# Prediction of Punctuation Marks for Classical and Modern Standard Arabic

**Thesis** · January 2016

DOI: 10.13140/RG.2.1.4946.7287



**1 author:**

Ala'a Alkreshah
University of Jordan
**1** PUBLICATION **0** CITATIONS

SEE PROFILE

# PREDICTION OF PUNCTUATION MARKS FOR CLASSICAL AND MODERN STANDARD ARABIC

**By**
**Ala'a M. Al-sulman**

**Supervisor**
**Dr. Majdi S. Sawalha**

**Co-Supervisor**
**Dr. Sane M. Yagi, Prof**

**This Thesis was Submitted in Partial Fulfillment of the Requirements for the Master's Degree of Science in Information Systems**

**Faculty of Graduate Studies**
**The University of Jordan**

**Jan, 2016**

# Committee Decision

**This thesis/Dissertation (Prediction of Punctuation Marks for Classical and Modern Standard Arabic) Was Successfully Defended and Approved on 5/1/2016**

| **Examination Committee** | **Signature** |
|---|---|
| Dr. Majdi Shaker Sawalha (Supervisor)<br>Assist. Prof. of Computer Information Systems | ---------- |
| Prof. Sane Mohammed Yagi (Co-Supervisor)<br>Prof. of Computational Linguistic | ---------- |
| Dr. Bassam Hassan Hammo (Member)<br>Assoc. Prof. of Computer Information Systems | ---------- |
| Dr. Mohammad A. M. Abushariah (Member)<br>Assist. Prof. of Computer Information Systems | ---------- |
| Dr. Samer Mohammed Jamil Samarah(Member)<br>Assist. Prof. of Computer Information Systems<br>(Yarmouk University) | ---------- |

# Dedication

I dedicate this thesis to my family, to my father and mother who taught me how to work hard, to my brothers and my friends. Also, I dedicate this thesis to all the muslims in the world.

## Acknowledgment

First of all, I would like to express my sincere gratitude to my advisor Dr. Majdi Sawalha for this continuous support of my MS research, his patience, motivation, and interest. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my MS study.

In addition, I would like to thank Prof. Sane Yagi for his insightful and valuable comments. My sincere thanks also go to my colleges and friends for their motivations and prayers to Allah.

Last but not the least, I would like to thank my family specially my parents and brothers for their continuous support and interest all the time of my study.

# Table of Contents

# List of Tables

# LIST OF FIGURES

# List of Abbreviations

| | |
|---|---|
| BAQ | Boundary Annotated Qur'an |
| NLP | Natural Language processing |
| NLTK | Natural Language Toolkit |
| MSA | Modern Standard Arabic |
| ML | Machine Learning |
| MLE | Maximum Likelihood Estimation |
| CRF | Conditional Random Field |
| HMM | Hidden Markov Model |
| POS | Part Of Speech |
| 3-POS T | Three Part of Speech Tag (Coarse Annotation) |
| 10-POS T | Ten Part of Speech Tag (Fine Annotation) |
| BCR | Balanced Accuracy Rate |
| TP | True Positive |
| TN | True Negative |
| FP | False Positive |
| FN | False Negative |

# PREDICTION OF PUNCTUATION MARKS FOR CLASSICAL AND MODERN STANDARD ARABIC

By
**Ala'a M. Al-sulman**

Supervisor
**Dr. Majdi S. Sawalha**

Co-Supervisor
**Dr. Sane M. Yagi, Prof**

## Abstract

Arab writers have serious deficiencies in using punctuation marks. Linguists call for Arabic punctuation to be revised and the rules better-defined. Some believe that punctuation should be anchored in grammatical rules, but what level of competence should the language user possess to be able to correctly punctuate a piece of text? Can Machine Learning (ML) algorithms handle automatic punctuation annotation of Arabic texts? Which of the machine learning algorithms could be the best suited for this task? Could machine learning algorithms produce a model that would be used to punctuate any Arabic text? In attempt to answer these questions, three tagging algorithms have been experimented with here: Conditional Random Fields (CRF), N-gram and Hidden Markov Model (HMM). They have been trained and tested on the Boundary Annotated Qur'an (BAQ) Corpus after it was appended with Modern Standard Arabic (MSA) style punctuation annotation. The three ML algorithms have been tested on the corpus in the two conditions of coarse and fine annotation, using three Part Of Speech (3-POST): noun, verb and particle, and ten part of speech (10-POST): noun, nominal, verb, pronoun, et.al respectively. The results show that machine learning algorithms can perform automatic punctuation and that implementing the CRF model with coarse annotation can yield slightly better punctuation performance than fine annotation can. This supports the conclusion that the language user need not possess a high level of linguistic competence to learn how to correctly punctuate a piece of text. The CRF model scored 93.4% and 86.4% for the accuracy and Balanced Accuracy Rate (BCR) respectively with the 3-POS tags. On the other hand, the CRF model scored 92.2% and 85.8% for the accuracy and BCR metrics for the 10-POS tags respectively. Furthermore, it has been found out that training the CRF tagger on Classical Arabic benefits greatly its automatic punctuation marks prediction of Modern Standard Arabic texts. For the sentence terminal prediction task, the N-gram model has the best performance compared with the two other ML algorithms. The N-gram model scored 91.8% and 70.0% for the accuracy and BCR respectively with the 3-POS tags, while it scored 91.8 and 69.6% for the accuracy and the BCR metrics respectively with the 10-POS tags.

# Chapter 1

# Introduction

## 1.1 Thesis Overview

This research aims to investigate and argue that automatic punctuation annotation including sentence terminal prediction is possible using machine learning algorithms on Classical Arabic and Modern Standard Arabic (MSA) alike. In addition, this research aims to examine which machine learning algorithm is best for this task. It also seeks to establish which level of linguistic competence the Arabic language user needs to have to punctuate Arabic text correctly. Furthermore, this study aims to develop an automatic system for inserting punctuation marks and sentence terminals in the Holy Qur'an. Then, it will use the knowledge gained there to punctuate any written MSA text. Note that this research does not taking into account the grammatical rules of the Arabic in the experiments.

To achieve these goals, usage of modern punctuation marks was investigated. Then, machine learning algorithms were trained and tested on the Boundary Annotated Qur'an (BAQ) Corpus (Sawalha, et al., 2012) after some modification. Modern punctuation marks were extracted from سَيِّد قُطّب Sayyid Qutb في ظلال القران "*fī ẓilāl al-qur'ān*" (Qutb, 1991) and then inserted in the BAQ Corpus. The results of testing different machine learning algorithms were then studied and comparisons were conducted using standard evaluation metrics. Finally, the best model generated by machine learning algorithm that was trained on the Qur'an was used to annotate MSA texts with modern punctuation marks.

## 1.2 Background on Arabic Punctuation

Punctuation marks are special marks such as (comma “,”, full stop “.”, colon “:”, semicolon “;”, question mark “?”, exclamation mark “!”). They appear in texts to help readers correctly understand the contents. These marks are useful for enhancing interpretation of texts. They also define text divisions i.e. clauses, sentences, and paragraphs. Punctuation marks appeared 200 B.C in the Greek language. Modern English has a well-defined set of punctuation marks. Other languages also developed standards for using punctuation marks. Many researches were conducted to automatically punctuate texts of other languages such as Vietnamese, Chinese, and Japanese.

Adding punctuation marks to a text improves its readability (Beaglehole and Yates, 2010). In addition, they provide rich information useful for Natural Language Processing applications, such as syntactic analysis, segmentation, part of speech tagging, machine translation, information extraction, parsing and phrasing.

## 1.3 Punctuation Annotation Challenges in Arabic

In Arabic readers emphasize the meaning of texts using pitch accent when reading a text aloud. Pitch accented words help listeners understand the meaning of a text. In silent reading readers need punctuation marks to highlight the pitch accented words in the text, so that the reader can have better understanding of it (Gordon, 2014). Early on, Arabic linguists realized the importance of introducing punctuation marks to Arabic and defining the rules for using each mark. In 1917 punctuation marks were introduced to Arabic for the first time by Ahmad Zaki Basha (Zaki, 1930).

However, Arab readers and writers have serious deficiencies in using suitable punctuation marks (Khafaji, 2001). Therefore, punctuation marks for Arabic need to be revised and their usage rules better defined.

## 1.4 Using Punctuation Marks in the Holy Qur’an

The Holy Qur'an is the central religious text of Islam and the original text of the final revelation of Allah. The Holy Qur’an is an excellent source of ideas and it is a fertile ground for Islamic studies, and for the development of Arabic linguistic theory. The Qur'an is considered as a perfect gold standard text for developing, modeling, and evaluating Arabic NLP applications. The Qur'an as a corpus consists of 114 chapters which are made up of 6,243 verses and 77,430 words according to (Sawalha, et al., 2012). Adding punctuation marks to the Qur'anic text can help readers decompose longer verses into shorter sentences that are easier to understand their meaning. For example, inserting quotation marks indicates to the reader that speech is a direct speech. The Qur’an already has prosodic (stops and starts) marks that scholars inserted into the text. These marks help the reciter while reading the text of Qur’an. The locations of pause and start prosodic marks are meaning determined. We would talk more about these marks in a subsequent section. Table 1.4.1 shows two examples of punctuated Qur’anic texts. The first example includes 1 to 3 verses of Surat Al-Baqarah. The second shows verses 12 and 13 of Surat Al-Imran.

Table 1.4.1: Examples of punctuated text from the Qur'an

| Saheeh International translation | Punctuated Verses according to Sayyed Qutb | Not punctuated |
|---|---|---|
| Alif, Lam, Meem. This is the Book about which there is no doubt, guidance for those conscious of Allah. Who believe in the unseen, establish prayer, and spend out of what We have provided for them. | الم. ذَالِكَ الْكِتَابُ لَا رَيْبَ فِيهِ، هُدًى لِّلْمُتَّقِين. الَّذِينَ يُؤْمِنُونَ بِالْغَيْبِ، وَيُقِيمُونَ الصَّلَاةَ، وَمِمَّا رَزَقْنَاهُمْ يُنفِقُونَ. | الم ﴿١﴾ ذَالِكَ الْكِتَابُ لَا رَيْبَ ۛ فِيهِ ۛ هُدًى لِّلْمُتَّقِينَ ﴿٢﴾ الَّذِينَ يُؤْمِنُونَ بِالْغَيْبِ وَيُقِيمُونَ الصَّلَاةَ وَمِمَّا رَزَقْنَاهُمْ يُنفِقُونَ ﴿٣﴾ سورة البقرة (١-٣) |
| Say to those who disbelieve, "You will be overcome and gathered together to Hell, and wretched is the resting place.". Already there has been for you a sign in the two armies which met - one fighting in the cause of Allah and another of disbelievers. They saw them [to be] twice their [own] number by [their] eyesight. But Allah supports with His victory whom He wills. Indeed in that is a lesson for those of vision. | قُل لِّلَّذِينَ كَفَرُوا: سَتُغْلَبُونَ وَتُحْشَرُونَ إِلَىٰ جَهَنَّمَ وَبِئْسَ الْمِهَادُ. قَدْ كَانَ لَكُمْ آيَةٌ فِي فِئَتَيْنِ الْتَقَتَا: فِئَةٌ تُقَاتِلُ فِي سَبِيلِ اللَّهِ، وَأُخْرَىٰ كَافِرَةٌ، يَرَوْنَهُم مِّثْلَيْهِمْ رَأْيَ الْعَيْنِ. وَاللَّهُ يُؤَيِّدُ بِنَصْرِهِ مَن يَشَاءُ. إِنَّ فِي ذَٰلِكَ لَعِبْرَةً لِّأُولِي الْأَبْصَارِ. | قُل لِّلَّذِينَ كَفَرُوا سَتُغْلَبُونَ وَتُحْشَرُونَ إِلَىٰ جَهَنَّمَ ۚ وَبِئْسَ الْمِهَادُ ﴿١٢﴾ قَدْ كَانَ لَكُمْ آيَةٌ فِي فِئَتَيْنِ الْتَقَتَا ۖ فِئَةٌ تُقَاتِلُ فِي سَبِيلِ اللَّهِ وَأُخْرَىٰ كَافِرَةٌ يَرَوْنَهُم مِّثْلَيْهِمْ رَأْيَ الْعَيْنِ ۚ وَاللَّهُ يُؤَيِّدُ بِنَصْرِهِ مَن يَشَاءُ ۗ إِنَّ فِي ذَٰلِكَ لَعِبْرَةً لِّأُولِي الْأَبْصَارِ ﴿١٣﴾ سورة آل عمران (١٢-١٣) |

## 1.5 Research Motivation

Since punctuation marks are helpful for better understanding of any text (Gordon, 2014), this research aims to develop an automatic punctuation system that can insert punctuation marks into the Holy Qur'an using ML algorithms. The knowledge gained from this punctuation annotation would be used to punctuate text.

Having a new version of the BAQ Corpus annotated with punctuation marks will make a useful gold standard for machine learning algorithms. Machine learning algorithms applied to the BAQ Corpus will learn patterns (transferable knowledge) and then use these patterns (knowledge) to help in automatically punctuating MSA texts.

This study will help Muslims around the world to read and understanding the Qur'an. This study is a multidisciplinary study that contributes not only to Information Technology including Computational Linguistics, Language Engineering, and Machine Learning but also to other disciplines such as Islamic Studies, Linguistics, and Arabic language learning. Punctuation annotation transferable knowledge would be useful for many NLP applications, such as POS tagging, name entity recognition, phrasing, segmentation etc.

## 1.6 Research Aim and Objectives

The main aim of this research is to develop an automatic punctuation annotation system including sentence terminal prediction for Qur'anic text using ML algorithms. The transferable knowledge of punctuation annotation from the BAQ Corpus would then be used to punctuate any Arabic text.

To achieve this goal, we will study modern punctuation marks, annotate the BAQ Corpus with the modern punctuation marks used in سيد قُطّب Sayyid Qutb explanation في ظلال القران "*fī ẓilāl al-qur'ān*" (Qutb, 1991), train and test ML algorithms on the BAQ Corpus, compare between the produced results, and finally use the best BAQ trained ML algorithm to punctuate MSA texts. The BAQ Corpus consists of 77430 words including its linguistic features such as; part of speech tags (noun, verb, and pronoun et.al). The BAQ corpus structured of 8230 sentences of the Holy Qur'an.

The main objectives for this study as follows:

1. Surveying the research on punctuation marks prediction.
2. Modeling punctuation marks prediction and sentence terminal prediction on the Qur'anic text using ML algorithms.
3. Comparing the results offered by the ML algorithms.
4. Using the ML algorithm that performed best on the BAQ Corpus to punctuate a piece of MSA text with modern punctuation marks.
5. Evaluating the obtained results.

## 1.7 Research Methodology

The methodology that we will use for building an automatic punctuation annotation system relies on the use of ML algorithms. We will extract modern punctuation marks from "*fī ẓilāl al-qur'ān*" (Qutb, 1991), and then insert them into the BAQ Corpus. The BAQ Corpus then will be split into training and testing datasets and ML algorithms would be trained and tested on them. The obtained results would be compared and the best ML model would be used to punctuate MSA texts after training on the BAQ Corpus. We will discuss this methodology in detail in Section 3.1.

## 1.8 Thesis Structure

This thesis is structured as follows:

1. Chapter one: an overview of the thesis is presented. It defines the main aims and objectives of this research. A brief introduction of modern punctuation marks is made and punctuation challenges are identified. In addition, prosodic punctuation marks in the Holy Qur'an are discussed. Finally, the research motivation and its methodology are delineated.
2. Chapter two: describes modern punctuation marks and their usage in Arabic texts. A review for prosodic (Starts and Stops) marks and their usage in the Qur'an is presented. The usage of punctuation marks by Sayyid Qutb was discussed. A comparison between the Makki and Madani chapters was presented. A literature review for the researches of punctuation annotation was presented.
3. Chapter three: presents our methodology for building an automatic punctuation system for the Qur'an and MSA text, in addition, it discusses the methods and tools used for implementing the methodology i.e. NLTK, BAQ corpus and ML algorithms.
4. Chapter four: presents the design and the plan for implementing the experiments, declaring the problem of punctuation and sentence terminal prediction.
5. Chapter five: discusses the evaluation metrics used to measure the performance of the ML algorithms and the problem of skewed data. In addition, presents and discuss the results of applying the ML algorithms on the BAQ Corpus and MSA text for punctuation and sentence terminal prediction.
6. Chapter six: summarize the conclusion of this research and the future work.

# Chapter 2

# Literature Review

Several researches were tended to discuss the issues of punctuation annotation and sentence terminals prediction, due to the importance of these topics in many of Natural Language processing applications, such as; information extraction, segmentation, POS tagging, automatic speech recognition systems et al.

The conducted researches for different languages aimed to use machine learning algorithms for punctuation and sentence terminal prediction. One of the first researches was (Beeferman, et al., 1998) for English language, while for Arabic language many researches discussed the sentence terminal detection (Sawalha, et al., 2012) issue but not punctuation annotation.

This chapter addresses different topics; the difference between punctuation marks and its usage in Arabic language, the prosodic marks and its usage in the Qur'an and the difference between Madani and Makki chapters in the Qur'an, also it presents the usage of punctuation marks in the explanation of Sayyid Qutb. In addition, it surveys the researches of punctuation annotation and sentence terminal prediction experimented with different machine learning algorithms.

## 2.1 Punctuation Marks Overview

### 2.1.1 Punctuation Marks for Arabic

Readers need special tones and symbols in order to facilitate understanding and realizing written texts. While other nations (Foreigners) realized the importance of this task they were the first how proceeded to use special symbols aim to declare written

texts by separating sentences to enable the readers to diversifies his tones while reading in order to distinguish different type sentences such as; stops, starts, questions, etc. (Zaki 1930).

Arab language was suffering shortages in this area, as readers feel difficulty to understand the meanings and the purposes of written Arabic texts in the absence of such signs and symbols. From here the need to use special was revealed, as Arab language scientists proceeded to put some of these. One of the scientists was "Ahmad Zaki Basha", who proceeded to combine punctuation marks used in foreign languages and modifies them according the rules of Arabic language. These marks are: (Comma "،" الفاصلة, Semi Colon "؛" الفاصلة المنقوطة, Full Stop "." النقطة, Question Mark "؟" علامة الإستفهام, Exclamation Mark "!" علامة التعجب, Colon ":" النقطتان الرأسيتان, Ellipsis Mark "…" علامة الحذف, Hyphen mark "-" علامة الشرطة, Quotation mark " )) " علامة التنصيص, Parentheses mark " () "علامة القوسان ). Table 2.1.1.1 declares these punctuation marks with examples.

Table 2.1.1.1: Types of punctuation marks and its usage in Arabic language.

| # | Punctuation Mark | Usage | Example |
|---|---|---|---|
| **1** | Comma " ، " | The purpose is to keep silent reader for a little while to differentiate sentences from another and it's used between multiple clauses in compound sentences. | ذَلِكَ الْكِتَابُ لَا رَيْبَ فِيهِ،هُدًى لِّلْمُتَّقِينَ (٢) البقرة<br>This is the Book about which there is no doubt, a guidance for those conscious of Allah (2) |
| **2** | Semi Colon " ؛ " | The purpose is to keep silent reader for a period more than for a comma. It's used between long sentences formed a full meaning or between two sentences where the second sentence is the causative for the first sentence. | هُوَ الَّذِي خَلَقَ لَكُم مَّا فِي الْأَرْضِ جَمِيعًا؛ ثُمَّ اسْتَوَىٰ إِلَى السَّمَاءِ فَسَوَّاهُنَّ سَبْعَ سَمَاوَاتٍ؛ وَهُوَ بِكُلِّ شَيْءٍ عَلِيمٌ (٢٩)البقرة<br>It is He who created for you all of that which is on the earth; then He directed Himself to the heaven [His being above all creation] and made them seven heavens; and He is Knowing of all things. |
| **3** | Full Stop". " | The purpose is to use at the end of a full meaning sentences. | الْحَمْدُ لِلَّـهِ رَبِّ الْعَالَمِينَ. (٢)الفاتحة<br>[All] praise is [due] to Allah Lord of the worlds. (2) |
| **4** | Question Mark "؟" | The purpose is to use at the end of the question sentence. | هَلْ أَتَاكَ حَدِيثُ الْغَاشِيَةِ ؟(١)الغاشية<br>Has there reached you the report of the Overwhelming [event]? (1) |
| **5** | Exclamation Mark "!" | It's used at the end of sentences to express excitement, surprise, astonishment or any strong emotion. | وَمَا أَدْرَاكَ مَا الْعَقَبَةُ!(١٢) البلد<br>And what can make you know what is [breaking through] the difficult pass! (12) |

| | | | |
|---|---|---|---|
| **6** | Colon " : " | It's used to precede say or to explain, proves, defines or list elements. | فَقَالَ لَهُمْ رَسُولُ اللَّـهِ: نَاقَةَ اللَّـهِ وَسُقْيَاهَا (١٣)الشمس<br>And the messenger of Allah [Salih] said to them: "[Do not harm] the she-camel of Allah or [prevent her from] her drink." (13) |
| **7** | Ellipsis mark " … " | It's used in place of omitted speech. | وَإِن كَانَ ذُو عُسْرَةٍ فَنَظِرَةٌ إِلَىٰ مَيْسَرَةٍ ۚ وَأَن تَصَدَّقُوا خَيْرٌ لَّكُمْ... إِن كُنتُمْ تَعْلَمُونَ(٢٨٠)البقرة<br>And if someone is in hardship, then [let there be] postponement until [a time of] ease. But if you give [from your right as] charity, then it is better for you… if you only knew. (280) |
| **8** | Hyphen mark " - " | It's used between two sentences if the first sentence is too long and between the number and the numbered. | قُلْ أُؤُنَبِّئُكُم بِخَيْرٍ مِّن ذَٰلِكُمْ ۚ لِلَّذِينَ اتَّقَوْا عِندَ رَبِّهِمْ جَنَّاتٌ تَجْرِي مِن تَحْتِهَا الْأَنْهَارُ - خَالِدِينَ فِيهَا - وَأَزْوَاجٌ مُّطَهَّرَةٌ وَرِضْوَانٌ مِّنَ اللَّـهِ ۗ وَاللَّـهُ بَصِيرٌ بِالْعِبَادِ (١٥)آل عمران<br>SAY, "SHALL I INFORM YOU OF [SOMETHING] BETTER THAN THAT? FOR THOSE WHO FEAR ALLAH WILL BE GARDENS IN THE PRESENCE OF THEIR LORD BENEATH WHICH RIVERS FLOW - WHEREIN THEY ABIDE ETERNALLY - AND PURIFIED SPOUSES AND APPROVAL FROM ALLAH. AND ALLAH IS SEEING OF [HIS] SERVANTS - (15) |
| **9** | Quotation Mark " )) " or " " " | It's used to represent direct speech, quotation or phrase. | قال لي: ((لن أذهب معكم في الحافلة)).<br>He said to me "I will not go with you in the bus". |
| **10** | Parentheses mark "( )" | It's used to include speech that is not related to the sentence | عمان (عاصمة الأردن) أكثر مدن الاردن تعداداً للسكان.<br>Amman (Jordan's capital) more cities populous. |

### 2.1.2 Prosodic Marks (Starts and Stops) in the Qur'an

Qur'an punctuation was proposed by Arabian linguistic scientists in the (16 century), where they worked to produce distinctive punctuation system to facilitate the recitation of the Qur'an, these marks of stops and starts placed in the middle of the verses, at the end of the word they refer to. Table 2.1.2.1 defines different types of starts and stops signs.

The reciter of the Holy Qur'an must have a good knowledge of the pause and the starts (Al Waqf and Al Ibtida') science which has close association with the syntax and the meanings of verses of the Holy Qur'an. Prosodic marks science has two main important points for the reciter and the listener of the Qur'an:

1. Clarifying the meanings of the verses to the listener whenever the reciter knows where he has to pause and starts.
2. Disparity of the Pause degrees has a real relation with the understandings of the Holy Qur'an and the amount of knowledge with it.

Ibin Al Jazari said (one of the greatest Muslims Scientists) "Many previous Muslim imams stipulated that no one can has a license in the Holy Qur'an unless he has a good knowledge in the starts and stops science".

As an example of the relation between the pause and starts science and the meanings of the verses in the Holy Qur'an:-

*(قَالَ فَإِنَّهَا مُحَرَّمَةٌ عَلَيْهِمْ ∴ أَرْبَعِينَ سَنَةً ∴ يَتِيهُونَ فِي الْأَرْضِ ۖ فَلَا تَأْسَ عَلَى الْقَوْمِ الْفَاسِقِينَ) (٢٦ : المائدة )*

[Allah] said, “Then indeed, it is forbidden to them for forty years [in which] they will wander throughout the land. So do not grieve over the defiantly disobedient people." (26**:** Al-Ma'ida).

The above verses has two Interchangeable Pause mark (∴) which means that the reciter has to stop on either of them which will has an impact on the meaning of the verses i.e. if the reciter stopped at the first site this means that the land is forbidden for the people (the Jews) forever and they will wander throughout the land for forty years. While, if the reciter stopped at the second site this means that it is forbidden for them for forty years and they will wander throughout the land (Shaker, Youssef et.al 2012).

Table 2.1.2.1: Pause and starts signs.

| # | pause mark | Pause type | Definition | Example |
|---|---|---|---|---|
| **1** | م | compulsory | The reciter is mandatory to pause. | لَقَدْ سَمِعَ اللَّـهُ قَوْلَ الَّذِينَ قَالُوا إِنَّ اللَّـهَ فَقِيرٌ وَنَحْنُ أَغْنِيَاءُ ۘ سَنَكْتُبُ مَا قَالُوا وَقَتْلَهُمُ الْأَنبِيَاءَ بِغَيْرِ حَقٍّ وَنَقُولُ ذُوقُوا عَذَابَ الْحَرِيقِ (١٨١:آل عمران)<br>Allah has certainly heard the statement of those [Jews] who said, "Indeed, Allah is poor, while we are rich." We will record what they said and their killing of the prophets without right and will say, "Taste the punishment of the Burning Fire. (181:Al-i-Imran) |
| **2** | ج | Permissible | The reciter is permissible to pause or continue. | رَّبَّنَا إِنَّنَا سَمِعْنَا مُنَادِيًا يُنَادِي لِلْإِيمَانِ أَنْ آمِنُوا بِرَبِّكُمْ فَآمَنَّا ۚ رَبَّنَا فَاغْفِرْ لَنَا ذُنُوبَنَا وَكَفِّرْ عَنَّا سَيِّئَاتِنَا وَتَوَفَّنَا مَعَ الْأَبْرَارِ (١٩٣: آل عمران)<br>Our Lord, indeed we have heard a caller calling to faith, [saying], 'Believe in your Lord,' and we have believed. Our Lord, so forgive us our sins and remove from us our misdeeds and cause us to die with the righteous.(193: Al-i-Imran ) |
| **3** | صلي | Continuation preferred | The reciter is allowed to pause, however continue is preferable. | ذَرْهُمْ يَأْكُلُوا وَيَتَمَتَّعُوا وَيُلْهِهِمُ الْأَمَلُ ۖ فَسَوْفَ يَعْلَمُونَ(٣: الحجر)<br>Let them eat and enjoy themselves and be diverted by [false] hope, for they are going to know. (3: Al-Hijr) |
| **4** | قلي | Pause preferred | The reciter is allowed to continue, however pause is preferable. | حَتَّىٰ إِذَا بَلَغَ مَغْرِبَ الشَّمْسِ وَجَدَهَا تَغْرُبُ فِي عَيْنٍ حَمِئَةٍ وَوَجَدَ عِندَهَا قَوْمًا ۗ قُلْنَا يَا ذَا الْقَرْنَيْنِ إِمَّا أَن تُعَذِّبَ وَإِمَّا أَن تَتَّخِذَ فِيهِمْ حُسْنًا (٨٦:الكهف)<br>Until, when he reached the setting of the sun, he found it [as if] |

| | | | | |
|---|---|---|---|---|
| | | | | setting in a spring of dark mud, and he found near it a people. Allah said, "O Dhul-Qarnayn, either you punish [them] or else adopt among them [away of] goodness." (86: Al-Kahf) |
| **5** | لا | Not permissible | The reciter is not permissible to pause. | وَإِذَا قِيلَ لَهُم مَّاذَا أَنزَلَ رَبُّكُمْ ۙ قَالُوا أَسَاطِيرُ الْأَوَّلِينَ(٢٤:النحل)<br>And when it is said to them, "What has your Lord sent down?" They say, "Legends of the former peoples," (24: An-Nahl) |
| **6** | ∴ ∴ | Interchangeable | These marks placed in two sites. The reciter can pause on either of them but not on both. | ذَٰلِكَ الْكِتَابُ لا رَيْبَ ۛ فِيهِ ۛ هُدًى لِّلْمُتَّقِينَ (٢:البقرة)<br>This is the Book about which there is no doubt, a guidance for those conscious of Allah - (2:Al-Baqara) |

### 2.1.3 Sayyid Qutb and the Usage of Punctuation Marks

While the science of Pause and Starts (Al Waqf and Al Ibtida') is somehow hard to learn and to be familiar with, an urgent need for an interpretation of the Holy Qur'an that includes explicit form of Pause and Start. One of the greatest books for interpretation of the Holy Qur'an is *fī ẓilāl al-qur'ān* (Qutb, 1991), who interpreted the Holy Qur'an in distinctive manner using Arabic punctuation marks.

While Sayyid Qutb was one of the greatest jurists, thinkers and interpreters of the Holy Qur'an, his interpretation and using of the punctuation marks was upon his understanding of the Holy Qur'an, the science of the Hadith, and his informed of other explanations books of Qur'an (Iben Katheer explanation). Said Qutb Used a set of punctuation marks in his interpretation (*fī ẓilāl al-qur'ān*) of the Holy Qur'an, these marks are (Comma "،", Full Stop ".", Question mark "؟", Exclamation mark "!", Semicolon "؛", Colon ":", Hyphen mark " - ", Ellipsis mark "..") (Alkhaldi, 2000).

Punctuation marks in " *fī ẓilāl al-qur'ān* " used to replace the original Pause and Starts marks (Al Waqf and Al Ibtida' marks) and such as a tool to explain the meanings of the speech of Allah. *fī ẓilāl al-qur'ān* book is one of the best explanation books of Qur'an, which can be used by many of the researchers in Islamic Shari'a. Sayyid Qutb did not talk explicitly about his usage of punctuation marks in his explanation (Alkhaldi, 2000).

### 2.1.4 Makki and Madani Chapters of the Holy Qur'an

The Holy Qur'an was revealed over twenty three years from the begging of preach. The first thirteen years was in Makkah where the general chapters of the Quran revealed there, which then called Makki chapters, then the prophet Mohammad migrated to madinah where the rest of the chapters of the Qur'an revealed there, which then called

Madani chapters. The Makki and Madani chapters have many substantive differences; the Makki chapters concentrate on the adoration only one God and to dismiss of the dominant religion in Makkah before, while the Madani chapters started to declare the laws of Islam to regulate the life of people. Makki and Madani chapters have many other characteristics, table 2.1.4.1 describes some of key differences between them:

All these important differences between chapters of the Qur'an forced us to split the BAQ corpus into two datasets and ensures that this process would be rearrange and interchanged repeatedly to make the cross validation experiments and guarantee a fair experimentation for all the verses of the Qur'an

Table 2.1.4.1: Differences of Makki and Madani chapters.

| # | Makki chapters | Madani chapters |
|---|---|---|
| **1.** | Chapters that have verses that command Muslims to prostrate to Allah. | Commit the Jihad and mention its rules. |
| **2.** | Chapters with verses that contains the word of "Kalla" (كلا) in the second half of the Qur'an. | Mention the details of legal Islamic system that governing people in the community. |
| **3.** | Short verses. | Long verses. |
| **4.** | Hard oratorical style. | Easy vocabulary. |
| **5.** | Arguments with the gentile and objection their connection of partners with Allah. | Argument with the people of the Book i.e. Jews and Christians. |
| **6.** | Chapter with the verses that have the phrase (يا أيها الناس) "O Mankind". | Chapters with verses that starts ( يا أيها الذين امنوا) "O you who believe". |

## 2.2 Punctuation Marks Annotation for Foreign Languages

### 2.2.1 Punctuation Annotation as a Classification Problem

Data Analysis is a concept concerned with extracting models to describe class models or to predict some continuous values. Classification models are related with predicting categorical class labels, namely assigning class labels to a set of unclassified case (David and Balakrishnan, 2010). As example of classification, a bank loan officer wants to analyze customer's data to determine which of them deserves a loan or not.

Classification modeling has two processes (phases):

1- Building the classifier model.

2- Using the classifier model for classification.

In the first phase, the classifier built using the classification algorithm and a set of training data consisting of tuples, where each tuple is associated with it's a class label. A set of classification rules produced of this phase which is called training phase. In the second phase, classification rules are used to assign class labels to a set of test data lacking of these labels. Accuracy of classification model is measured through this phase.

Prediction of punctuation marks is one of the classification problem applications. Where a predefined training set must be exist, that is consisting of a set of tuples such as; words, POS and punctuation tags, these tuples present the features that would be used to solve such problems of punctuation annotation.

Many of machine learning algorithms such as; Hidden Markov Model (HMM), N-gram model, TNT model, Conditional Random Field (CRF) et al., are used to solve problems

in Natural Language Processing such as; information retrieval, machine learning, speech recognition, machine translation, intelligent character recognition et al., and one of these problems are a classification problems i.e. punctuation marks annotation and sentence terminal prediction.

### 2.2.2 Related Works

One of the first systems, although not the first is the system of predicting punctuation marks in speech text using ML algorithm i.e. Trigram language model, with the application of the Viterbi algorithm is the system proposed by (Beeferman, Berger et al. 1998) for the purpose of punctuation restoration. The proposed system was applied based only on lexical information of Penn Treebank corpus for English language with condoning of other linguistic features such as Part Of Speech tags, as it can helpful in the disambiguation of the word categories which can be considered as one of the disadvantages of the system. Another lack of the system is relying on the trigram model which can be feeble in elicitation of long-range dependencies between the words along the sentences. Anyhow, the experiments on the proposed system reveals good results of sentence accuracy (about 54.0%) as it considered more denotative for real-world interests, also it have achieved high Precision and Recall scores.

Another applied model for punctuation annotation is the CRF model that has a reliability for labeling sequential data due its functioning in predicting punctuation marks in many different languages (Lafferty, et al., 2001), where it takes into account the context of the observation and its ability to recept multi-layer of functions for taking observations. (Lu and Ng, 2010), proposed a model of a multifunction task: sentence boundary, sentence type prediction, and punctuation prediction for a speech utterance. The proposed approach has been applied in English and Chinese languages, using

International Workshop on Spoken Language Translation (IWLST) corpus (Paul, et al., 2010) which consists of conversational speech texts, where short and more question sentences appeared compared with other corpora.

