

CLASSIFYING ARABIC TEXT BASED ON TEMPORAL PERIODS

BY

ZAINAB ALHATHLOUL

A Thesis Presented to the
DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

COMPUTER SCIENCE

DECEMBER 2021

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS
DHAHRAN 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

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

Thesis Committee

Dr. Irfan Ahmad (Adviser)

Dr. Sajjad Mahmood (Member)

Dr. Hamoud Aljamaan (Member)

Dr. Hamoud Aljamaan

Department Chairman

Dr. Suliman Al-Homidan

Dean of Graduate Studies

Date



©Zainab Alhathloul
2021

I dedicate this work to my beloved mother Fatemah and my dear father Hathloul; thank you for your support and care. I also dedicate this thesis to my daughter Amal, my husband Ahmad, and my lovely sisters. I appreciate their sacrifices, and after Allah, my thesis would not be completed without them.

ACKNOWLEDGMENTS

Praise to Allah, Lord of the Worlds, for His guidance and blessings, without which this thesis would not have been possible. My honest appreciation to my thesis advisor, Dr. Irfan Ahmad, for his advice and invaluable assistance during my thesis journey. Also, thanks to my thesis committee members, Dr. Sajjad Mahmood and Dr. Hamoud Aljamaan, for their helpful comments and discussions. Lastly, special thanks to my mother and my husband; without their help and support, I could not go through this thesis.

TABLE OF CONTENTS

ACKNOWLEDGMENTS	v
LIST OF TABLES	x
LIST OF FIGURES	xiv
ABSTRACT (ENGLISH)	xvii
ABSTRACT (ARABIC)	xix
CHAPTER 1 INTRODUCTION	1
1.1 Motivation	4
1.2 Background	7
1.2.1 NLP and ANLP	7
1.2.2 Text Classification	10
1.2.3 Division of Literature Arabic eras	11
1.2.4 Overview of Some Arabic Literature Eras	13
1.3 Problem Statement	14
1.4 Overview of The Thesis	15
CHAPTER 2 SURVEY	16
2.1 Introduction	16
2.2 Dataset	18
2.2.1 English Datasets	18
2.2.2 Datasets for Arabic and Other Languages	22

2.3	Features for Temporal Prediction	25
2.3.1	Stylistic Features	26
2.3.2	Lexical Features	26
2.3.3	Grammatical Features	27
2.3.4	Length Meta-Features	27
2.3.5	Other Features	27
2.4	The Temporal Period Predictions Task	28
2.5	Methods of Predicting Texts Based on Temporal Periods	30
2.5.1	Works Related to English Texts	30
2.5.2	Works Related to Arabic Text and Other Languages Texts . . .	36
2.6	Discussion	40
CHAPTER 3 DATASET		51
3.1	Data Collection	51
3.1.1	OpenITI Dataset For Temporal Text Classification	52
3.1.2	APCD Dataset For Temporal Poem Classification	59
3.2	Data Analysis	66
3.2.1	Unique Words	66
3.2.2	Common Words	72
CHAPTER 4 METHODOLOGY		76
4.1	Dataset Preparation	77
4.1.1	Data Pre-Processing	77
4.1.2	Tokenization	79
4.2	Temporal Text Classification Model Design	79
4.2.1	ANN Base System For The Temporal Classification	80
4.2.2	RNNs	90
4.2.3	Other Models	100
4.2.4	Models Tuning Process	104
4.3	Evaluation Metrics	108

CHAPTER 5	EXPERIMENTS AND RESULTS	111
5.1	Temporal Classification with Authors Separated	112
5.1.1	Temporal Classification Using ANN	113
5.1.2	Temporal Classification Using RNNs	121
5.1.3	Temporal Classification Using Other Models	127
5.2	Temporal Classification with Authors Merged	133
5.3	The Effect of Lemmatization	135
5.4	Summary of The Best Results	138
5.5	Result Analysis and Discussion	142
5.5.1	OpenITI dataset	143
5.5.2	APCD	146
5.6	Comparison to Other Works in The Literature	149
5.7	Summary	151
CHAPTER 6	CONCLUSIONS AND FUTURE WORKS	157
6.1	Contribution	159
6.2	Limitations	160
6.3	Future Work	161
APPENDIX A	THE RESULTS CONFUSION MATRICES	163
A.1	The confusion matrices for the OpenITI dataset results using ANN model	163
A.2	The confusion matrices for the APCD dataset results using ANN model	163
A.3	The confusion matrices for the OpenITI dataset results using RNN model	164
A.4	The confusion matrices for the APCD dataset results using RNN model	164
APPENDIX B	HUMAN EVALUATION	174
B.1	Human Evaluation Setup	174
B.2	Human Evaluation Results	176
B.2.1	Conclusions Based on Human Evaluation	177
REFERENCES		179

