrent layer (e.g., the first LSTM layer) as an argument.

It also allows you to specify the merge mode, that is how the forward and backward outputs should be combined before being passed on to the next layer.

4.3.4.3 Keras-CRF

The Keras-CRF-Layer module implements a linear-chain CRF layer for learning to predict tag sequences. This variant of the CRF is factored into unary potentials for every element in the sequence and binary potentials for every transition between output tags.

4.3.4.4 Keras-Embedding

Keras offers an Embedding layer that can be used for neural networks on text data. It requires that the input data be integer encoded, so that each word is represented by a unique integer. This data preparation step can be performed using the Tokenizer API also provided with Keras.

The Embedding layer is initialized with random weights and will learn an embedding for all of the words in the training dataset. It is a flexible layer that can be used in a variety of ways, such as:

- ❖ *It can be used alone to learn a word embedding that can be saved and used in another model later.*
- ❖ *It can be used as part of a deep learning model where the embedding is learned along with the model itself.*
- ❖ *It can be used to load a pre-trained word embedding model, a type of transfer learning.*

The Embedding layer is defined as the first hidden layer of a network. It must specify 3 arguments:

- ❖ **Input dim**: *This is the size of the vocabulary in the text data. For example, if the data is integer encoded to values between 0-10, then the size of the vocabulary would be 11 words.*

- ❖ **Output dim**: *This is the size of the vector space in which words will be embedded. It defines the size of the output vectors from this layer for each word. For example, it could be 32 or 100 or even larger. Test different values for the problem.*
- ❖ **Input length**: *This is the length of input sequences, as you would define for any input layer of a Keras model. For example, if all of the input documents are comprised of 1000 words, this would be 1000.*

4.3.5 System Architecture

The functional implementation of our system is represented by the **Figure 4.6**.

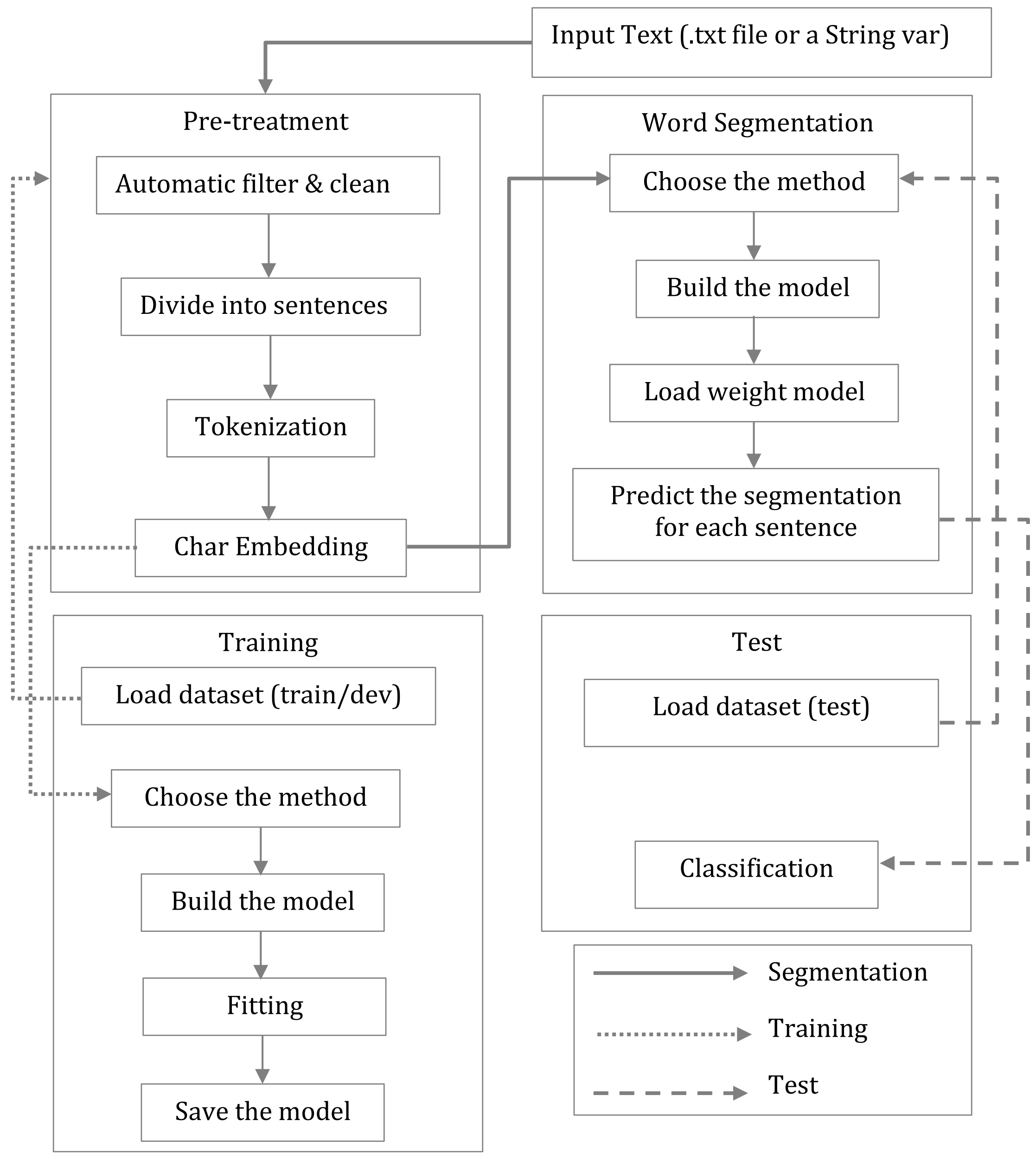

Figure 4.6 *Our system architecture.*

4.3.6 Pre-treatment modules

This phase seeks to standardize the text in order to make its use easier.

4.3.6.1 Automatic filter & clean

This step involves removing special and unknown characters, and digits and removing diacritics in the case of partially vowel words.

4.3.6.2 Tokenization

Before to apply this phase on the text, we segmenting the text into sentences using punctuation marks and language expressions acting as borders. such as;

- *The reverse comma « ، ».*
- *The inverted semicolon « ؛ ».*
- *The inverted question mark « ؟ ».*
- *The final point « . ».*

The elements of the border list do not always play a border role. For this reason, we propose a solution for the both method, which to define a maximum length of sentences, which is fixed at between (50, 100) words for each sentence in the second method, while the first use the maximum length of sentence in the corpus.

After this module, the tokenization seeks to transform each sentence into a series of individual tokens. In the idea, each token represents a word, and identifying words seems like a relatively simple task.

4.3.7 Main modules

4.3.7.1 Creation the dataset

This step consists to convert the Holy Quran text are tagged morphologically into a char labelled for a classical Arabic dataset, we follow a form general of word (proclitics + (noun/verb) + enclitics) in our dataset, to define the boundaries of morphemes.

4.3.7.2 Choose the method

This module is a necessary process which make to choose which method is in charge of the following process such as training, test, or Word Segmentation.

4.3.7.3 Build the model & configuration

It consists to create a model sequential which contains three layers, input layer: Embedding, hidden layer: Bigenerational (LSTM), output layer: CRF. Thus, it consists in fixing the value of dropout, batch size, and max length, and number of label (5 or 4).