Results have been concluded throw experiments referring that punctuation annotation depends on both words and sentence levels have better results compared with depending on just a set of contiguous observations (words). Accordingly, Linear-chain conditional random fields (Lafferty, et al., 2001) would not be useful based on its function of labeling sequences of words. Based on that, they have proposed the usage of Factorial CRF (F-CRF) (Sutton and McCallum, 2006) to do the task of word level labeling, and both sentence boundary detection and sentence type prediction level. The word label layer is used for annotating punctuation symbol (e.g. NONE, COMMA, PERIOD), and the sentence layer used for both declaring sentence boundaries (DEBEG, DEIN, QNBEG, QNIN, EXBEG, EXIN) and sentence type (Declarative, Question, Exclamatory) such that; DEBEG refers to declarative sentence begin, where the GRMM package (Sutton and McCallum, 2006) was used for building and training the Linear-Chain CRF and the Factorial CRF models. The conducted experiments approved that using a Factorial CRF model has many advantages over using Linear-chain CRF model, because of its ability for handling multi-layer of sequence tags which can be used simultaneously into determining the inner words of sentences (sentence boundaries) and the tags for the observations (words). The F-CRF model has more significant improvements for English more than Chinese, because of the linguistic diversity of the Chinese. For example the indicative words of question sentences can be located in any place within the sentence of the question.

Another research for predicting punctuation marks for Chinese was (Zhao, et al., 2012) where a proposed the method uses Conditional Random Field (CRF) model as in

(Lafferty, et al., 2001). They configured the task of punctuation prediction by extracting some linguistic features at three levels such as: word, phrase and function (a combination of the current, the preceding and the following words and their corresponding phrase tag) levels. They assumed that using more linguistic features will help full and outperforming the systems with fewer features used, where the CRF sequence labeling method has been applied to be the framework of the proposed system. The overall methodology stands for using already punctuated Chinese text from Tsinghua Chinese Treebank corpus (Qiang, 2004) for training the proposed model which has many Chinese linguistic features. They removed all punctuation marks, then create three levels of linguistic information (word POS, phrase and functional chunks) to restore punctuation marks using CRF-based sequence labeling method. Several experiments were conducted using mixed features of the extracted levels with different length of sentences. The experiments showed that combining features of phrase and functional levels obtained best results of inserting punctuation marks into the Chinese text compared with using phrase-level or functional-level features alone. The proposed system has a discriminative approach for annotating punctuation marks using only lexical information, also its ability of working on a phrase level, which considered as one of the most helpful features for annotating punctuation marks into texts.

(Pham, et al., 2014) investigated the prediction of punctuation marks for Vietnamese. They have proposed a system for punctuation annotation using linear-chain Conditional Random Fields (CRF) model as in (Lafferty, et al., 2001). Because of Vietnamese has no lexical corpus for the task of punctuation mark prediction, a corpus has to be configured for this purpose. Accordingly, online news journals and papers have been adopted after some preprocessing steps such as; deleting redundant data and modifying digital numbers to written numbers, to produce 240,000 words and a set of seven

punctuation marks i.e. comma (,), period (.), colon (:), semicolon (;), ellipse (…), question mark (?) and exclamation mark (!). Two systems of tagging have been proposed (concise and expanded tagging sets). The first system (concise) aimed to label the non-punctuated word with O and punctuated word with one of punctuation marks i.e. comma, period et.al. The second system concentrates on the fact that there is a relation between current punctuation mark and the previous one. Accordingly, each non-punctuated word was labeled with O with the type of the previous punctuation mark (e.g. O/comma), in the other hand, the punctuated word was labeled with the type of punctuation mark. Based on the two previous systems a set of features has been proposed and trained with the CFR++ toolkit (Kudo, 2005). The best set of features was selected to train and test the proposed system. A Group of experiments have been conducted on the two labeling systems depending on the selected features, then compared with each other. They conclude that using the best set of features with the expanded system they could get the best $F_1$ Scores (52.89%), and that is because of the high dependability of the current punctuation mark and the previous one. This conclusion was proved by applying the expanded system. The proposed system has got a good results score regardless to the lack and the shortage of the training data. In addition, the system does not depend on any other linguistic feature such as; POS tags, which could be considered as a disadvantage of the system.

Punctuation annotation also has an important application in the Automatic Speech recognition systems; which is responsible for translating spoken words into written texts in a real time (Stuckless, 1994). Because of the inability of such systems of automatic inserting of punctuation marks into transcribed texts, which has a direct relation with the information extraction process; that is comprising of many features extractions, such as; Part Of Speech tagging and named entity tagging and many other features. (Hillard,

et al., 2006) investigated the impact of automatically comma annotation on the Part-Of-Speech tagging and named entity tagging through an Automatic Speech Recognition (ASR) system for Mandarin language. The proposed system consists of many phases; automatic speech recognition, punctuation annotation, Part-Of-Speech tagging, and finally name tagging. Various techniques and algorithm taggers were used for each of the previous phases; SRI Decipher recognizer (Stolcke, et al., 2006) used for speech recognition system, while the comma annotation and sentence boundary detection are implemented based on the ICSI+ multilingual sentence segmentation System tools (Zimmerman, et al., 2006), and a two taggers were used for Part-Of-Speech; Viterbi and N-gram taggers, and finally name tagging was performed based on the HMM tagger. The training corpora were built basically based on Mandarin transcribed news and Chinese textbooks and also benefits from many other resources. The proposed system proved the hypothesis that comma prediction will improve the POS and name tagging.

Another investigation that illustrates the importance of sentence boundary detection and punctuation annotation on the translation quality using Automatic Speech Recognition (ASR) system was proposed by (Matusov, et al., 2006). The proposed model that was based on translating the output text of the ASR (long sequence of words) from the source language to the target language after two consecutive steps; sentence boundary detection (segmentation) and punctuation prediction. Three strategies were investigated for the proposed task. The first strategy was based on detecting sentence boundaries, and then translates the segmented sentence to the target language. The produced sentence would be tagged with punctuation marks, taking into account that punctuation marks were removed from the source and the target languages. One of the drawbacks of this system that is punctuation annotation process has to be done based on just using lexical information. Another drawback is that punctuation annotation would be based on

a given translation that is will not be free of mistakes, which can lead of increasing errors of punctuation prediction. The second strategy, consisting of two stages; the first strategy is detecting sentence boundaries, the second is integrating the process of translation and the prediction of punctuation marks for the target language, taking into account that punctuation marks were removed only from the training source corpus, consequently, the translation method would produce two type of phrases, one type without punctuation marks and another type with punctuation marks. Log-linear method was used to select between these two phrases. The third strategy, aimed to integrate the process of detecting sentence boundaries and punctuation predictions together in the source language, then translating the output to the target language, taking care of maintaining the punctuation marks in both of the source and the target training corpora. Because of punctuation rules of the bilingual languages could be different from each other that could be one of the drawbacks of this strategy. After they have analyzed the results of the previous three methods, method number two was the most appropriate for the proposed system. Measuring the quality of sentence segmentation was conducted using Precision and Recall metrics on TC-STAR task (English-to-Spanish) and IWSLT (Chinese-to-English) task. Experiments revealed good results on the TC-STAR task that outperforms the baseline, which used the pause model to improve the segmentation process. On the other hand, experiments showed modest results for the IWSLT task, because of the lack of recognizing the identical words at the start and the end of sentences.

Winnow algorithm (Blum, 1997) is a machine learning algorithm for feature extraction. (Charoenpornsawat and Sornlertlamvanich, 2004) used this algorithm to extract sentence breaks from a paragraph by determining the sentence breaking spaces by taking into account the context around the spaces, whether it’s a sentence break or not,

where the space is the only punctuation mark used in Thai language to determine break sentences. A training data consists of a set of a segmented paragraph and a segmented sentence where each word is tagged with the appropriate POS tag, these training data is passed to the Winnow algorithm to learn from and build the model, then the testing set is fed to the produced model to evaluate the system. Trigram model was used for the process of word segmentation and POS tagging and the Winnow algorithm was used for the sentence break determination. Table 2.2.2.1 presents a conclusion of the related works section and its results, while table 2.2.2.2 presents the advantages and disadvantages for each work.

Table 2.2.2.1: Summary of the related works and their results.

| Method | Paper | Purpose | Language | Corpus | Data Size | Accuracy |
|---|---|---|---|---|---|---|
| **Trigram model using Viterbi algorithm for two proposed algorithms (A and B)** | Cyberpunc: A lightweight punctuation annotation system for speech | A speech recognition system for automatically punctuation annotation into text based on lexical information. | English | Penn Treebank corpus | 1,265,577 trigrams, with 185,420 commas. | **-Algorithm A:** P: 75.6% R: 65.6% $F_1$: 70.2% Sentences Accuracy: 53.3% **-Algorithm B:** P: 78.4% R: 62.4% $F_1$: 69.4% Sentence Accuracy: 54.0% |
| **Factorial Conditional Random Filed Model (F-CRF)** | Better Punctuation Prediction with Dynamic Conditional Random Fields | A multifunction model for: sentence boundary, sentence type prediction and punctuation annotation for Speech utterance. | English and Chinese languages | IWLST09 corpus (BTEC dataset and CT dataset) | A total of 31,000 of Chinese-English utterance pairs. | -Chinese: P: 93.82% R: 89.01% $F_1$: 91.35% P:93.72% R:92.68% $F_1$:93.19% |

| | | | | | | |
|---|---|---|---|---|---|---|
| **Conditional Random Field (CRF)** | A CRF Sequence Labeling Approach to Chinese Punctuation Prediction | Punctuation annotation of Chinese texts using three different levels of features. | Chinese Languages | Tsinghua Chinese Treebank corpus | Training Dataset consists of 57,865 Punctuation Marks<br>Testing Dataset consists of 13,515 Punctuation Marks | P: 82.00%<br>R: 64.90%<br>$F_1$:72.50% |
| **Linear-chain Conditional Random Fields** | Punctuation Prediction for Vietnamese Texts Using Conditional Random Fields | Configuring a system for punctuation prediction of Vietnamese language. | Vietnamese languages | Built corpus of online news journals and papers | 240,000 words with 66% for training and 34% for testing | Best set of features:<br>P: 81.24%<br>R: 39.21%<br>$F_1$:52.89% |

| | | | | | | |
|---|---|---|---|---|---|---|
| **A combination of Hidden-event language model and boostexter classifier.** | Impact Of Automatic Comma Prediction On POS/Name Tagging Of Spanish | Investigating the influence of comma annotation on the Part-Of-Speech and name tagging in a speech recognition system. | Mandarin language | | 60,000 words for training and | |
| **The log-linear model.** | Automatic Sentence Segmentation and Punctuation Prediction for Spoken Language Translation. | Investigation of the importance of sentence boundary detection and punctuation annotation on the quality of translation using the Automatic Speech Recognition system. | English, Spanish and Chinese languages. | -TC-STAR task (English-to-Spanish) corpus And IWSLT (Chinese-to-English) corpus | 28,000 words for training.<br><br>5550 words for testing. | P : 69.9%<br>R: 70.3% |
| **Winnow algorithm** | Automatic sentence break disambiguation for Thai. | Extracting sentence breaking space while spaces are the only punctuation marks used in Thai language. | Thai language | ORCHID corpus | 10,864 sentence (90% for training and 10% for testing) | |

Table 2.2.2.2: Advantages and disadvantages for each the used algorithm.

| Method | Advantages | Disadvantages |
|---|---|---|
| **Trigram Model with using Viterbi algorithm/ Cyberpunc: A lightweight punctuation annotation system for speech** | - Punctuation annotation based only on lexical information. | - Poor performance in case of long range dependencies between the words required for the punctuation annotation purpose.<br>- The lack of accreditation on lexical linguistic features such as Part Of Speech tags which can be of benefit in word category disambiguation. |
| **Factorial Conditional Random Fields model/ Better Punctuation Prediction with Dynamic Conditional Random Fields** | - Overcome traditional Linear-Chine Conditional Random Fields for Investigating long-range dependencies between the observations in the texts, because of its ability to handle multi-layer of sequence tags.<br>- Discriminative model for Predicting punctuation into English language texts. | - Has modest scores in predicting punctuation for Chinese language texts. |

| | | |
|---|---|---|
| **Conditional Random Filed model/ A CRF Sequence Labeling Approach to Chinese Punctuation Prediction** | - Punctuation annotation based only on lexical information.<br>- The ability to benefit from a phrase level feature in order to facilitate the investigation of sentence type. | - The linguistic structure of the Chinese language could be one of the main challenges of this type of methods. |
| **Linear-chain Conditional Random Fields/ Punctuation Prediction for Vietnamese Texts**<br>**Using Conditional Random Fields** | - Ability to capture long-range dependencies which could the most influential feature for the punctuation annotation task.- | -Relying on a limited number of resources to build the training data which could cause lack and shortage of the training data.<br>- Omission of some linguistic features such as POS which could be helpful to improve the scores of the system. |

# Chapter 3

# Methodology

This chapter describes our proposed model for predicting punctuation marks in Arabic text. It describes the methodologies, tools and resources used to predict punctuation marks in Arabic text. Our methodology was based on comparing between 3 commonly used ML algorithms namely; CFR, HMM, and N-gram. We based our experiments on the BAQ corpus (Sawalha, et al., 2012). To implement ML algorithms used in this research we used the Natural Language Toolkit (NLTK) (Bird, et al., 2009).

An overview of our proposed model is described in Section 3.1. NLTK is described in Section 3.2. The text source used in this research is the BAQ Corpus with added tiers of punctuation marks and sentence terminals. The new version of the BAQ Corpus is described and in Sections 3.3. Section 3.4 discusses in great details the ML algorithms. Examples of the different ML algorithms focused in using them for Arabic text.

## 3.1 Methodology of Predicting Punctuation Marks for Arabic Text

Our proposed model is based on using ML algorithms for predicting punctuation marks for Classical and Modern Standard Arabic text. In order to apply ML algorithms to predict punctuation marks in Arabic text, annotated training/testing data with punctuation marks is required.

We have chosen the BAQ Corpus because it is suitable for training/testing ML algorithms and it has many linguistic features appropriate for our task such POS information. Section 1.5 discusses the motivation for choosing Qur'an as training/testing Corpus. Section 3.3 describes the BAQ Corpus in more details.

However, the BAQ Corpus does not include modern punctuation marks information. The only resource that adds modern punctuation marks to the Qur'an text is (fī ẓilāl al-qurān) (Qutb, 1991). We add a new tier of modern punctuation marks to the BAQ Corpus Qutb's punctuation marks placement as in his book.

The new version of the BAQ Corpus with added modern punctuation marks was used to train and test three commonly used ML algorithms. These algorithms are HMM, N-gram and CRF.

Figure 3.1 shows an overview of our model. It is divided into two parts. The first part is data preparation where modern punctuation marks were added to the BAQ Corpus. The second part presents the application of ML algorithms to predict punctuation marks. The following are the major steps that have been achieved to build our methodology.

**Data Preparation:**

1. Annotating of the Qur'an by manually adding punctuation marks to the BAQ Corpus in accordance with Sayyid Qutb explanation (Qutb 1991). These punctuation marks are: Comma "،", Semi Colon "؛", Full Stop ".", Question Mark "؟", Exclamation Mark "!", Colon ":", Ellipsis Mark "…", Hyphen mark "-".
2. Four punctuation marks (*i.e.* Full stop, Question, Exclamation and semi-colon) are used to mark the end of sentences (Zaki 1930). These marks were used to identify sentence terminals in the Qur'an. The corpus then is divided into 8366 sentences. Defining sentence terminals would be used for punctuation annotation and sentence terminal prediction tasks.

**Applying the ML Algorithms:**

1. Splitting the Qur'an corpus for the cross validation experiments into two datasets: training and testing parts, such that the training and testing parts would always be rearranged and interchanged. The training part would always occupy 90% and the testing part would also occupy 10% of the corpus in any one experiment.
2. Training the machine learning algorithms (HMM, N-gram and CRF Taggers) on the training datasets this will utilize two categories of Part-Of-Speech features i.e. three-POS tag set (noun, verb and particle) and ten-POS tag set (noun, verb, nominal, conjunction et al.) of each word in the corpus.
3. Applying the trained models using the test datasets. Then, the obtained results by these algorithms are compared with the already punctuated text (gold datasets).
4. Predicting punctuation marks to MSA text using the most accurate model obtained from the previous step.
5. Finally, obtained results are evaluated.

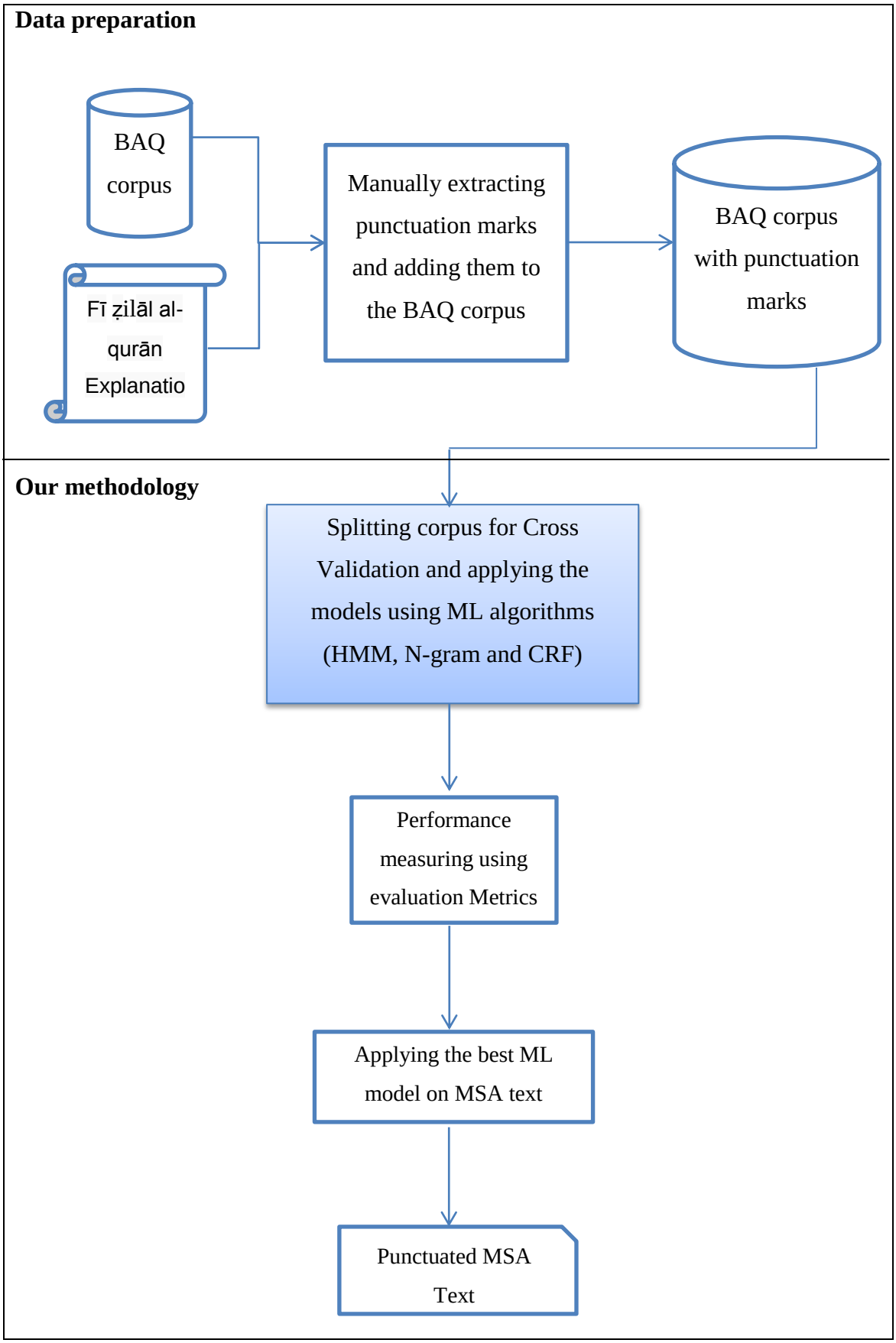


Figure 3.1.1: The proposed model for predicting punctuation marks for Arabic text.

## 3.2 Natural Language Toolkit (NLTK)

To implement the ML algorithms in this research we used the Natural Language Toolkit (NLTK). NLTK is a platform used for building Python programs in order to process human language data. NLTK firstly was created in 2001 at the University of Pennsylvania and it has been developed and improved by many contributors. While NLTK was characterized with its simplicity, consistency, extensibility and modularity, it has been used with many of NLP tasks such as; POS tagging, Text Classification, Text Chunking, Parsing et.al (Bird, et al., 2009).

## 3.3 Dataset (Corpus)

In order to accomplish any NLP task such as information extraction, machine translation, speech recognition or punctuation marks prediction, a corpus (dataset) is needed. This corpus must contain text data for the language under investigation and it is often constructed such that it would have lexical information on each word e.g. POS tags such as noun, verb, particle, etc. or morphological annotation such as root, stem, prefix, suffix, etc.

Because the Holy Qur'an is the central religious text of Islam, it has received the attention of researchers throughout the ages. The Qur'an is positioned as a gold standard corpus in present NLP research with tens of postgraduate theses and journal articles using it to train, test, and develop computational linguistic resources. We used an existing corpus, the Boundary Annotated Quran (BAQ) Corpus (Brierley, et al., 2012). "BAQ Corpus is constructed of 77,430 words and 8,230 sentences, where each word is tagged with syntactic and prosodic information" (Sawalha, et al., 2012). A description of this corpus will follow shortly.

Since the Quran was punctuated hundreds of years ago in a style that is unique to it and BAQ reflected this punctuation system in its annotation, it became necessary that this corpus be also annotated in the modern European-inspired style of punctuation commonly used in MSA. The present research, therefore, modified the BAQ Corpus by adding the punctuation marks that were used in Sayyid Qutb's *fī ẓilāl al-qurān* Exegesis. The reason why this particular book was used is that it is the only authoritative modern interpretation that reproduced the Quran text in an MSA orthographic form, with modern spelling and modern punctuation. Sayyid Qutb, as a modern exegesis, used his understanding of the Quran and his knowledge of Arabic grammar to punctuate the Quran text in a European-inspired style using the same punctuation marks of commas, full stops, exclamation marks, etc. that MSA currently uses. Based on the above, we have extracted punctuation marks from Sayyid Qutb's explanation and added them to the BAQ Corpus in one column i.e. one punctuation mark for each word and "nopunc" for the words that does not have any punctuation.

One aim of the current research project is to investigate the most suitable machine-learning algorithm for automatic punctuation of Arabic texts, but the question is how detailed should the knowledge available for training be? POS tags are critical in any learning because knowledge of the parts of speech, word order, and syntactic structures guides sentential semantics and that guides punctuation. To answer the question, the BAQ Corpus contains two columns; one with coarse POS annotation, and the other with fine POS annotation. The Coarse POS (3-POS) tag sets are: noun, verb and particle, while the Fine POS (10-POS) tag sets are: noun, pronoun, nominal, adverb, verb, preposition, 'lām prefixes, conjunction, particle, disconnected, letters.

Part of the BAQ Corpus version 2.0 is shown in Figure 3.3.1. The figure is structured as an 14-column table, such that columns 1-4 contain tracking information with the 1st column being for the Quran chapter reference; the 2nd for the surah (passage) reference; the 3rd for the ayah (verse) reference; the 4th for the index of the word within the verse (1 for the 1st word in the verse, 2 for the 2nd, 3 for the 3rd, etc.). Since each word is stored in one cell in the 4th column, its annotation is stored in the opposite cells in the rest of columns. Thus, the 5th column contains the orthographic representation of a word entry in Othmani script; the 6th the word's MSA orthographic representation. The 7th and the 8th columns are respectively for the category in a three POS tag classification and a ten POS tag classification for each word in the 5th column. the 9th column the punctuation mark that follows it in Sayyid Qutb's fī ẓilāl al-qurān Exegesis. The annotations used there are: Full-Stop '.'; comma ','; semi-colon ';'; exclamation '!'; question '?'; colon ':'; ellipsis '…'; and hyphen '_'. When no punctuation mark follows the word, the annotation entered is 'nopunc'. The 10th column represents the terminal annotation for each sentence in the corpus, where we adopted four punctuation marks to indicate the ending of sentences in the BAQ corpus; these punctuation marks are: Full-Stop '.'; ; semi-colon ';'; exclamation '!'; question '?', 8366 sentences were produced from this step. The 11th column is used for denoting the ending of the ayah according to the Othmani scripts of the Qur'an. The 12th, 13th and 14th columns specify the phrase boundary annotation (break or non-break annotation) used in the research of (Sawalha, Brierley et al. 2012).

| Chapter No. | Verses No. | Verses ref. | Index of Words in Othmani script | Words in MSA script | 3 POS | 10 POS | Punctuation tags | Sentence terminal tags | End of verses Othmani script | Phrase boundary annotation | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 77 | 44 | 1 | 1 | إِنَّا | إِنَّا | P | PARTICLE | nopunc | - | - | - | non-break | - |
| 77 | 44 | 1 | 2 | كَذَٰلِكَ | كَذَلِكَ | N | PRONOUN | nopunc | - | - | - | non-break | - |
| 77 | 44 | 1 | 3 | نَجْزِى | نَجْزِي | V | VERB | nopunc | - | - | - | non-break | - |
| 77 | 44 | 1 | 4 | ٱلْمُحْسِنِينَ | الْمُحْسِنِينَ | N | NOUN | . | terminal | ۞ | \|\| | break | t |
| 77 | 45 | 1 | 1 | وَيْلٌ | وَيْلٌ | N | NOUN | nopunc | - | - | - | non-break | - |
| 77 | 45 | 1 | 2 | يَوْمَئِذٍ | يَوْمَئِذٍ | N | ADVERB | nopunc | - | - | - | non-break | - |
| 77 | 45 | 1 | 3 | لِّلْمُكَذِّبِينَ | لِلْمُكَذِّبِينَ | N | NOUN | ! | terminal | ۞ | \|\| | break | t |
| 77 | 46 | 1 | 1 | كُلُوا۟ | كُلُوا | V | VERB | nopunc | - | - | - | non-break | - |
| 77 | 46 | 1 | 2 | وَتَمَتَّعُوا۟ | وَتَمَتَّعُوا | V | VERB | nopunc | - | - | - | non-break | - |
| 77 | 46 | 1 | 3 | قَلِيلًا | قَلِيلاً | N | NOMINAL | nopunc | - | - | - | non-break | - |
| 77 | 46 | 1 | 4 | إِنَّكُم | إِنَّكُمْ | P | PARTICLE | nopunc | - | - | - | non-break | - |
| 77 | 46 | 1 | 5 | مُّجْرِمُونَ | مُجْرِمُونَ | N | NOUN | . | terminal | ۞ | \|\| | break | t |
| 77 | 47 | 1 | 1 | وَيْلٌ | وَيْلٌ | N | NOUN | nopunc | - | - | - | non-break | - |
| 77 | 47 | 1 | 2 | يَوْمَئِذٍ | يَوْمَئِذٍ | N | ADVERB | nopunc | - | - | - | non-break | - |
| 77 | 47 | 1 | 3 | لِّلْمُكَذِّبِينَ | لِلْمُكَذِّبِينَ | N | NOUN | ! | terminal | ۞ | \|\| | break | t |
| 77 | 48 | 1 | 1 | وَإِذَا | وَإِذَا | N | ADVERB | nopunc | - | - | - | non-break | - |
| 77 | 48 | 1 | 2 | قِيلَ | قِيلَ | V | VERB | nopunc | - | - | - | non-break | - |
| 77 | 48 | 1 | 3 | لَهُمُ | لَهُمُ | N | PRONOUN | nopunc | - | - | - | non-break | - |
| 77 | 48 | 1 | 4 | ٱرْكَعُوا۟ | ارْكَعُوا | V | VERB | nopunc | - | - | - | non-break | - |
| 77 | 48 | 1 | 5 | لَا | لَا | P | PARTICLE | nopunc | - | - | - | non-break | - |
| 77 | 48 | 1 | 6 | يَرْكَعُونَ | يَرْكَعُونَ | V | VERB | . | terminal | ۞ | \|\| | break | t |

Figure 3.3.1: A sample of the BAQ corpus with added modern punctuation marks.

## 3.4 Machine Learning Algorithms

Three ML algorithms were used in the research i.e. N-gram, HMM and CRF algorithms. These algorithms use different approaches for prediction. ML algorithms are probabilistic Computational Linguistics models. Probability represents the measuring of the portion of how certain events will occur. This portion is limited between 0 and 1. Computational Linguistic is a field concerned with having statistical measurements in the natural language field from computational perspective (Daniel and James, 2000). In this section we will describe the each algorithm in detail.

### 3.4.1 N-GRAM Model

The N-gram model is one of the probabilistic language models that are concerned for predicting the next item in a sequence of states (e.g. word prediction as application of Natural Language Processing). Prediction using N-gram model is performed by computing the probability of a set of sequence states and selecting the highest probability sequence. The N-gram model has several uses in Natural Language Processing such as; Part-Of-Speech tagging, word similarity and predictive text input systems. In other fields such as Speech Recognition, it has been applied in some applications for Automatic Speech Recognition systems (Jurafsky and Martin 2007). In our study, we will employ the N-gram model for predicting punctuation marks for Arabic text by extracting a model which will be trained and tested on the Boundary Annotated Corpus (BAQ) of the Holy Quran.

Probabilistic models used in Natural Language Processing applications is simply based on **counting** words, POS tags or other features such as punctuation marks for this research (Jurafsky and Martin, 2007). We developed an N-gram model that accomplishes a counting task for predicting punctuation marks for Arabic text. Our N-gram model uses the BAQ Corpus for training and testing. The BAQ Corpus was divided into two portions, 90% for training and 10% for testing.

For example, to predict the punctuation marks after the word "الْوَارِثِينَ" which is a noun in the sentence [21:89] "رَبِّ لَا تَذَرْنِي فَرْدًا وَأَنْتَ خَيْرُ الْوَارِثِينَ.". This word is followed by a Full stop ".". Then the probability $P(.|(\text{الْوَارِثِينَ}, \text{noun}))$ represents the count of this word is a noun and is followed by Full Stop in the training set divided by the count of this word appeared in the training corpus as a noun (**Unigrams**). Equation 1 represents this model:

$$P(.|\text{الْوَارِثِينَ}) = \frac{C(\text{الْوَارِثِينَ(noun) .})}{C(\text{الْوَارِثِينَ(noun)})} \quad (1)$$

The principle of the N-gram model is that we compute the probability of a given word or symbol based on the last few words. A number of models could be derived from the general N-gram model depending on the number of preceding words that would be included in the process of probability approximation. Based on that, the **Bigram model** could be used to compute the conditional probability of the current word taking into account the pervious word $P(w_n|w_{n-1})$.

Furthermore, **the Trigram model** is used to approximate the probability of the current word based on the previous two words $P(w_n|w_{n-2}w_{n-1})$. N-gram models use **Maximum Likelihood Estimation (MLE)** to compute the conditional probabilities. In a Bigram model, the probability of the word $w_i$ is computed through counting the number of occurrences of the current word $w_n$ preceded by the word $w_{n-1}$ (i.e. the count of the two words $w_n$ and $w_{n-1}$ occurring together in the training dataset), and then it is divided by the number of times the word $w_{n-1}$ has occurred in the training dataset. Equation 2 presents the MLE formula for the Bigram model.

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})} \quad (2)$$

In general, we can represent the MLE estimation for the N-gram models by equation 3:

$$P(w_n|w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n-1}w_n)}{C(w_{n-N+1}^{n-1})} \quad (3)$$

Where $C(w_{n-N+1}^{n-1}w_n)$ represent the number of times the sequence of the current word $w_n$ and the previous words $w_{n-N+1}^{n-1}$ (depends on which N-gram model is used) are occurring in the training corpus. $C(w_{n-N+1}^{n-1})$ represents the number of times the

sequence of the previous *N* words has occurred in the training corpus (Jurafsky and Martin 2007).

This study aims to apply the N-gram models (Unigram, Bigram and Trigram models) on the BAQ Corpus in order to produce a model for predicting punctuation marks for Arabic text. So, how can we implement the N-gram models in this work?

As we will explain in the next chapter, the BAQ Corpus is divided into two portions (training and testing portions), We will use the training part in order to produce our N-gram model. The N-gram model uses the word $w_n$ and its Part-Of-Speech tag $p_n$ and the punctuation tags $t_n$ as a feature set $((w_n, p_n)\, t_n)$ for the task of predicting punctuation marks for Arabic text. The estimated probability of the tag $t_n$ given the word $w_n$ and its POS tag $p_n$ using the Unigram model is represented as:

$$P\big(t_n \,|(w_n, p_n)\big) = P\big((w_n, p_n)|t_n\big).\, p(t_n) \qquad (4)$$

Equation 4 computes the estimation probability of the punctuation annotation $t_n$, for the word $w_n$ which has the Part-Of-Speech tag $p_n$. Equation 4 computes the estimation probability of the $P\big((w_n, p_n)|t_n\big)$ using Maximum Likelihood Estimation (MLE) as presented in equation 5:

$$P\big((w_n, p_n)|t_n\big) = \frac{C((w_n, p_n), t_n)}{C((w_n, p_n))} \qquad (5)$$

Where the $C\big((w_n, p_n), t_n\big)$ represents the number of times the word $w_n$ which has the POS tag $p_n$ was followed with punctuation mark $t_n$ in the training corpus. The $C((w_n, p_n))$ represents the number of times the word $w_n$ which has the POS tag $p_n$

occurred in the training corpus. Notice that in equation 5, the probability of the punctuation tag itself $P(t_n)$ is equal to $\frac{C(t_n)}{C(t_n)}$ which is equal to 1.

Now, we use the Bigram model to compute the estimation probability of the $P\big((w_n, p_n)\big|t_n\big)$ as an equation 6 and equation 7, where the $P(t_n|t_{n-1})$ computes the probability of the current tag given the previous tag:

$$P\big(t_n\big|(w_n, p_n)\big) = P\big((w_n, p_n)\big|t_n\big).P(t_n|t_{n-1}) \qquad (6)$$

$$P\big(t_n\big|(w_n, p_n)\big) = \frac{C((w_n, p_n), t_n)}{C((w_n, p_n))} * \frac{C(t_{n-1}\, t_n)}{C(t_{n-1})} \qquad (7)$$

The Trigram model is represented in equation 8 and equation 9 and the figure 3.4.1:

$$P\big(t_n\big|(w_n, p_n)\big) = P\big((w_n, p_n)\big|t_n\big).P(t_n|t_{n-2}t_{n-1}) \qquad (8)$$

$$P\big(t_n\big|(w_n, p_n)\big) = \frac{C((w_n, p_n), t_n)}{C((w_n, p_n))} * \frac{C(t_{n-2}\, t_{n-1}\, t_n)}{C(t_{n-2}\, t_{n-1})} \qquad (9)$$

For example, to compute the estimated probability of the Bigram model for the word "الْوَارِثِينَ" which is a noun (POS tag) and followed with a Full stop "." in the verses ". رَبِّ لَا تَذَرْنِي فَرْدًا وَأَنْتَ خَيْرُ الْوَارِثِينَ" using equations 6 and 7, we use the following equations:

$$P\left(.\left|\big(\text{الْوَارِثِينَ}, \text{noun}\big)\right.\right) = P\left(\big(\text{الْوَارِثِينَ}, \text{noun}\big)\right|.).P(.|nopunc)$$

$$P\left(.\left|\big(\text{الْوَارِثِينَ}, \text{noun}\big)\right.\right) = \frac{C(\big(\text{الْوَارِثِينَ}, \text{noun}\big), .)}{C(\big(\text{الْوَارِثِينَ}, \text{noun}\big))} * \frac{C(nopunc\ .)}{C(nopunc)}$$

And for the Trigram model estimated probability is computed using equations 8 and equations 9:

$$P\left(.\left|\left(\text{الْوَارِثِينَ}, \text{noun}\right)\right.\right) = P\left(\left(\text{الْوَارِثِينَ}, \text{noun}\right)\right|.).P(.|nopunc\ nopunc)$$

$$P\left(.\left|\left(\text{الْوَارِثِينَ}, \text{noun}\right)\right.\right) = \frac{C(\left(\text{الْوَارِثِينَ}, \text{noun}\right),.)}{C(.)} * \frac{C(\ nopunc\ nopunc\ .)}{C(nopunc\ nopunc)}$$

One of the challenges for using the N-gram model is **Sparse Data**. Such challenge happens when data is not well represented in the training corpus. Therefore, rare cases will have zero estimation probability (Jelinek, 1980). Using a large training dataset could solve this problem, but while we are restricted with a limited corpus (BAQ) we have to think of another solution.

**Back-off models** which also called **Katz Back-off models** (Katz, 1987) are one of the proposed models to solve sparse data problem. This model is based on consulting previous models (lower models) in order. This means, if we are looking to compute the estimated probability of a certain word in the training corpus using the Trigram model and we have found that this word has a zero estimation probability, then we would back-off to consult the lower model (i.e. the Bigram model), and so on until we find an estimation probability of that word greater than zero. For example, if we have a trigram of the parameters (x, y and z) then the estimated probability using the Back-off model will be represented in equation 10:

$$P_{katz}(z|x,y) = \begin{cases} P_{katz}(z|x,y); & if\ C(x,y,z) > 0 \\ \alpha(x,y)P_{katz}(z|y); & else\ if\ C(x,y) > 0 \\ P_{katz}(z); & otherwise \end{cases} \quad (10)$$

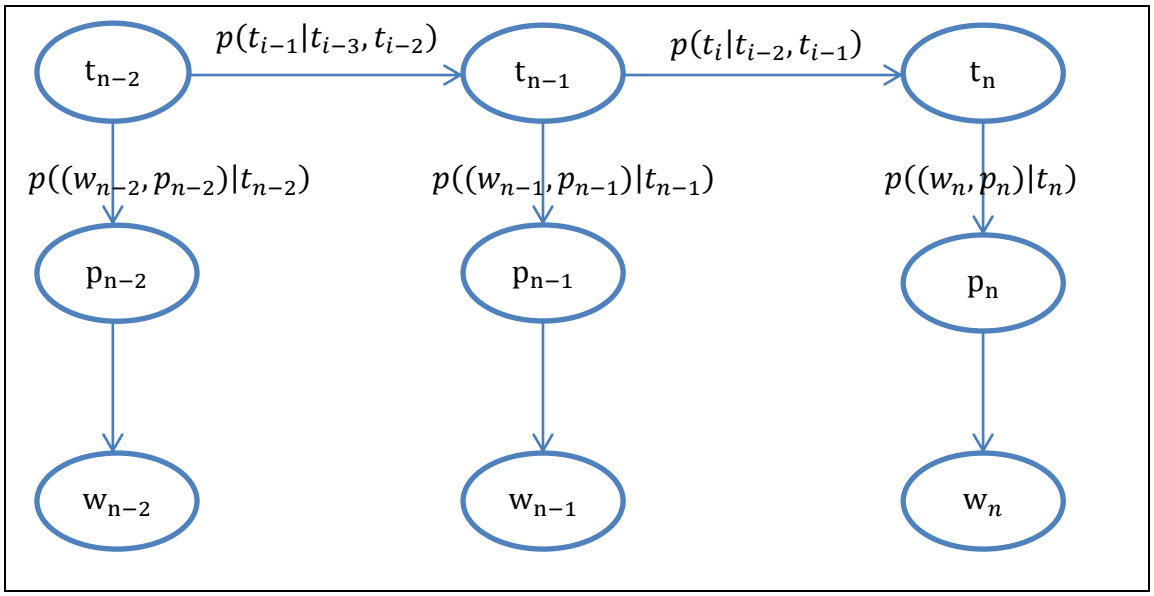


Figure 3.4.1.1: The Trigram model.

### 3.4.2 Hidden Markov Model

The Hidden Markov Model (HMM) is one of the probabilistic sequence classifier models and one of the Marko's models that are used to compute the probabilistic distribution for a sequence of words and assign the best sequence of labels with the highest probability estimation (Blunsom, 2004).

HMM model has many applications in Natural Language Processing such as; Part-Of-Speech tagging, speech recognition systems, sentence segmentation, information extraction and of course punctuation marks prediction (Jurafsky and Martin, 2007)

One of the prominent features of the HMM model is its ability of referring to the observed and hidden events jointly, where the word $w_n$ in a sequence of words ($w_1 w_2 \ldots w_n$) and the Part-Of-Speech tags $p_n$ could be presented as an observed event, and the punctuation annotation tag $t_n$ could be referred as hidden events in our task. The Hidden Markov Model consists of set of components as declared in table 3.4.2.1.

The Hidden Markov Model could be characterized in three stages:

1. **Computing likelihood:** Given $\lambda = (A, B)$ denotes for a set of hidden states (punctuation marks in our case) and a sequence of observations (O) represented with words ($w_n$) and it's Part Of Speech ($p_n$) labels. Then we compute the likelihood of $p(O|\lambda)$.
2. **Decoding:** Finding out the best hidden states (best path) based on the given hidden states $\lambda = (A, B)$ and the observations (O) (words ($w_n$) with its POS labels ($p_n$)).
3. **Learning:** Learning the Hidden Markov Model how to tag the states A and B, based on the given sequence of observations (O) (words ($w_n$) with its POS labels ($p_n$)) and the set of hidden states $\lambda = (A, B)$ of the HMM.

Table 3.4.2.1: Components of Hidden Markov Model.

| ID | Symbol | Meaning |
|---|---|---|
| 1. | $Q = q_1 q_2 q_3 \dots q_N$ | A set on N nodes. |
| 2. | $A = a_{11} a_{12} \dots a_{n1} \dots a_{nn}$ | A transition matrix probability A, where $a_{ij}$ represents the probability of transitioning from node $i$ to node $j$, $\sum_{j=1}^{n} a_{ij} = 1$ , $\forall i$. |
| 3. | $O = o_1 o_2 o_3 \dots o_T$ | A sequence of T observations. |
| 4. | $B = b_i(o_t)$ | A sequence of observations likelihood (emission probability), denotes for the probability of observation $o_t$ to be produced from state $b_i$. |
| 5. | $q_0 q_F$ | A special start and end nodes, where $q_0$ node is connected with probably first hidden nodes with a transition probability $a_{0i}$ , and $q_F$ is connected with the last hidden nodes with a transition probability $a_{iF}$ without any association with any observations (O). |

The following subsections will describe each of these phases and explains their application for predicting punctuation marks for Arabic text using the BAQ corpus.

### 3.4.2.1 Computing Likelihood using Forward Algorithm:

As we said before HMM is one of the Markov's models, as well as Markov chain is also one of the Markov's models, but the key difference between the two models (HMM and Markov chain) is that Markov chain does not have any hidden states, so to compute the probability of any sequence of observations ($o_1 o_2 .. o_T$) we could easily multiply the probabilities for all these observations together. But for the HMM where there are a set of possible hidden states proposed for each observation ($q_1 q_2 .. q_N$) the procedures is different. The procedure is illustrated in Figure 4 represents.

Based on the above we could use the joint probability to compute the likelihood of a sequence of observations ($O$) and a certain sequence of hidden states ($Q$) based on the equation 11:

$$P(O,Q) = P(O|Q) * P(Q) = \prod_{i=1}^{n} p(o_i|q_i) * \prod_{i=1}^{n} p(q_i|q_{i-1}) \qquad (11)$$

But, because we do not know the certain sequence of hidden states we have to compute the likelihood of the sequence of observations with all possible sequences of hidden states which will cause a big complexity of ($N^T$). Instead of using this highly expensive approach the Forward algorithm was used.

The Forward algorithm is one of the dynamic programming algorithms with a complexity of $O(N^2 T)$. This approach aims to compute the observation probability based on; (i) the likelihood (emission) probability of the observation $b_j(o_t)$, (ii) the

transition probability of the node that produces this observation $a_{ij}$ and (iii) the previous path probability. Equation 12 represents the forward algorithm:

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i)\, a_{ij}\, b_j(o_t) \qquad (12)$$

Where:

$\boldsymbol{\alpha_t(j)}$**:** The Forward probability of the observation $o_t$ given the hidden state $j$.

$\boldsymbol{\alpha_{t-1}(i)}$**:** The Forward probability of the previous observation $o_{t-1}$.

$\boldsymbol{a_{ij}}$**:** The transition probability from the previous hidden state $q_i$ to the current state $q_j$.

$\boldsymbol{b_j(o_t)}$**:** The state observation likelihood (emission probability) given the hidden state $j$.

Figure 3.4.2.1 represents a sample of the Forward algorithm for computing the Likelihood for a sequence of hidden states ($q_1 q_2$) and the corresponding set of observations ($o_1 o_2 o_3$), where the length of the sequence of the hidden states equal to the length of the sequence of the observations, taking into consideration that each hidden state ($q_N$) is responsible for generating only one observation ($o_T$) at each time step.

Figure 3.4.2.2 represents the Forward trellis for computing the likelihood of a sequence of hidden states for eight punctuation marks and the non-punctuation (nopunc) denoted with a circles and a sequence of observations representing the words with their three Part Of Speech tags for the first verse in the Holy Qur'an (الْحَمْدُ لِلَّهِ رَبِّ الْعَالَمِينَ.) denoted with squares.

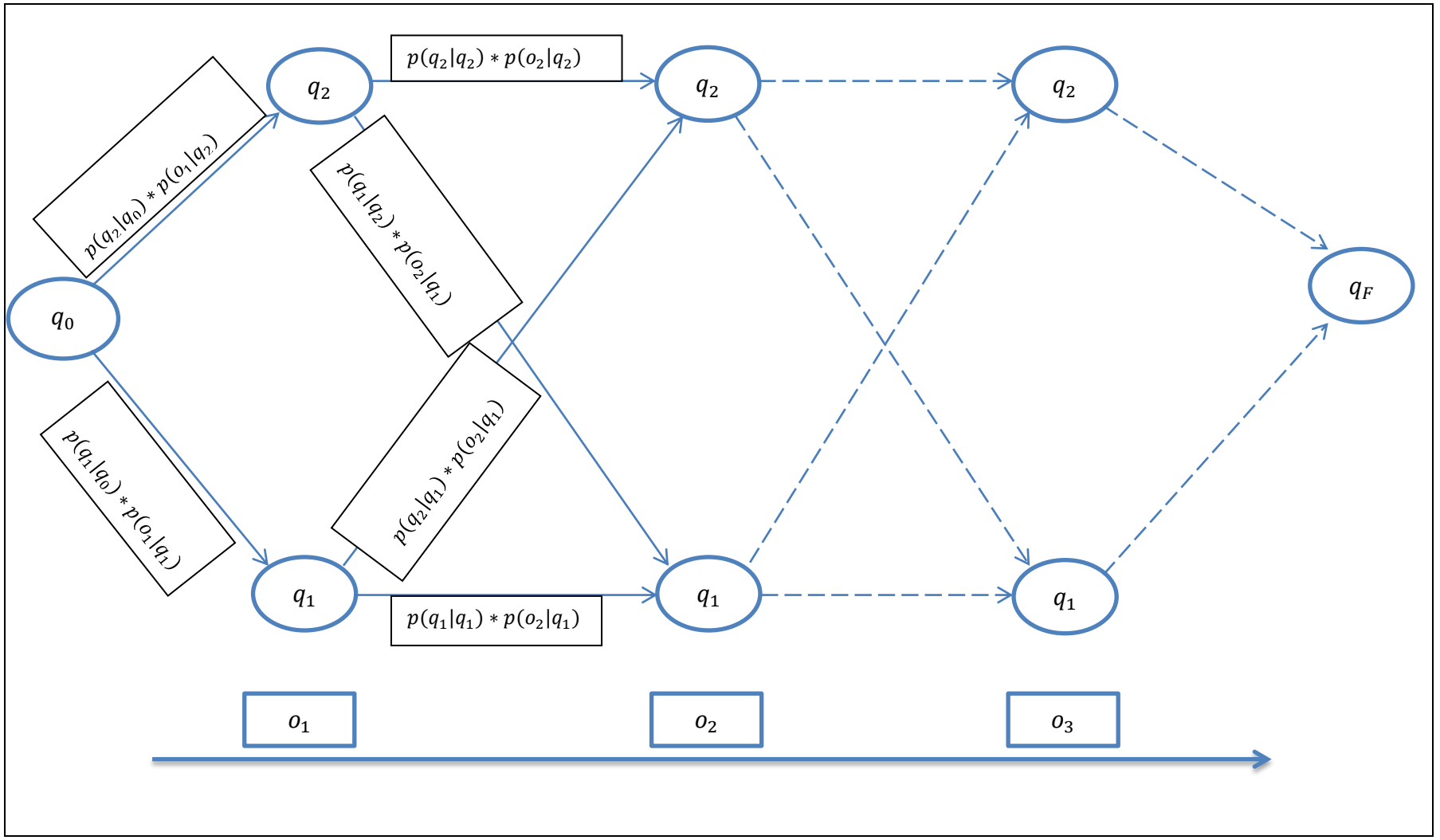


Figure 3.4.2.1.1: Computing Likelihood using the Forward algorithm.

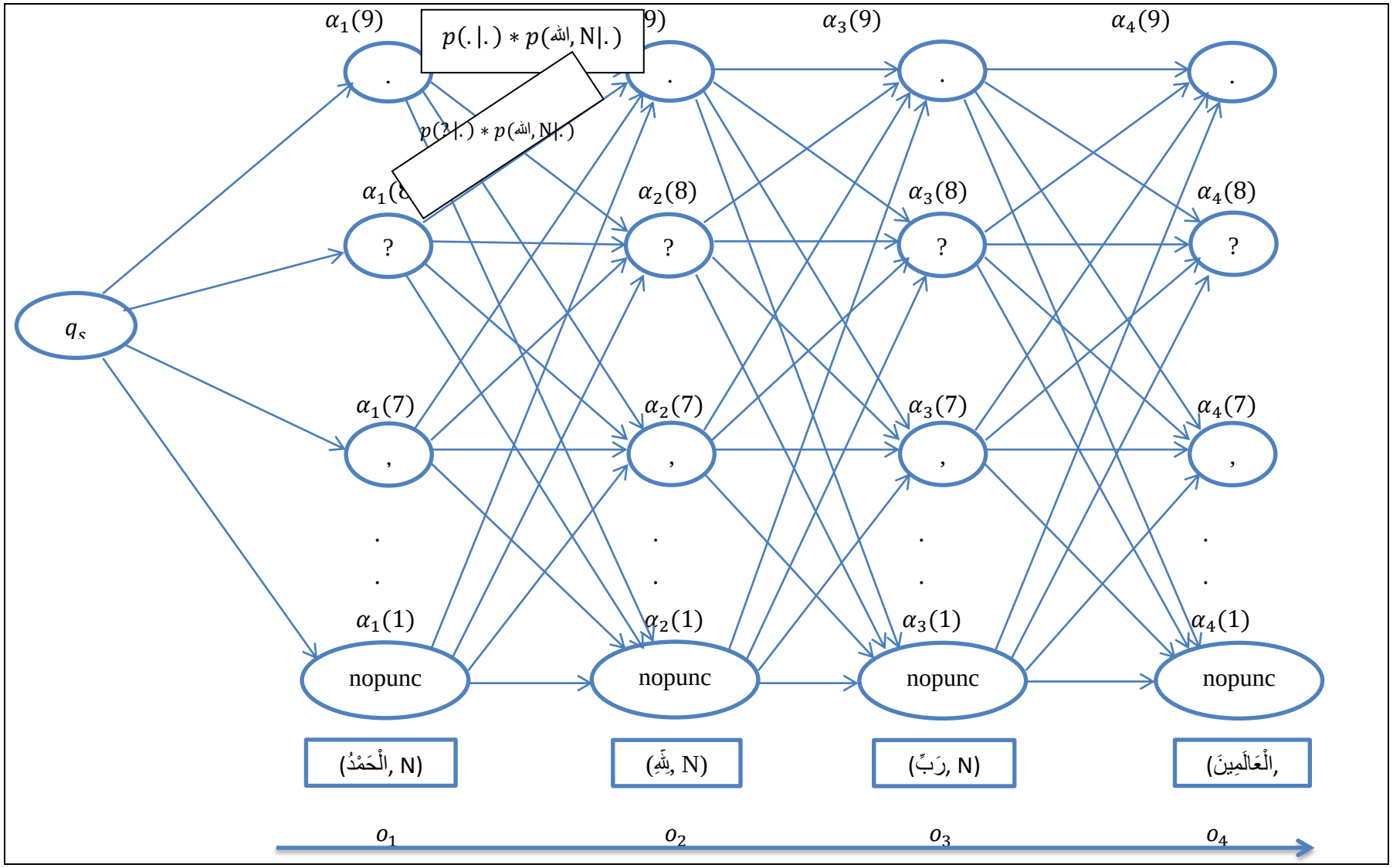


Figure 3.4.2.2.2: An example of computing the likelihood of the punctuation marks with the first verses in the Qur'an.

### 3.4.2.2 Decoding using the Viterbi Algorithm:

Decoding is the process of detecting the best path of a sequence of hidden states ($Q = q_1 q_2 \dots q_N$) that are responsible for generating a sequence of observations ($O = o_1 o_2 \dots o_T$) based on the likelihood probability computed from the previous step (computing likelihood). Viterbi algorithm is used for the decoding process.

Viterbi algorithm computes the maximum probability for each path that leading to the current hidden state (represented with the continuous black line in Figure 3.4.2.2.1) and store it in a Back pointers to keep track for each path (represented with the dashed line). Figure 3.4.2.2.1 presents the Viterbi algorithm. Equation 13 shows how to compute the Viterbi algorithm:

$$v_t(j) = max_{i=1}^{N} v_{t-1}(i) a_{ij}\, b_j(o_t) \qquad (13)$$

Where:

$v_{t-1}(i)$: The previous Viterbi path probability for the previous time step.

$a_{ij}$: The transition probability from the previous state $q_i$ to the current state $q_j$.

$b_j(o_t)$: The state observation likelihood for the current observation $o_t$ and the corresponding hidden state $q_j$.

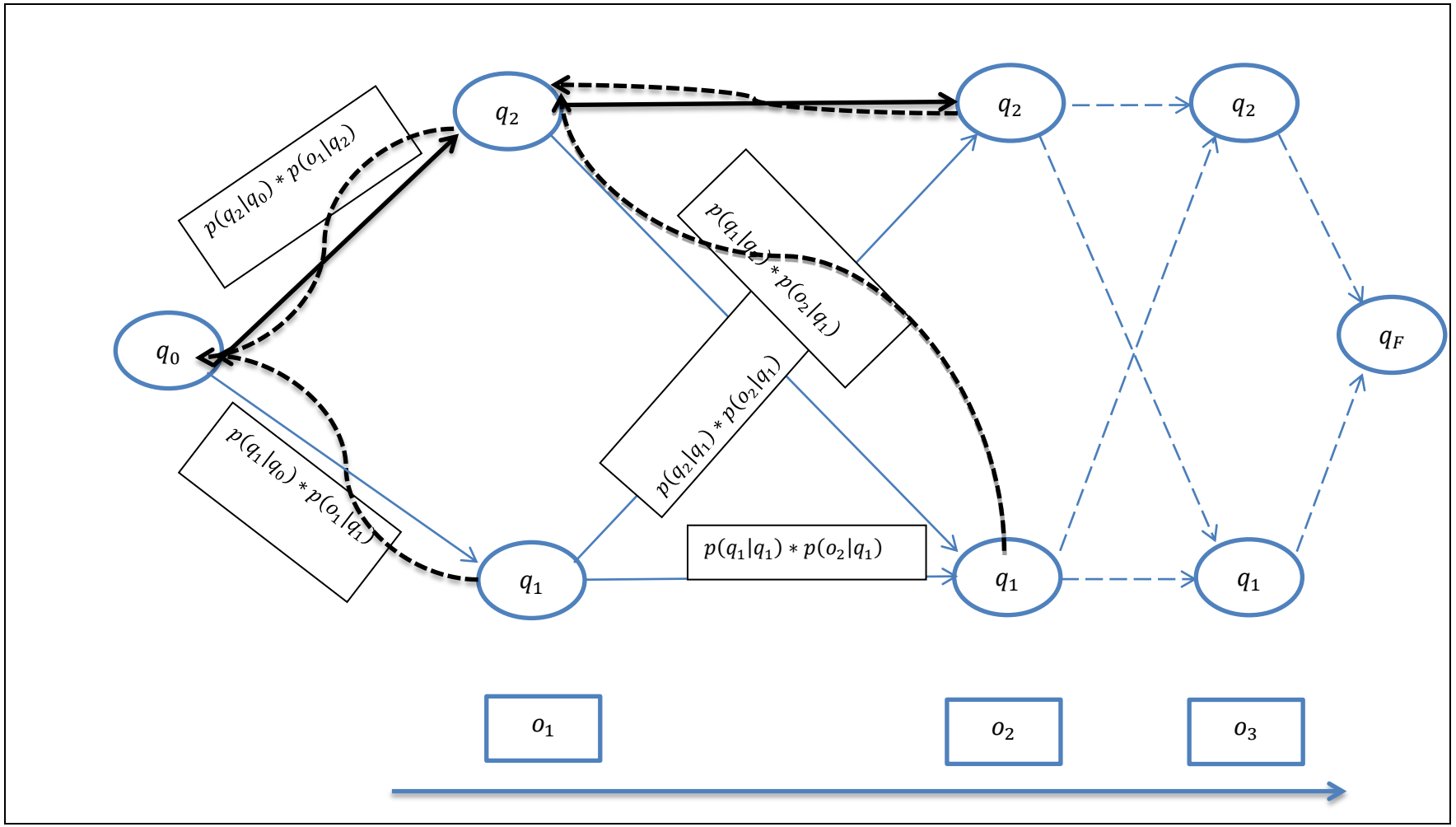


Figure 3.4.2.2.1: Computing maximum probability for each cell in the Viterbi trellis.

### 3.4.2.3 Learning (Training) the HMM using the Forward-Backward Algorithm:

Training is the process of learning the HMM the parameters ***A*** (Transition probabilities) and ***B*** (Emission probabilities), given the sequence of observations ***O*** (unlabeled observations) and the set of possible hidden states ***Q*** using the Forward-Backward algorithm (Baum, 1972). The Forward-Backward algorithm can be used to find the most probable state in the sequence at any time step. In previous section we explain the Forward algorithm, know we will present the Backward algorithm as represented in equation 14. Backward algorithm computes the probability of observing the sequence of observations from $t+1$ to the end, given we are in state $i$ at time $t$.

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij}\, b_j(o_{t+1})\, \beta_{t+1}(j) \qquad (14)$$

Where:

$\beta_t(i)$: The Backward probability of the state $i$ at time $t$.

$a_{ij}$: The transition probability from state $i$ to state $j$.

$b_j(o_{t+1})$: The emission probability of observation ***o*** at time $t+1$ in state $j$.

$\beta_{t+1}(j)$: The Backward probability of state $j$ at time $t+1$.

Figure 3.4.2.3.1 shows the computation of the Backward probability of a sequence of observations at time $t$ given ***s*** set of hidden states $N$.

In general, HMM have some drawbacks. These drawbacks are: (i) inability to deal with multiple interacting features and (ii) its weakness of detecting long-range dependencies along sequence of observations (Lafferty, et al., 2001). In our task, there is a set of

interacting features such as; words, POS tags and punctuation tags. In addition, our task depends on a long-range of dependencies where the nature of punctuation marks prediction task depends on long sequences of words. Therefore, we expect some deficits in the performance of punctuation marks prediction task when we apply HMM algorithm.

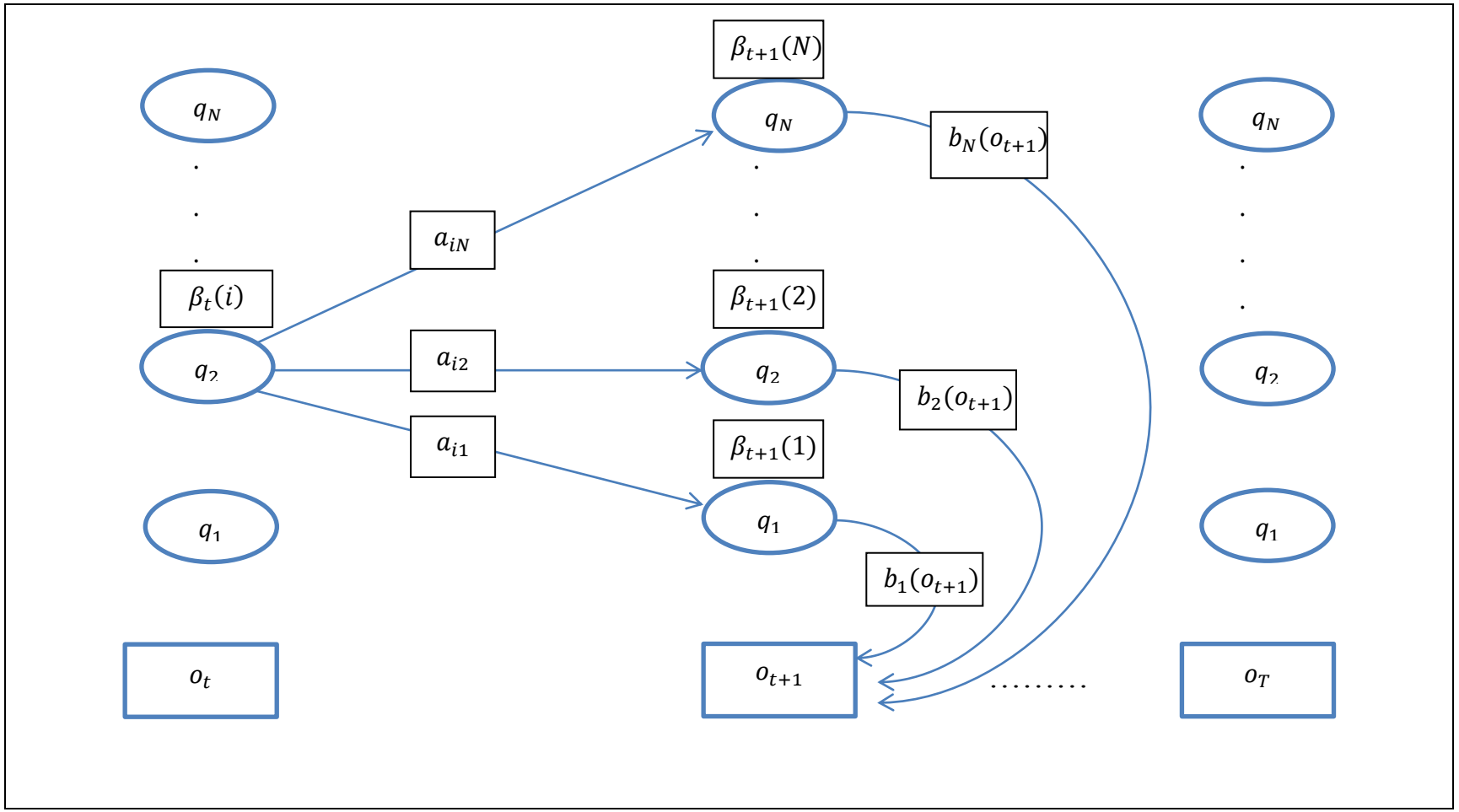


Figure 3.4.2.3.1: Computing the Backward probability of a sequence of observations at time $T$ given state $I$.

### 3.4.3 Conditional Random Fields (CRF)

Conditional Random Fields (CRF) is a framework used for building probabilistic models for sequence labeling and segmentation. CRF model solves the problem of generative models such as; Hidden Markov model, where these generative models try to maximize the joint probability over the observations and sequence labels. The joint probability problem is defined as making the computation of the emission probabilities of the observations (words) given a sequence of labels (hidden states) as the source problem through counting all possible sequences of observations. In contrast, conditional models such as, CRF, concentrates on computing the conditional probability of a sequence of labels where the sequence of observations is given.

Furthermore, HMM model is restricted with a constant number of feature functions in order to compute the transition and emission probabilities. On the other hand, CRF model uses a varied number of later and earlier feature functions for inspecting a sequence of observations. These feature functions are used to compute probabilities of a sequence of labels. Using a number of feature functions to compute conditional probability in CRF model is an additional advantage over HMM model. These feature functions play a key role for detecting long range of dependencies between a sequence of hidden states and corresponding sequence of observations.

We assume two random variables X and Y, where $X = (x_1, x_2, x_3, \dots x_T)$ presents the sequence of observations to be labeled and $Y = (y_1, y_2, y_3, \dots y_T)$ presents the sequence of labels (hidden states), X and Y have the same length. We will assume that the components of X are the sequence of words and there POS tags from the BAQ corpus, where the components of Y are the Punctuation or sentence terminal labels that we want to tag the observations X with them. Then, the condition of variable Y over variable X

is the conditional distribution of $p(Y|X)$ without explicitly counting the probability of the observations $p(x)$.

We will define a graph $G = (V, E)$ such that $Y = (Y_v)_{v \in V}$, such that $Y$ presented with the vertices of G, then the CRF model that is conditioned on the random variables X with random variables $Y_v$ could be defined as: $p(Y_v|X, Y_w, w \neq v)$, where $w \neq v$ means that $w$ and $v$ are neighbors (Lafferty, McCallum et al. 2001). Equation 15 presents the general form of the CRF model:

$$p(y|x) = \frac{1}{Z(x)} \exp\left(\sum_{k=1}^{K}\sum_{t=1}^{T} \lambda_k f_k(y_t, y_{t-1}, x_t)\right) \qquad (15)$$

Where $Z(x) = \sum_y exp\,(\sum_{k=1}^{K}\sum_{t=1}^{T} \lambda_k f_k(y_t, y_{t-1}, x_t))$ is the normalization function and the $f_k(y_t, y', x_t)$ is a set of real valued feature functions where each feature function $f_k$ has a corresponding weight $\lambda_k$ (Sutton and McCallum 2006).

As we mentioned before, the feature function $f_k$ of the CRF model depends on the whole sequence of the observations $x$ rather than one observation $x_t$. Figure 3.4.3.1 presents the undirected graph of the CRF model for a sequence of observations X and the corresponding sequence of observations Y.

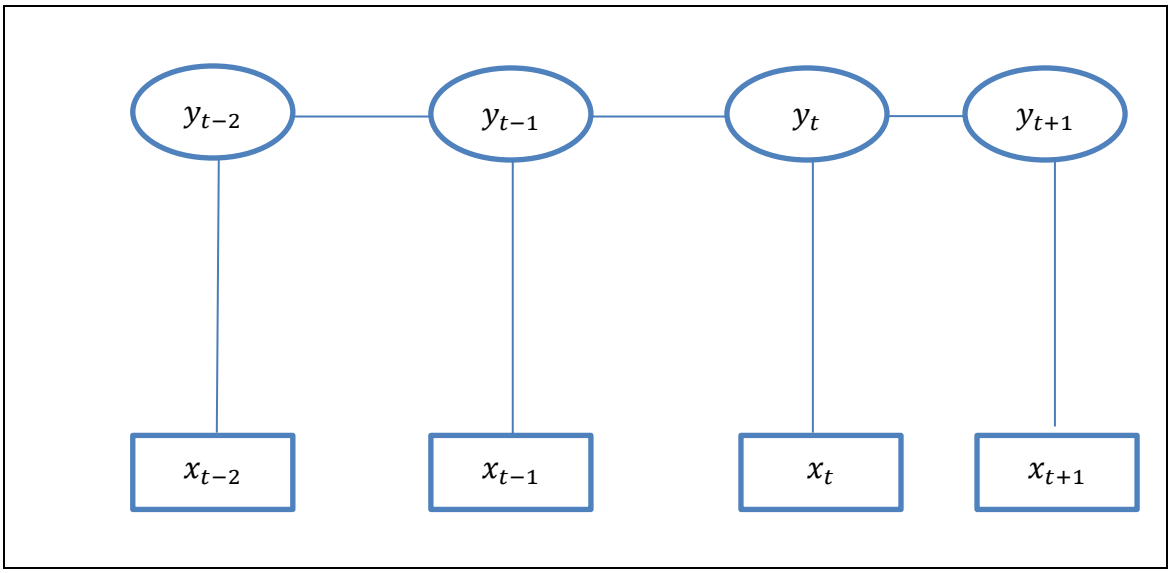


Figure 3.4.3.1: The CRF model.

# Chapter 4

# Design of Experiments

This chapter discusses the implementation of the ML algorithms in the two tasks: punctuation marks prediction and sentence terminal prediction. At the beginning, it talks about the general form of the experiments and preparation of the BAQ corpus for the experiments. Then it declares the two classification problems (*i.e.* punctuation and sentence terminal prediction), and the design of NLTK code for processing these two tasks using three ML algorithms. An experiment of MSA text punctuation marks prediction is conducted using one of the ML algorithms that has the best performance accuracy. A conclusion is drawn at the end of all the experiments.

## 4.1 Experiments in general

### 4.1.1 Cross Validation Experiments

As we explained in section 2.1.4 that verses of the Holy Qur'an were divided into Makki and Madani chapters; these verses differ in their characteristics. We have conducted cross validation experiments for the BAQ corpus for each phase of the experiments to make sure that each chapter of the Qur'an is fairly experimented, such that the text used for training and the text used for testing would be rearranged and interchanged as in the schematic permutation below, preserving that the training part and the testing part would always occupy 90% and 10% of the corpus respectively. Figure 4.1.1.1 presents the structure of the training and testing sets for each experiment of the ML algorithms in the Cross validation experiments.