4.3.7.4 Word segmenter

4.3.7.4.1 Load weight of model

After each training, we are saved the model into json file, to fast loaded on the next time when segmenting or testing. This process which makes our system very fast to test and very fast to segmenting.

4.3.7.4.2 Predict the right segmentation

After loading the weight of the chosen model, after char-encoding the input text, then we can predict the right segmentation for each word given (see **Figure 4.7**).

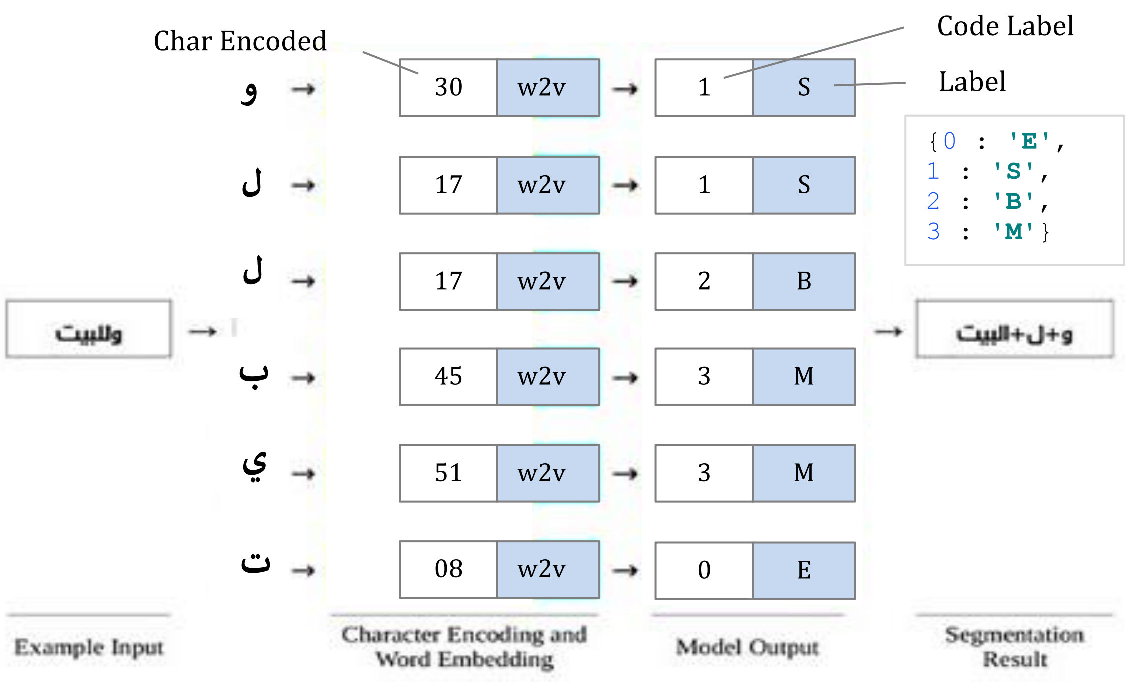

Figure 4.7 An example prediction for segmentation.

4.3.8 Model Fit (training)

Understanding model fit is important for understanding the root cause for poor model accuracy. This understanding will guide you to take corrective steps. We can determine whether a predictive model is underfitting or overfitting the training data by looking at the prediction error on the training data and the evaluation data.

The goal of fitting is to build a model capable of predicting output for unknown data away from the training or development datasets. In order to achieve this goal, must be avoid overfitting or underfitting (see **Figure 4.8**).

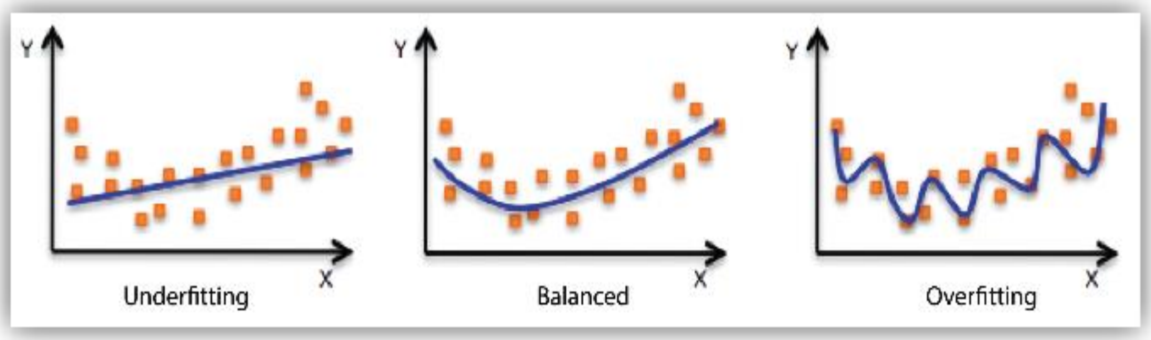

Figure 4.8 Types of model fit.

4.3.8.1 Overfitting

If the model performs poorly on the training data. The model is overfitting the dataset this is because the model is unable to capture the relationship between the input examples (often called X) and the target values (often called Y).

4.3.8.2 Underfitting

In underfitting, the model performs well on the training data but does not perform well on the evaluation data. This is because the model is memorizing the data it has seen and is unable to generalize to unseen examples.

Poor performance on the training data could be because the model is too simple (the input features are not expressive enough) to describe the target well. Performance can be improved by increasing model flexibility.

4.4 Experiments and results

4.4.1 Dataset

The dataset used in our experiments was a holy Quran text is tagged morphologically. It contains approximately 8K sentences that include approximately 111K words. For the experiments, we split the data into three datasets: training (60%), development (20%), and evaluation (20%).

The **Table 4.3** shows all most clitics in the corpus and their counts. The count may not be very precise, because of the ambiguity of some clitics. However, these numbers provide an excellent estimation of the distribution of all the clitics in the corpus.

Table 4.3 Counts of most clitics in corpus.

Clitics	Count	%	Example
'و+ '	2481	25.17 %	و+قد
'ل+' ، '+ل+'	1132	11.46 %	ل+كلام
'ب+'، '+ب+'	592	6.00 %	ب+مسجد
'+ه'	912	9.25 %	توحيد+ه
'+ها'	427	4.33 %	توحيد+ها
'+هم'	878	8.91%	توحيد+هم
' ف+'	1024	10.39 %	ف+لله
'+نا'	526	5.33 %	دين+نا
'ك+'، '+ك+'، '+ك'	464	4.70 %	ك+مشكاة
'+ما'	30	0.30 %	عند+ما
'+هما'	71	0.72 %	في+هما
'+ي'	150	1.52 %	إسلام+ي
'+ني'	127	1.29 %	يحزن+ني
'+كم'	627	6.36 %	دين+كم
'+ا'	85	0.86 %	أنّ+ا
'+هن'	116	1.77 %	مستقبل+هن
'+كن'	9	0.09 %	كيد+كن
'+كما'	19	0.19 %	في+كما
'أ+'	186	1.89 %	أ+أنذرت
Total	9896		

4.4.1.1 Data format

We extracted the segmentation data and stored it in text files, sentence by sentence separated by ***EOS*** line, the file contained a number of ***EOS*** equal to number of sentences, which contained a number of lines equal to the number of characters in the sentence. In each line, we stored the input vector data and output label.