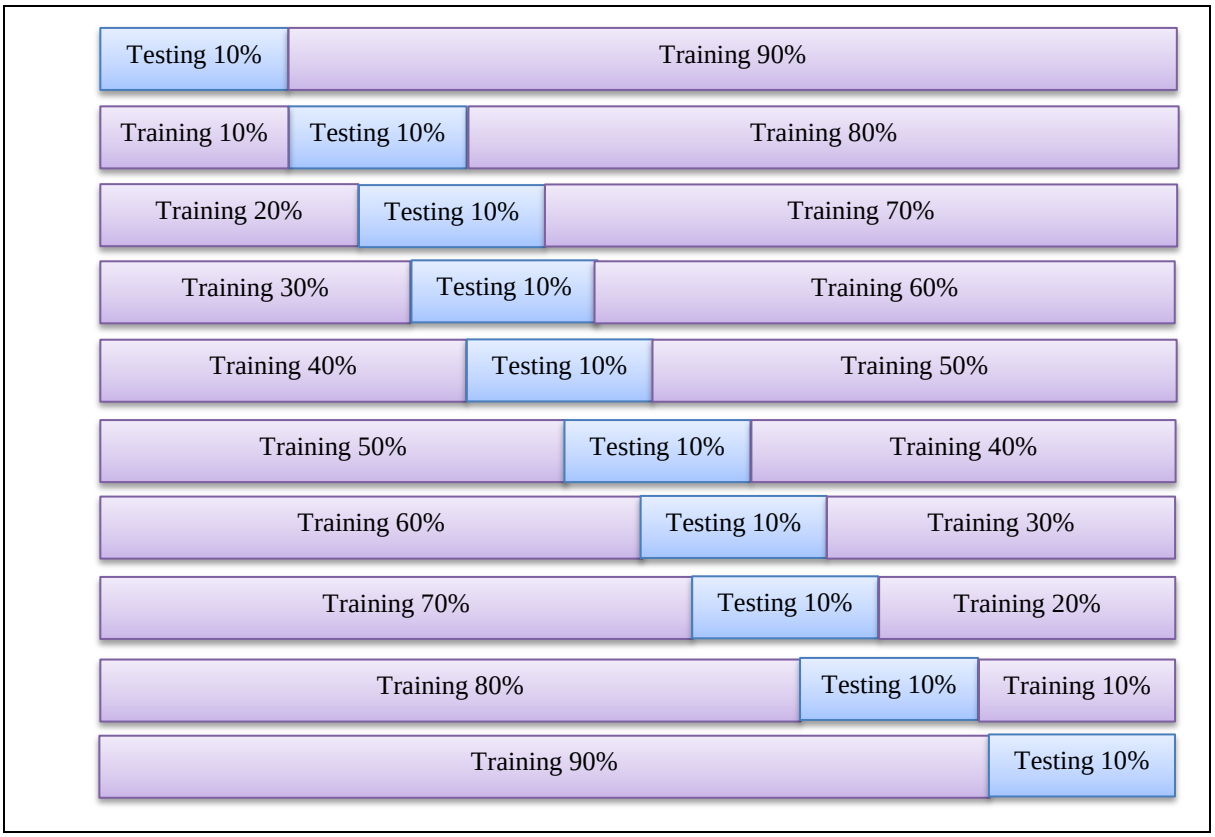


Figure 4.1.1.1: The cross validation experiments.

### 4.1.2 Preparing the Dataset (Training and Testing Datasets)

In order to process the machine learning algorithms we have first of all to prepare the dataset. Therefore, we have two stages; firstly, breaking the BAQ corpus into sentences, secondly, splitting the BAQ corpus into training and testing parts for the cross validation experiments.

#### 4.1.2.1 Breaking the Dataset into Sentences

Breaking the dataset into sentences requires reading the corpus and then determining where sentences end. We have adopted four punctuation marks to indicate the end of sentences (*i.e.* question "?", exclamation "!", full-stop "." and semicolon ";"). Based on that, 8366 sentences resulted. Each sentence consists of a sequence of words, where each word is connected with its POS and punctuation tags (*i.e.* word, POS tag, and

punctuation tag) in the nine class problem (*i.e.* punctuation marks prediction) and (*i.e.* word, POS tag, terminal tag) in the two class problem (*i.e.* sentence terminal prediction).

For the punctuation marks prediction task (nine class problem), we have conducted two types of experiments for each category of POS tags (*i.e.* 3 POS tags and 10 POS tags). Based on that, the feature set for the 3 POS experiments consists of the word itself and its 3 POS tag and punctuation tag (*i.e.* word 3 POS tag, and punctuation tag). On the other hand, the feature set for the 10 POS experiments consists of the word itself and its 10 POS tag and punctuation tag (*i.e.* word, 10 POS tag, and punctuation tag).

In contrast, for the sentence terminal prediction task (two class problem), the feature set of the 3 POS tags experiment consists of the word itself and its 3 POS tag and terminal tags respectively (*i.e.* word, 3 POS tag, terminal tag), on the other hand, the feature set for the 10 POS experiment consists of the word itself and its 10 POS and terminal tags respectively (*i.e.* word, 10 POS tag, terminal tag). In order to break the BAQ Corpus into sentences an NLTK python code was written. The first step is to read the corpus and defining two variables sent_list and quran_list. The sent_list variable used to store the sequence of words with their features (word, POS tag, and punctuation tag), that constructs a complete sentence. Each sentence terminal was defined based on the four punctuation marks; full-stop, exclamation, question and semicolon marks. All of the sentences would be stored in the quran_list variable. Figure 4.1.2.1.1 presents the NLTK code for breaking the BAQ corpus. Figure 4.1.2.1.2 declares the process of breaking the BAQ Corpus.

```
def breaking_quran_corpus():
    lines = codecs.open (r'BAQ_Corpus_v2_with_punctuations.txt', 'r',
'utf-8').readlines( )
    outfile = codecs.open (r'HMM_pos3\quran_list.txt','w','utf-8')
    sent_list = [ ]
    quran_list = [ ]
    for line in lines:
        if line [0] == u'\ufeff':
            line = line[1:]
        if len(line)<=4:
            pass
        else:
            tokens = line.rstrip ( ).lstrip ( ).split ( )
            word, punc, pos3, terminal, pos10 = tokens[5], tokens[6],
tokens[7], tokens[8], tokens[9]
            sent_list.append ((word, pos3, punc))
            if terminal == u'terminal':
                quran_list.append (sent_list)
                for sent in sent_list:
                    outfile.write ('%s\t%s\t%s\n' % (sent[0], sent[1],
sent[2]))
                sent_list = [ ]
    return (quran_list)
```

Figure 4.1.2.1.1: NLTK code for breaking the BAQ Corpus into sentences.

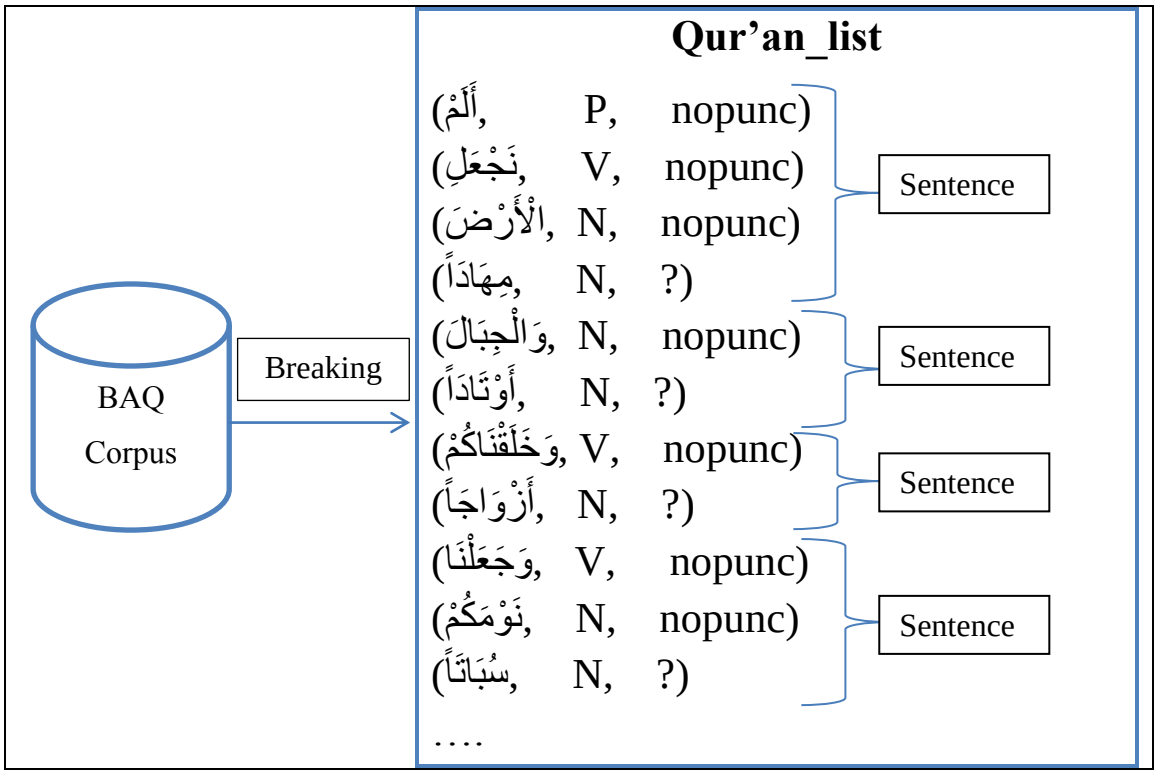


Figure 4.1.2.1.2: Breaking the BAQ Corpus into sentences.

#### 4.1.2.2 Splitting the Corpus (quran_list) into Training, Testing and Gold Parts

The second stage of preparing the dataset includes splitting the quran_list file that was produced from the previous stage into training, testing and gold sets considering that the quran_list file is the input file for this process. Splitting the corpus was repeated 10 times taking into account that the training set would always occupy 90% of the dataset and the testing set would always occupy 10% of the dataset, while the two datasets would be always rearranged and interchanged. Figure 4.1.2.2.1 presents the process used at this stage.

Training part consists of a set of sentences; each sentence consists of a set of tokens (words) and each word connected with its corresponding POS and punctuation tags, tags that constitute the features that ML algorithms use for training. e.g. (الرَّحْمَنِ , N, nopunc).

The gold part and test parts have the same size and same sentences, except that the elements that form each sentence in the test set consist of the word itself and its POS tag e.g. (النَّاسِ ,N), while the gold part consists of the word, its POS tag, and its punctuation tag e.g. (النَّاسِ , N, .).

After training ML algorithms using training sets in each run of the 10_fold experiments, the produced model is used to tag each word in the test set with its appropriate punctuation tag. The produced set of words, POS tag and punctuation tag in the test set would then be compared with the corresponding word, POS tag, and punctuation tag in the gold set to measure the performance accuracy for each machine learning algorithm.

Figure 4.1.2.2.2 presents a piece of code for splitting the quran_list to generate training, testing and gold sets for the cross validation experiments runs, while Figure 4.1.2.2.3

presents the code for generating the training, testing, and gold datasets for each run of the cross validation experiments.

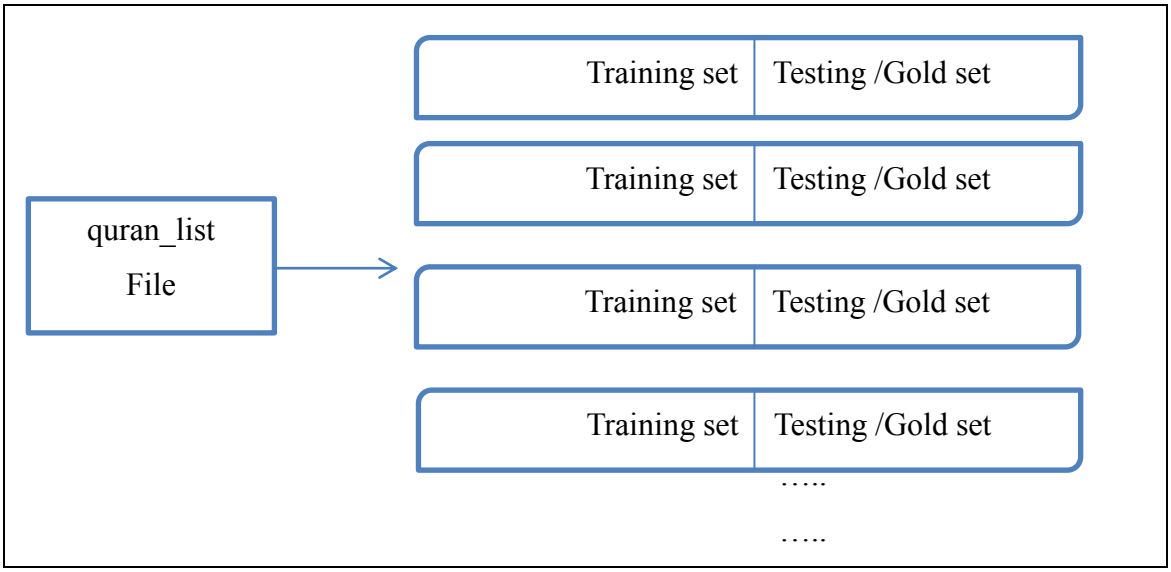


Figure 4.1.2.2.1: Splitting the quran_list for the cross validation experiments to generate the training and testing and gold datasets.

```
def splitting_quran_list (quran_corpus):
    start = 0
    len_test_lst = int (len (quran_corpus) * 0.10)
    test_list = [ ]
    train_list = [ ]
    gold_list = [ ]
    for i in range (10):
        if i == 9:
            end = len (quran_corpus)
        else:
            end = start + len_test_lst
        print ('start: %d\t End: %d ' % (start, end))
        test_list = quran_corpus[start:end]
        train_list = quran_corpus[:start] + quran_corpus[end:]
        print ('Test: %d\t Train: %d ' % ( len (test_list), len (
train_list)))
        start = end
        test_sents = generate_test_file (test_list, i)
        train_sents = generate_train_file (train_list, i)
        gold_sents = generate_gold_file (test_list, i)
```

Figure 4.1.2.2.2: Splitting the quran_list file for 10 times into Training and Testing /Gold sets.

```
def generate_train_file (train_list, i):
    outfile = codecs.open (r'HMM_pos3\train\train_file_%d.txt' %
i,'w','utf-8')
    for sent in train_list:
        for token in sent:
            outfile.write ('%s %s %s\n' % (token[0], token[1],
token[2]))
    outfile.close( )
def generate_test_file (test_list, i):
    outfile = codecs.open (r'HMM_pos3\test\test_file_%d.txt' %
i,'w','utf-8')
    for sent in test_list:
        for token in sent:
            outfile.write ('%s %s\n' % (token[0], token[1]))
    outfile.close ( )
def generate_gold_file (test_list, i):
    outfile = codecs.open (r'HMM_pos3\gold\gold_file_%d.txt' %
i,'w','utf-8')
    for sent in test_list:
        for token in sent:
            outfile.write ('%s %s %s\n' % (token[0], token[1],
token[2]))
    outfile.close ( )
```

Figure 4.1.2.2.3: Codes for generating the train, test and gold dataset for each of the cross validation experiments.

## 4.2 Punctuation Marks Prediction (Nine-Class Problem) and Sentence Terminal Prediction (Two-Class Problem)

Punctuation marks prediction (Nine class problem) task is corresponding with experimenting three machine learning algorithms (*i.e.* N-Gram, HMM, and CRF), on the dataset corpus with both categories of POS tag sets (*i.e.* three-POS tag and ten-POS tag). After preparing the dataset, the three ML algorithms have to be applied each time of the ten experiments, by training the data set to produce a model that would be used to tag the test data set taking into account that the feature set composed of: the word itself, POS tag and punctuation. In this section we will present the way of work for each of the algorithms.

Sentence terminal prediction (Tow class problem) is one of the main topics in NLP. Breaking the sentences is the process concerned with braking long sentences into smaller ones, while the presence or absence of breaks between sentences causes a change in the meaning of the text (Agüero, et.al, 2003).

For the sentence terminal prediction problem we have two types of classes (*i.e.* Terminal class denoted with "terminal", and non-terminal class denoted with "non"). The terminal type indicates to the end or terminal of the sentence. We have adopted four punctuation marks that indicate sentence terminals i.e. Full stop ".", Question mark "?", Exclamation mark "!" and Semi-colon mark ";". The non-terminal type replaced inside of the sentence to indicate no terminals or breaks.

As we have used BAQ corpus for punctuation annotation that consists of 77430 words, we will also use it for phrase break prediction task. Because of adopting four punctuation marks to indicate phrase breaks we have got 8366 breaks or sentences in the corpus. These data set we will be trained and tested using the three machine learning algorithms (taggers) (*i.e.* N-gram, HMM, and CRF) in order to investigate the best performance of the algorithms.

Cross validation experiments would be conducted and two categories of POS tag set i.e. three-POS tag set and ten-POS tag set, would be experimented for each of the taggers. The corpus (dataset) also would be splited into two parts; training set that always will occupy 90% of the dataset and testing set that will occupy 10% of the dataset. The two sets of training and testing would be rearranged and interchanged as in the schematic in figure 4.1.1.1. We denote that the number of terminals in each of the reference datasets is equal to 836 terminals except for the last which was 842 terminals.

Two types of experiment would be conducted for phrase break prediction i.e. three-POS tag set experiments and ten-POS tag set experiments. The features used for training the three algorithms are: the word itself, three-POS tag set and terminal tag for the three-POS tag experiment, while the word itself, ten-POS tag set and terminal tag would be for the ten-POS tag experiment. Figure 4.2.2 presents the experiments of the punctuation marks prediction and sentence terminal prediction. An example of the BAQ corpus is presented in the below Figure 4.2.1, where the words are colored with yellow, three-POS tags are colored with green, the ten-POS tags are in orange, sentence breaks are colored with blue while the corresponding punctuation marks are colored with gray. A full description of the structure for the BAQ Corpus is presented in section 3.3.

| Chapter num. | Verses Num. | Verses ref. | Index of the word number in the Verse | Words in Othmani script | Words in MSA script | 3 POS tags | 10 POS tags | Punctuation tags | Sentence terminal tags |
|---|---|---|---|---|---|---|---|---|---|
| 78 | 1 | 1 | 1 | عَمَّ | عَمَّ | P | PREPOSITION | nopunc | - |
| 78 | 1 | 1 | 2 | يَتَسَآءَلُونَ | يَتَسَاءَلُونَ | V | VERB | ? | terminal |
| 78 | 2 | 1 | 1 | عَنِ | عَنِ | P | PREPOSITION | nopunc | - |
| 78 | 2 | 1 | 2 | ٱلنَّبَإِ | النَّبَإِ | N | NOUN | nopunc | - |
| 78 | 2 | 1 | 3 | ٱلْعَظِيمِ | الْعَظِيمِ | N | NOMINAL | . | terminal |
| 78 | 3 | 1 | 1 | ٱلَّذِى | الَّذِي | N | PRONOUN | nopunc | - |
| 78 | 3 | 1 | 2 | هُمْ | هُمْ | N | PRONOUN | nopunc | - |
| 78 | 3 | 1 | 3 | فِيهِ | فِيهِ | P | PREPOSITION | nopunc | - |
| 78 | 3 | 1 | 4 | مُخْتَلِفُونَ | مُخْتَلِفُونَ | N | NOUN | . | terminal |
| 78 | 4 | 1 | 1 | كَلَّا | كَلَّا | P | PARTICLE | ! | terminal |
| 78 | 4 | 1 | 2 | سَيَعْلَمُونَ | سَيَعْلَمُونَ | V | VERB | . | terminal |
| 78 | 5 | 1 | 1 | ثُمَّ | ثُمَّ | P | CONJUNCTION | nopunc | - |
| 78 | 5 | 1 | 2 | كَلَّا | كَلَّا | P | PARTICLE | ! | terminal |
| 78 | 5 | 1 | 3 | سَيَعْلَمُونَ | سَيَعْلَمُونَ | V | VERB | . | terminal |
| 78 | 6 | 1 | 1 | أَلَمْ | أَلَمْ | P | PARTICLE | nopunc | - |
| 78 | 6 | 1 | 2 | نَجْعَلِ | نَجْعَلِ | V | VERB | nopunc | - |
| 78 | 6 | 1 | 3 | ٱلْأَرْضَ | الْأَرْضَ | N | NOUN | nopunc | - |

Figure 4.2.1: An example of sentence terminals from the BAQ cropus.

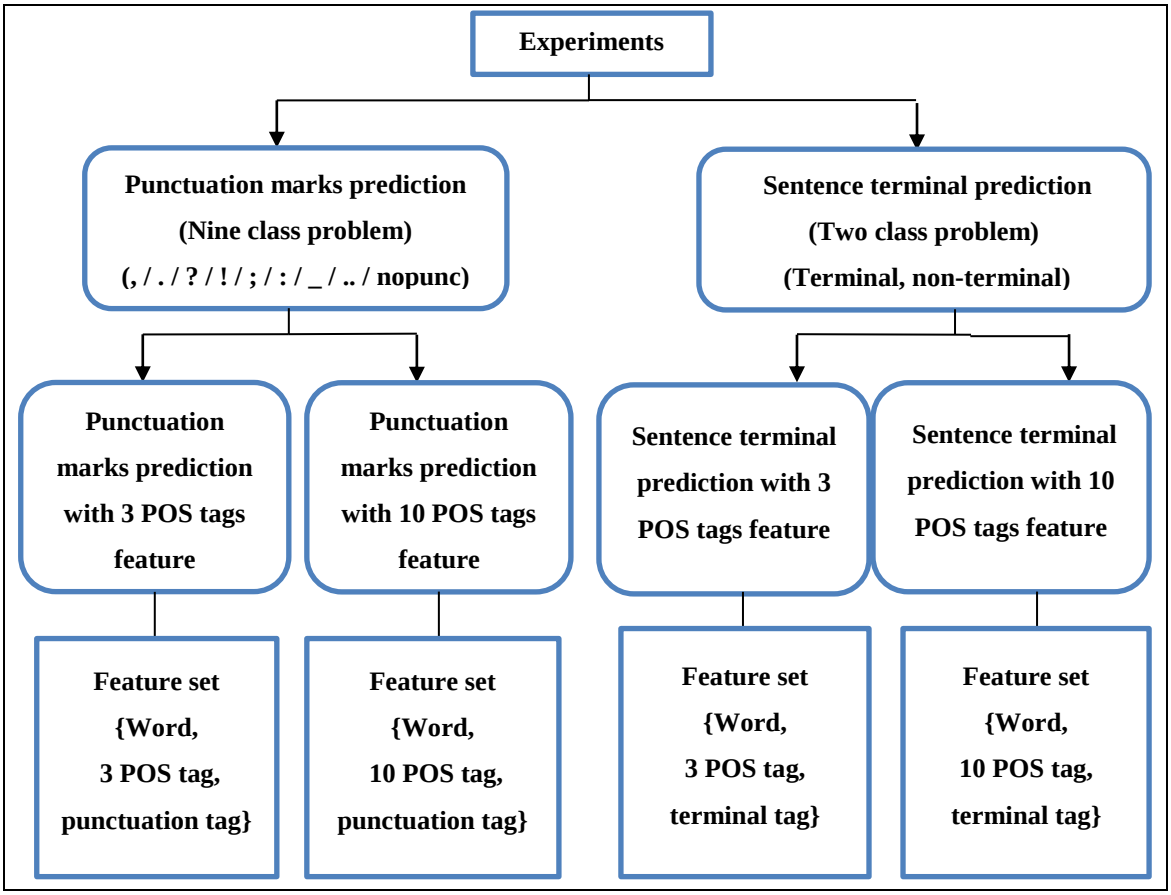


Figure 4.2.2: The expirements of punctuation marks prediction and sentence terminal prediction.

### 4.2.1 N-GRAM Model

In this subsection we will describe applying the N-gram algorithm for predicting punctuation marks for Arabic text. N-gram algorithm was discussed in detail in Section 3.4.1. This Section details the application of N-gram to predict punctuation marks and sentence terminals for Arabic text. N-gram algorithm will be trained and tested using the BAQ Corpus. The feature set is used by the N-gram algorithm includes; (i) the word itself, (ii) 3 POS tags or 10 POS tags, and (iii) the punctuation or sentence terminal tags.

Applying the N-gram algorithm requires implicitly building three models; a Trigram model, a Bigram model, and a Unigram model. These models are built by computing

the estimated probability for each token in a sequence of observations. Building three N-gram models enables the tagger to get the benefit of the Back-off technique. This technique enables the higher model to consult the lower model (e.g. Trigram model (the higher model) consults the Bigram model (the lower model)) in the cases where the estimated probability for a certain word $P\big((w_n, p_n)\big|t_n\big)$ in a sequence of observations tends to the value of zero. After building the N-gram model, it is applied to predict punctuation marks in the test set. Then, the results are evaluated using the gold standard dataset. The evaluation process compares the results of the N-gram model against their equivalent in the gold standard dataset and reports on a set of standard evaluation metrics.

Figure 4.2.1.1 presents the NLTK code for training the N-gram model, where the *default_tagger* value of "nopunc" is used when the estimated probability of any observation is equal to zero. The N-gram model is built by training on 90% of the BAQ corpus sentences. After building the N-gram model, it is used to predict punctuation marks and sentence terminals on the test dataset which represents 10% of the BAQ Corpus sentences. Figure 4.2.1.2 shows the code for testing the N-gram model.

```
default_tagger = nltk.DefaultTagger ('nopunc')
unigram_tagger=nltk.UnigramTagger(train_sents,backoff=default_tagger)
bigram_tagger=nltk.BigramTagger(train_sents,backoff = unigram_tagger)
trigram_tagger=nltk.TrigramTagger(train_sents,backoff= bigram_tagger)
```

Figure 4.2.1.1: The code for training the N-gram models.

```
tagged_sents = [ ]
for i in range ( len ( test_sents )):
      tagged_sents.append ( trigram_tagger.tag ( test_sents[i] ))
```

Figure 4.2.1.2: The code for testing the N-gram model.

### 4.2.2 HMM Model

This section presents the application of the HMM tagger for predicting punctuation marks and sentence terminals of Arabic text. The HMM model is trained and tested using the BAQ corpus. The feature set is used by the HMM algorithm includes; (i) the word itself, (ii) 3 POS tags or 10 POS tags, and (iii) the punctuation or sentence terminal tags.

The application of the HMM tagger includes two main functions; *load_pun* and *run_HMM_tagger*. The *load_pun* function used for splitting and storing the words and their POS tags into a variable called *symbols* set, and the punctuation tags into a variable called *tag_set*. The two variables are stored in a *cleaned_sentences* variable. Three variables *i.e. symbols, tag_set* and *cleaned_sentences* are passed the *run_HMM_tagger* function. Running the HMM tagger produces a model for each of the ten experiments. The produced model used for tagging the test dataset for each run of the ten experiments. Figure 4.2.2.1 presents the two functions load_pun and run_HMM_tagger.

```
def load_pun(train_sents):

    sentences = train_sents
    tag_re = re.compile (r'[*]|--|[^+*-]+')
    tag_set = set ()
    symbols = set ()
    sent = [ ]
    cleaned_sentences = [ ]
    print (len (sentences))
    for sentence in sentences:
        for element in sentence:
            word, tag = element [0],element [1]
            symbols.add ((word))     # store each word with its POS
tag
            tag_set.add (tag)    # punctuation marks
            sent.append  ((word,  tag))    #  store  cleaned-up  tagged
token              (word , POS tag , Punc tag)
        cleaned_sentences += [sent]
        sent = [ ]
    return cleaned_sentences, list (tag_set),  list (symbols)

def run_HMM_tagger (train_sents, gold_sents):

    labelled_sequences, tag_set, symbols = load_pun (train_sents)
    trainer = nltk.HiddenMarkovModelTrainer (tag_set, symbols)
    hmm = trainer.train_supervised (labelled_sequences,
                                    estimator=lambdafd,bins:
LidstoneProbDist(fd, 0.1, bins))
    tagged_sents = hmm.test (gold_sents, verbos = True)
    return (tagged_sents)
```

Figure 4.2.2.1: HMM tagger code.

### 4.2.3 CRF Model

This section presents the application of the CRF model for predicting punctuation marks and sentence terminals for Arabic text. The CRF model was trained and tested using the BAQ Corpus. In contrast to the HMM tagger which is restricted with a constant number of feature functions, the CRF model is characterized in its advantage of using varied number of preceding and succeeding feature set. Because of these advantage, the CRF model is expected to be the best model for punctuation marks and sentence terminal prediction tasks for the Arabic text.

A set of preceding and succeeding features are defined to be CRF model feature set. These features are: the current word, and its POS tag, the succeeding word at position (+1), and its POS tag, the preceding word at position (-1), and its POS tag, the preceding word at position (-2), and its POS tag, and the preceding word at position (-3), and its POS tag. If the current word is the first word in the sentence then a "BOS" is printed, also if the current word is the last word in the sentence, then an "EOS" is printed. The essential feature is the current word and its POS tag. Table 4.2.3.1 shows these features.

Table 4.2.3.1: The set of features used in the CRF model.

| ID | Features | Definition |
|---|---|---|
| **1** | **%x [i, 0]** | **The current word.** |
| **2** | **%x [i, 1]** | **The POS tag of the current word.** |
| **3** | **%x [i-1, 0]** | **The previous word at position (-1).** |
| **4** | **%x [i-1, 1]** | **The POS tag of the word at position (-1).** |
| **5** | **%x [i-2, 0]** | **The previous word at position (-2).** |
| 6 | **%x [i-2, 1]** | **The POS tag of the word at position (-2).** |
| 7 | **%x [i-3, 0]** | **The previous word at position (-3).** |
| 8 | **%x [i-3, 1]** | **The POS tag of the word at position (-3).** |
| 9 | **%x [i+1, 0]** | **The succeeding word at position (+1).** |
| 10 | **%x [i+1, 1]** | **The POS tag of the word at position (+1).** |

An NLTK python code was prepared for enrolling the selected feature set in the CRF model. Three functions code were designed using the NLTK python for preparing the feature set to be used in the CRF model. These features are: *sent2features*, *word2features*, and *sent2punc*. The *sent2features* function is used to pass each word in the sentence to the *word2features* function. Figure 4.2.3.1 presents the *sent2features* function. The *word2features* function is used to extract the feature set for each word in the sentence. Figure 4.2.3.2 presents the *word2features* function. The *sent2punc*

function is used for extracting punctuation tag for each word in the sentence. Figure 4.2.3.3 presents the *sent2punc* function.

```
def sent2features (sent):

    return [word2features (sent, i) for i in range (len (sent))]
```

Figure 4.2.3.1: sent2feature function for passing each function to wor2feature function.

```
def word2features (sent, i):
    word = sent [i][0]
    postag = sent [i][1]
    features = [
        'word =' + word,
        'postag =' + postag, ]
    if i > 0:
        word1 = sent [i-1][0]
        postag1 = sent [i-1][1]
        word2 = sent [i-2][0]
        postag2 = sent [i-2][1]
        word3 = sent [i-3][0]
        postag3 = sent [i-3][1]
        features.extend ([
            '-1:word1 =' + word1,
            '-1:postag1 =' + postag1,
            '-2:word2 =' + word2,
            '-2:postag =' + postag2,
            '-3:word3 =' + word3,
            '-3:postag3 =' + postag3,])
    else:
        features.append ('BOS')
    if i < len(sent)-1:
        word1 = sent [i+1][0]
        postag1 = sent [i+1][1]
        features.extend ([
            '+1:word1 =' + word1,
            '+1:postag1 =' + postag1,])
    else:
        features.append ('EOS')
    return (features)
```

Figure 4.2.3.2: word2feature function for extracting features of each word in the sentence.

```
def sent2punc(sent):

    return [punctag for word, postag, punctag in sent]
```

Figure 4.2.3.3: sent2punc for extracting punctuation tags from each observation.

These two functions *i.e.* sen2features and sent2punc, are appended together to run the CRF trainer. Figure 4.2.3.4 presents the CRF tagger function

```
def crf_tagger (train_sents, test_sents, fileno):
    x_train = [sent2features(s) for s in train_sents]
    y_train = [sent2punc(s) for s in train_sents]
    x_test = [sent2features(s) for s in test_sents]
    y_test = [sent2punc(s) for s in test_sents]
    trainer = pycrfsuite.Trainer (verbose = False)
    for xseq, yseq in zip (x_train, y_train):
        trainer.append (xseq, yseq)

    trainer.set_params ({
        'c1' : 1.0,
        'c2' : 1e-3,
        'max_iterations' :50,
        'feature.possible_transitions': True
        })
    trainer.train ('BAQ_crf_test.crfsuite') # Training the model
    tagger = pycrfsuite.Tagger()
    tagger.open ('BAQ_crf_test.crfsuite')
    example_sent = test_sents[0]
    print ('  '.join (sent2word (example_sent)), end='\n\n')
    print ("Predicted:", ' '.join (tagger.tag (sent2features
(example_sent))))
    print ("Correct:  ", ' '.join (sent2punc (example_sent)))

    y_pred = [tagger.tag (xseq) for xseq in x_test]
    outfile = codecs.open (r'result\crf_results_%d.txt'%fileno, 'w',
'utf-8')
```

Figure 4.2.3.4: CRF tagger function.

After training and creating a model for each run of the ten experiments, these models are used to tag each sentence in the test sets. As an example, Figure 4.2.3.5 presents a sample of a sentence *i.e.* "وَمَا كَانَ لِنَفْسٍ أَنْ تَمُوتَ إِلَّا بِإِذْنِ اللَّهِ كِتَابًا مُؤَجَّلًا", which was tagged using the CRF model with punctuation marks.

وَمَا كَانَ لِنَفْسٍ أَنْ تَمُوتَ إِلَّا بِإِذْنِ اللَّهِ كِتَابًا مُؤَجَّلًا

Predicted: nopunc nopunc nopunc nopunc nopunc nopunc nopunc nopunc nopunc.

Gold : nopunc nopunc nopunc nopunc nopunc nopunc nopunc nopunc nopunc.

Figure 4.2.3.5: A sample of tagged sentence with punctuation marks using the CRF model.

## 4.3 Design of the Modern Standard Arabic Text Punctuation Marks Prediction Experiment

This section presents the design of experiment for tagging the MSA text with punctuation marks using the ML algorithm that has the best performance scores.

A text from سَيِّد قُطْب *Sayyid Qutb* book معالم في الطريق"*Mallem Fittareek*" (Qutb 1979) were selected. The selected text consists of 3859 words without counting punctuation marks. A prerequisite for running the ML algorithm the text need to be tagged with coarse POS tags (*i.e.* 3 POS [noun, verb, and particle]), such that each word would be annotated with one of the POS tags. The Stanford POS tagger has been used for the tagging the text with POS tags. The obtained POS tags (*i.e.* 3-POS tags) for the MSA text was inaccurate; also it was tagged with fine POS tags (10-POS tags) which will not help the argument that the coarse POS tags (*i.e.* 3-POS tags) would be good for tagging an MSA text. Therefore, we have tended to tag the MSA text with coarse POS tags manually. Afterward, the MSA text was tokenized and isolated such that; each word, its POS tag, its and punctuation marks tag, are sorted into three opposite columns. Figure 4.3.1 presents a sample structure of the selected MSA text after processing.

| | | |
|---|---|---|
| تقف | V | nopunc |
| البشرية | N | nopunc |
| اليوم | N | nopunc |
| على | P | nopunc |
| حافة | N | nopunc |
| الهاوية | N | . |
| لا | P | nopunc |
| بسبب | N | nopunc |
| التهديد | N | nopunc |

Figure 4.3.1: sample of the text structure after processing.

The produced file from the previous step which contains the words, POS tags, and punctuation tags is considered as the gold file. The gold file is passed to the ML algorithm. The ML algorithm will ignore the last column (punctuation marks) and tag the text with punctuation marks. The tagged text would be compared with the original file (gold file) to produce a confusion matrix with four values (*i.e.* TPs, TNs, FPs and FNs), to measure the accuracy of the ML model using the performance evaluation metrics.

## 4.4 Summary of the Experiments

To conclude, two tasks *i.e.* punctuation marks predicton (nine-class problem) and sentence terminal prediction (two-class problem), are expermiented using three ML algorithms (*i.e.* N-gram, HMM, and CRF algorithms). Two types of POS tags categories (*i.e.* 3-POS tags and 10-POS tags) are experimented for each task. The ML algorithms are trained and tested using the BAQ Corpus for both task. The training dataset would always occupy 90% and the test dataset would always occupy 10% of the BAQ Corpus. The ML algorithms performance are measured using perfomance evaluation metrics (*i.e.* Accuracy and BCR). An MSA text are tagged with punctuation marks using one of the ML algorithms that has the best performnce evaluation. The ML algorithm are trained on the whole BAQ Coprus to predict punctuation marks for the MSA text.

Therefor, the total number of conducted experiments is thirteen experiment. Table 4.4.1 shows a short look for the design of all the experiments.