The output label is the tag of the current character, which can be a label from {B, M, E, S}.

4.4.1.2 Rewrite information

Character rewrite information is not provided in the dataset; the information in the dataset only marks the split positions, we identified most of the rewrite cases in the corpus and used a lexicon technique with a dictionary to correct these issues.

Table below shows a several examples for all the eight rewrite cases corrected in our system.

Table 4.4 Cases rewriting with examples.

Cases	Examples
The « لل»	ل+ال+مدرس → للمدرس | ل+ال+متقين → للمتقين
The « للّ»	ل+ال+لاجئين → للاجئين | ل+الله → لله
The « ي»	أنجى+نا+كم → أنجيناكم | على+هم → عليهم
The « ت»	أداة+ه → أداته | ل+دلالة+ها → لدلاتها
The « و»	ف+ادعوا+ه → فادعوه
The « ا»	سمى+ه → سماه | مدى+ه → مداه
The « آ»	رأى+ه → رآه
The « ئ/ ؤ»	مساء+ك → مساؤك | مساء+ك → مسائك

4.4.2 Development environment

In this section, we present the programming language Python 3 used and the PyCharm environment. In order to achieve our objective; we took the initiative to operate and implement our program on the version: Microsoft Windows 10 Pro, and in order to carry out this project, we used a set of materials whose main characteristics are as follows:

- ***Processor:*** Intel(R) Core (TM) i5-4210H CPU @ 2.90 ~ 3.40GHz, 2 Core(s), 4 Logical Processor(s).
- ***RAM***: 8.00 GB
- ***GPU***: NVidia 940m, 6 GB

4.4.2.1 Python 3

4.4.2.1.1 Definition

Python is the most widely used open-source programming language among computer scientists. This language has propelled itself to the forefront of infrastructure management, data analysis or software development.

Indeed, among its qualities, Python allows developers to focus on what they do rather than how they do it. It freed developers from the form constraints that occupied their time with older languages. Thus, developing code with Python is faster than with other languages [63].

4.4.2.1.2 What is the Python language for?

The main uses of Python by developers are:

- *Programming applications.*
- *The creation of web services.*
- *Code generation.*
- *Metaprogramming.*

4.4.2.1.3 Version

Technically, this language will mainly be used for scripting and automation (interaction with web browsers). There are two versions: Python 2 and Python 3. Python 2, the old version offers updates until 2020. Python 3 is the current version. Its interpreter is more efficient, as is its concurrency control [63].

For our system, we used the version 3.9 which is the stable one, and RNNs packages which corresponds to our approach, such as: ***TensorFlow*** [64], ***Keras*** [61].

4.4.2.2 PyCharm

PyCharm is an integrated development environment used for programming in Python. It allows code analysis and contains a graphical debugger. It also allows the management of unit tests, integration of version management software, and supports web development with Django.

Developed by the Czech company JetBrains, it is a cross-platform software that runs on Windows, Mac OS X and Linux. It's available in a professional edition, distributed under a proprietary license, and in community edition distributed under the Apache license [65].

4.4.3 Measure of quality

In the fields of pattern recognition, information retrieval and automatic classification, ***precision*** (or positive predictive value) is the proportion of relevant items among the set of items proposed; ***recall*** (or sensitivity) is the proportion of relevant items proposed among all relevant items. These two concepts thus correspond to a conception and a measure of relevance.

Where *precision* can thus be understood as a measure of accuracy or quality, while *recall* is a measure of completeness or quantity.

Precision is used with *recall*, the percent of all relevant documents that is returned by the search. The two measures are sometimes used together in the *F1-Score* (or *F-measure*) to provide a single measurement for a system [66] (see **Figure 4.9**).

4.4.3.1 Precision

It's the fraction of retrieved documents that are relevant to the query.

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|} \tag{4.11}$$

For instance, for a text search on a set of documents, precision is the number of correct results divided by the number of all returned results.

4.4.3.2 Recall

It's the fraction of the relevant documents that are successfully retrieved.

$$\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|} \tag{4.12}$$

For instance, for a text search on a set of documents, recall is the number of correct results divided by the number of results that should have been returned.

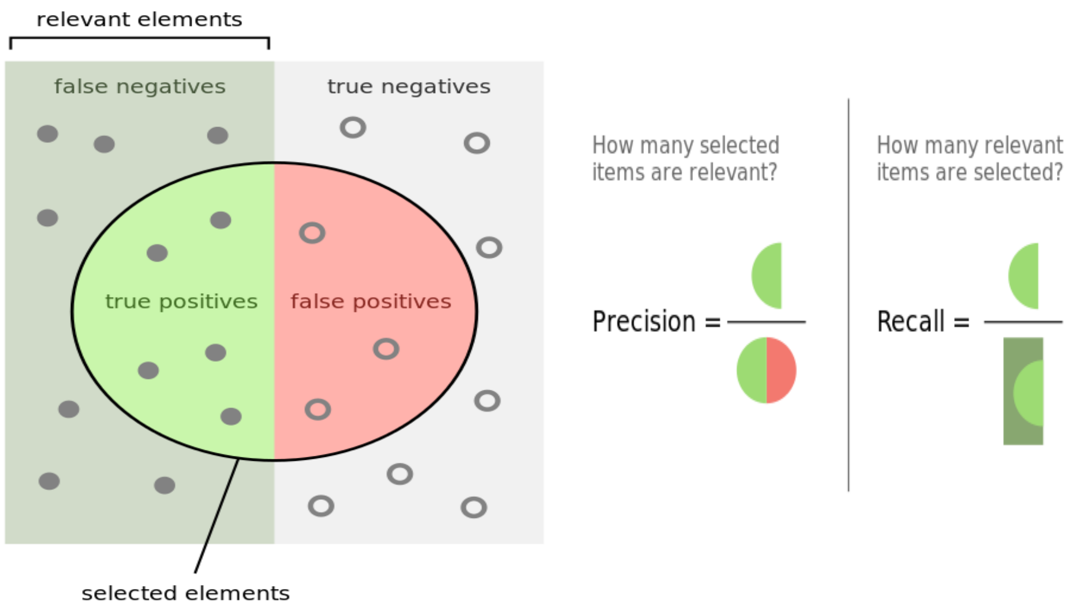

Figure 4.9 Precision and recall [66].

4.4.3.3 Accuracy

Accuracy has two definitions:

- *More commonly, it is a description of systematic errors, a measure of statistical bias; low precision results in a difference between a result and a "true" value.* ***ISO*** *calls it trueness.*
- *Alternatively,* ***ISO*** *defines accuracy as describing a combination of both types of random and systematic measurement error, so high accuracy requires both high precision and high trueness* [67].