Table 4.4.1: Summary of the experiments.

| Exp. # | Nine/ Two class problem | ML Algorithm | POS Categoriy | Train / BAQ | Test / BAQ Or MSA |
|---|---|---|---|---|---|
| **1.** | Nine class | N-gram | 3-pos | 90% | 10% |
| **2.** | Nine class | N-gram | 10-POS | 90% | 10% |
| **3.** | Nine class | HMM | 3-POS | 90% | 10% |
| **4.** | Nine class | HMM | 10-POS | 90% | 10% |
| **5.** | Nine class | CRF | 3-POS | 90% | 10% |
| **6.** | Nine class | CRF | 10-POS | 90% | 10% |
| **7.** | Two class | N-gram | 3-POS | 90% | 10% |
| **8.** | Two class | N-gram | 10-POS | 90% | 10% |
| **9.** | Two class | HMM | 3-POS | 90% | 10% |
| **10.** | Two class | HMM | 10-POS | 90% | 10% |
| **11.** | Two class | CRF | 3-POS | 90% | 10% |
| **12.** | Two class | CRF | 10-POS | 90% | 10% |
| **13.** | Nine class | CRF | 3-POS | 100% BAQ | 100% MSA |

# Chapter 5

# Experiments Results and Evaluation Discussion

## 5.1 Introduction

After preparing the BAQ corpus, breaking the corpus into sentences, splitting it into training and testing datasets, and designing the NLTK code for training and testing the ML algorithms to experiment with the corpus, the experiments of punctuation annotation and sentence terminal prediction were conducted.

Two types of experiments are presented in this chapter; (i) punctuation marks prediction and (ii) sentence terminal prediction. Each of these experiments were applied using three ML algorithms (*i.e.* N-gram, HMM, and CRF). The ML algorithms use the word and POS tags as features for predicting punctuation marks and sentence terminals. Different evaluation metrics were used to compare the results of experiments. The algorithm that has the best performance results is used to punctuate an MSA text with punctuation marks.

This chapter presents and discusses the results of the experiments. Section 5.2 discusses the evaluation metrics used for measuring the performance of the ML algorithms used in this research. Section 5.3 presents the problem of skewed data and the suitable evaluation metrics. Section 5.4 describes the format of the presentation of results. Section 5.5 presents the results of the conducted experiments. Finally, section 5.6 presents a discussion of the obtained results.

## 5.2 Evaluation Metrics

Evaluation process is “concerned with measuring the differences between the expected and the final results … such as the evaluation value is restricted between 0 and 1” (Nakache, et al., 2005)

Many machine learning algorithms used in NLP applications are used to solve the **classification problems.** Classification is the process of identifying the category of an instance, the category is from a set of predefined categories, and a classifier is trained on a dataset where the category for each instance is already known. Algorithms used in this research are considered classification algorithms. Two types of classification tasks are conducted in this research: (i) a nine-class problem (Punctuation marks prediction task) and (ii) a two-class problem (sentence terminal prediction task).

The performance of classification algorithms is measured using evaluation metrics such as; Precision, Recall, F-score, Accuracy Rate and Balanced Accuracy Rate (BCR).. We are going to employ each of these measurements in order to compute the success rate in the performance of the algorithms which have been used in this research.

**Precision** is defined as the ratio of the retrieved instances that are relevant to the query, while **Recall** is the ratio of the relevant instances that are retrieved. For example; if our algorithm correctly predicted 5 full-stop marks in a document which contains 50 sentences. 10 sentences in the document are actually ended with full stop. The precision of our algorithm is the ratio of correct prediction of full-stop mark to the number of full-stops in the document (5/10). Recall is the ration of correct predictions to the total number of sentences in the document (5/50).

One of the tools used to visualize the performance of an algorithm is the **Confusion Matrix.** Columns of confusion matrix present the predicted observations while rows present the actual (gold) observations. Figure 5.2.1 and Figure 5.2.2 show the confusion matrix for the two classification problems.

Precision, Recall, and other evaluation metrics are clearly defined on the basis of the confusion matrix and the number of **True Positives**, **False Positives**, **True Negatives** and **False Negatives**. For the nine-class problem, as shown in Figure 26, the four terms were defined as:

1. **True Positive (TP):** Is the number of punctuation marks which are correctly predicted, *e.g.* a Full-stop is correctly predicted as a Full-stop and a comma is correctly predicted as a comma.
2. **False Positive (FP):** Is the number of punctuation and non-punctuation marks that are incorrectly predicted, *e.g.* a comma is predicted as a full-stop and a non-punctuation mark (*nopunc*) is predicted as an exclamation mark.
3. **True Negative (TN):** Is the number of non-punctuation (*nopunc*) marks that are correctly predicted as non-punctuation marks (*nopunc*).
4. **False Negative (FN):** Is the number of punctuation marks that are predicted as non-punctuation marks, *e.g.* a comma is predicted as non-punctuation mark (*nopunc*).

For the two-class problem as shown in Figure 27 the four terms were defined as follows:

1. **True Positive (TP):** Is the number of sentence breaks predicted correctly as sentence breaks.
2. **False Positive (FP):** Is the number of non-breaks predicted as sentence breaks.

3. **True Negative (TN):** Is the number of non-breaks predicted correctly as non-breaks.
4. **False Negative (FN):** Is the number of sentence breaks predicted as non-breaks.

Predicted

| Gold \ Predicted | . | ? | ! | ; | , | : | .. | - | nopunc |
|---|---|---|---|---|---|---|---|---|---|
| . | TP | FP | FP | FP | FP | FP | FP | FP | FN |
| ? | FP | TP | FP | FP | FP | FP | FP | FP | FN |
| ! | FP | FP | TP | FP | FP | FP | FP | FP | FN |
| ; | FP | FP | FP | TP | FP | FP | FP | FP | FN |
| , | FP | FP | FP | FP | TP | FP | FP | FP | FN |
| : | FP | FP | FP | FP | FP | TP | FP | FP | FN |
| .. | FP | FP | FP | FP | FP | FP | TP | FP | FN |
| - | FP | FP | FP | FP | FP | FP | FP | TP | FN |
| nopunc | FP | FP | FP | FP | FP | FP | FP | FP | TN |

Figure 5.2.1: TPs, TNs, FPs and FNs values for a confusion matrix of nine-class problem.

Predicted

| Gold \ Predicted | Breaks | Non-Breaks |
|---|---|---|
| Breaks | TP | FN |
| Non-Breaks | FP | TN |

Figure 5.2.2: TPs, TNs, FPs and FNs values for a confusion matrix of the two-class problem.

Based on the previous definitions, the performance measurements are defined as in the following equations:

$$Precision = \frac{Tp}{TP+FP} \quad (1)$$

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

$$F - score = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \quad (3)$$

$$Specificity = \frac{TN}{FP+TN} \quad (4)$$

$$Accuracy = \frac{TP+TN}{TP+FN+FP+TN} \quad (5)$$

$$Balanced\ Accuracy\ Rate\ (BCR) = \frac{0.5 * TP}{TP+FN} + \frac{0.5 * TN}{TN+FP} \quad (6)$$

Precision, recall and F-score measures presented in equations 1, 2 and 3 respectively, focus only on positive predictions and omit any information about negative predictions (Powers, 2011). This means, these metrics will yield misleading results if the data is unbalanced (the number of classes vary greatly, *i.e.* number of *nopunc* is over presented in the dataset). We expect low precision, recall, and F-score rates because the true negative cases are the majority of the real and the predicted cases (observations) in our experiments. As so do, the Specificity measure; equation 4 omits information about the true positive cases and concentrates only on the true negative cases. We expect high Specificity rate. More detail about the problem of imbalanced data is discussed in Section 5.3.

Based on that, we need performance measurements that take into account the correct positive and negative predictions. Such performance measurements are Accuracy and the BCR (equations 5 and 6). In this research we will give more concentration on the values of these two measurements.

## 5.3 Skewed Data

Skewed data is the problem of imbalanced data distribution; “a type of classification problem, where some classes are highly underrepresented compared to other classes” (Wang and Yao, 2012).

The skewed data causes problems to machine learning algorithms; *i.e.* they limit the ability to predict less under represented data. Classes in the skewed data are two types; (i) multi-minority classes, *i.e.* classes which are represented with a small ratio of data, and (ii) multi-majority classes, *i.e.* classes which are represented with a great ratio of data. Misclassification of the multi-minority in machine learning could harm the general performance of these algorithms. Two-class imbalanced data and multi-class imbalanced data are two types of the skewed data problem (Wang and Yao, 2012).

Many applications experience the problem of skewed data. In the medical field, for instance, symptoms are used to recognize patient’s diseases; hence, misclassifying these symptoms could harm the prediction of rare diseases. Furthermore, skewed data problems in banking the systems could mislead in the prediction fraud (Longadge and Dongre, 2013).

Data in the current research is also considered skewed. Two types of experiments have been conducted here: punctuation marks prediction (multi-class problem) and sentence break prediction (two-class problem). The dataset in the task of the multi-class problem are distributed into nine classes, but the class of non-punctuation (*nonpunc*) presents the big ratio of the data in the BAQ corpus. While, in the two-class problem, the dataset were distributed into two classes: sentence terminal and non-terminal, where the non-terminal (non) class also presents the big ratio of the data in the BAQ corpus.

The BAQ Corpus is used in this research for training and testing the three machine learning algorithms. To facilitate measurement of the performance of the machine learning algorithms, the predicted data are grouped into four categories: TPs, FPs, TNs, and FNs. Several performance metrics are used to measure the performance of these algorithms. Some of the used metrics such as; Precision, Recall, and F-Score omit TNs from their calculation (see section 5.2). Therefore, we need to use metrics that would deal with the problem of imbalanced data distribution without omitting any of the important information about the results of classification. The two measurements we will use are Accuracy and BCR.

## 5.4 Format of Result Presentation

This section describes the structure of the table of results. This table summarizes the results of cross validation experiments and rating of 6 performance metrics. The same table format is used to summarize the results of testing the three ML algorithms. The table is constructed of 16 columns and 24 rows. The 1st column lists the experiment numbers. The baseline in this column, presents the number of words that are not followed with punctuation marks in the testing parts. The 2nd column presents the number of observations, *i.e.* the number of words in the testing dataset. The 3rd states the number of punctuation marks or sentence terminals in the gold datasets, *i.e.* the number of words that are followed by punctuation marks (this is for punctuation marks prediction experiments), and the number of words followed by sentence terminal (this is for sentence terminal prediction experiments). The 4th column shows the number of non-punctuation marks (nopunc) in the gold datasets, *i.e.* number of words that are not followed by any punctuation mark (this is for punctuation marks prediction experiments), and the number of non-terminals in the gold datasets (this is for sentence

terminal prediction experiments). The 5th and the 6th columns present the number of predicted punctuation and non-punctuation marks respectively.. The 7th, 8th, 9th and 10th present the values of TP, TN, FP and FN respectively. Note that, the values of TN and FN in the baseline rows are the same values as the non-punctuation and punctuation marks in the gold datasets respectively. These values were added to measure the baseline accuracy for each experiment. Baseline accuracy for each experiment is represented in the 11th column. The boldface values in the 11th column are the accuracy rates of the ML algorithm in each experiment. The 12th, 13th, 14th columns list the values of obtained for Recall, Precision and F-score in each experiment. For these three columns, the baseline recall, precision and F-score are equal to zero. The 15th column lists the Specificity value in each experiment. The baseline Specificity equals one. The 16th column, the last one, presents the BCR value in each experiment; the baseline BCR value being equals to 0.5. The bottom two rows in the table present the average performance metric and the average baseline accuracy and BCR. Table 5.4.1 presents an example of the HMM model results for punctuation marks prediction formatted in a table.

Table 5.4.1: Example of results table for testing HMM on the BAQ Corpus.

| | | Gold | | Prediction | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Exp. #** | **N** | **Punctuation** | **Non-Punc** | **Punctuation** | **Non-Punc** | **TP** | **TN** | **FP** | **FN** | **Accuracy** | **Recall** | **Precision** | **F-score** | **Specificity** | **BCR** |
| Baseline | 8523 | 1619 | 6904 | 0 | 8523 | 0 | 6904 | 0 | 1619 | 0.810 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **1** | **8523** | **1619** | **6904** | **895** | **7628** | **536** | **6679** | **359** | **949** | **0.847** | **0.361** | **0.599** | **0.450** | **0.949** | **0.655** |
| Baseline | 9378 | 1900 | 7478 | 0 | 9378 | 0 | 7478 | 0 | 1900 | 0.797 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **2** | **9378** | **1900** | **7478** | **845** | **8533** | **491** | **7280** | **354** | **1253** | **0.829** | **0.282** | **0.581** | **0.379** | **0.954** | **0.618** |
| Baseline | 9810 | 1853 | 7957 | 0 | 9810 | 0 | 7957 | 0 | 1853 | 0.811 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **3** | **9810** | **1853** | **7957** | **1107** | **8703** | **642** | **7670** | **465** | **1033** | **0.847** | **0.383** | **0.580** | **0.462** | **0.943** | **0.663** |
| Baseline | 8517 | 1659 | 6858 | 0 | 8517 | 0 | 6858 | 0 | 1659 | 0.805 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **4** | **8517** | **1659** | **6858** | **874** | **7643** | **546** | **6671** | **328** | **972** | **0.847** | **0.360** | **0.625** | **0.457** | **0.953** | **0.656** |
| Baseline | 8050 | 1490 | 6560 | 0 | 8050 | 0 | 6560 | 0 | 1490 | 0.815 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **5** | **8050** | **1490** | **6560** | **753** | **7297** | **433** | **6365** | **320** | **932** | **0.844** | **0.317** | **0.575** | **0.409** | **0.952** | **0.635** |
| Baseline | 6737 | 1345 | 5392 | 0 | 6737 | 0 | 5392 | 0 | 1345 | 0.800 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **6** | **6737** | **1345** | **5392** | **874** | **5863** | **556** | **5214** | **318** | **649** | **0.856** | **0.461** | **0.636** | **0.535** | **0.943** | **0.702** |
| Baseline | 7129 | 1344 | 5785 | 0 | 7129 | 0 | 5785 | 0 | 1344 | 0.811 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **7** | **7129** | **1344** | **5785** | **833** | **6296** | **506** | **5595** | **327** | **701** | **0.856** | **0.419** | **0.607** | **0.496** | **0.945** | **0.682** |
| Baseline | 7163 | 1382 | 5781 | 0 | 7163 | 0 | 5781 | 0 | 1382 | 0.807 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **8** | **7163** | **1382** | **5781** | **794** | **6369** | **511** | **5606** | **283** | **763** | **0.854** | **0.401** | **0.644** | **0.494** | **0.952** | **0.677** |
| Baseline | 6919 | 1347 | 5572 | 0 | 6919 | 0 | 5572 | 0 | 1347 | 0.805 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **9** | **6919** | **1347** | **5572** | **632** | **6287** | **353** | **5385** | **279** | **902** | **0.829** | **0.281** | **0.559** | **0.374** | **0.951** | **0.616** |
| Baseline | 5204 | 1206 | 3998 | 0 | 5204 | 0 | 3998 | 0 | 1206 | 0.768 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **10** | **5204** | **1206** | **3998** | **462** | **4742** | **216** | **3869** | **246** | **873** | **0.785** | **0.198** | **0.468** | **0.279** | **0.940** | **0.569** |
| | | | | | **Performance Metrics Average** | | | | | **0.839** | **0.346** | **0.587** | **0.433** | **0.948** | **0.647** |
| | | | | | **Average Accuracy Baseline** | | | | | **0.803** | **Average BCR Baseline** | | | | **0.500** |

## 5.5 Experiments Results

After preparing the BAQ Corpus and designing the ML algorithms, the proposed experiments were conducted. This section presents the results for the two classification tasks; (i) prediction of punctuation marks and (ii) prediction of sentence terminals.

### 5.5.1 Prediction of Punctuation Marks (Nine Class Problem)

Here are the results of punctuation mark prediction experiments for each of the ML algorithms, *i.e.* N-gram, HMM and CRF respectively, for both POS-tag categories (*i.e.* 3 POS and 10 POS).

#### 5.5.1.1 N-gram Algorithm

The N-gram model is one of the probabilistic language models that are concerned with predicting the next item in a sequence of observations. Predicting with the N-gram model is performed by computing the probability of a sequence of observations and selecting the highest probability sequence. The N-gram model has many applications in Natural Language processing, such as segmentation, Part Of Speech tagging, etc. More detail about this model can be found in section 3.4.1. The N-gram model was trained and tested on the BAQ Corpus for the prediction of punctuation marks. The obtained results for both categories of POS-tags are described in the next two subsections.

#### 5.5.1.1.1 Results of N-gram Punctuation Marks Prediction using a Three-POS Tag Set

After training and testing the N-gram tagger on the BAQ Corpus for the cross validation experiments, table 5.5.1.1.1.1 was obtained.

The N-gram algorithm in these experiments uses the word and POS tag (*i.e.* 3 POS [noun, verb, and particle]) to predict the punctuation mark or *nopunc* after each word in the test dataset. The average accuracy rate for the 10 experiments is 83.9% with a 3.6% increment above the average baseline accuracy of 80.3%. We also gain an increment of 14.7% in average BCR; *i.e.* a score of 64.7% above the BCR baseline of 50%.

As explained in section 5.2, the recall, precision, and F-score yield misleading results when the data is imbalanced. The results were very low because these metrics take into account only correct positive predictions. Positive cases (*i.e.* punctuation marks) represent less than 20% compared to more than 80% negative cases (*i.e. nopunc*) in the sample since the majority of words in a standard piece of text are followed by no punctuation marks whatsoever. Average recall scored 34.6%, average precision 58.7% and average, and F-score 43.3%. On the other hand, because specificity represents correct predictions of negative cases (*i.e.* how many *nopunc* was predicted as *nopunc*), the obtained specificity was score 94.8%.

Because our dataset is skewed towards *nopunc*, we will rely, in our evaluation of ML algorithm's performance, on Accuracy rate and BCR as they do take correct negative predictions into account. For each experiment, we listed the positive and negative prediction values. TPs and TNs represent the correct predictions of punctuation marks and *nopunc* respectively. FPs and FNs represent false predictions of punctuation marks and *nopunc* respectively. TPs are a primitive measure of an algorithm's performance.

#### 5.5.1.1.2 Results of N-gram Punctuation Marks Prediction using a Ten-POS Tag Set

Table 5.5.1.1.1.2 below displays the results of N-gram's model punctuation prediction in 10 experiments. To predict a punctuation mark or *nopunc* after each word in the dataset in each experiment, the N-gram model uses the word and POS tag (*i.e.* 10 POS [Noun, Verb, Nominal, etc.]) as features. The average accuracy rate obtained in the 10 experiments is 83.3% at a 3.3% increment above the average baseline accuracy of 80.3%. Similarly, the average BCR was a score of 65% at an increment of 15% above the average BCR baseline of 50%.

As explained in the previous section and section 5.2, the recall, precision, and F-score accuracy measures are misleading because the dataset is imbalanced. The average recall score is 36.5%, average precision score 53.8%, and the average F-score 43.3%. The specificity score, on the other hand, is 93.4%. We will concentrate on the accuracy and BCR measures when comparing results.

Figure 5.5.1.1.1 below shows a sample of text from the Qur'an where punctuation marks were automatically added using the N-gram algorithm. In this example, the N-gram algorithm used 3 POS tags and the word as features for predicting the punctuation marks. Corresponding between the original Qur'anic punctuation and the automatic punctuation are highlighted in green. Cases of discrepancy are highlighted in red.

**Original punctuated Quran text**

الم. ذَلِكَ الْكِتَابُ لَا رَيْبَ فِيهِ، هُدًى لِلْمُتَّقِينَ. الَّذِينَ يُؤْمِنُونَ بِالْغَيْبِ، وَيُقِيمُونَ الصَّلَاةَ وَمِمَّا رَزَقْنَاهُمْ يُنْفِقُونَ، وَالَّذِينَ يُؤْمِنُونَ بِمَا أُنْزِلَ إِلَيْكَ وَمَا أُنْزِلَ مِنْ قَبْلِكَ، وَبِالْآخِرَةِ هُمْ يُوقِنُونَ. أُولَئِكَ عَلَى هُدًى مِنْ رَبِّهِمْ، وَأُولَئِكَ هُمُ الْمُفْلِحُونَ. إِنَّ الَّذِينَ كَفَرُوا سَوَاءٌ عَلَيْهِمْ أَأَنْذَرْتَهُمْ أَمْ لَمْ تُنْذِرْهُمْ لَا يُؤْمِنُونَ. خَتَمَ اللَّهُ عَلَى قُلُوبِهِمْ وَعَلَى سَمْعِهِمْ، وَعَلَى أَبْصَارِهِمْ غِشَاوَةٌ، وَلَهُمْ عَذَابٌ عَظِيمٌ. وَمِنَ النَّاسِ مَنْ يَقُولُ: آمَنَّا بِاللَّهِ وَبِالْيَوْمِ الْآخِرِ، وَمَا هُمْ بِمُؤْمِنِينَ. يُخَادِعُونَ اللَّهَ وَالَّذِينَ آمَنُوا، وَمَا يَخْدَعُونَ إِلَّا أَنْفُسَهُمْ وَمَا يَشْعُرُونَ. فِي قُلُوبِهِمْ مَرَضٌ فَزَادَهُمُ اللَّهُ مَرَضًا وَلَهُمْ عَذَابٌ أَلِيمٌ بِمَا كَانُوا يَكْذِبُونَ. وَإِذَا قِيلَ: لَهُمْ لَا تُفْسِدُوا فِي الْأَرْضِ، قَالُوا: إِنَّمَا نَحْنُ مُصْلِحُونَ. (البقرة ١-١١)

**Automatic punctuation of the Quran text**

الم. ذَلِكَ الْكِتَابُ لَا رَيْبَ فِيهِ هُدًى لِلْمُتَّقِينَ. الَّذِينَ يُؤْمِنُونَ بِالْغَيْبِ، وَيُقِيمُونَ الصَّلَاةَ وَمِمَّا رَزَقْنَاهُمْ يُنْفِقُونَ. وَالَّذِينَ يُؤْمِنُونَ بِمَا أُنْزِلَ إِلَيْكَ وَمَا أُنْزِلَ مِنْ قَبْلِكَ وَبِالْآخِرَةِ هُمْ يُوقِنُونَ. أُولَئِكَ عَلَى هُدًى مِنْ رَبِّهِمْ وَأُولَئِكَ هُمُ الْمُفْلِحُونَ. إِنَّ الَّذِينَ كَفَرُوا سَوَاءٌ عَلَيْهِمْ أَأَنْذَرْتَهُمْ أَمْ لَمْ تُنْذِرْهُمْ لَا يُؤْمِنُونَ. خَتَمَ اللَّهُ عَلَى قُلُوبِهِمْ وَعَلَى سَمْعِهِمْ وَعَلَى أَبْصَارِهِمْ غِشَاوَةٌ وَلَهُمْ عَذَابٌ عَظِيمٌ. وَمِنَ النَّاسِ مَنْ يَقُولُ آمَنَّا بِاللَّهِ وَبِالْيَوْمِ الْآخِرِ، وَمَا هُمْ بِمُؤْمِنِينَ. يُخَادِعُونَ اللَّهَ وَالَّذِينَ آمَنُوا وَمَا يَخْدَعُونَ إِلَّا أَنْفُسَهُمْ وَمَا يَشْعُرُونَ. فِي قُلُوبِهِمْ مَرَضٌ فَزَادَهُمُ اللَّهُ مَرَضًا وَلَهُمْ عَذَابٌ أَلِيمٌ بِمَا كَانُوا يَكْذِبُونَ. وَإِذَا قِيلَ لَهُمْ لَا تُفْسِدُوا فِي الْأَرْضِ قَالُوا: إِنَّمَا نَحْنُ مُصْلِحُونَ. (البقرة ١-١١)

Figure 5.5.1.1.1: Automatic prediction of punctuation marks using the N-gram model with 3-POS tags.

Table 5.5.1.1.1.1: Results of punctuation marks prediction using the N-gram algorithm with 3-POS tags.

| | | Gold | | Prediction | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Exp. #** | **N** | **Punctuation** | **Non-Punc** | **Punctuation** | **Non-Punc** | **TP** | **TN** | **FP** | **FN** | **Accuracy** | **Recall** | **Precision** | **F-score** | **Specificity** | **BCR** |
| Baseline | 8523 | 1619 | 6904 | 0 | 8523 | 0 | 6904 | 0 | 1619 | 0.810 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **1** | **8523** | **1619** | **6904** | **895** | **7628** | **536** | **6679** | **359** | **949** | **0.847** | **0.361** | **0.599** | **0.450** | **0.949** | **0.655** |
| Baseline | 9378 | 1900 | 7478 | 0 | 9378 | 0 | 7478 | 0 | 1900 | 0.797 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **2** | **9378** | **1900** | **7478** | **845** | **8533** | **491** | **7280** | **354** | **1253** | **0.829** | **0.282** | **0.581** | **0.379** | **0.954** | **0.618** |
| Baseline | 9810 | 1853 | 7957 | 0 | 9810 | 0 | 7957 | 0 | 1853 | 0.811 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **3** | **9810** | **1853** | **7957** | **1107** | **8703** | **642** | **7670** | **465** | **1033** | **0.847** | **0.383** | **0.580** | **0.462** | **0.943** | **0.663** |
| Baseline | 8517 | 1659 | 6858 | 0 | 8517 | 0 | 6858 | 0 | 1659 | 0.805 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **4** | **8517** | **1659** | **6858** | **874** | **7643** | **546** | **6671** | **328** | **972** | **0.847** | **0.360** | **0.625** | **0.457** | **0.953** | **0.656** |
| Baseline | 8050 | 1490 | 6560 | 0 | 8050 | 0 | 6560 | 0 | 1490 | 0.815 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **5** | **8050** | **1490** | **6560** | **753** | **7297** | **433** | **6365** | **320** | **932** | **0.844** | **0.317** | **0.575** | **0.409** | **0.952** | **0.635** |
| Baseline | 6737 | 1345 | 5392 | 0 | 6737 | 0 | 5392 | 0 | 1345 | 0.800 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **6** | **6737** | **1345** | **5392** | **874** | **5863** | **556** | **5214** | **318** | **649** | **0.856** | **0.461** | **0.636** | **0.535** | **0.943** | **0.702** |
| Baseline | 7129 | 1344 | 5785 | 0 | 7129 | 0 | 5785 | 0 | 1344 | 0.811 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **7** | **7129** | **1344** | **5785** | **833** | **6296** | **506** | **5595** | **327** | **701** | **0.856** | **0.419** | **0.607** | **0.496** | **0.945** | **0.682** |
| Baseline | 7163 | 1382 | 5781 | 0 | 7163 | 0 | 5781 | 0 | 1382 | 0.807 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **8** | **7163** | **1382** | **5781** | **794** | **6369** | **511** | **5606** | **283** | **763** | **0.854** | **0.401** | **0.644** | **0.494** | **0.952** | **0.677** |
| Baseline | 6919 | 1347 | 5572 | 0 | 6919 | 0 | 5572 | 0 | 1347 | 0.805 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **9** | **6919** | **1347** | **5572** | **632** | **6287** | **353** | **5385** | **279** | **902** | **0.829** | **0.281** | **0.559** | **0.374** | **0.951** | **0.616** |
| Baseline | 5204 | 1206 | 3998 | 0 | 5204 | 0 | 3998 | 0 | 1206 | 0.768 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **10** | **5204** | **1206** | **3998** | **462** | **4742** | **216** | **3869** | **246** | **873** | **0.785** | **0.198** | **0.468** | **0.279** | **0.940** | **0.569** |
| | | | | | **Performance Metrics Average** | | | | | **0.839** | **0.346** | **0.587** | **0.433** | **0.948** | **0.647** |
| | | | | | **Average Accuracy Baseline** | | | | | **0.803** | **Average BCR Baseline** | | | | **0.500** |

Table 5.5.1.1.1.2: Results of punctuation annotation using the N-gram algorithm with 10-POS tags.

| | | Gold | | Prediction | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Exp. #** | **N** | **Punctuation** | **Non-Punc** | **Punctuation** | **Non-Punc** | **TP** | **TN** | **FP** | **FN** | **Accuracy** | **Recall** | **Precision** | **F-score** | **Specificity** | **BCR** |
| Baseline | 8523 | 1619 | 6904 | 0 | 8523 | 0 | 6904 | 0 | 1619 | 0.810 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **1** | **8523** | **1619** | **6904** | **1021** | **7502** | **553** | **6600** | **468** | **902** | **0.839** | **0.380** | **0.542** | **0.447** | **0.934** | **0.657** |
| Baseline | 9378 | 1900 | 7478 | 0 | 9378 | 0 | 7478 | 0 | 1900 | 0.797 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **2** | **9378** | **1900** | **7478** | **990** | **8388** | **514** | **7193** | **476** | **1195** | **0.822** | **0.301** | **0.519** | **0.381** | **0.938** | **0.619** |
| Baseline | 9810 | 1853 | 7957 | 0 | 9810 | 0 | 7957 | 0 | 1853 | 0.811 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **3** | **9810** | **1853** | **7957** | **1245** | **8565** | **685** | **7587** | **560** | **978** | **0.843** | **0.412** | **0.550** | **0.471** | **0.931** | **0.672** |
| Baseline | 8517 | 1659 | 6858 | 0 | 8517 | 0 | 6858 | 0 | 1659 | 0.805 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **4** | **8517** | **1659** | **6858** | **967** | **7550** | **567** | **6618** | **400** | **932** | **0.844** | **0.378** | **0.586** | **0.460** | **0.943** | **0.661** |
| Baseline | 8050 | 1490 | 6560 | 0 | 8050 | 0 | 6560 | 0 | 1490 | 0.815 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **5** | **8050** | **1490** | **6560** | **883** | **7167** | **453** | **6270** | **430** | **897** | **0.835** | **0.336** | **0.513** | **0.406** | **0.936** | **0.636** |
| Baseline | 6737 | 1345 | 5392 | 0 | 6737 | 0 | 5392 | 0 | 1345 | 0.800 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **6** | **6737** | **1345** | **5392** | **968** | **5769** | **565** | **5144** | **403** | **625** | **0.847** | **0.475** | **0.584** | **0.524** | **0.927** | **0.701** |
| Baseline | 7129 | 1344 | 5785 | 0 | 7129 | 0 | 5785 | 0 | 1344 | 0.811 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **7** | **7129** | **1344** | **5785** | **941** | **6188** | **530** | **5534** | **411** | **654** | **0.851** | **0.448** | **0.563** | **0.499** | **0.931** | **0.689** |
| Baseline | 7163 | 1382 | 5781 | 0 | 7163 | 0 | 5781 | 0 | 1382 | 0.807 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **8** | **7163** | **1382** | **5781** | **889** | **6274** | **542** | **5554** | **347** | **720** | **0.851** | **0.429** | **0.610** | **0.504** | **0.941** | **0.685** |
| Baseline | 6919 | 1347 | 5572 | 0 | 6919 | 0 | 5572 | 0 | 1347 | 0.805 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **9** | **6919** | **1347** | **5572** | **743** | **6176** | **368** | **5319** | **375** | **857** | **0.822** | **0.300** | **0.495** | **0.374** | **0.934** | **0.617** |
| Baseline | 5204 | 1206 | 3998 | 0 | 5204 | 0 | 3998 | 0 | 1206 | 0.768 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **10** | **5204** | **1206** | **3998** | **506** | **4698** | **209** | **3821** | **297** | **877** | **0.774** | **0.192** | **0.413** | **0.263** | **0.928** | **0.560** |
| | | | | | **Performance Metrics Average** | | | | | **0.833** | **0.365** | **0.538** | **0.433** | **0.934** | **0.650** |
| | | | | | **Average Accuracy Baseline** | | | | | **0.803** | **Average BCR Baseline** | | | | **0.500** |

#### 5.5.1.2 HMM Algorithm

The Hidden Markov Model (HMM) is one of the probabilistic sequence classifiers. HMM is used to compute the probabilistic distribution of a sequence of observations and assign the best sequence of labels that are with the highest probability for occurrences. The HMM model has many applications in Natural Language Processing (NLP), such as; part of speech tagging, sentence segmentation, and information extraction. More detail about this model is in section 3.4.2.

The next two sections present the results of punctuation marks prediction using the BAQ Corpus for training and testing the HMM. Both categories of POS tag (*i.e.* 3 POS and 10 POS) are used together with word sequences as features in the experiments.

##### 5.5.1.2.1 Results of HMM Punctuation Marks Prediction using a Three-POS Tag Set

Table 5.5.1.2.1.1 presents the results of predicting the punctuation marks in the BAQ Corpus using the HMM model in 10 experiments. The word and POS tag (*i.e.* 3 POS [noun, verb, and particle]) are used by the HMM model for predicting punctuation mark or *nopunc* after each word in the dataset. The average accuracy scored is 84.1% with at a 3.8% increment above the average baseline accuracy of 80.3% in the 10 experiments. The average BCR rate is 63.9% at an increment of 13.9% above the average BCR baseline of 50%.

On the other hand, the average recall is scored at 32%, the average precision at 62%, and the average F-score scored at 41.9%. Furthermore, the average specificity is scored at 95.8%. Since the dataset used in this research is imbalanced (as explained in section 5.2), the recall, precision, F-score, and specificity measurements are misleading;

therefore, the measurements for comparing experiment results shall be are the accuracy and BCR measurements.

#### 5.5.1.2.2 Results of HMM Punctuation Marks Prediction using a Ten-POS Tag Set

This subsection presents the results of punctuation mark prediction using the ten POS tags feature for training and testing the HMM algorithm on the BAQ Corpus.

Table 5.5.1.2.2.1 shows the results of this punctuation mark in the 10 experiments. The HMM model uses the word and POS tag (i.e. 10 POS [noun, adverb, pronoun, etc.]) features for predicting whether each word is to be followed by a punctuation mark or by *nopunc*. The average accuracy rate for the 10 experiments is scored at 84.1% at a 3.8 increment above the average baseline accuracy of 80.3%. We also gain an increment of 13.7% in average BCR value above the 50% baseline, the attained BCR rate being 63.7%.

The average rates of recall, precision, and F-score are respectively 31.6%, 62.4%, and 41.7%. Specificity, on the other hand, averages 95.9%. As explained in earlier sections, none of these metrics will be commented on here because they are essentially misleading when the data set is skewed. We will use instead the accuracy and BCR measurements for comparing experiment results.

Figure 5.5.1.2.1 illustrates HMM's automatic punctuation prediction in a piece of Qur'anic text. The HMM model uses the word and the 3-POS tag set as features for predicting punctuation marks. Agreement between the original Qur'anic punctuation and automatically punctuate highlighted in green. Cases of disagreement are highlighted in red.

**Original punctuated Quran text**

وَيَوْمَ يَحْشُرُهُمْ كَأَنْ لَمْ يَلْبَثُوا إِلَّا سَاعَةً مِنَ النَّهَارِ يَتَعَارَفُونَ بَيْنَهُمْ. قَدْ خَسِرَ الَّذِينَ كَذَّبُوا بِلِقَاءِ اللَّهِ، وَمَا كَانُوا مُهْتَدِينَ.
وَإِمَّا نُرِيَنَّكَ بَعْضَ الَّذِي نَعِدُهُمْ أَوْ نَتَوَفَّيَنَّكَ فَإِلَيْنَا مَرْجِعُهُمْ، ثُمَّ اللَّهُ شَهِيدٌ عَلَى مَا يَفْعَلُونَ. وَلِكُلِّ أُمَّةٍ رَسُولٌ، فَإِذَا
جَاءَ رَسُولُهُمْ قُضِيَ بَيْنَهُمْ بِالْقِسْطِ وَهُمْ لَا يُظْلَمُونَ. وَيَقُولُونَ: مَتَى هَذَا الْوَعْدُ إِنْ كُنْتُمْ صَادِقِينَ؟ قُلْ: لَا أَمْلِكُ لِنَفْسِي
ضَرًّا وَلَا نَفْعًا إِلَّا مَا شَاءَ اللَّهُ، لِكُلِّ أُمَّةٍ أَجَلٌ، إِذَا جَاءَ أَجَلُهُمْ فَلَا يَسْتَأْخِرُونَ سَاعَةً وَلَا يَسْتَقْدِمُونَ. سورة يونس
(٤٥-٤٩)

**Automatic punctuation of the Quran text**

وَيَوْمَ يَحْشُرُهُمْ كَأَنْ لَمْ يَلْبَثُوا إِلَّا سَاعَةً مِنَ النَّهَارِ يَتَعَارَفُونَ بَيْنَهُمْ قَدْ خَسِرَ الَّذِينَ كَذَّبُوا بِلِقَاءِ اللَّهِ وَمَا كَانُوا مُهْتَدِينَ.
وَإِمَّا نُرِيَنَّكَ بَعْضَ الَّذِي نَعِدُهُمْ أَوْ نَتَوَفَّيَنَّكَ فَإِلَيْنَا مَرْجِعُهُمْ: ثُمَّ اللَّهُ شَهِيدٌ. عَلَى مَا يَفْعَلُونَ. وَلِكُلِّ أُمَّةٍ رَسُولٌ فَإِذَا
جَاءَ رَسُولُهُمْ قُضِيَ بَيْنَهُمْ بِالْقِسْطِ، وَهُمْ لَا يُظْلَمُونَ. وَيَقُولُونَ: مَتَى هَذَا الْوَعْدُ إِنْ كُنْتُمْ صَادِقِينَ. قُلْ: لَا أَمْلِكُ
لِنَفْسِي.. ضَرًّا وَلَا نَفْعًا إِلَّا مَا شَاءَ اللَّهُ لِكُلِّ أُمَّةٍ أَجَلٌ إِذَا جَاءَ أَجَلُهُمْ فَلَا يَسْتَأْخِرُونَ. سَاعَةً وَلَا يَسْتَقْدِمُونَ.
سورة يونس (٤٥-٤٩)

Figure 5.5.1.2.1: HMM automatic punctuation mark prediction in a Qur'anic text using a 3-POS tag set.

Table 5.5.1.2.1.1: HMM punctuation mark prediction a 3-POS tag set.

| | | Gold | | Prediction | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Exp. #** | **N** | **Punctuation** | **Non-Punc** | **Punctuation** | **Non-Punc** | **TP** | **TN** | **FP** | **FN** | **Accuracy** | **Recall** | **Precision** | **F-score** | **Specificity** | **BCR** |
| Baseline | 8523 | 1619 | 6904 | 0 | 8523 | 0 | 6904 | 0 | 1619 | 0.810 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **1** | **8523** | **1619** | **6904** | **786** | **7737** | **494** | **6721** | **292** | **1016** | **0.847** | **0.327** | **0.628** | **0.430** | **0.958** | **0.643** |
| Baseline | 9378 | 1900 | 7478 | 0 | 9387 | 0 | 7478 | 0 | 1900 | 0.797 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **2** | **9378** | **1900** | **7478** | **740** | **8638** | **458** | **7326** | **282** | **1312** | **0.830** | **0.259** | **0.619** | **0.365** | **0.963** | **0.611** |
| Baseline | 9810 | 1853 | 7957 | 0 | 9810 | 0 | 7957 | 0 | 1853 | 0.811 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **3** | **9810** | **1853** | **7957** | **954** | **8856** | **587** | **7746** | **367** | **1110** | **0.849** | **0.346** | **0.615** | **0.443** | **0.955** | **0.650** |
| Baseline | 8517 | 1659 | 6858 | 0 | 8517 | 0 | 6858 | 0 | 1659 | 0.805 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **4** | **8517** | **1659** | **6858** | **773** | **7744** | **523** | **6719** | **250** | **1025** | **0.850** | **0.338** | **0.677** | **0.451** | **0.964** | **0.651** |
| Baseline | 8050 | 1490 | 6560 | 0 | 8050 | 0 | 6560 | 0 | 1490 | 0.815 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **5** | **8050** | **1490** | **6560** | **675** | **7375** | **399** | **6401** | **276** | **974** | **0.845** | **0.291** | **0.591** | **0.390** | **0.959** | **0.625** |
| Baseline | 6737 | 1345 | 5392 | 0 | 6737 | 0 | 5392 | 0 | 1345 | 0.800 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **6** | **6737** | **1345** | **5392** | **788** | **5949** | **517** | **5243** | **271** | **706** | **0.855** | **0.423** | **0.656** | **0.514** | **0.951** | **0.687** |
| Baseline | 7129 | 1344 | 5785 | 0 | 7129 | 0 | 5785 | 0 | 1344 | 0.811 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **7** | **7129** | **1344** | **5785** | **756** | **6373** | **478** | **5623** | **278** | **750** | **0.856** | **0.389** | **0.632** | **0.482** | **0.953** | **0.671** |
| Baseline | 7163 | 1382 | 5781 | 0 | 7163 | 0 | 5781 | 0 | 1382 | 0.807 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **8** | **7163** | **1382** | **5781** | **736** | **6427** | **501** | **5638** | **235** | **789** | **0.857** | **0.388** | **0.681** | **0.495** | **0.960** | **0.674** |
| Baseline | 6919 | 1347 | 5572 | 0 | 6919 | 0 | 5572 | 0 | 1347 | 0.805 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **9** | **6919** | **1347** | **5572** | **569** | **6350** | **330** | **5411** | **239** | **939** | **0.830** | **0.260** | **0.580** | **0.359** | **0.958** | **0.609** |
| Baseline | 5204 | 1206 | 3998 | 0 | 5204 | 0 | 3998 | 0 | 1206 | 0.768 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **10** | **5204** | **1206** | **3998** | **383** | **4821** | **200** | **3900** | **183** | **921** | **0.788** | **0.178** | **0.522** | **0.266** | **0.955** | **0.567** |
| | | | | | **Performance Metrics Average** | | | | | **0.841** | **0.320** | **0.620** | **0.419** | **0.958** | **0.639** |
| | | | | | **Average Accuracy Baseline** | | | | | **0.803** | **Average BCR Baseline** | | | | **0.500** |

Table 5.5.1.2.2.1: HMM punctuation mark prediction a 10-POS tag set.

| | | Gold | | Prediction | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Exp. #** | **N** | **Punctuation** | **Non-Punc** | **Punctuation** | **Non-Punc** | **TP** | **TN** | **FP** | **FN** | **Accuracy** | **Recall** | **Precision** | **F-score** | **Specificity** | **BCR** |
| Baseline | 8523 | 1619 | 6904 | 0 | 8523 | 0 | 6904 | 0 | 1619 | 0.810 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **1** | **8523** | **1619** | **6904** | **779** | **7744** | **491** | **6725** | **288** | **1019** | **0.847** | **0.325** | **0.630** | **0.429** | **0.959** | **0.642** |
| Baseline | 9378 | 1900 | 7478 | 0 | 9378 | 0 | 7478 | 0 | 1900 | 0.797 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **2** | **9378** | **1900** | **7478** | **727** | **8651** | **453** | **7328** | **274** | **1323** | **0.830** | **0.255** | **0.623** | **0.362** | **0.964** | **0.610** |
| Baseline | 9810 | 1853 | 7957 | 0 | 9810 | 0 | 7957 | 0 | 1853 | 0.811 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **3** | **9810** | **1853** | **7957** | **961** | **8849** | **593** | **7744** | **368** | **1105** | **0.850** | **0.349** | **0.617** | **0.446** | **0.955** | **0.652** |
| Baseline | 8517 | 1659 | 6858 | 0 | 8517 | 0 | 6858 | 0 | 1659 | 0.805 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **4** | **8517** | **1659** | **6858** | **769** | **7748** | **521** | **6720** | **248** | **1028** | **0.850** | **0.336** | **0.678** | **0.450** | **0.964** | **0.650** |
| Baseline | 8050 | 1490 | 6560 | 0 | 8050 | 0 | 6560 | 0 | 1490 | 0.815 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **5** | **8050** | **1490** | **6560** | **672** | **7378** | **392** | **6399** | **280** | **979** | **0.844** | **0.286** | **0.583** | **0.384** | **0.958** | **0.622** |
| Baseline | 6737 | 1345 | 5392 | 0 | 6737 | 0 | 5392 | 0 | 1345 | 0.800 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **6** | **6737** | **1345** | **5392** | **784** | **5953** | **519** | **5246** | **265** | **707** | **0.856** | **0.423** | **0.662** | **0.516** | **0.952** | **0.688** |
| Baseline | 7129 | 1344 | 5785 | 0 | 7129 | 0 | 5785 | 0 | 1344 | 0.811 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **7** | **7129** | **1344** | **5785** | **728** | **6401** | **471** | **5640** | **257** | **761** | **0.857** | **0.382** | **0.647** | **0.481** | **0.956** | **0.669** |
| Baseline | 7163 | 1382 | 5781 | 0 | 7163 | 0 | 5781 | 0 | 1382 | 0.807 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **8** | **7163** | **1382** | **5781** | **723** | **6440** | **494** | **5642** | **229** | **798** | **0.857** | **0.382** | **0.683** | **0.490** | **0.961** | **0.672** |
| Baseline | 6919 | 1347 | 5572 | 0 | 6919 | 0 | 5572 | 0 | 1347 | 0.805 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **9** | **6919** | **1347** | **5572** | **552** | **6367** | **324** | **5416** | **228** | **951** | **0.830** | **0.254** | **0.587** | **0.355** | **0.960** | **0.607** |
| Baseline | 5204 | 1206 | 3998 | 0 | 5204 | 0 | 3998 | 0 | 1206 | 0.768 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **10** | **5204** | **1206** | **3998** | **353** | **4851** | **188** | **3908** | **165** | **943** | **0.787** | **0.166** | **0.533** | **0.253** | **0.959** | **0.563** |
| | | | | | **Performance Metrics Average** | | | | | **0.841** | **0.316** | **0.624** | **0.417** | **0.959** | **0.637** |
| | | | | | **Average Accuracy Baseline** | | | | | **0.803** | **Average BCR Baseline** | | | | **0.500** |

#### 5.5.1.3 CRF Algorithm

CRF model is one of the statistical modeling methods. CRF model used for building a model used in structure prediction. Many types of CRF models such as; linear-chine CRF, dynamic CRF, skip-chine models, have many applications in NLP such as; sentence phrasing, named entity recognition, POS tagging, etc. More details about this model in section 3.4.3.

The next two sections discuss the experiments results of punctuation marks prediction. The CRF model was trained and tested using the BAQ Corpus for ten experiments for both categories of POS tags (*i.e.* 3 POS tags and 10 POS tags).

##### 5.5.1.3.1 Results of CRF Punctuation Mark Prediction using a Three-POS Tag Set

This subsection presents the results of training and testing the CRF model for the ten experiments using the BAQ Corpus with 3-POS tags selected as a feature for prediction.

The CRF model uses the word and its POS tags (*i.e.* 3 POS [noun, verb, and particle]) as features to predict punctuation marks or *nopunc* after each word in the dataset for each of the ten experiments. Table 5.5.1.3.1.1 shows the results of punctuation marks prediction using the CRF model. The average BCR scored 86.4% with increment of 36.4% over the average BCR baseline 50%. Also, the average accuracy gained 93.4% with increment of 13.1% over average accuracy baseline 80.3%.

On the other hand, the recall average scored 75.5%, the average precision scored 86.4%, and the average f-score scored 80.6%. Furthermore, the average specificity scored 97.4%. As we explained in section 5.2, these measures yield misleading results when

the data is imbalanced. Therefore, we will depend on the accuracy and BCR measures to compare the results of the experiments.

#### 5.5.1.3.2 Results of CRF Punctuation Mark Prediction using a Ten-POS Tag Set

This subsection presents the results of training and testing the CRF model using the BAQ Corpus with ten POS tag selected as a feature for prediction for the ten experiments.

Table 5.5.1.3.2.1 presents the results of the punctuation marks prediction using the CRF model for ten experiments. The CRF model in these experiments uses the word and POS tag (*i.e.* 10 POS [nominal, pronoun, verb, etc.]) to predict punctuation marks or *nopunc* after each word in the dataset. The average accuracy scored 92.2% with increment of 11.9% over the average baseline accuracy 80.3%. We also gained and increment of 35.8% in average BCR that scored 85.8% over the average BCR baseline of 50%.

As we explained in section 5.2, the precision, recall, f-score, and specificity yield misleading when the data is imbalanced. The average recall scored 76.0%.The average precision scored 79.1%, and the average f-score scored 77.4%. On the other hand, the average specificity scored 95.7%.

The CRF algorithm results with 3-POS tags for the prediction of punctuation marks proved its superiority over other algorithms (*i.e.* N-gram and HMM). This supremacy of the CRF model back to its feature selection characteristic, which enables this model to use different number of earlier and later features (*i.e.* words and their corresponding POS tag) that helps to investigate the tested dataset.

Figure 5.5.1.3.1 presents an example of the Qur'an text where it was automatically punctuated with punctuation marks using the CRF model with 3 POS tags. The consensuses between the original Qur'an text and the automatically punctuated text are highlighted in green. Controversies cases are highlighted in red.

**Original punctuated Quran text**

المص. كِتَابٌ أُنْزِلَ إِلَيْكَ، فَلَا يَكُنْ فِي صَدْرِكَ حَرَجٌ مِنْهُ، لِتُنْذِرَ بِهِ، وَذِكْرَى لِلْمُؤْمِنِينَ. اتَّبِعُوا مَا أُنْزِلَ إِلَيْكُمْ مِنْ رَبِّكُمْ، وَلَا تَتَّبِعُوا مِنْ دُونِهِ أَوْلِيَاءَ. قَلِيلًا مَا تَذَكَّرُونَ. وَكَمْ مِنْ قَرْيَةٍ أَهْلَكْنَاهَا، فَجَاءَهَا بَأْسُنَا بَيَاتًا أَوْ هُمْ قَائِلُونَ. فَمَا كَانَ دَعْوَاهُمْ إِذْ جَاءَهُمْ بَأْسُنَا إِلَّا أَنْ قَالُوا: إِنَّا كُنَّا ظَالِمِينَ. فَلَنَسْأَلَنَّ الَّذِينَ أُرْسِلَ إِلَيْهِمْ، وَلَنَسْأَلَنَّ الْمُرْسَلِينَ. سورة الأعراف (١-7)

**Automatic punctuation of the Quran text**

المص. كِتَابٌ أُنْزِلَ إِلَيْكَ فَلَا يَكُنْ فِي صَدْرِكَ حَرَجٌ مِنْهُ، لِتُنْذِرَ بِهِ وَذِكْرَى لِلْمُؤْمِنِينَ. اتَّبِعُوا مَا أُنْزِلَ إِلَيْكُمْ مِنْ رَبِّكُمْ، وَلَا تَتَّبِعُوا مِنْ دُونِهِ أَوْلِيَاءَ. قَلِيلًا مَا تَذَكَّرُونَ. وَكَمْ مِنْ قَرْيَةٍ أَهْلَكْنَاهَا فَجَاءَهَا بَأْسُنَا بَيَاتًا أَوْ هُمْ قَائِلُونَ. فَمَا كَانَ دَعْوَاهُمْ إِذْ جَاءَهُمْ بَأْسُنَا إِلَّا أَنْ قَالُوا: إِنَّا كُنَّا ظَالِمِينَ. فَلَنَسْأَلَنَّ الَّذِينَ أُرْسِلَ إِلَيْهِمْ وَلَنَسْأَلَنَّ الْمُرْسَلِينَ. سورة الأعراف (١-7)

Figure 5.5.1.3.1: Automatic prediction of punctuation mark for a Qur'an text using CRF model with the 3-POS tags.

Table 5.5.1.3.1.1: Punctuation marks prediction using CRF algorithm with 3-POS tag set.

| | | Gold | | Prediction | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Exp. #** | **N** | **Punctuation** | **Non-Punc** | **Punctuation** | **Non-Punc** | **TP** | **TN** | **FP** | **FN** | **Accuracy** | **Recall** | **Precision** | **F-score** | **Specificity** | **BCR** |
| Baseline | 8523 | 1619 | 6904 | 0 | 8523 | 0 | 6904 | 0 | 1619 | 0.810 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **1** | **8523** | **1619** | **6904** | **1276** | **7247** | **1109** | **6859** | **167** | **388** | **0.935** | **0.741** | **0.869** | **0.800** | **0.976** | **0.859** |
| Baseline | 9378 | 1900 | 7478 | 0 | 9378 | 0 | 7478 | 0 | 1900 | 0.797 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **2** | **9378** | **1900** | **7478** | **1519** | **7859** | **1242** | **7383** | **277** | **476** | **0.920** | **0.723** | **0.818** | **0.767** | **0.964** | **0.843** |
| Baseline | 9810 | 1853 | 7957 | 0 | 9810 | 0 | 7957 | 0 | 1853 | 0.811 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **3** | **9810** | **1853** | **7957** | **1431** | **8379** | **1212** | **7896** | **219** | **483** | **0.928** | **0.715** | **0.847** | **0.775** | **0.973** | **0.844** |
| Baseline | 8517 | 1659 | 6858 | 0 | 8517 | 0 | 6858 | 0 | 1659 | 0.805 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **4** | **8517** | **1659** | **6858** | **1320** | **7197** | **1146** | **6800** | **174** | **397** | **0.933** | **0.743** | **0.868** | **0.801** | **0.975** | **0.859** |
| Baseline | 8050 | 1490 | 6560 | 0 | 8050 | 0 | 6560 | 0 | 1490 | 0.815 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **5** | **8050** | **1490** | **6560** | **1179** | **6871** | **1032** | **6527** | **147** | **344** | **0.939** | **0.750** | **0.875** | **0.808** | **0.978** | **0.864** |
| Baseline | 6737 | 1345 | 5392 | 0 | 6737 | 0 | 5392 | 0 | 1345 | 0.800 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **6** | **6737** | **1345** | **5392** | **1121** | **5616** | **966** | **5359** | **155** | **257** | **0.939** | **0.790** | **0.862** | **0.824** | **0.972** | **0.881** |
| Baseline | 7129 | 1344 | 5785 | 0 | 7129 | 0 | 5785 | 0 | 1344 | 0.811 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **7** | **7129** | **1344** | **5785** | **1072** | **6057** | **939** | **5763** | **133** | **294** | **0.940** | **0.762** | **0.876** | **0.815** | **0.977** | **0.869** |
| Baseline | 7163 | 1382 | 5781 | 0 | 7163 | 0 | 5781 | 0 | 1382 | 0.807 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **8** | **7163** | **1382** | **5781** | **1136** | **6027** | **995** | **5749** | **141** | **278** | **0.942** | **0.782** | **0.876** | **0.826** | **0.976** | **0.879** |
| Baseline | 6919 | 1347 | 5572 | 0 | 6919 | 0 | 5572 | 0 | 1347 | 0.805 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **9** | **6919** | **1347** | **5572** | **955** | **5964** | **952** | **5543** | **116** | **308** | **0.939** | **0.756** | **0.891** | **0.818** | **0.980** | **0.868** |
| Baseline | 5204 | 1206 | 3998 | 0 | 5204 | 0 | 3998 | 0 | 1206 | 0.768 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **10** | **5204** | **1206** | **3998** | **995** | **4209** | **854** | **3983** | **141** | **226** | **0.929** | **0.791** | **0.858** | **0.823** | **0.966** | **0.878** |
| | | | | | **Performance Metrics Average** | | | | | **0.934** | **0.755** | **0.864** | **0.806** | **0.974** | **0.864** |
| | | | | | **Average Accuracy Baseline** | | | | | **0.803** | **Average BCR Baseline** | | | | **0.500** |

Table 5.5.1.3.2.1: Punctuation marks prediction using CRF algorithm with 10-POS tag set.

| | | Gold | | Prediction | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Exp. #** | **N** | **Punctuation** | **Non-Punc** | **Punctuation** | **Non-Punc** | **TP** | **TN** | **FP** | **FN** | **Accuracy** | **Recall** | **Precision** | **F-score** | **Specificity** | **BCR** |
| Baseline | 8523 | 1619 | 6904 | 0 | 8523 | 0 | 6904 | 0 | 1619 | 0.810 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **1** | **8523** | **1619** | **6904** | **1377** | **7146** | **1107** | **6781** | **270** | **365** | **0.925** | **0.752** | **0.804** | **0.777** | **0.962** | **0.857** |
| Baseline | 9378 | 1900 | 7478 | 0 | 9378 | 0 | 7478 | 0 | 1900 | 0.797 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **2** | **9378** | **1900** | **7478** | **1385** | **7993** | **1076** | **7374** | **309** | **619** | **0.901** | **0.635** | **0.777** | **0.699** | **0.960** | **0.797** |
| Baseline | 9810 | 1853 | 7957 | 0 | 9810 | 0 | 7957 | 0 | 1853 | 0.811 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **3** | **9810** | **1853** | **7957** | **1474** | **8336** | **1146** | **7802** | **328** | **534** | **0.912** | **0.682** | **0.777** | **0.727** | **0.960** | **0.821** |
| Baseline | 8517 | 1659 | 6858 | 0 | 8517 | 0 | 6858 | 0 | 1659 | 0.805 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **4** | **8517** | **1659** | **6858** | **1391** | **7126** | **1133** | **6729** | **258** | **397** | **0.923** | **0.741** | **0.815** | **0.776** | **0.963** | **0.852** |
| Baseline | 8050 | 1490 | 6560 | 0 | 8050 | 0 | 6560 | 0 | 1490 | 0.815 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **5** | **8050** | **1490** | **6560** | **1369** | **6681** | **1076** | **6397** | **293** | **284** | **0.928** | **0.791** | **0.786** | **0.789** | **0.956** | **0.874** |
| Baseline | 6737 | 1345 | 5392 | 0 | 6737 | 0 | 5392 | 0 | 1345 | 0.800 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **6** | **6737** | **1345** | **5392** | **1228** | **5509** | **961** | **5273** | **267** | **236** | **0.925** | **0.803** | **0.783** | **0.793** | **0.952** | **0.877** |
| Baseline | 7129 | 1344 | 5785 | 0 | 7129 | 0 | 5785 | 0 | 1344 | 0.811 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **7** | **7129** | **1344** | **5785** | **1240** | **5889** | **1000** | **5663** | **240** | **226** | **0.935** | **0.816** | **0.806** | **0.811** | **0.959** | **0.888** |
| Baseline | 7163 | 1382 | 5781 | 0 | 7163 | 0 | 5781 | 0 | 1382 | 0.807 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **8** | **7163** | **1382** | **5781** | **1278** | **5885** | **1045** | **5662** | **233** | **223** | **0.936** | **0.824** | **0.818** | **0.821** | **0.960** | **0.892** |
| Baseline | 6919 | 1347 | 5572 | 0 | 6919 | 0 | 5572 | 0 | 1347 | 0.805 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **9** | **6919** | **1347** | **5572** | **1208** | **5711** | **952** | **5452** | **256** | **259** | **0.926** | **0.786** | **0.788** | **0.787** | **0.955** | **0.871** |
| Baseline | 5204 | 1206 | 3998 | 0 | 5204 | 0 | 3998 | 0 | 1206 | 0.768 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **10** | **5204** | **1206** | **3998** | **1043** | **4161** | **792** | **3925** | **251** | **236** | **0.906** | **0.770** | **0.759** | **0.765** | **0.940** | **0.855** |
| | | | | | **Performance Metrics Average** | | | | | **0.922** | **0.760** | **0.791** | **0.774** | **0.957** | **0.858** |
| | | | | | **Average Accuracy Baseline** | | | | | **0.803** | **Average BCR Baseline** | | | | **0.500** |

### 5.5.2 Prediction of Sentence Terminals (Two Class Problem)

This section presents the results for the ten experiments for the task of sentence terminal prediction after training and testing three ML algorithms (*i.e.* N-gram, HMM, and CRF) using the BAQ Corpus. Two categories of the POS tags (*i.e.* 3 POS and 10 POS) were selected as features for the three algorithms.

#### 5.5.2.1 N-gram Algorithm

The N-gram model is one of the probabilistic language models that are concerned with predicting the next item in a sequence of observations. Predicting using the N-gram model is performed by computing the probability of a sequence of observations and selecting the highest probability sequence. The N-gram model has many applications in the Natural Language processing field such as; segmentation, Part Of Speech tagging, etc. More detail about this model is found in section 3.4.1. The N-gram model was trained and tested on the BAQ Corpus for sentence terminal prediction. The obtained results for experiments, using both categories of POS-tags as features, are described in the next two sections.

##### 5.5.2.1.1 Results of N-gram Sentence Terminal Prediction using a Three-POS Tag Set

This subsection presents the results of sentence terminal prediction after training and testing the N-gram model using the BAQ Corpus with 3 POS tags selected as a feature for prediction.

Table 5.5.2.1.1.1 shows the results of sentence terminal prediction using the N-gram model for ten experiments. The N-gram model was trained and tested using the BAQ

Corpus for sentence terminal prediction, where the N-gram model used the word and POS tags (*i.e.* 3 POS [noun, verb, and particle]) to predict the sentence *terminal* (*i.e.* the word located at the end of the sentence) or *non* (*i.e.* the word does not located at the end of the sentence). The average accuracy scored 91.8% with increment of 3.0% over the average accuracy baseline 88.8%. The average BCR gained increment of 20.0% over the average BCR baseline 50.0% which scored 70.0%.

The average recall scored 41.7%, the average precision scored 73.5%, the average f-score scored 52.8%, and the average specificity scored 98.2%. Positive cases (*i.e.* sentence terminals) present less than 12% compared to more than 88% of negative cases (*i.e. non*) in the dataset (*i.e.* BAQ Corpus). The previous two percentages indicate that our dataset is skewed (*i.e.* imbalanced) data towards *non* (*i.e.* no sentence terminal). The recall, precision, and f-score present correct predictions for positive cases (*i.e.* how many sentence terminals were predicted as sentence terminals). On the other hand, the specificity presents the correct predictions for negative cases (*i.e.* how many *non* was predicted as non). Because our dataset is imbalanced, this means that the four metrics (*i.e.* recall, precision, f-score, and specificity) yield misleading results, more details about this four metrics is found in section 5.2. While the accuracy rate and BCR metrics can deal with the problem of skewed data, therefore, we will depend on the accuracy rate and BCR metrics for comparing the results of the three ML algorithms of sentence terminal prediction experiments. More information about skewed data is found in section 5.3.

For each experiment we listed the positive and negative prediction values representing elements of confusion matrix. TPs, TNs represent the correct predictions of sentence terminals and *non* (*i.e*. no sentence terminal) respectively. FPs and FNs represent the

wrong (false) predictions of sentence terminals and *non* (*i.e.* no sentence terminal) respectively. The number of TPs indicates primitive measure of the algorithm performance.

#### 5.5.2.1.2 Results of N-gram Sentence Terminal Prediction using a Ten-POS Tag Set

This subsection presents the results of ten experiments after training and testing the N-gram model using the BAQ Corpus with ten POS tags selected as a feature for prediction..

Table 5.5.2.1.2.1 shows the results of predicting sentence terminals using the N-gram model for ten experiments. The N-gram model in these experiments uses the word and POS tag (*i.e.* 10 POS [nominal, verb, pronoun, etc.]) to predict sentence terminals or *non* (*i.e.* the word is not the end of the sentence) after each word in the tested dataset. The average accuracy scored 91.8% with increment of 3.0% over the average accuracy baseline 88.8%. Also we gained an increment of 19.6% over the average BCR baseline 50% which scored 69.6%.

The average recall scored 41%, the average precision scored 74.2%, and the average precision scored 52.3%. On the other hand, the average specificity scored 98.3%. As we explained in section 5.2, the recall, precision, f-score, and specificity yield misleading when the dataset is imbalanced. Therefore, we will concentrate on the accuracy and BCR for comparing the results of the conducted experiments with different ML algorithms.

The discrepancy between the results of using both categories of POS tags (*i.e.* 3 POS and 10 POS) as features for experiments insures the assumption that the language reader

need not possess a high level of linguistic competence to learn how to correctly punctuate a piece of text.

Figure 5.5.2.1.1 shows a sample of text from the Qur'an where sentence terminals were automatically added using N-gram algorithm. In this example, the N-gram algorithm uses 3 POS tags and the word as features for predicting the sentence terminals in the sample. The consensuses between the original Qur'an text and the automatically punctuated text are highlighted in green. Controversies cases are highlighted in red. Where the (*) symbol indicates sentence terminal.

**Original punctuated Quran text**

حم * وَالْكِتَابِ الْمُبِينِ * إِنَّا أَنْزَلْنَاهُ فِي لَيْلَةٍ مُبَارَكَةٍ إِنَّا كُنَّا مُنْذِرِينَ * فِيهَا يُفْرَقُ كُلُّ أَمْرٍ حَكِيمٍ * أَمْرًا مِنْ عِنْدِنَا إِنَّا كُنَّا مُرْسِلِينَ * رَحْمَةً مِنْ رَبِّكَ إِنَّهُ هُوَ السَّمِيعُ الْعَلِيمُ * رَبِّ السَّمَاوَاتِ وَالْأَرْضِ وَمَا بَيْنَهُمَا إِنْ كُنْتُمْ مُوقِنِينَ * لَا إِلَهَ إِلَّا هُوَ يُحْيِي وَيُمِيتُ رَبُّكُمْ وَرَبُّ آبَائِكُمُ الْأَوَّلِينَ * بَلْ هُمْ فِي شَكٍّ يَلْعَبُونَ * فَارْتَقِبْ يَوْمَ تَأْتِي السَّمَاءُ بِدُخَانٍ مُبِينٍ يَغْشَى النَّاسَ هَذَا عَذَابٌ أَلِيمٌ * رَبَّنَا اكْشِفْ عَنَّا الْعَذَابَ إِنَّا مُؤْمِنُونَ * أَنَّى لَهُمُ الذِّكْرَى وَقَدْ جَاءَهُمْ رَسُولٌ مُبِينٌ * سورة الدخان (١-١٣)

**Automatic punctuation of the Quran text**

حم * وَالْكِتَابِ الْمُبِينِ * إِنَّا أَنْزَلْنَاهُ فِي لَيْلَةٍ مُبَارَكَةٍ إِنَّا كُنَّا مُنْذِرِينَ * فِيهَا يُفْرَقُ كُلُّ أَمْرٍ حَكِيمٍ أَمْرًا مِنْ عِنْدِنَا إِنَّا كُنَّا مُرْسِلِينَ * رَحْمَةً مِنْ رَبِّكَ إِنَّهُ هُوَ السَّمِيعُ الْعَلِيمُ * رَبِّ السَّمَاوَاتِ وَالْأَرْضِ وَمَا بَيْنَهُمَا إِنْ كُنْتُمْ مُوقِنِينَ * لَا إِلَهَ إِلَّا هُوَ يُحْيِي وَيُمِيتُ رَبُّكُمْ وَرَبُّ آبَائِكُمُ الْأَوَّلِينَ * بَلْ هُمْ فِي شَكٍّ يَلْعَبُونَ * فَارْتَقِبْ يَوْمَ تَأْتِي السَّمَاءُ بِدُخَانٍ مُبِينٍ * يَغْشَى النَّاسَ هَذَا عَذَابٌ أَلِيمٌ * رَبَّنَا اكْشِفْ عَنَّا الْعَذَابَ إِنَّا مُؤْمِنُونَ أَنَّى لَهُمُ الذِّكْرَى وَقَدْ جَاءَهُمْ رَسُولٌ مُبِينٌ * سورة الدخان (١-١٣)

Figure 5.5.2.1.1: Automatic sentence terminal prediction for a Qur'an text using N-gram model with the 3-POS tags.