It's also used as a statistical measure of how well a binary classification test correctly identifies or excludes a condition. That is, the accuracy is the proportion of correct predictions (both true positives and true negatives) among the total number of cases examined. The formula for quantifying binary accuracy is:

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn} \quad (4.13)$$

4.4.3.4 F-measure

A measure that combines precision and recall is the harmonic mean, called F-measure or F-score, it's also known as the F_1 metric because precision and recall are weighted equally. The formula for F_1 is:

$$F = 2 \cdot \frac{(\text{précision} \cdot \text{rappel})}{(\text{précision} + \text{rappel})} \quad (4.14)$$

4.4.4 Network Configuration

To select the best configuration for RNN model (second method), we lance some tests based on three important settings, such as:

- *Embedding input length (see 4.3.4.4).*
- *Learning rate: It controls the step-size in updating the weights.*
- *LSTM dimension.*

Before we choose the final configuration of our method to compare it with another segmenters, we will reveal the results that allowed us to choose the best settings values, and compare them, we use for each test the same optimisation as mentioned above (see 4.3.2), and the corpus used for evaluation is the dataset test (20 %).

4.4.4.1 Embedding input length

The table below shows different tests with different values for the input length parameter of the *keras embedding*.

Table 4.5 *Comparison between input length of Keras Embedding.*

Score / Tests	**Test 1**	**Test 2**	**Test 3**	**Test 4**
Value input length	10	50	80	100
Char-based Accuracy	96.9 %	98.5 %	**98.6 %**	98.3 %
Word-based Accuracy	94.2 %	96.9 %	**97.2 %**	96.5 %

We had made the comparison between results of Embedding input length in the figure bellow, it illustrates the histogram of accuracy values for two evaluation char-based and word-based for each test.

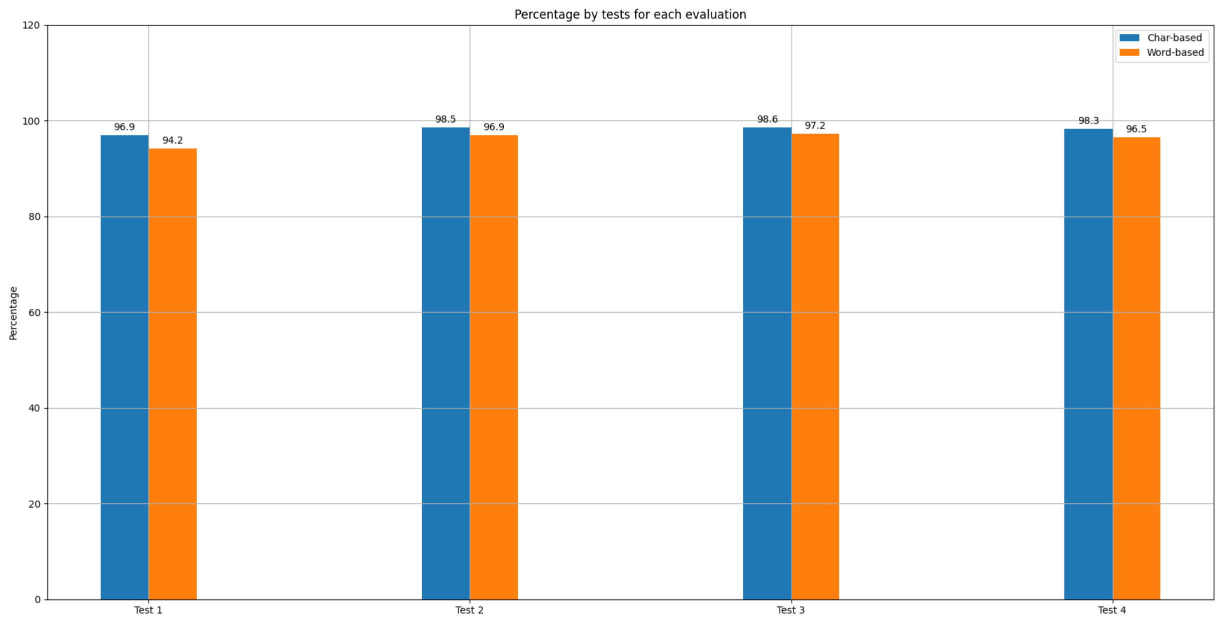

Figure 4.10 *Histogram of both accuracy values for each test.*

4.4.4.2 Learning rate

The table below shows different tests with different values for the learning rate parameter of the *RMSprop optimizer*.

Table 4.6 *Comparison between values of learning rate.*

Score / Tests	Test 1	Test 2	Test 3	Test 4
Value learning rate	0.01	0.005	0.001	0.0001
Min loss	0.0239	0.0215	0.0212	0.2596
Sparse categorical accuracy	93.02 %	94.20 %	94.28 %	92.89 %
Char-based Accuracy	97.2 %	97.2 %	**97.9 %**	94.3 %
Word-based Accuracy	94.1 %	93.4 %	**95.4 %**	84.5 %

We had made the comparison between results of learning rate in two figures, the first **Figure 4.11** illustrates the loss values per iteration for each test, while the second **Figure 4.12** illustrates the variation of sparse categorical accuracy.

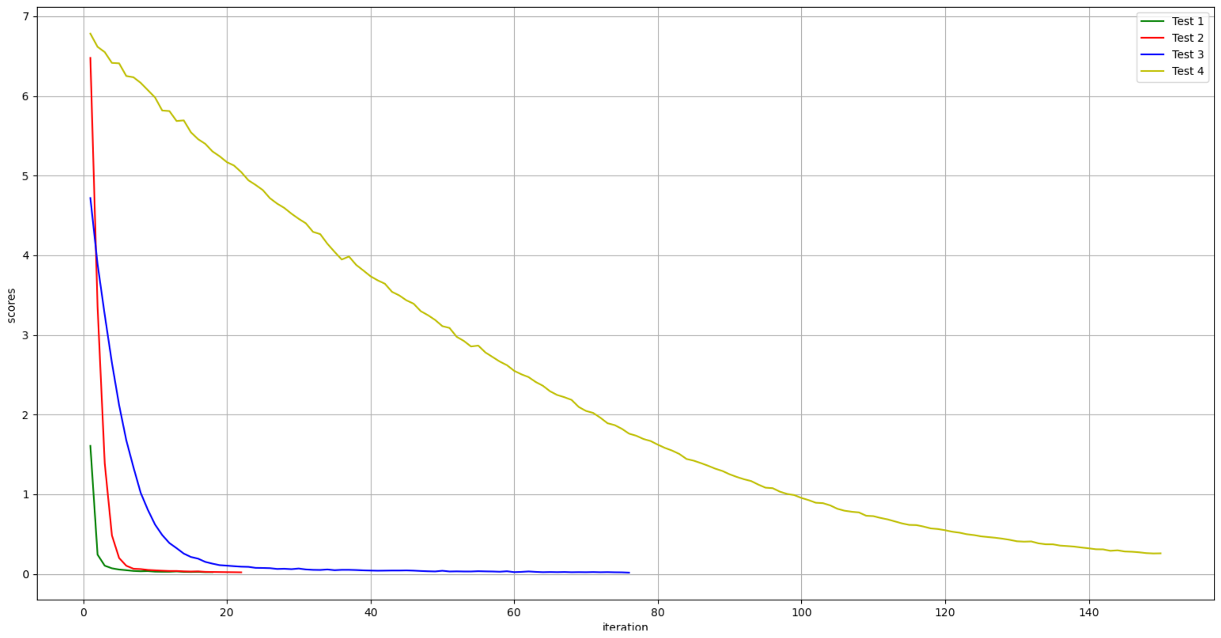

Figure 4.11 *Chart of variation loss per iteration (learning rate).*

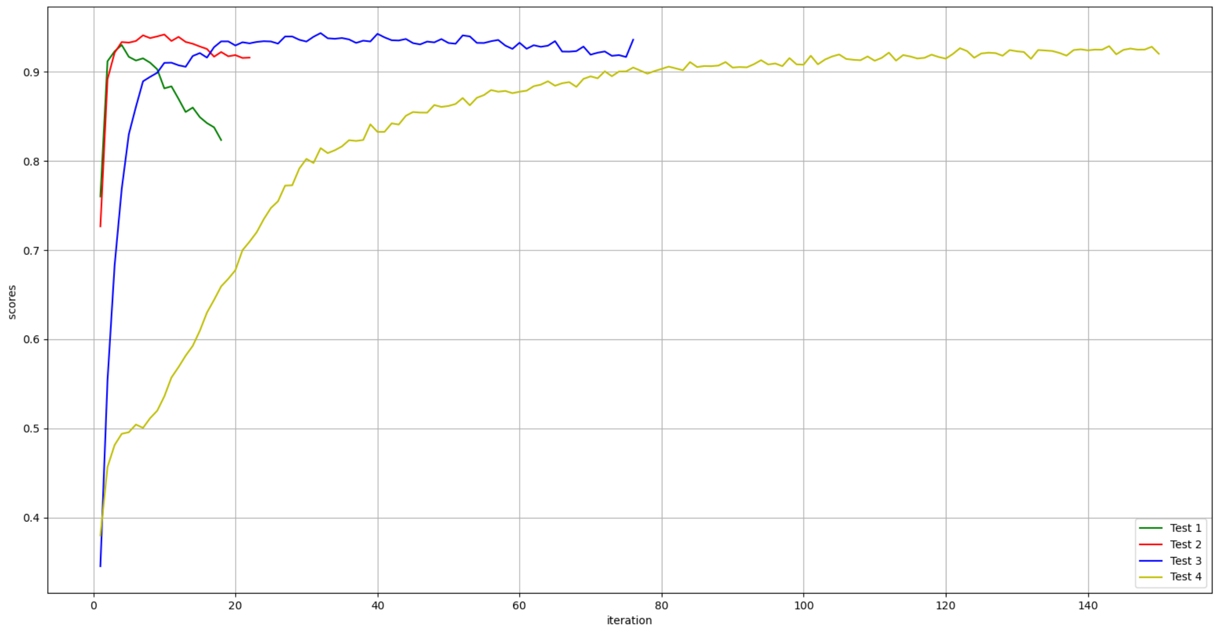

Figure 4.12 *Variation of sparse categorical accuracy per iteration (learning rate).*

4.4.4.3 LSTM dimension

The table below shows different tests with different values for the LSTM dimension parameter of the *Keras* model.

Table 4.7 Comparison between different values of the LSTM dimension.

Score / Tests	Test 1	Test 2	Test 3	Test 4
Value LSTM dim	50	100	200	400
Min loss	0.0259	0.0173	0.0176	0.0286
Sparse categorical accuracy	92.72 %	94.20 %	95.25 %	92.46 %
Char-based Accuracy	96.3 %	97.2 %	97.8 %	**98.2 %**
Word-based Accuracy	92.3 %	93.4 %	95.0 %	**96.0 %**

We had made the comparison between results of LSTM dimension test in two figures, the first **Figure 4.13** illustrates the loss values per iteration for each test, while the second **Figure 4.14** illustrates the variation of sparse categorical accuracy.

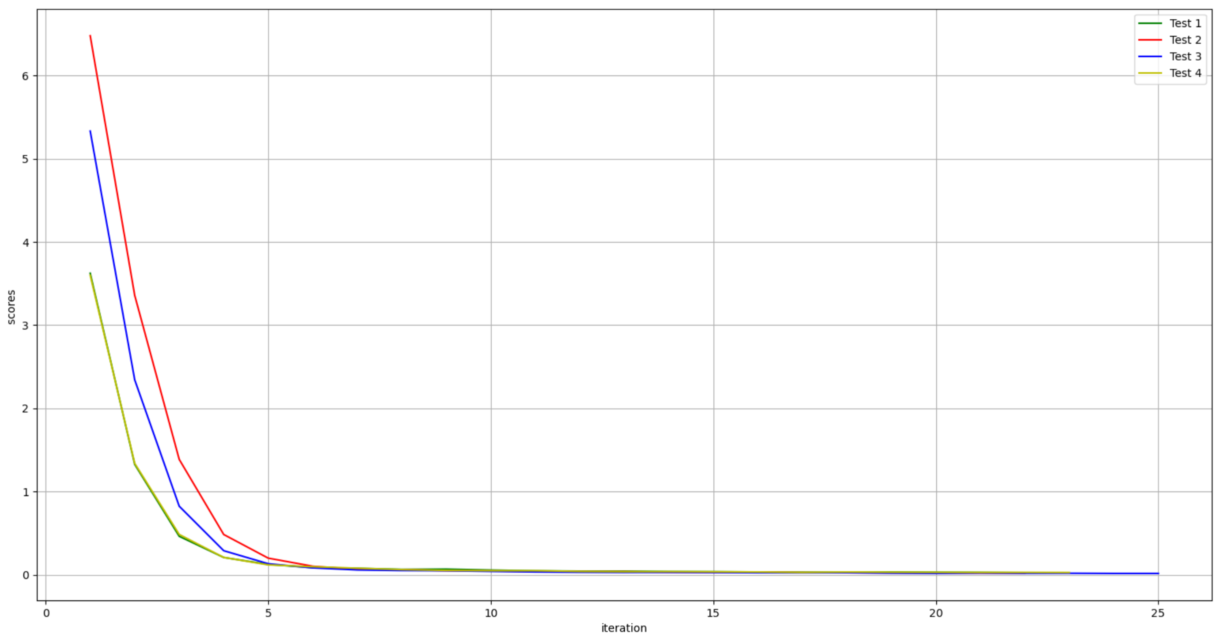

Figure 4.13 Chart of variation loss per iteration (LSTM dim).