Table 5.5.2.1.1.1: Sentence terminal prediction using N-gram algorithm with 3-POS tag set.

| | | Gold | | Prediction | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Exp. # | N | Terminal | Non-Terminal | Terminal | Non-Terminal | TP | TN | FP | FN | Accuracy | Recall | Precision | F-score | Specificity | BCR |
| Baseline | 8523 | 836 | 7687 | 0 | 8523 | 0 | 7687 | 0 | 836 | 0.902 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **1** | **8523** | **836** | **7687** | **467** | **8056** | **354** | **7574** | **113** | **482** | **0.930** | **0.423** | **0.758** | **0.543** | **0.985** | **0.704** |
| Baseline | 9378 | 836 | 8542 | 0 | 9378 | 0 | 8542 | 0 | 836 | 0.911 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **2** | **9378** | **836** | **8542** | **496** | **8882** | **347** | **8393** | **149** | **489** | **0.932** | **0.415** | **0.700** | **0.521** | **0.983** | **0.699** |
| Baseline | 9810 | 836 | 8974 | 0 | 9810 | 0 | 8974 | 0 | 836 | 0.915 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **3** | **9810** | **836** | **8974** | **615** | **9195** | **448** | **8807** | **167** | **388** | **0.943** | **0.536** | **0.728** | **0.618** | **0.981** | **0.759** |
| Baseline | 8517 | 836 | 7681 | 0 | 8517 | 0 | 7681 | 0 | 836 | 0.902 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **4** | **8517** | **836** | **7681** | **527** | **7990** | **399** | **7553** | **128** | **437** | **0.934** | **0.477** | **0.757** | **0.585** | **0.983** | **0.730** |
| Baseline | 8050 | 836 | 7214 | 0 | 8050 | 0 | 7214 | 0 | 836 | 0.896 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **5** | **8050** | **836** | **7214** | **348** | **7702** | **233** | **7099** | **115** | **603** | **0.911** | **0.279** | **0.670** | **0.394** | **0.984** | **0.631** |
| Baseline | 6737 | 836 | 5901 | 0 | 6737 | 0 | 5901 | 0 | 836 | 0.876 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **6** | **6737** | **836** | **5901** | **524** | **6213** | **423** | **5800** | **101** | **413** | **0.924** | **0.506** | **0.807** | **0.622** | **0.983** | **0.744** |
| Baseline | 7129 | 836 | 6293 | 0 | 7129 | 0 | 6293 | 0 | 836 | 0.883 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **7** | **7129** | **836** | **6293** | **513** | **6616** | **399** | **6179** | **114** | **437** | **0.923** | **0.477** | **0.778** | **0.592** | **0.982** | **0.730** |
| Baseline | 7163 | 836 | 6327 | 0 | 7163 | 0 | 6327 | 0 | 836 | 0.883 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **8** | **7163** | **836** | **6327** | **497** | **6666** | **386** | **6216** | **111** | **450** | **0.922** | **0.462** | **0.777** | **0.579** | **0.982** | **0.722** |
| Baseline | 6919 | 836 | 6083 | 0 | 6919 | 0 | 6083 | 0 | 836 | 0.879 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **9** | **6919** | **836** | **6083** | **405** | **6514** | **288** | **5966** | **117** | **548** | **0.904** | **0.344** | **0.711** | **0.464** | **0.981** | **0.663** |
| Baseline | 5204 | 842 | 4362 | 0 | 5204 | 0 | 4362 | 0 | 842 | 0.838 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **10** | **5204** | **842** | **4362** | **321** | **4883** | **212** | **4253** | **109** | **630** | **0.858** | **0.252** | **0.660** | **0.365** | **0.975** | **0.613** |
| | | | | | **Performance Metrics Average** | | | | | **0.918** | **0.417** | **0.735** | **0.528** | **0.982** | **0.700** |
| | | | | | **Accuracy Baseline Average** | | | | | **0.888** | **BCR Baseline Average** | | | | **0.500** |

Table 5.5.2.1.2.1: Sentence terminal prediction using N-gram algorithm with 10-POS tag set.

| | | Gold | | Prediction | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Exp. #** | **N** | **Terminal** | **Non-Terminal** | **Terminal** | **Non-Terminal** | **TP** | **TN** | **FP** | **FN** | **Accuracy** | **Recall** | **Precision** | **F-score** | **Specificity** | **BCR** |
| Baseline | 8523 | 836 | 7687 | 0 | 8523 | 0 | 7687 | 0 | 836 | 0.902 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **1** | **8523** | **836** | **7687** | **469** | **8054** | **355** | **7573** | **114** | **481** | **0.930** | **0.425** | **0.757** | **0.544** | **0.985** | **0.705** |
| Baseline | 9378 | 836 | 8542 | 0 | 9378 | 0 | 8542 | 0 | 836 | 0.911 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **2** | **9378** | **836** | **8542** | **470** | **8908** | **332** | **8404** | **138** | **504** | **0.932** | **0.397** | **0.706** | **0.508** | **0.984** | **0.690** |
| Baseline | 9810 | 836 | 8974 | 0 | 9810 | 0 | 8974 | 0 | 836 | 0.915 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **3** | **9810** | **836** | **8974** | **615** | **9195** | **448** | **8807** | **167** | **388** | **0.943** | **0.536** | **0.728** | **0.618** | **0.981** | **0.759** |
| Baseline | 8517 | 836 | 7681 | 0 | 8517 | 0 | 7681 | 0 | 836 | 0.902 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **4** | **8517** | **836** | **7681** | **520** | **7997** | **397** | **7558** | **123** | **439** | **0.934** | **0.475** | **0.763** | **0.586** | **0.984** | **0.729** |
| Baseline | 8050 | 836 | 7214 | 0 | 8050 | 0 | 7214 | 0 | 836 | 0.896 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **5** | **8050** | **836** | **7214** | **333** | **7717** | **223** | **7104** | **110** | **613** | **0.910** | **0.267** | **0.670** | **0.382** | **0.985** | **0.626** |
| Baseline | 6737 | 836 | 5901 | 0 | 6737 | 0 | 5901 | 0 | 836 | 0.876 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **6** | **6737** | **836** | **5901** | **513** | **6224** | **421** | **5809** | **92** | **415** | **0.925** | **0.504** | **0.821** | **0.624** | **0.984** | **0.744** |
| Baseline | 7129 | 836 | 6293 | 0 | 7129 | 0 | 6293 | 0 | 836 | 0.883 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **7** | **7129** | **836** | **6293** | **490** | **6639** | **391** | **6194** | **99** | **445** | **0.924** | **0.468** | **0.798** | **0.590** | **0.984** | **0.726** |
| Baseline | 7163 | 836 | 6327 | 0 | 7163 | 0 | 6327 | 0 | 836 | 0.883 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **8** | **7163** | **836** | **6327** | **481** | **6682** | **378** | **6224** | **103** | **458** | **0.922** | **0.452** | **0.786** | **0.574** | **0.984** | **0.718** |
| Baseline | 6919 | 836 | 6083 | 0 | 6919 | 0 | 6083 | 0 | 836 | 0.879 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **9** | **6919** | **836** | **6083** | **387** | **6532** | **280** | **5976** | **107** | **556** | **0.904** | **0.335** | **0.724** | **0.458** | **0.982** | **0.659** |
| Baseline | 5204 | 842 | 4362 | 0 | 5204 | 0 | 4362 | 0 | 842 | 0.838 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **10** | **5204** | **842** | **4362** | **298** | **4906** | **200** | **4264** | **98** | **642** | **0.858** | **0.238** | **0.671** | **0.351** | **0.978** | **0.608** |
| | | | | | **Performance Metrics Average** | | | | | **0.918** | **0.410** | **0.742** | **0.523** | **0.983** | **0.696** |
| | | | | | **Accuracy Baseline Average** | | | | | **0.888** | **BCR Baseline Average** | | | | **0.500** |

### 5.5.2.2 HMM Algorithm

This section presents the results of training and testing the HMM model using the BAQ Corpus for ten experiments to predict sentence terminals. Two categories of POS tags were experimented (*i.e.* 3 POS and 10 POS)as features for the HMM algorithm.

### 5.4.2.1 Results of HMM Sentence Terminal Prediction using a Three-POS Tag Set

This subsection presents the results of training and testing the HMM model for ten experiments using the BAQ Corpus with three POS tags.

Table 5.4.2.1.1 presents the results of predicting sentence terminals using the HMM model for ten experiments. The HMM model uses the word and POS tag (*i.e.* 3 POS [noun, verb, and particle]) to predict sentence *terminal* or *non* (*i.e.* the word does not located at the end of the sentence) for the words in test dataset for the ten experiments.

The results show, that the average accuracy scored 91.8% with increment of 3.0% over the average accuracy baseline 88.8%, while the average BCR scored 69.2% over the average BCR baseline 50.0% with increment of 19.2%. The average recall scored 40.0%, the average precision scored 75.0%, and the average f-score scored 51.8%. Furthermore, the average specificity scored 98.4%.

### 5.4.2.2 Results of HMM Sentence Terminal Prediction using a Ten-POS Tag Set

This subsection presents the results of sentence terminal prediction after training and testing the HMM model using BAQ Corpus with ten POS tags selected as a feature for perdition.

The HMM model was trained and tested using the BAQ Corpus with 10 POS tags (nominal, verb, particle, pronoun, etc.). The HMM model uses the word and POS tags

(10 POS) for predicting sentence terminals. Table 19 shows the results of predicting sentence terminals using the HMM model for ten experiments. The average accuracy scored 91.8% over the average accuracy baseline 88.8% with increment of 3.0%. The average BCR rate scored 69.0% with increment of 19.0% over the average BCR baseline 50.0%. The average recall scored 39.4%, the average precision scored 76.1%, and the average f-score scored 51.4%. On the hand, the average specificity scored 98.5%. While our dataset in this research is imbalanced, and the four metrics (*i.e.* recall, precision, f-score, and specificity) do not have the ability to deal with imbalanced data, we would concentrate on the accuracy and BCR to compare the results of our experiments.

Figure 5.5.2.2.1 shows a sample of text from the Qur'an where sentence terminals were automatically added using HMM model with 3 POS tags. The consensuses between the original Qur'an text and the automatically punctuated text are highlighted in green. Controversies cases are highlighted in red. Star (*) symbol indicates sentence terminal.

**Original punctuated Quran text**

حم * وَالْكِتَابِ الْمُبِينِ * إِنَّا أَنْزَلْنَاهُ فِي لَيْلَةٍ مُبَارَكَةٍ إِنَّا كُنَّا مُنْذِرِينَ * فِيهَا يُفْرَقُ كُلُّ أَمْرٍ حَكِيمٍ * أَمْرًا مِنْ عِنْدِنَا إِنَّا كُنَّا مُرْسِلِينَ * رَحْمَةً مِنْ رَبِّكَ إِنَّهُ هُوَ السَّمِيعُ الْعَلِيمُ * رَبِّ السَّمَاوَاتِ وَالْأَرْضِ وَمَا بَيْنَهُمَا إِنْ كُنْتُمْ مُوقِنِينَ * لَا إِلَهَ إِلَّا هُوَ يُحْيِي وَيُمِيتُ رَبُّكُمْ وَرَبُّ آبَائِكُمُ الْأَوَّلِينَ * بَلْ هُمْ فِي شَكٍّ يَلْعَبُونَ * فَارْتَقِبْ يَوْمَ تَأْتِي السَّمَاءُ بِدُخَانٍ مُبِينٍ يَغْشَى النَّاسَ هَذَا عَذَابٌ أَلِيمٌ * رَبَّنَا اكْشِفْ عَنَّا الْعَذَابَ إِنَّا مُؤْمِنُونَ * أَنَّى لَهُمُ الذِّكْرَى وَقَدْ جَاءَهُمْ رَسُولٌ مُبِينٌ * سورة الدخان (١-١٣)

**Automatic punctuation of the Quran text**

حم | وَالْكِتَابِ الْمُبِينِ * إِنَّا أَنْزَلْنَاهُ فِي لَيْلَةٍ مُبَارَكَةٍ إِنَّا كُنَّا مُنْذِرِينَ * فِيهَا يُفْرَقُ كُلُّ أَمْرٍ حَكِيمٍ | أَمْرًا مِنْ عِنْدِنَا إِنَّا كُنَّا مُرْسِلِينَ * رَحْمَةً مِنْ رَبِّكَ إِنَّهُ هُوَ السَّمِيعُ الْعَلِيمُ * رَبِّ السَّمَاوَاتِ وَالْأَرْضِ وَمَا بَيْنَهُمَا إِنْ كُنْتُمْ مُوقِنِينَ * لَا إِلَهَ إِلَّا هُوَ يُحْيِي وَيُمِيتُ رَبُّكُمْ وَرَبُّ آبَائِكُمُ الْأَوَّلِينَ * بَلْ هُمْ فِي شَكٍّ يَلْعَبُونَ * فَارْتَقِبْ يَوْمَ تَأْتِي السَّمَاءُ بِدُخَانٍ مُبِينٍ * يَغْشَى النَّاسَ هَذَا عَذَابٌ أَلِيمٌ * رَبَّنَا اكْشِفْ عَنَّا الْعَذَابَ إِنَّا مُؤْمِنُونَ | أَنَّى لَهُمُ الذِّكْرَى * وَقَدْ جَاءَهُمْ رَسُولٌ مُبِينٌ * سورة الدخان (١-١٣)

Figure 5.5.2.2.1: Automatic sentence terminal prediction for a Qur'an text using HMM model with the 3-POS tags.

Table 5.4.2.1.1: Sentence terminal prediction using HMM algorithm with 3-POS tag set.

| | | Gold | | Prediction | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Exp. #** | **N** | **Terminal** | **Non-Terminal** | **Terminal** | **Non-Terminal** | **TP** | **TN** | **FP** | **FN** | **Accuracy** | **Recall** | **Precision** | **F-score** | **Specificity** | **BCR** |
| Baseline | 8523 | 836 | 7687 | 0 | 8523 | 0 | 7687 | 0 | 836 | 0.902 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **1** | **8523** | **836** | **7687** | **458** | **8065** | **349** | **7578** | **109** | **487** | **0.930** | **0.417** | **0.762** | **0.539** | **0.986** | **0.702** |
| Baseline | 9378 | 836 | 8542 | 0 | 9378 | 0 | 8542 | 0 | 836 | 0.911 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **2** | **9378** | **836** | **8542** | **452** | **8926** | **330** | **8420** | **122** | **506** | **0.933** | **0.395** | **0.730** | **0.512** | **0.986** | **0.690** |
| Baseline | 9810 | 836 | 8974 | 0 | 9810 | 0 | 8974 | 0 | 836 | 0.915 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **3** | **9810** | **836** | **8974** | **564** | **9246** | **426** | **8836** | **138** | **410** | **0.944** | **0.510** | **0.755** | **0.609** | **0.985** | **0.747** |
| Baseline | 8517 | 836 | 7681 | 0 | 8517 | 0 | 7681 | 0 | 836 | 0.902 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **4** | **8517** | **836** | **7681** | **493** | **8024** | **384** | **7572** | **109** | **452** | **0.934** | **0.459** | **0.779** | **0.578** | **0.986** | **0.723** |
| Baseline | 8050 | 836 | 7214 | 0 | 8050 | 0 | 7214 | 0 | 836 | 0.896 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **5** | **8050** | **836** | **7214** | **337** | **7713** | **228** | **7105** | **109** | **608** | **0.911** | **0.273** | **0.677** | **0.389** | **0.985** | **0.629** |
| Baseline | 6737 | 836 | 5901 | 0 | 6737 | 0 | 5901 | 0 | 836 | 0.876 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **6** | **6737** | **836** | **5901** | **500** | **6237** | **411** | **5812** | **89** | **425** | **0.924** | **0.492** | **0.822** | **0.615** | **0.985** | **0.738** |
| Baseline | 7129 | 836 | 6293 | 0 | 7129 | 0 | 6293 | 0 | 836 | 0.883 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **7** | **7129** | **836** | **6293** | **490** | **6639** | **391** | **6194** | **99** | **445** | **0.924** | **0.468** | **0.798** | **0.590** | **0.984** | **0.726** |
| Baseline | 7163 | 836 | 6327 | 0 | 7163 | 0 | 6327 | 0 | 836 | 0.883 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **8** | **7163** | **836** | **6327** | **476** | **6687** | **372** | **6223** | **104** | **464** | **0.921** | **0.445** | **0.782** | **0.567** | **0.984** | **0.714** |
| Baseline | 6919 | 836 | 6083 | 0 | 6919 | 0 | 6083 | 0 | 836 | 0.879 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **9** | **6919** | **836** | **6083** | **375** | **6544** | **278** | **5986** | **97** | **558** | **0.905** | **0.333** | **0.741** | **0.459** | **0.984** | **0.658** |
| Baseline | 5204 | 842 | 4362 | 0 | 5204 | 0 | 4362 | 0 | 842 | 0.838 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **10** | **5204** | **842** | **4362** | **274** | **4930** | **179** | **4267** | **95** | **663** | **0.854** | **0.213** | **0.653** | **0.321** | **0.978** | **0.595** |
| | | | | | **Performance Metrics Average** | | | | | **0.918** | **0.401** | **0.750** | **0.518** | **0.984** | **0.692** |
| | | | | | **Accuracy Baseline Average** | | | | | **0.888** | **BCR baseline Average** | | | | **0.500** |

Table 5.5.2.2.1: Sentence terminal prediction using HMM algorithm with 10-POS tag set.

| | | Gold | | Prediction | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Exp. #** | **N** | **Terminal** | **Non-Terminal** | **Terminal** | **Non-Terminal** | **TP** | **TN** | **FP** | **FN** | **Accuracy** | **Recall** | **Precision** | **F-score** | **Specificity** | **BCR** |
| Baseline | 8523 | 836 | 7687 | 0 | 8523 | 0 | 7687 | 0 | 836 | 0.902 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **1** | **8523** | **836** | **7687** | **456** | **8067** | **350** | **7581** | **106** | **486** | **0.931** | **0.419** | **0.768** | **0.542** | **0.986** | **0.702** |
| Baseline | 9378 | 836 | 8542 | 0 | 9378 | 0 | 8542 | 0 | 836 | 0.911 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **2** | **9378** | **836** | **8542** | **437** | **8941** | **322** | **8427** | **115** | **514** | **0.933** | **0.385** | **0.737** | **0.506** | **0.987** | **0.686** |
| Baseline | 9810 | 836 | 8974 | 0 | 9810 | 0 | 8974 | 0 | 836 | 0.915 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **3** | **9810** | **836** | **8974** | **563** | **9247** | **426** | **8837** | **137** | **410** | **0.944** | **0.510** | **0.757** | **0.609** | **0.985** | **0.747** |
| Baseline | 8517 | 836 | 7681 | 0 | 8517 | 0 | 7681 | 0 | 836 | 0.902 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **4** | **8517** | **836** | **7681** | **481** | **8036** | **382** | **7582** | **99** | **454** | **0.935** | **0.457** | **0.794** | **0.580** | **0.987** | **0.722** |
| Baseline | 8050 | 836 | 7214 | 0 | 8050 | 0 | 7214 | 0 | 836 | 0.896 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **5** | **8050** | **836** | **7214** | **319** | **7731** | **219** | **7114** | **100** | **617** | **0.911** | **0.262** | **0.687** | **0.379** | **0.986** | **0.624** |
| Baseline | 6737 | 836 | 5901 | 0 | 6737 | 0 | 5901 | 0 | 836 | 0.876 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **6** | **6737** | **836** | **5901** | **486** | **6251** | **407** | **5822** | **79** | **429** | **0.925** | **0.487** | **0.837** | **0.616** | **0.987** | **0.737** |
| Baseline | 7129 | 836 | 6293 | 0 | 7129 | 0 | 6293 | 0 | 836 | 0.883 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **7** | **7129** | **836** | **6293** | **470** | **6659** | **385** | **6208** | **85** | **451** | **0.925** | **0.461** | **0.819** | **0.590** | **0.986** | **0.724** |
| Baseline | 7163 | 836 | 6327 | 0 | 7163 | 0 | 6327 | 0 | 836 | 0.883 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **8** | **7163** | **836** | **6327** | **465** | **6698** | **368** | **6230** | **97** | **468** | **0.921** | **0.440** | **0.791** | **0.566** | **0.985** | **0.712** |
| Baseline | 6919 | 836 | 6083 | 0 | 6919 | 0 | 6083 | 0 | 836 | 0.879 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **9** | **6919** | **836** | **6083** | **361** | **6558** | **271** | **5993** | **90** | **565** | **0.905** | **0.324** | **0.751** | **0.453** | **0.985** | **0.655** |
| Baseline | 5204 | 842 | 4362 | 0 | 5204 | 0 | 4362 | 0 | 842 | 0.838 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **10** | **5204** | **842** | **4362** | **247** | **4957** | **165** | **4280** | **82** | **677** | **0.854** | **0.196** | **0.668** | **0.303** | **0.981** | **0.589** |
| | | | | | **Performance Metrics Average** | | | | | **0.918** | **0.394** | **0.761** | **0.514** | **0.985** | **0.690** |
| | | | | | **Accuracy Baseline Average** | | | | | **0.888** | **BCR baseline Average** | | | | **0.500** |

#### 5.5.2.3 CRF Algorithm

This section presents the results of training and testing the CRF model for sentence terminal prediction using the BAQ Corpus. Two categories of POS tag sets (*i.e.* 3 POS and 10 POS) were selected as features for prediction sentence terminals.

##### 5.5.2.3.1 Results of CRF Sentence Terminal Prediction using a Three-POS Tag Set

This subsection presents the results of the conducted experiments for sentence terminal prediction by applying the CRF model. The CRF model was trained and tested using the BAQ Corpus with 3 POS tags were selected as a feature for prediction.

Table 5.5.2.3.1.1 shows the results of sentence terminal prediction using the CRF model for ten experiments. The CRF model uses the words and POS tags (*i.e.* 3 POS) as features for predicting sentence *terminal* (*i.e.* the word is located at the end of the sentence) or *non* (*i.e.* the word is not located at the end of the sentence) for the ten experiments. The average accuracy scored 91.4% with increment of 2.6% over the average accuracy baseline 88.8%. The average BCR scored 64.6% over the average BCR baseline 50.0% with increment of 16.6%.

On the other hand, the average recall scored 29.9%, the average precision scored 81.2%, and the f-scored scored 43.1%. Furthermore, the average specificity scored 99.2%. The dataset in our experiments is imbalanced. As we explained in section 5.2, these four metrics yield misleading when the data is imbalanced. Therefore, we will depend on the accuracy and BCR metrics for comparing the conducted experiments.

#### 5.5.2.3.2 Results of CRF Sentence Terminal Prediction using Ten-POS Tag Set

This subsection presents the results of the ten experiments for the sentence terminal prediction using CRF model.

The CRF model was trained and tested using the BAQ Corpus with ten POS tags were selected as a feature for prediction. Table 5.5.2.3.2.1 presents the results of the conducted experiments for sentence terminal prediction. The average accuracy scored 91.4% over the average accuracy baseline 88.8% with increment of 2.6%. On the other hand, the average BCR scored 65.0% with increment of 15.0% over the average BCR baseline 50.0%.

In return, the average recall rate scored 30.9%, the average precision scored 80.5%, the average f-score scored 44.0%, and the average specificity scored 99.2%.

Figure 5.5.2.3.1 shows a sample of text from the Qur'an where sentence terminals were automatically added using CRF model with 3 POS tags. The consensuses between the original Qur'an text and the automatically punctuated text are highlighted in green. Controversies cases are highlighted in red. The star (*) symbol indicates sentence terminal.

**Original punctuated Quran text**

حم * وَالْكِتَابِ الْمُبِينِ * إِنَّا أَنْزَلْنَاهُ فِي لَيْلَةٍ مُبَارَكَةٍ إِنَّا كُنَّا مُنْذِرِينَ * فِيهَا يُفْرَقُ كُلُّ أَمْرٍ حَكِيمٍ * أَمْرًا مِنْ عِنْدِنَا إِنَّا كُنَّا مُرْسِلِينَ * رَحْمَةً مِنْ رَبِّكَ إِنَّهُ هُوَ السَّمِيعُ الْعَلِيمُ * رَبِّ السَّمَاوَاتِ وَالْأَرْضِ وَمَا بَيْنَهُمَا إِنْ كُنْتُمْ مُوقِنِينَ * لَا إِلَهَ إِلَّا هُوَ يُحْيِي وَيُمِيتُ رَبُّكُمْ وَرَبُّ آبَائِكُمُ الْأَوَّلِينَ * بَلْ هُمْ فِي شَكٍّ يَلْعَبُونَ * فَارْتَقِبْ يَوْمَ تَأْتِي السَّمَاءُ بِدُخَانٍ مُبِينٍ يَغْشَى النَّاسَ هَذَا عَذَابٌ أَلِيمٌ * رَبَّنَا اكْشِفْ عَنَّا الْعَذَابَ إِنَّا مُؤْمِنُونَ * أَنَّى لَهُمُ الذِّكْرَى وَقَدْ جَاءَهُمْ رَسُولٌ مُبِينٌ * سورة الدخان (١-١٣)

**Automatic punctuation of the Quran text**

حم | وَالْكِتَابِ الْمُبِينِ * إِنَّا أَنْزَلْنَاهُ فِي لَيْلَةٍ مُبَارَكَةٍ إِنَّا كُنَّا مُنْذِرِينَ | فِيهَا يُفْرَقُ كُلُّ أَمْرٍ حَكِيمٍ | أَمْرًا مِنْ عِنْدِنَا إِنَّا كُنَّا مُرْسِلِينَ | رَحْمَةً مِنْ رَبِّكَ إِنَّهُ هُوَ السَّمِيعُ الْعَلِيمُ * رَبِّ السَّمَاوَاتِ وَالْأَرْضِ وَمَا بَيْنَهُمَا إِنْ كُنْتُمْ مُوقِنِينَ | لَا إِلَهَ إِلَّا هُوَ يُحْيِي وَيُمِيتُ رَبُّكُمْ وَرَبُّ آبَائِكُمُ الْأَوَّلِينَ * بَلْ هُمْ فِي شَكٍّ يَلْعَبُونَ * فَارْتَقِبْ يَوْمَ تَأْتِي السَّمَاءُ بِدُخَانٍ مُبِينٍ * يَغْشَى النَّاسَ هَذَا عَذَابٌ أَلِيمٌ * رَبَّنَا اكْشِفْ عَنَّا الْعَذَابَ إِنَّا مُؤْمِنُونَ | أَنَّى لَهُمُ الذِّكْرَى وَقَدْ جَاءَهُمْ رَسُولٌ مُبِينٌ * سورة الدخان (١-١٣)

Figure 5.5.2.3.1: Automatic sentence terminal prediction for a Qur'an text using CRF model with the 3-POS tags.

Table 5.5.2.3.1.1: Sentence terminal prediction using CRF algorithm with 3-POS tags.

| | | Gold | | Prediction | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Exp. #** | **N** | **Terminal** | **Non-Terminal** | **Terminal** | **Non-Terminal** | **TP** | **TN** | **FP** | **FN** | **Accuracy** | **Recall** | **Precision** | **F-score** | **Specificity** | **BCR** |
| Baseline | 8523 | 836 | 7687 | 0 | 8523 | 0 | 7687 | 0 | 836 | 0.902 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **1** | **8523** | **836** | **7687** | **350** | **8173** | **298** | **7635** | **52** | **538** | **0.931** | **0.356** | **0.851** | **0.503** | **0.993** | **0.675** |
| Baseline | 9378 | 836 | 8542 | 0 | 9378 | 0 | 8542 | 0 | 836 | 0.911 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **2** | **9378** | **836** | **8542** | **329** | **9049** | **256** | **8469** | **73** | **580** | **0.930** | **0.306** | **0.778** | **0.439** | **0.991** | **0.649** |
| Baseline | 9810 | 836 | 8974 | 0 | 9810 | 0 | 8974 | 0 | 836 | 0.915 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **3** | **9810** | **836** | **8974** | **419** | **9391** | **335** | **8890** | **84** | **501** | **0.940** | **0.401** | **0.800** | **0.534** | **0.991** | **0.696** |
| Baseline | 8517 | 836 | 7681 | 0 | 8517 | 0 | 7681 | 0 | 836 | 0.902 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **4** | **8517** | **836** | **7681** | **346** | **8171** | **276** | **7611** | **70** | **560** | **0.926** | **0.330** | **0.798** | **0.467** | **0.991** | **0.661** |
| Baseline | 8050 | 836 | 7214 | 0 | 8050 | 0 | 7214 | 0 | 836 | 0.896 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **5** | **8050** | **836** | **7214** | **173** | **7877** | **127** | **7168** | **46** | **709** | **0.906** | **0.152** | **0.734** | **0.252** | **0.994** | **0.573** |
| Baseline | 6737 | 836 | 5901 | 0 | 6737 | 0 | 5901 | 0 | 836 | 0.876 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **6** | **6737** | **836** | **5901** | **368** | **6369** | **329** | **5862** | **39** | **507** | **0.919** | **0.394** | **0.894** | **0.547** | **0.993** | **0.693** |
| Baseline | 7129 | 836 | 6293 | 0 | 7129 | 0 | 6293 | 0 | 836 | 0.883 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **7** | **7129** | **836** | **6293** | **343** | **6786** | **297** | **6247** | **46** | **539** | **0.918** | **0.355** | **0.866** | **0.504** | **0.993** | **0.674** |
| Baseline | 7163 | 836 | 6327 | 0 | 7163 | 0 | 6327 | 0 | 836 | 0.883 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **8** | **7163** | **836** | **6327** | **328** | **6835** | **284** | **6283** | **44** | **552** | **0.917** | **0.340** | **0.866** | **0.488** | **0.993** | **0.666** |
| Baseline | 6919 | 836 | 6083 | 0 | 6919 | 0 | 6083 | 0 | 836 | 0.879 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **9** | **6919** | **836** | **6083** | **258** | **6661** | **207** | **6032** | **51** | **629** | **0.902** | **0.248** | **0.802** | **0.378** | **0.992** | **0.620** |
| Baseline | 5204 | 842 | 4362 | 0 | 5204 | 0 | 4362 | 0 | 842 | 0.838 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **10** | **5204** | **842** | **4362** | **130** | **5074** | **95** | **4327** | **35** | **747** | **0.850** | **0.113** | **0.731** | **0.195** | **0.992** | **0.552** |
| | | | | | **Performance Metrics Average** | | | | | **0.914** | **0.299** | **0.812** | **0.431** | **0.992** | **0.646** |
| | | | | | **Accuracy Baseline Average** | | | | | **0.888** | **BCR Baseline Average** | | | | **0.500** |

Table 5.5.2.3.2.1: Sentence terminal prediction using CRF algorithm with 10-POS tags.

| | | Gold | | Prediction | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Exp. #** | **N** | **Terminal** | **Non-Terminal** | **Terminal** | **Non-Terminal** | **TP** | **TN** | **FP** | **FN** | **Accuracy** | **Recall** | **Precision** | **F-score** | **Specificity** | **BCR** |
| Baseline | 8523 | 836 | 7687 | 0 | 8523 | 0 | 7687 | 0 | 836 | 0.902 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **0** | **8523** | **836** | **7687** | **357** | **8166** | **303** | **7633** | **54** | **533** | **0.931** | **0.362** | **0.849** | **0.508** | **0.993** | **0.678** |
| Baseline | 9378 | 836 | 8542 | 0 | 9378 | 0 | 8542 | 0 | 836 | 0.911 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **1** | **9378** | **836** | **8542** | **362** | **9016** | **278** | **8458** | **84** | **558** | **0.932** | **0.333** | **0.768** | **0.464** | **0.990** | **0.661** |
| Baseline | 9810 | 836 | 8974 | 0 | 9810 | 0 | 8974 | 0 | 836 | 0.915 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **2** | **9810** | **836** | **8974** | **427** | **9383** | **339** | **8886** | **88** | **497** | **0.940** | **0.406** | **0.794** | **0.537** | **0.990** | **0.698** |
| Baseline | 8517 | 836 | 7681 | 0 | 8517 | 0 | 7681 | 0 | 836 | 0.902 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **3** | **8517** | **836** | **7681** | **352** | **8165** | **283** | **7612** | **69** | **553** | **0.927** | **0.339** | **0.804** | **0.476** | **0.991** | **0.665** |
| Baseline | 8050 | 836 | 7214 | 0 | 8050 | 0 | 7214 | 0 | 836 | 0.896 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **4** | **8050** | **836** | **7214** | **190** | **7860** | **140** | **7164** | **50** | **696** | **0.907** | **0.167** | **0.737** | **0.273** | **0.993** | **0.580** |
| Baseline | 6737 | 836 | 5901 | 0 | 6737 | 0 | 5901 | 0 | 836 | 0.876 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **5** | **6737** | **836** | **5901** | **386** | **6351** | **341** | **5856** | **45** | **495** | **0.920** | **0.408** | **0.883** | **0.558** | **0.992** | **0.700** |
| Baseline | 7129 | 836 | 6293 | 0 | 7129 | 0 | 6293 | 0 | 836 | 0.883 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **6** | **7129** | **836** | **6293** | **358** | **6771** | **305** | **6240** | **53** | **531** | **0.918** | **0.365** | **0.852** | **0.511** | **0.992** | **0.678** |
| Baseline | 7163 | 836 | 6327 | 0 | 7163 | 0 | 6327 | 0 | 836 | 0.883 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **7** | **7163** | **836** | **6327** | **340** | **6823** | **295** | **6282** | **45** | **541** | **0.918** | **0.353** | **0.868** | **0.502** | **0.993** | **0.673** |
| Baseline | 6919 | 836 | 6083 | 0 | 6919 | 0 | 6083 | 0 | 836 | 0.879 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **8** | **6919** | **836** | **6083** | **259** | **6660** | **204** | **6028** | **55** | **632** | **0.901** | **0.244** | **0.788** | **0.373** | **0.991** | **0.617** |
| Baseline | 5204 | 842 | 4362 | 0 | 5204 | 0 | 4362 | 0 | 842 | 0.838 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| **9** | **5204** | **842** | **4362** | **139** | **5065** | **99** | **4322** | **40** | **743** | **0.850** | **0.118** | **0.712** | **0.202** | **0.991** | **0.554** |
| | | | | | **Performance Metrics Average** | | | | | **0.914** | **0.309** | **0.805** | **0.440** | **0.992** | **0.650** |
| | | | | | **Accuracy Baseline Average** | | | | | **0.888** | **BCR Baseline Average** | | | | **0.500** |

### 5.5.3 Predicting Punctuation Marks in MSA Texts

This section applies the best ML algorithm based on the previous results to predict punctuation marks for MSA text. The selected algorithm uses the BAQ Corpus as training dataset. For the purpose of experiment, we selected MSA text from the book معالم في الطريق "*ma'ālim fī aṭ-ṭarīq*" (Qutb 1979) for Sayyid Qutb. Sayyid Qutb is also the author of في ظلال القرآن "*fī ẓilāl al-qur'ān*" (Qutb 1991) which we used to add punctuation marks to the BAQ Corpus. Our purpose of doing this experiment is to prove that we can transfer knowledge from the Quran to MSA.

The chosen text consists of 3859 words without counting punctuation marks; each word was manually tagged with coarse POS (*i.e.* 3 POS). The text was tokenized into rows such that each row consists of: word, POS and punctuation tag. The algorithm that has the best performance results for predicting punctuation marks in the BAQ Corpus was used to predict the punctuation marks in the MSA text. The punctuated text was compared with the original tagged text using different metrics. More detail about the design of experiment punctuation marks prediction for the MSA text is found in section 4.3.

The results showed that the CRF algorithm obtained the highest performance results using the word and 3-POS tags as features for prediction. The CRF algorithm was trained on the BAQ Corpus to predict punctuation marks for the MSA sample.

Table 5.5.3.1 presents the results of punctuation marks prediction for the MSA text using the CRF algorithm. The CRF algorithm uses the word and POS tags (*i.e.* 3 POS) to predict punctuation marks or *nopunc* after each word in the MSA text. The average accuracy scored 90.4% with increment of 5.3% over the average accuracy baseline of

85.1%. On the other hand, the average BCR scored 69.0% over the average BCR 50.0% with increment of 19.0%. Furthermore, the average recall scored 39.7%, the average precision 78.4%, the average f-score scored 52.7%, and the average specificity scored 98.3%.

Several conclusions are drawn from the experiment of punctuation marks prediction for MSA text: (i) Knowledge learned using ML algorithms can be transferred from the Qur'an text to the MSA text. (ii) The Qur'an and MSA texts share grammatical structures that are enough for training and testing ML algorithms. (iii) Using 3 POS tags (*i.e.* coarse annotation) as a feature indicates that this level of linguistic competence can be enough for the ML algorithms as well as for the language user to predict punctuation marks for any MSA text. Figure 5.5.3.1 presents a sample of the selected MSA text which was punctuated after training the CRF algorithm using BAQ Corpus with 3 POS tags as a feature.

Table 5.5.3.1: Results of MSA text punctuation marks prediction using CRF model.

| | | Gold | | Prediction | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Exp. #** | **N** | **Punctuation** | **Non-Punc** | **Punctuation** | **Non-Punc** | **TP** | **TN** | **FP** | **FN** | **Accuracy** | **Recall** | **Precision** | **F-score** | **Specificity** | **BCR** |
| Baseline | 3859 | 574 | 3285 | 0 | 3859 | 0 | 3285 | 0 | 574 | 0.851 | 0.000 | 0.000 | 0.000 | 1.000 | 0.500 |
| | **3859** | **574** | **3285** | **264** | **3595** | **207** | **3280** | **57** | **315** | **0.904** | **0.397** | **0.784** | **0.527** | **0.983** | **0.690** |

**Original MSA Text**

تقف البشرية اليوم على حافة الهاوية. لا بسبب التهديد بالفناء المعلق على رأسها. فهذا عَرَضٌ للمرض وليس هو المرض. ولكن بسبب إفلاسها في عالم القيم التي يمكن أن تنمو الحياة الإنسانية في ظلالها نمواً سليما وترتقي ترقياً صحيحاً. وهذا واضح كل الوضوح في العالم الغربي، الذي لم يعد لديه ما يعطيه للبشرية من القيم، بل الذي لم يعد لديه ما يُقنع ضميره بإستحقاقه للوجود،

**Punctuated MSA Text**

تقف البشرية اليوم على حافة الهاوية. لا بسبب التهديد بالفناء المعلق على رأسها. فهذا عَرَضٌ للمرض وليس هو المرض. ولكن بسبب إفلاسها في عالم القيم التي يمكن أن تنمو الحياة الإنسانية في ظلالها نمواً سليما وترتقي ترقياً صحيحاً. وهذا واضح كل الوضوح في العالم الغربي الذي لم يعد لديه ما يعطيه للبشرية من القيم بل الذي لم يعد لديه ما يُقنع ضميره بإستحقاقه للوجود

Figure 5.5.3.1 : Automatic punctuation mark prediction for MSA text using the CRF model with the 3-POS tags.