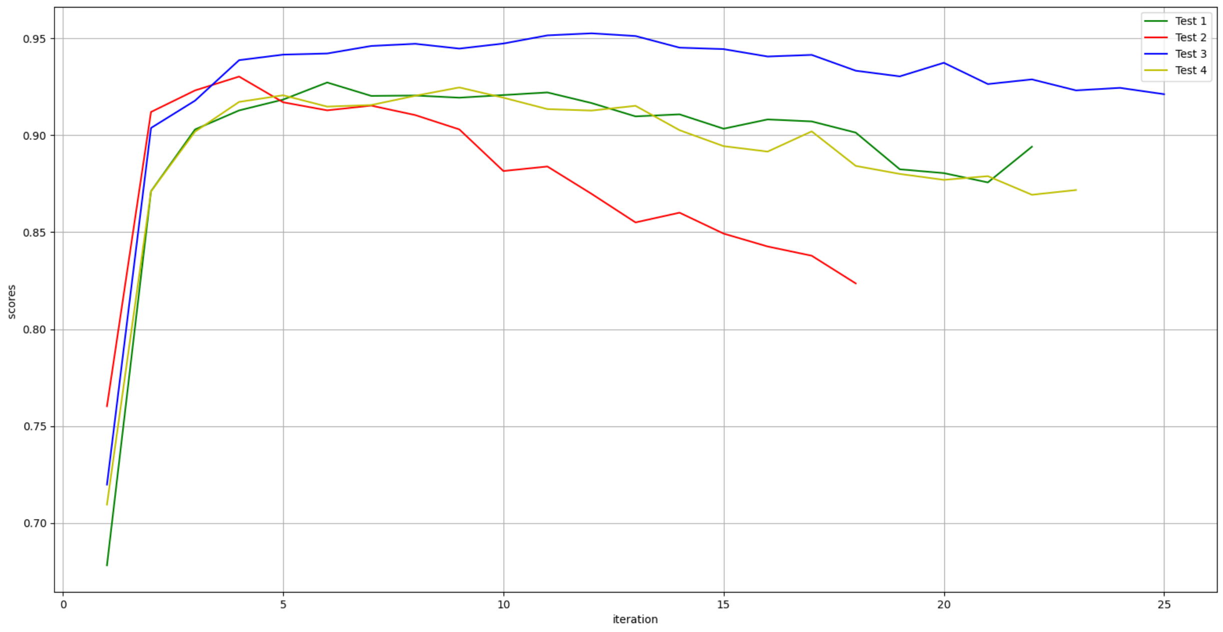

Figure 4.14 Chart of variation sparse categorical accuracy per iteration (LSTM dim).

4.4.4.4 Best configuration

After the different results of several testing as illustrated in previous section, we have found the best configurations for our method as shown by the table below.

Table 4.8 Values of best settings.

Settings	Best values
Epochs	50
Steps per epoch	None
Word Embedding dim	200
Embedding Input length	80
Learning Rate	(0.001, 0.01)
LSTM dim	(200, 400)

4.4.5 Comparison between both methods

In this section, we compare between both methods used, with the best settings for the both, and we have run four tests with different size of dataset test, table below shows all configuration and results for each method.

Table 4.9 Settings and Results of several tests.

Settings		Met 1 (5 labels)	Met 2 (4 labels)
LSTM dim		200	200
Embed Input length		200	200
Learning rate		0.01	0.01
Test 1 (5%)	Char-Based Accuracy	97.4 %	**98.7 %**
	Word-Based Accuracy	92.8 %	**97.3 %**
Test 2 (10%)	Char-Based Accuracy	97.3 %	**98.6 %**
	Word-Based Accuracy	92.6 %	**97.3 %**
Test 3 (20%)	Char-Based Accuracy	97.4 %	**98.7 %**
	Word-Based Accuracy	93.0 %	**97.4%**
Test 4 (30%)	Char-Based Accuracy	97.3 %	**98.7 %**
	Word-Based Accuracy	92.7 %	**97.3 %**

We had made the comparison of accuracy values for each label in each test, the **Figure 4.15** illustrates this variation.

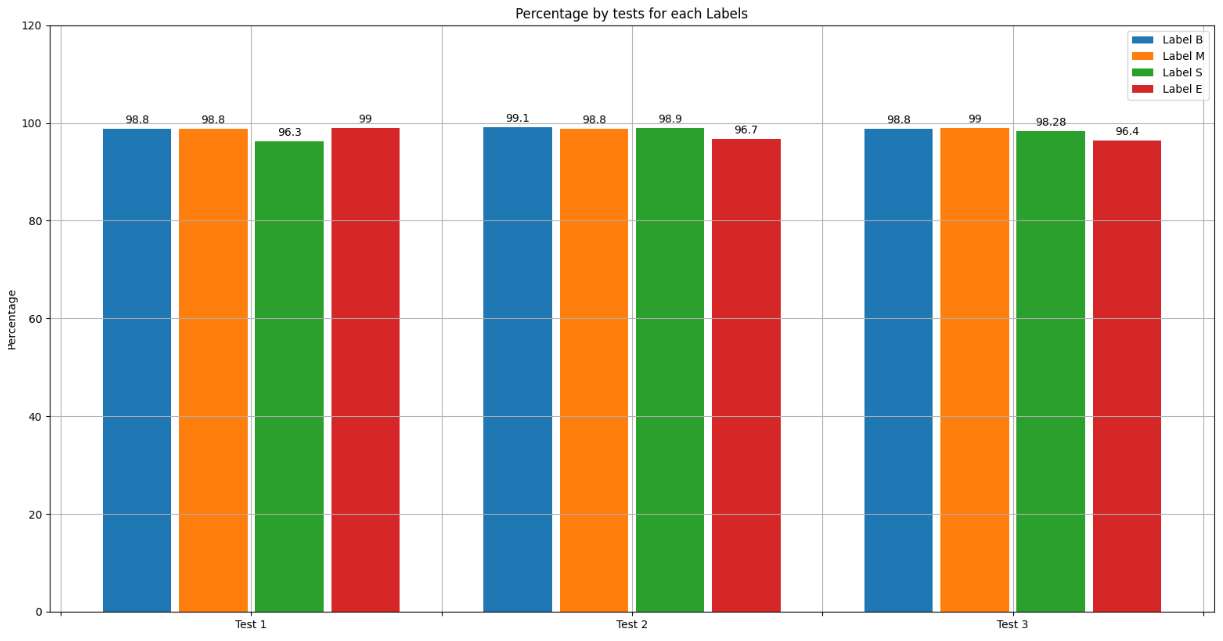

Figure 4.15 Accuracy values for each label in each test.

4.4.6 Comparison with state-of-the-art of the Arabic word segmenters

In this section, we compare our proposed method with the main and state-of-the-art methods in the literature. Most of these methods, including in this study, were trained and/or tested using ATB.

The results were compared with the following state-of-the-art methods:

- *Farasa* [9]
- *Seg-Eldesouki* [10]
- *Our segmenter (second method)*

For this experiment, we chose a text made its as difficult as possible. The text was a classical Arabic advisory opinion by Al-Imam Abdul-Aziz Ibn Baz (رحمه الله), it contained different vocabulary and syntax patterns that may never have been seen in the training data, which contains 481 tokens with 278 segmentation cases. So, we preformed the required word segmentations manually and used them for comparison. After we ran the test using the three methods, the results showed in table below.

Out of the 278 cases, there were 27 segmentation cases with errors based on one or more methods. Farasa at last with 33 segmentation errors. Seg-*Eldesouki* ranked second with 29 errors. and our proposed method achieved a best accuracy with 7 segmentation errors (see **Table 4.10**), the **Table 4.11** shows the unique errors for each method of the first test.

Table 4.10 *Results of the comparison.*

Methods	**Farasa**	**Seg-*Eldesouki***	**Our segmenter**
Accuracy	93.13 %	93.97 %	**98.54 %**

[9] Online at Farasa (qcri.org)
[10] Segmenter inspired by *Eldesouki et al.* [52].

Table 4.11 *Results of test « Unique segmentation errors are shown for each method ».*

Methods	Unique Segmentations Errors	Total Errors
Farasa	*علي+ه، ال+حفا+ة، ال+عرا+ة، يكثرن، رب+ت+ها ال+صلا+ة، ال+نبو+ة، جه+ة، ممن، أيض+ا و+ثني+ة، فيستقر*	*33*
Seg-Eldesouki	*علي+ه، ل+فظ، ربت+ها، و+لدت يعن+ي، بن+تا، اب+نا، و+قع، ال+معاص+ي الفتن، أ+ما، بعدما، ينت+هي، فل+سطين*	*29*
Our Method	*و+لدت، ل+فظ، س+لم، اب+نا، و+ح+كم، و+ثنية يح+كم*	7

4.5 Conclusion

In this chapter, we have described the process of building our application by specifying the development environment, the libraries used and the approach we adopted to carry out our project. Then to evaluate our system, we run a several tests to choose a best network configuration, this allowed us to calculate the three metrics (recall, precision and measure) in order to obtain more precise measurements of the quality of our summary.

5 General Conclusion

Throughout this thesis, we have addressed the issue of segmentation of Arabic words which is considered as a POS tagging problem, followed by NLP techniques for divide word into units which are morphemes (clitics, affixes, stem).

To implement the segmentation system, we proposed a statistical approach, which is a Bi-LSTM RNN coupled with the CRF model, during the segmentation process, based on the detection the limits of each morpheme in each word.

We consider that we have succeeded in achieving a large part of the objectives of this project « despite the time criterion », and that we have made good choices regarding the implementation tools, therefore our work will constitute a very good lead for other future projects.

For future work, we want to perform domain adaptation using large MSA data, such as ATB, to improve segmentation results. Further, we plan to investigate building a new model capable of segmenting the Arabic word with rewriting without use any dictionary or lexicon with minimal loss in accuracy.

Bibliography

[1] L. Audibert, "Traitement Automatique du Langage Naturel (TALN), Outils d'analyse de données textuelles.," Université Paris 13 – Laboratoire d'Informatique de Paris-Nord., France, 2010.

[2] X. Tannier, "Extraction et recherche d'information en langage naturel dans des documents semi-structurés," Universié de Jean Monnet, Thése de doctorat, France, 2006.

[3] K. Shaalan and A. Farghaly, "Arabic Natural Language Processing: Challenges and Solutions.," *ACMTransactions on Asian Language Information Processing,* vol. 8, no. 14, 2009.

[4] Y. Jaafar and K. Bouzoubaa, "Arabic Natural Language Processing from Software Engineering to Complex Pipeline," in *First International Conference on Arabic Computational Linguistics (ACLing)*, 2015.

[5] "Semitic languages," [Online]. Available: https://en.wikipedia.org/wiki/Semitic_languages. [Accessed 16 06 2021].

[6] J. Akesson, Arabic Morphology and Phonology: Based on the Marāḥ Al-arwāḥ by Aḥmad Ibn 'Alī Ibn Mas'ūd., Leiden: Brill, 2011.

[7] IONOS, "Le traitement automatique du langage naturel (TALN)," Digital Guide, 09 04 2019. [Online]. Available: https://www.1and1.fr/digitalguide/web-marketing/vendre-sur-internet/le-traitement-automatique-du-langage-naturel-taln/. [Accessed 12 04 2021].

[8] B. Bigi, "TALN Informatique.," CLIPS- Equipe GEOD, ITC, 2006.

[9] F. Carton, Introduction à la phonétique du français, Dunod, Ed., Paris: Bordas, 1974.

[10] F. Katamba, "An introduction to phonology," *Journal of the International Phonetic Association,* vol. 22, no. 1-2, pp. 70-72, 1989.

[11] H. P. Matthews, "Morphology," Cambridge University Press, 1991.

[12] J. R. Hurford and al., “Semantics : A Coursebook,” 1983.

[13] S. C. Levinson, “Pragmatics,” Cambridge University Press, 1983.

[14] Ibn-Manzûr, “Lisàn al-'arab (1232-1311)”.

[15] N. Y. Habash, “ Introduction to Arabic natural language processing (Synthesis lectures on human language technologies).,” *Machine Translation,* vol. 24, pp. 285-289, 2010.

[16] W. Zaghouani, “Le développement de corpus annotés pour la langue arabe.,” Thèse de doctorat en Sciences du langage, France, 2015.

[17] J. Owens, “Arabica, The Syntactic Basis of Arabic Word Classification,” pp. 211-234, 1989.

[18] F. A.-N. M. Muhammad, “Concept du nom, ses caractéristiques, ses types et ses représentations pour les grammairiens,” 2016.

[19] K. Dukes, “Statistical Parsing by Machine Learning,” The University of Leeds, School of Computing, PhD thesis, 2013.

[20] B. Salih, al-i’rāb al-mufaṣṣal likitāb allāh al-murattal, Beirut: Dar Al-Fikr, 2007.

[21] A. Ben Gharbia, “Les schémas de phrase en arabe et en français,” *Syntaxe et Sémantique,* pp. 49-72, 1 2011.

[22] J. Dichy, “On lemmatization in Arabic, A formal definition of the Arabic entries of multilingual lexical databases.,” pp. 23-30, 08 2001.

[23] K. F. Hammouda and A. A. Almarimi, “Heuristic Lemmatization for Arabic texts Indexation and Classification.,” *Journal of Computer Science,* pp. 660-665, 2010.

[24] A. A. Neme and E. Laporte, “Pattern-and-root inflectional morphology: the Arabic broken plural.,” vol. 40, pp. 221-250, 2013.

[25] L. E. Neme A. A., “Les informaticiens et la morphologie computationnelle arabe.,” 2011.

[26] Cambridge, "Cambridge dictionary," [Online]. Available: https://dictionary.cambridge.org/fr/dictionnaire/anglais/segmentation. [Accessed 10 06 2021].

[27] D. Jurafsky and J. Martin, Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, Prentice Hall, 2008.

[28] A. Aliwy, "Tokenization as Preprocessing for Arabic Tagging System," *International Journal of Information and Education Technology.,* vol. 2, no. 4, pp. 348-353, August 2012.

[29] A. ALMUHAREB and al., "Arabic Word Segmentation With Long Short-Term Memory Neural Networks and Word Embedding," vol. 7, January 21, 2019.

[30] N. Y. Habash, "Introduction to Arabic Natural Language Processing Synthesis Lectures on Human Language Technologies," Morgan and Claypool Publishers, 2010.

[31] S. Khoja, "APT: Arabic part-of-speech tagger," Ph.D. dissertation, Comput. Dept., Lancaster Univ., Lancaster, U.K., 2003.

[32] C. B. O. Zribi, A. Torjmen and M. B. Ahmed, "A multi-agent system for POS-tagging vocalized Arabic texts.," *Int. Arab J. Inf. Technol,* vol. 4, no. 4, p. 322–329, 2007.