## 5.6 Discussion Of The Results

Below is a comparison between the Accuracy and BCR ratios of the three ML algorithms in the 3-POS and 10-POS sets of experiments across the two tasks of word and sentence terminal punctuation. Consider Tables 5.6.1 and 5.6.2 below:

Table 5.6.1: ML algorithms Compared: Punctuation marks prediction (9-class problem).

| ML alg. / Metrics | N-gram 3-POS | N-gram 10-POS | HMM 3-POS | HMM 10-POS | CRF 3-POS | CRF 10-POS |
|---|---|---|---|---|---|---|
| **Accuracy** | 0.839 | 0.833 | 0.841 | 0.841 | 0.934 | 0.922 |
| **BCR** | 0.647 | 0.650 | 0.639 | 0.637 | 0.864 | 0.858 |

Table 5.6.2: ML algorithms Compared: Sentence terminal prediction (2-class problem).

| ML alg. / Metrics | N-gram 3-POS | N-gram 10-POS | HMM 3-POS | HMM 10-POS | CRF 3-POS | CRF 10-POS |
|---|---|---|---|---|---|---|
| **Accuracy** | 0.918 | 0.918 | 0.918 | 0.918 | 0.914 | 0.914 |
| **BCR** | 0.700 | 0.696 | 0.692 | 0.690 | 0.646 | 0.650 |

To start with, there appears to be no significant difference in accuracy or BCR between an ML algorithm's handling of our two-class and nine-class problems. Notice in Tables 5.6.3 and 5.6.4 that the differences between two ML algorithm's accuracy and BCR rates vis-à-vis the 3-POS and their counterparts in the 10-POS sets of experiments ranges between 0 and 0.006. The differences between an algorithm's performances in the two sets of experiments in the 2-class and in the 9-class punctuation tasks were extremely insignificant. This implies that detailed annotation is of little importance as far as these ML algorithms are concerned.

CRF had a similar behavior and there was no significant difference between its BCR rates in the 3-POS and the 10-POS sets whether while punctuation mark prediction or sentence terminal prediction. Its accuracy rates for the two types of POS annotation in the two sets of experiments were in fact identical in sentence terminal prediction as well. The only discrepancy is the accuracy rate obtained when CRF punctuation mark prediction in the 3-POS and 10-POS annotation contexts; the difference between the two rates being 0.012.

Table 5.6.3: Differences between N-gram, HMM, and CRF's performances in punctuation marks prediction and sentence terminal punctuation in 3-POS vis-à-vis 10-POS experiments

| ML alg. / Metrics | N-gram 3-POS | N-gram 10-POS | HMM 3-POS | HMM 10-POS | CRF 3-POS | CRF 10-POS |
|---|---|---|---|---|---|---|
| **Accuracy** | 0.006 | --- | 0 | --- | 0.012 | --- |
| **BCR** | -0.003 | --- | 0.002 | --- | 0.006 | --- |

Table 5.6.4: Differences between N-gram, HMM, and CRF's performances in word punctuation and sentence terminal punctuation in 3-POS vis-à-vis 10-POS experiments

| ML alg. / Metrics | N-gram 3-POS | N-gram 10-POS | HMM 3-POS | HMM 10-POS | CRF 3-POS | CRF 10-POS |
|---|---|---|---|---|---|---|
| **Accuracy** | 0 | --- | 0 | --- | 0 | --- |
| **BCR** | 0.004 | --- | 0.002 | --- | -0.004 | --- |

One would have concluded that annotation details cause CRF's performance to decline had it not been the case that the BCR measurement was in contradiction and so were CRF's accuracy and BCR rates in sentence terminal prediction experiments. We are compelled to consider the 0.012 value an anomaly that goes against the trend

established by N-gram, HMM, and CRF itself. In other words, the two types of annotation (3-POS and 10-POS) yield similar results regardless of which ML algorithm is used for automatic punctuation. In other words, detailed POS annotation does not improve punctuation annotation. We conclude that users can eaisly punctuate a piece of Arabic text with modest knowledge of Arabic grammar. They only need to know the main POS of the word to accomplish the punctuation task.

Having established that there is no significant difference in the performance of the three ML algorithms vis-à-vis the 3-POS and 10-POS experimental conditions, we will cease to refer to them in the discussion below to make it clearer even though we will continue to report their results. Now, let's find out whether there are any differences in performance between the three ML models. Consider N-gram and HMM did not perform much differently from one another. With 0.002 and 0.008 differences in accuracy between them, one may venture to claim that they have similar performances since these differences are so insignificant. What may set them apart is the difference in BCR. With N-gram performing better than HMM in the 3-POS set of experiments by 0.008 and in the 10-POS set by 0.013, one may venture to claim that N-gram performed slightly better than HMM did. Most probably this difference is not significant either. It would probably be more accurate to claim that the two algorithms are of equal capability when it comes to annotating words with punctuation marks. CRF, on the other hand, outperformed these two in terms of accuracy by 0.093 and 0.081 in the 3-POS and 10-POS sets of experiments respectively and in terms of BCR by 0.225 and 0.221 respectively as well. Clearly then, CRF is the most efficient algorithm for automatic punctuation if it is desired to punctuate with commas, semicolons, hyphens, and sentence terminals. If the punctuation of concern is sentence-terminal only, however, N-gram and HMM are identical in capability with regard to accuracy and

almost identical in terms of BCR (the difference being 0.008 in the 3-POS set of experiments and 0.006 in the 10-POS set). This confirms our conclusion above that there is insignificant difference between these two algorithms. When they are compared with CRF, however, they are similar in terms of accuracy (the difference being a mere 0.004 in favor of the two algorithms) but they are superior to it in terms of BCR, with the difference being higher than 0.040.

Tables 5.6.5 and 5.6.6 below where the accuracy rate of HMM in the 3-POS condition was deducted from N-gram's corresponding accuracy rate in each table, its 3-POS BCR rate is deducted from its N-gram corresponding BCR rate, etc.

Table 5.6.5: N-gram vs. HMM vs. CRF in punctuation mark prediction.

| ML alg. / Metrics | N-gram 3-POS | N-gram 10-POS | HMM 3-POS | HMM 10-POS | CRF 3-POS | CRF 10-POS |
|---|---|---|---|---|---|---|
| **Accuracy** | -0.002 | -0.008 | --- | --- | -0.093 | -0.081 |
| **BCR** | 0.008 | 0.013 | --- | --- | -0.225 | -0.221 |

Table 5.6.6: N-gram vs. HMM vs. CRF in sentence terminal prediction.

| ML alg. / Metrics | N-gram 3-POS | N-gram 10-POS | HMM 3-POS | HMM 10-POS | CRF 3-POS | CRF 10-POS |
|---|---|---|---|---|---|---|
| **Accuracy** | 0 | 0 | --- | --- | 0.004 | 0.004 |
| **BCR** | 0.008 | 0.006 | --- | --- | 0.046 | 0.040 |

When the three ML algorithms' performances are compared in terms of punctuation mark prediction vs. sentence terminal punctuation, as in Table 5.6.7, N-gram's and HMM's accuracy and BCR soar by nearly 8% and 5% respectively. CRF's accuracy, however, declines by 2% and its BCR plunges by around 21%.

Table 5.6.7: N-gram vs. HMM vs. CRF vis-à-vis the punctuation marks prediction and sentence terminal prediction.

| ML algo. / Metrics | N-gram 3-POS | N-gram 10-POS | HMM 3-POS | HMM 10-POS | CRF 3-POS | CRF 10-POS |
|---|---|---|---|---|---|---|
| **Accuracy** | -0.079 | -0.085 | -0.077 | -0.077 | 0.020 | 0.008 |
| **BCR** | -0.053 | -0.046 | -0.053 | -0.053 | 0.218 | 0.208 |

CRF does better at word level punctuation because it takes conditional distribution into account and they are only focused on the POS labels; if this POS is followed by that POS, then use or do not use this punctuation mark. HMM, on the other hand, does better at sentence level for two reasons: (1) it is more focused on joint distributions of POS tags as they join to form sentences; (2) it takes account of the distribution over both the word observations and their POS labels, while CRF does not. In other words, HMM benefits from pattern sequences in two sets of data, the words and their POS tags. Table 5.6.8 shows a summary of results for all the experiments in the research. Figures 5.6.1 and 5.6.2 shows the charts of the three ML algorithms for both, punctuation mark prediction and sentence terminal prdiction with 3-POS and 10-POS tag sets.

Table 5.6.8: Summary of expierments results.

| Exp. # | Nine/ Two class problem | ML Algorithm | POS Categoriy | Train / BAQ | Test / BAQ Or MSA | Accuracy Baseline Average | Accuracy Score Average | BCR baseline Average | BCR Score Average |
|---|---|---|---|---|---|---|---|---|---|
| **1.** | Nine class | N-gram | 3-pos | 90% | 10% | 0.803 | 0.839 | 0.500 | 0.647 |
| **2.** | Nine class | N-gram | 10-POS | 90% | 10% | 0.803 | 0.833 | 0.500 | 0.650 |
| **3.** | Nine class | HMM | 3-POS | 90% | 10% | 0.803 | 0.841 | 0.500 | 0.639 |
| **4.** | Nine class | HMM | 10-POS | 90% | 10% | 0.803 | 0.841 | 0.500 | 0.637 |
| **5.** | Nine class | CRF | 3-POS | 90% | 10% | 0.803 | 0.934 | 0.500 | 0.864 |
| **6.** | Nine class | CRF | 10-POS | 90% | 10% | 0.803 | 0.922 | 0.500 | 0.858 |
| **7.** | Two class | N-gram | 3-POS | 90% | 10% | 0.888 | 0.918 | 0.500 | 0.700 |
| **8.** | Two class | N-gram | 10-POS | 90% | 10% | 0.888 | 0.918 | 0.500 | 0.696 |
| **9.** | Two class | HMM | 3-POS | 90% | 10% | 0.888 | 0.918 | 0.500 | 0.692 |
| **10.** | Two class | HMM | 10-POS | 90% | 10% | 0.888 | 0.918 | 0.500 | 0.690 |
| **11.** | Two class | CRF | 3-POS | 90% | 10% | 0.888 | 0.914 | 0.500 | 0.646 |
| **12.** | Two class | CRF | 10-POS | 90% | 10% | 0.888 | 0.914 | 0.500 | 0.650 |
| **13.** | Nine class | CRF | 3-POS | 100% BAQ | 100% MSA | 0.851 | 0.904 | 0.500 | 0.690 |

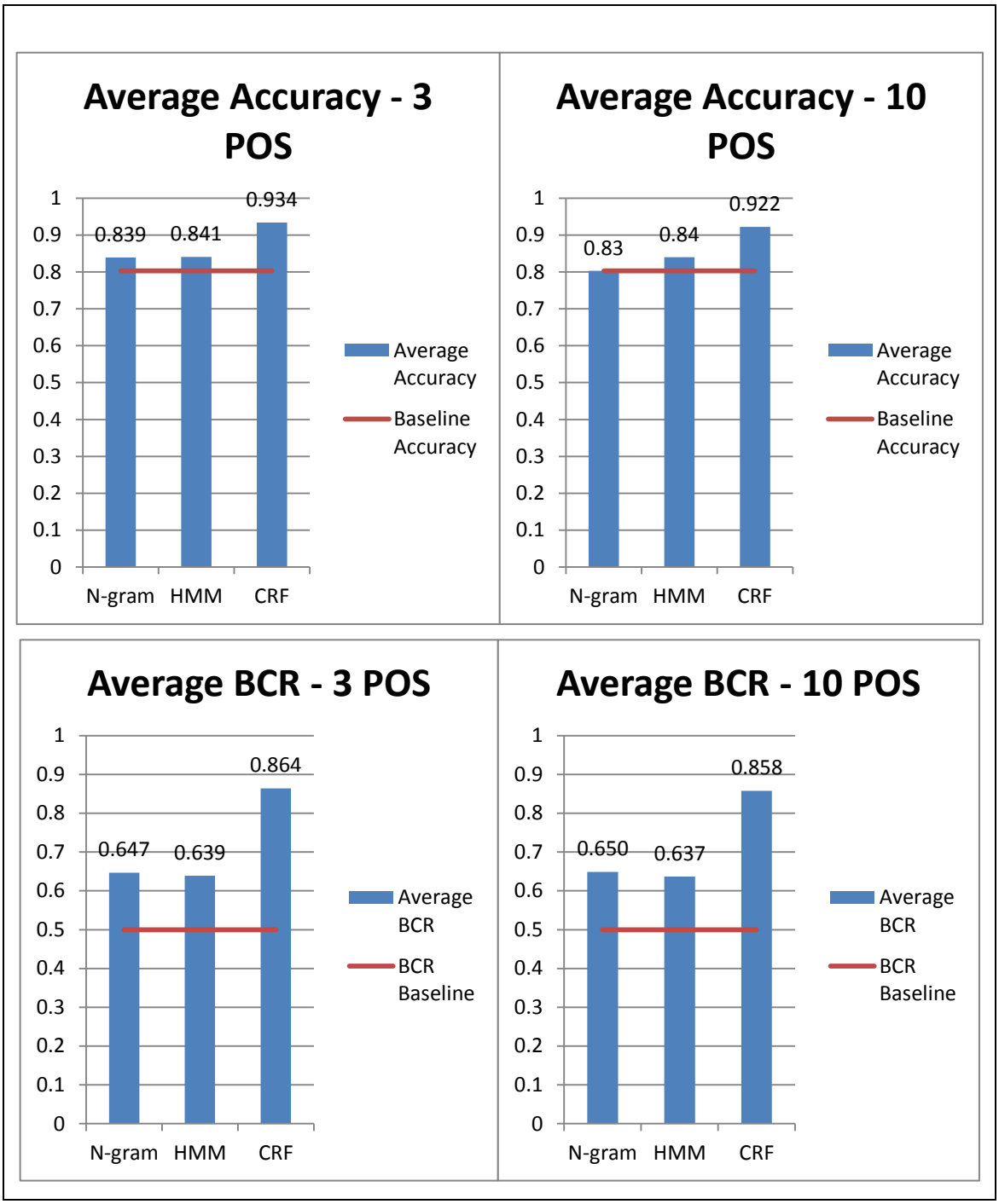


Figure 5.6.1: Charts for punctuation marks prediction using ML algorithms with 3 and 10 POS tags.

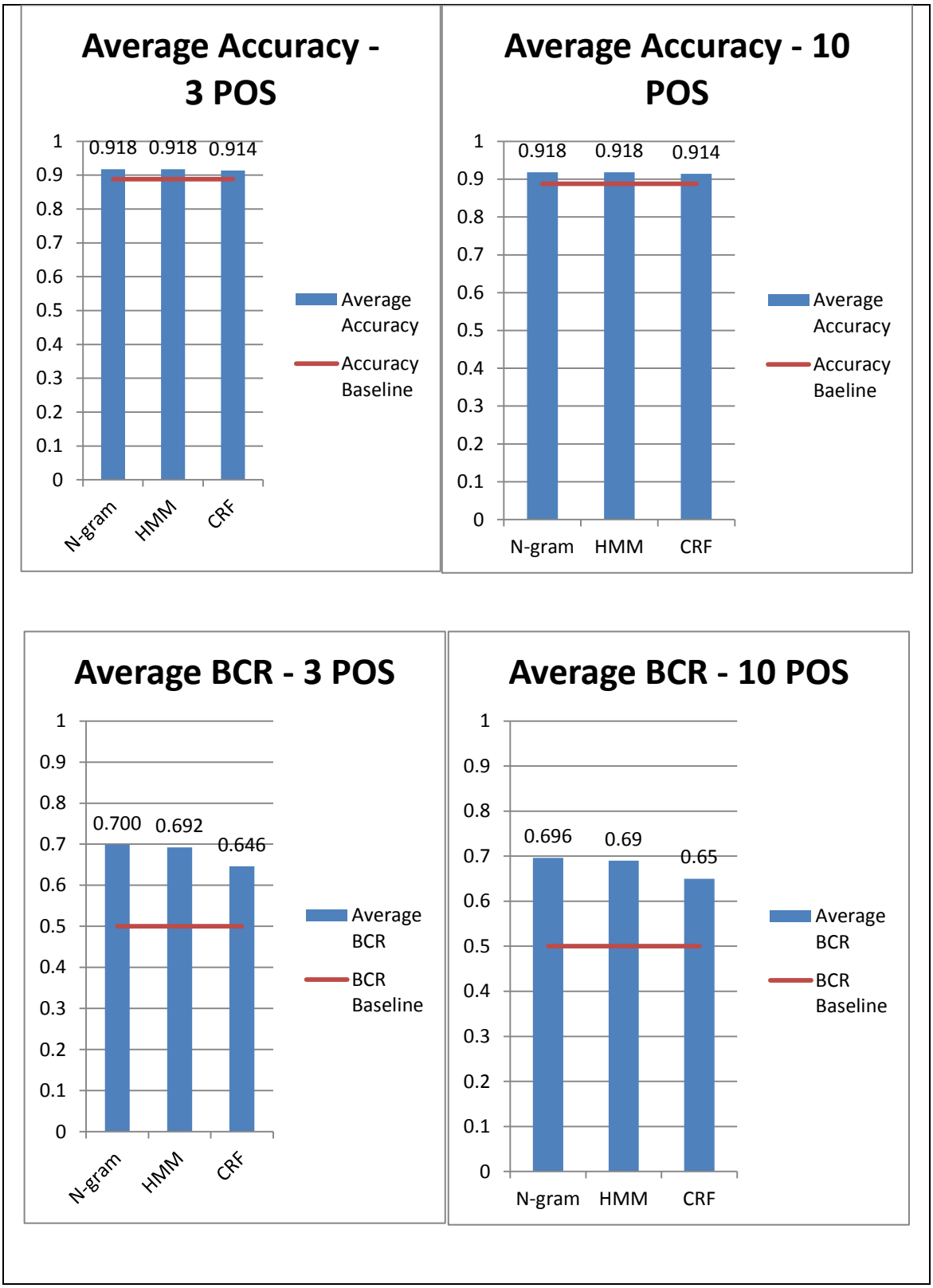


Figure 5.6.2: Charts for sentence terminal prediction using ML algorithms with 3 and 10 POS tags.

### 5.6.1 Results for Predicting Individual Punctuation Marks

This section investigates the results of predicting individual punctuation marks. We selected the punctuation marks (full stop, exclamation mark, comma, question mark, colon and simi-colon) to investigate the results of prediction for the 3 ML algorithms with 3-POS tags. These algorithms were trained and tested using the BAQ corpus.

The results showed that the CRF algorithm proved again its superiority for predicting all of the 6 punctuation marks. The highest results of F-score was for predicting full stop of 100% using CRF algorithm. The CRF scored F-score of 85% for predicting the colon. It scored F-score of 63% for comma. It also scored 37%, 20% and 10% for predicting exclamation mark, question mark and simi-colon respectively. Table 5.6.1.1 shows the results for predicting each punctuation mark using the 3 ML algorithms with 3 POS tags.

In conclusion, CRF algorithm can predict internal punctuation marks such as comma better than the other models. 63% of the commas in the test sample were correctly predicted using the CRF algorithm.

Table 5.6.1.1: F-score results for five punctuation marks from 3-POS punctuation marks prediction experiment.

| Punctuation mark | ML algorithm | F-score |
|---|---|---|
| ! | N-gram | 0 |
| | HMM | 0 |
| | CRF | 0.37 |
| , | N-gram | 0.16 |
| | HMM | 0.15 |
| | CRF | 0.63 |
| . | N-gram | 0.68 |
| | HMM | 0.57 |
| | CRF | 1.00 |
| : | N-gram | 0.76 |
| | HMM | 0.71 |
| | CRF | 0.85 |
| ; | N-gram | 0 |
| | HMM | 0 |
| | CRF | 0.10 |
| ? | N-gram | 0.07 |
| | HMM | 0 |
| | CRF | 0.20 |

# Chapter Six

# Conclusions and Recommendations

This chapter summarizes the main findings of this thesis. In addition, it describes our plan for future work in order to enhance our punctuation marks prediction system for Arabic text. Our proposal for the enhancement of the prediction system is (i) using other ML algorithms and (ii) enhancing the corpus with more features that will help ML algorithm to predict punctuation marks for Arabic text. The first section of this chapter presents the major conclusions of our research. The second section presents our proposal for future work.

## 6.1 Conclusion

This research discusses the problem of punctuation marks prediction for Arabic texts. We showed that the Arab readers have deficiencies in the usage of punctuation marks for Arabic text (Khafaji, 2001). In addition, our research highlighted the importance of punctuation marks for understanding Arabic text and Quran text. Also, we listed the Natural Language Processing Applications (NLP) were punctuation marks are useful. These applications are POS tagging, phrasing, information extraction etc. We also overviewed the contributions of other researches that implemented solutions for predicting punctuation marks for other foreign languages.

The Holy Quran is the central text of Islam and Arabic. In our research we considered the Quran as a gold standard for our experiments. Three Machine Learning (ML) algorithms (*i.e.* N-gram, HMM and CRF algorithms), were used to investigate the automatic prediction of punctuation marks and sentence terminals for Arabic text. To achieve our goal we used the Boundary Annotated Quran (BAQ) Corpus (Sawalha, et

al., 2012) for training and testing the 3 ML algorithms. We added new tier to the BAQ corpus containing punctuation marks and sentence terminals as used in Sayyid Qutb's Quran exegesis called في ظلال القرآن "*fī ẓilāl al-qur'ān*".

To test our hypothesis that ML algorithms can predict punctuation marks for Arabic text, we designed 13 different experiments. The first set of experiments, train and test the three ML algorithms on the BAQ corpus for predicting punctuation marks as a nine-class problem. The ML algorithms for this set of experiments use either 3 POS tags and the word or 10 POS tags and the word as features for prediction. The second set of experiments train and test the ML algorithm on BAQ corpus for predicting sentence terminals as two-class problem. The ML algorithm use either 3 POS tags and the word or 10 POS tags and the word as features for prediction. The last experiment, applies CRF algorithm to MSA text after been trained on the BAQ corpus. This experiment was designed to test our second hypothesis which is knowledge learnt from the Quran can be transferred to MSA text.

We have tried to examine which of the three machine learning algorithms is the best suited for automatically punctuating and sentence terminal predicting of the Modern Standard Arabic (MSA) texts. In addition, we have tried to answer the question of what type of linguistic annotation is required for the best automatic punctuation performance.

The results of experiments show that the CRF model has the best performance results for the punctuation marks prediction (*i.e.* nine-class problem) task for the two POS categories (*i.e.* 3-POS tags and 10-POS tags). The CRF model scored 93.4% and 86.4% for the average accuracy and average BCR respectively with the 3-POS tags, while it scored 92.2% and 85.8% for the average accuracy and average BCR respectively. The N-gram model was ranked second; it scored 83.9% and 64.7% for the average accuracy

and BCR average respectively with the 3-POS tags, while it scored 80.3% and 64.9% for the average accuracy and average BCR respectively with the 10-POS tags. The third model was the HMM model. It scored 84.1% for the average accuracy and 63.9% for the average BCR with the 3-POS tags. On the other hand, it scored 84.0% for the average accuracy and 63.7% for the average BCR with the 10-POS tags.

On the other hand, the results of experiments show the N-gram model has the best performance for the sentence terminal prediction (*i.e.* two-class problem) task for the both POS tags (*i.e.* 3-POS tags and 10-POS tags). The N-gram model scored 91.8% for the average accuracy; also it scored 70.0% for the average BCR with the 3-POS tags. In addition, the N-gram model scored 91.8% for the average accuracy and 69.6% for the average BCR with the 10-POS tags. The HMM model was ranked second. It scored 91.8% for the average accuracy and 69.2% for the average BCR with the 3-POS tags. On the other hand, the HMM model scored 91.8% for the average accuracy and 69.6% for the average BCR with the 10-POS tags. Finally, the CRF model was the last ranked model. The CRF model scored 91.4% for the average accuracy and 64.6% for the average BCR with the 3-POS tags. For the 10-POS tags, the CRF model scored 91.4% for the average accuracy and 65.05 for the average BCR.

The results of experiments showed the superiority of the CRF algorithm over other machine learning algorithms for automatic punctuation marks prediction. The superiority of the CRF algorithm is justified by its ability of investigating long range of dependencies between a sequence of observations (words and its POS tags) and their corresponding tags (punctuation). In addition, the CRF model proved its ability to investigate the internal punctuation marks with least number of errors compared with the other algorithms.

Furthermore, the experiments results proved that the machine learning algorithms performs slightly better when the annotation is coarse than when it is fine-grained. Apparently, because patterns can be more easily emerge with coarse than with fine-grained annotation. Therefore, we could say that the language users need not to possess a high level of linguistic knowledge to be able to punctuate a text with proper use of punctuation marks. This means, if the user only knows simple grammar rules or modest knowledge of Arabic grammar, then the user is able punctuate any Arabic texts.

## 6.2 Future Work

Further research may be required to improve accuracy of both punctuation marks prediction and sentence terminal prediction. This improvement could be achieved by adding more linguistic features to the used corpus. Many linguistic features can be added such as; grammatical annotation, parsed sentences or simple classification of words into content or function words.

Another way to improve the prediction of punctuation marks and sentence terminals is to use different Machine Learning algorithms such as the Dynamic CRF algorithm. Dynamic CRF algorithm used to investigate more than one feature at the same time such as; POS tags and punctuation marks.

# References

Alkhaldi, S, (2000), **Entrance to Fi Dhilal AlQur'an**, Dar Ammar puplishing: Amman.

Baum, L. E., (1972). **An equality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes**, Inequalities, Vol. 3, pages 1-8.

Beaglehole, V. J. and G. C. Yates, (2010), **The full stop effect: Using readability statistics with young writers**, Journal of Literacy and Technology, Vol. 11(4), pages 53-82.

Beeferman, D., A. Berger and J. Lafferty, (1998), **Cyberpunc: A lightweight punctuation annotation system for speech**, Acoustics, Speech and Signal Processing, 1998, Proceedings of the 1998 IEEE International Conference, Vol. 2, Pages 689-692.

Bird, S., E. Klein and E. Loper, (2009), **Natural language processing with Python**, 1st Edition, O'Reilly Media, Inc.: 1005 Gravenstein Highway North, Sebastopol, CA 95472.

Blum, A., (1997), **Empirical support for winnow and weighted-majority algorithms: Results on a calendar scheduling domain**, Kluwer Academic: Boston, Vol. 26, Issue 1, pages 5-23.

Blunsom, P., (2004), **Hidden markov models**, Lecture notes, Vol. 15, pages 18-19.

Brierley, C., M. Sawalha and E. Atwell, (2012), **Open-Source Boundary-Annotated Corpus for Arabic Speech and Language Processing**, Citeseer: LREC, pages 1011-1016.

Charoenpornsawat, P. and V. Sornlertlamvanich, (2004), **Automatic sentence break disambiguation for Thai**, National Electronics and Computer Technology Center.

David, J. M. and K. Balakrishnan, (2010), **Significance of Classification Techniques In Prediction Of Learning Disabilities**, Citeseer: International Journal of Artificial Intelligence & Applications (IJAIA), Vol. 1(4).

Gordon, M., (2014), **3 Disentangling stress and pitch-accent: a typology of prominence at different prosodic levels**, Cambridge university Press: Word Stress-Theoretical and Typological Issues, page 83.

Hillard, D., Z. Huang, H. Ji, R. Grishman, D. Hakkani-Tur, M. Harper, M. Ostendorf and W. Wang, (2006), **Impact of automatic comma prediction on POS/name tagging of speech**, Spoken Language Technology Workshop: IEEE, pages 58-61.

Jelinek, F., (1980), **Interpolated estimation of Markov source parameters from sparse data**, North-Holland: Pattern recognition in practice.

Jurafsky, D. and J. H. Martin, (2007), ***Speech and Language Processing: An introduction to natural language processing, computational linguistics, and speech recognition***, MIT Press.

Katz, S. M., (1987), **Estimation of probabilities from sparse data for the language model component of a speech recognizer,** IEEE: Acoustics, Speech and Signal Processing, Vol. 35(3), pages 400-401.

Khafaji, R., (2001), **Punctuation marks in original Arabic texts**, Harrassowitz: Zeitschrift für arabische Linguistik, number 40, pages 7-24.

Kudo, T., (2005), **CRF++: Yet another CRF toolkit**, Software available at http://crfpp. sourceforge. net.

Lafferty, J., A. McCallum and F. C. Pereira, (2001), **Conditional random fields: Probabilistic models for segmenting and labeling sequence data**, *Proceedings of the 18th International Conference on Machine Learning, pages 282-289.*

Longadge, R. and S. Dongre, (2013), **Class Imbalance Problem in Data Mining Review**, arXiv preprint arXiv:1305.1707.

Lu, W. and H. T. Ng, (2010), **Better punctuation prediction with dynamic conditional random fields**, Association for Computational Linguistics: Proceedings of the 2010 conference on empirical methods in natural language processing, pages 177-186.

Mohammed Shaker, Adnan Yousef, and Abed Alkareem Tori, **Stopping and starting science and its relationship to the interpretation of the Kor'an**,'Ulūm Islāmiyyah Journal, Vol. 8, pages 135-152.

Matusov, E., A. Mauser and H. Ney, (2006), **Automatic sentence segmentation and punctuation prediction for spoken language translation**, Citeseer: IWSLT, pages 158-165.

Nakache, D., E. Metais and J. F. Timsit, (2005), **Evaluation and NLP**, Springer: Database and Expert Systems Applications, pages 626-632.

Paul, M., M. Federico and S. Stüker, (2010), **Overview of the IWSLT 2010 evaluation campaign**, IWSLT, Vol. 10, pages 3-27.

Pham, Q. H., B. T. Nguyen and N. V. Cuong, (2014), **Punctuation Prediction for Vietnamese Texts Using Conditional Random Fields**, ACML Workshop: Machine Learning and Its Applications in Vietnam, pages 1-9.

Powers, D. M., (2011), **Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation**, Bioinfo Publications: Journal of Machine learning Technologies, Vol. 2(1), pages 37-63.

Qiang, Z., (2004), **Annotation scheme for chinese treebank**, Journal of Chinese information processing, Vol. 18(4), pages 1-8.

Qutb, S. (1979), **Mallem Fittareek**, 6th ed. Beirut & Cairo: Dar Al-Shurouq.

Qutb, S. (1991), **fī ẓilāl al-qurān**, Ed. 17. Beirut: Dar Al-Shurouq.

Sawalha, M., C. Brierley and E. Atwell, (2012), **Prosody Prediction for Arabic via the Open-Source Boundary-Annotated Qur'an Corpus**, Luso-Brazilian Association of Speech Sciences: Journal of Speech Sciences, Vol. 2(2), pages 175-191.

Stolcke, A., B. Chen, H. Franco, V. R. R. Gadde, M. Graciarena, M.-Y. Hwang, K. Kirchhoff, A. Mandal, N. Morgan and X. Lei, (2006), **Recent innovations in speech-to-text transcription at SRI-ICSI-UW**, IEEE: Audio, Speech, and Language Processing, Transactions, Vol. 14(5), pages 1729-1744.

Stuckless, R., (1994), **Developments in real-time speech-to-text communication for people with impaired hearing**, York Press Baltimore, MD: Communication access for people with hearing loss, pages 197-226.

Sutton, C. and A. McCallum, (2006), **An introduction to conditional random fields for relational learning**, MIT press: Introduction to statistical relational learning, pages 93-128.

Wang, S. and X. Yao, (2012), **Multiclass imbalance problems: Analysis and potential solutions**, IEEE: Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions, Vol. 42(4), pages 1119-1130.

Zaki, A., (1930), **Punctuation Marks for Ahmed Zaki**, Available from: http:// www. mediafire.com/?0mivmmozcwk.

Zhao, Y., C. Wang and G. Fu, (2012), **A CRF Sequence Labeling Approach to Chinese Punctuation Prediction**, 26th Pacific Asia Conference on Language, Information and Computation (PACLIC 26), page 508.

Zimmerman, M., D. Hakkani-Tür, J. Fung, N. Mirghafori, L. Gottlieb, E. Shriberg and Y. Liu, (2006), **The ICSI+ multilingual sentence segmentation system**, INTERSPEECH 2006 - ICSLP.

# التنبؤ بعلامات الترقيم للغة العربية الكلاسيكية والحديثة

**إعداد**
**علاء محمد الصلمان**

**المشرف**
**الدكتور مجدي شاكر صوالحه**

**المشرف المشارك**
**الأستاذ الدكتور حسين محمد ياغي**

**ملخص**

يجد الكتاب العرب صعوبة في عملية إستخدام علامات الترقيم الحديثة، لذلك إقترح علماء اللغة العربية بضرورة مراجعة قواعد إستخدام هذه العلامات. يعتقد البعض بأنه يجب التقيد عند إستخدام علامات الترقيم بقواعد نحوية، ولكن ما هو مستوى الكفاءة والمعرفة التي يجب أن يُلم بها مستخدم اللغة العربية كي يستطيع ترقيم نص عربي بعلامات ترقيم صحيحة؟ وهل تستطيع خوارزميات تعليم الآلة أن تتعامل مع مهمة الترقيم التلقائي للنصوص العربية؟ وهل تستطيع خوارزميات تعليم الآلة إنتاج نموذج يمكن من خلاله ترقيم أي من النصوص العربية؟ هذه الأسئلة يمكن إجابتها من خلال هذا البحث.

ثلاثة من خوارزميات تعليم الآلة تم إستخدامها في هذا البحث وهي: Conditional Random Felids (CRF) و N-gram و Hidden Markov Model (HMM). حيث أنه تم تدريبها وفحصها على الذخيرة القرآنية المعنونة بعلامات الوقف Boundary Annotated Qur'an (BAQ) من بعد أن قمنا بإدخال علامات الترقيم الحديثة لهذه الذخيرة. هذه الخوارزميات الثلاث تم فحصها على الذخيرة بإستخدام نوعين من علامات الخطاب الدقيقة والعامة وهي: عشرة من علامات الخطاب و ثلاثة من علامات الخطاب على الترتيب. ودلّت النتائج على أنه مع إستخدام علامات الخطاب العامة (ثلاثة من علامات الخطاب) فإنه سوف يكون هنالك تحسن بسيط مقارنة بإستخدام علامات الخطاب الدقيقة (عشرة من علامات الترقيم). وهذا يدلنا على أن مستخدم اللغة العربية لا يحتاج الى أن يكون ملماً كثيراً بالقواعد النحوية حتى يستطيع أن يقوم بترقيم أي نص عربي بعلامات الترقيم الحديثة. وعلاوة على ذلك، دلت النتائج بأنه حينما إستخدمنا خوارزمية التعليم الآلي (CRF) من أجل الترقيم الآلي لنص عربي حديث فإنه يعطينا نتائج جيده. علماً بأن تدريب الآلة على الترقيم الآلي لذخيرة قرآنية يفيد في ترقيم النصوص العربية الحديثة غير القرآنية.