[33] S. Alqrainy, H. M. AlSerhan and A. Ayesh, "Pattern-based algorithm for part-of-speech tagging Arabic text," Nov, 2008.

[34] A. Al-Taani and S. A. Al-Rub, "A rule-based approach for tagging non-vocalized Arabic words," *Int. Arab J. Inf. Technol.,* vol. 6, no. 3, p. 302–328, 2009.

[35] M. Hadni, S. A. Ouatik, A. Lachkar and M. Meknassi, "Hybrid part-of-speech tagger for non-vocalized Arabic text," vol. 2, no. 6, p. 1–15, 2013.

[36] S. AlGahtani, W. Black and J. McNaught, "Arabic part-of-speech tagging using transformation-based learning," in *Proc. 2nd Int. Conf. Arabic Lang Resour. Tools,*, Cairo, Egypt, Apr, 2009.

[37] S. Kulick, “Exploiting separation of closed-class categories for Arabic tokenization and part-of-speech tagging,” *ACM Transactions on Asian Language Information Processing,* vol. 10, no. 1, pp. 1-18, 2011.

[38] A. S. Fatma and A. Guessoum, “A hidden Markov model-based POS tagger for Arabic,” in *Proc. 8th Int. Conf. Stat. Anal. Textual Data*, Besançon, France, 2006.

[39] M. Diab, K. Hacioglu and D. Jurafsky, “Automatic tagging of Arabic text: From raw text to base phrase chunks,” in *Proc. HLT-NAACL*, Boston, MA, USA, 2004.

[40] M. Diab, “Second generation AMIRA tools for Arabic processing: Fast and robust tokenization, POS tagging, and base phrase chunking,” in *Proc. 2nd Int. Conf. Arabic Lang. Resour. Tools*, Cairo, Egypt, 2009.

[41] A. Pasha and al., “MADAMIRA: A fast, comprehensive tool for morphological analysis and disambiguation of Arabic,” in *Proc. LREC*, Reykjavik, Iceland, 2014.

[42] K. Darwish, A. Abdelali and H. Mubarak, “Using stem-templates to improve Arabic POS and gender/number tagging,” in *Proc. LREC*, Reykjavik, Iceland, 2014.

[43] A. Bosch, E. Marsi and A. Soudi, “Memory-based morphological analysis and part-of-speech tagging of Arabic,” vol. 38, p. 201–217, 2007.

[44] M. Abdul-Mageed, M. Diab and S. Kübler, “ASMA: A system for automatic segmentation and morpho-syntactic disambiguation of MSA,” in *Proc. Recent Adv. Natural Lang. Process*, Hissar, Bulgaria, 2013.

[45] S. Khalifa, N. Zalmout and N. Habash, “Yamama: Yet another multi-dialect Arabic morphological analyzer,” in *Proc. 26th Int. Conf. Comput. Linguistics, Syst. Demonstrations (COLING)*, Osaka, Japan, 2016.

[46] M. Diab, N. Habash, O. Rambow and R. Roth, “MADA+TOKAN A toolkit for Arabic tokenization, diacritization, morphological disambiguation, POS tagging, stemming and lemmatization,” April, 2009.

[47] N. Habash, R. Roth, O. Rambow, R. Eskander and N. Tomeh, “Morphological Analysis and Disambiguation for Dialectal Arabic,” in *The 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HTL'13)*, Atlanta, USA, 2013.

[48] N. Habash and O. Rambow, “Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop,” in *Proc. 43rd Annu. Meeting Assoc. Comput. Linguistics, Lansing*, MI, USA, 2005.

[49] R. Roth, O. Rambow, N. Habash, M. Diab and C. Rudin, “Arabic morphological tagging, diacritization, and lemmatization using lexeme models and feature ranking,” in *Proc. 46th Annu. Meeting Assoc. Comput. Linguistics Hum. Lang. Technol.*, Jun, 2008.

[50] A. A. Freihat, G. Bella, H. Mubarak and F. Giunchiglia, “A single-model approach for Arabic segmentation, POS tagging, and named entity recognition,” in *Proc. 2nd Int. Conf. Natural Lang. Speech Process*, Algiers, Algeria, Apr. 2018.

[51] N. Zalmout and N. Habash, “Don’t throw those morphological analyzers away just yet: Neural morphological disambiguation for Arabic,” in *Proc. Conf. Empirical Methods Natural Lang. Process*, Copenhagen, Denmark, 2017.

[52] M. Eldesouki and al., “Arabic Multi-Dialect Segmentation: bi-LSTM-CRF vs. SVM,” 2017.

[53] “ANNs,” [Online]. Available: https://fr.wikipedia.org/wiki/R%C3%A9seau_de_neurones_artificiels#Structure_du_r%C3%A9seau. [Accessed 16 06 2021].

[54] H. Oumaima, “Réseaux neuronaux récurrents et LSTM,” DataScienceToday, 26 juin 2018. [Online]. Available: https://datasciencetoday.net/index.php/fr/machine-learning/148-reseaux-neuronaux-recurrents-et-lstm. [Accessed 31 5 2021].

[55] G. V. Houdtn and al., “A review on the long short-term memory model,” Springer Nature, 13 May 2020. [Online]. Available: https://link.springer.com/article/10.1007/s10462-020-09838-1. [Accessed 31 5 2021].

[56] A. Graves, “Generating sequences with recurrent neural networks,” arXiv preprint arXiv, 2013.

[57] M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks,” vol. 45, no. 11, p. 2673–2681, 1997.

[58] Z. Cui and Y. Wang, “Deep Stacked Bidirectional and Unidirectional LSTM Recurrent Neural Network for Network-wide Traffic Speed Prediction,” 2017.

[59] J. Lafferty, A. McCallum and F. Pereira, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data,” in *Proc 18th International Conf. on Machine Learning*, Morgan Kaufmann, 2001.

[60] Ł. Augustyniak and al., “ASPECT DETECTION USING WORD AND CHAR EMBEDDINGS WITH (BI) LSTM AND CRF,” in *2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, 2019.

[61] “Keras - Models,” [Online]. Available: https://www.tutorialspoint.com/keras/keras_models.htm. [Accessed 11 06 2021].

[62] A. Graves and J. Schmidhuber, “Framewise Phoneme Classification with Bidirectional LSTM and Other Neural Network Architectures,” in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks*, 2005.

[63] “Python : définition et utilisation de ce langage informatique,” [Online]. Available: https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1445304-python-definition-et-utilisation-de-ce-langage-informatique/. [Accessed 15 06 2021].

[64] “Tensorflow,” [Online]. Available: https://www.tensorflow.org/. [Accessed 15 06 2021].

[65] “PyCharm,” [Online]. Available: https://fr.wikipedia.org/wiki/PyCharm.

[66] “Precision & Recall,” [Online]. Available: https://fr.wikipedia.org/wiki/Pr%C3%A9cision_et_rappel. [Accessed 16 06 2021].

[67] “Accuracy (trueness and precision) of measurement methods and results — Part 1: General principles and definitions,” ISO, 1994. [Online]. Available: https://www.iso.org/obp/ui/#iso:std:iso:5725:-1:ed-1:v1:en. [Accessed 26 06 2021].