LIST OF TABLES

Table 1.1	The semantic changes of the word أدب <i>literature</i> in the Arabic language.	2
Table 1.2	The frequency for all the synonyms of fighting word by [1].	6
Table 1.3	Some of the challenges that face the researchers in the process of developing the ANLP system proposed by Darwish et al. [2].	9
Table 1.4	The periods of six literature eras by Ghoniem [3].	12
Table 2.1	The formula for the four features proposed by [4].	26
Table 2.2	Summary of the covered studies in terms of data source, the language of the data, and the total covered years.	41
Table 2.3	The label type and interval length or years range for all the covered studies	43
Table 2.4	Number of features used by each study from each feature type and the total number of features per study.	46
Table 2.5	Survey summary in terms of the best models and their results.	48
Table 3.1	The datasheet for the OpenITI dataset.	56
Table 3.2	The periods for the five literature eras setup in OpenITI dataset. . . .	58
Table 3.3	The three periods for the Arabic language development proposed by [5] in the OpenITI dataset.	59
Table 3.4	The number of samples for all the class setups in OpenITI dataset. . .	59
Table 3.5	The number of authors and samples per author for the binary eras and the five literature eras setups in the OpenITI dataset.	60
Table 3.6	The datasheet for the APCD dataset.	62
Table 3.7	The total number of verses and poets per era in the APCD corpus. . .	63

Table 3.8	The samples distribution of all the class setups in APCD dataset. . . .	64
Table 3.9	The number of authors and number samples per author, for the binary eras and the five literature eras setups in the APCD dataset.	65
Table 3.10	The number of the unique words and vocabulary size for each era in OpenITI and APCD datasets.	66
Table 3.11	Example of the unique words per era for OpenITI and APCD datasets.	68
Table 3.12	The Top three common words in OpenITI dataset.	73
Table 3.13	The Top three common words in APCD dataset	73
Table 4.1	Example of pre-processed sample using remove stop words and lemmatization.	79
Table 4.2	Two samples from the poem dataset to show the representation using word occurrence and TF-IDF.	87
Table 5.1	The initial results of the five literature eras setup from the OpenITI dataset using the ANN model in terms of tokenizer and optimizer. . . .	115
Table 5.2	The initial results for the five literature eras setup (one-verse per sample) from the APCD dataset using the ANN model in terms of tokenizer and optimizer.	115
Table 5.3	The initial results for the five literature eras setup from the OpenITI dataset using the ANN model in terms of number of layers and tokenizer.	115
Table 5.4	The initial results for the five literature eras setup (one-verse per sample) from the APCD dataset using the ANN model in terms of number of layers and tokenizer.	115
Table 5.5	The results for OpenITI dataset using ANN model.	118
Table 5.6	The results of several configurations for the five literature eras setup from the APCD dataset using the ANN model.	119
Table 5.7	The results for APCD dataset using ANN model.	120
Table 5.8	The initial results for the five literature eras setup form the OpenITI dataset using RNNs model.	122

Table 5.9	The initial results for the five literature eras setup (one-verse per sample) form the APCD dataset using RNNs model.	124
Table 5.10	The results for the RNNs model using different vocabulary sizes for the five literature eras setup from OpenITI dataset.	125
Table 5.11	The results for OpenITI dataset using word-level RNNs model.	125
Table 5.12	The results for the RNNs model using different vocabulary sizes for the five literature eras setup from APCD dataset.	128
Table 5.13	The results for APCD dataset using the word-level RNNs model.	128
Table 5.14	The results for OpenITI in the other models experiments	131
Table 5.15	The results for APCD in the other models experiments.	133
Table 5.16	The results for all the models using OpenITI dataset with merging the authors.	136
Table 5.17	The results for all the models using APCD dataset without separating the authors	137
Table 5.18	The results for all the models using five literature eras setup in OpenITI and APCD datasets after applying lemmatization.	139
Table 5.19	Part of two samples from the OpenITI dataset before and after applying lemmatization.	140
Table 5.20	The best achieved results in OpenITI dataset with separating the authors using the ANN model.	141
Table 5.21	The best results for the APCD dataset with separating the authors using the ANN model.	141
Table 5.22	The best achieved results in OpenITI dataset with merging the authors.	142
Table 5.23	The best results for APCD with merging the authors using ANN model.	142
Table 5.24	Two samples from the OpenITI dataset for different eras being wrongly predicted as Islamic era.	145
Table 5.25	The actual era, predicted era, and the words frequency per era for some samples from OpenITI datasets.	147
Table 5.26	The actual era, predicted era, and the words frequency per era for some samples from APCD dataset	149

Table 5.27	Other works from the literature similar to our work.	150
Table 5.28	Comparison between our best model results and the CNN model proposed by Orabi et al. [6].	151
Table 5.29	The baseline results and the relative improvement to the baseline accuracy for each class setup in all the datasets.	154
Table B.1	The human experiment results for five literature eras setup in OpenITI and APCD datasets.	176
Table B.2	Participant-1 feedback for two predicted samples for the OpenITI and APCD datasets.	178

LIST OF FIGURES

Figure 1.1	Examples of Abbasid poems that concern about gray hair, the word in red represent the gray hair in Arabic [7]	5
Figure 3.1	OpenITI corpus structure	53
Figure 3.2	OpenITI corpus distribution in term of authors and books	54
Figure 3.3	Frequency of the top 10 common words in OpenITI dataset	74
Figure 3.4	Frequency of the top 10 common words in APCD dataset	75
Figure 4.1	The elements of a neural networks architecture	83
Figure 4.2	The structure of the neural networks neuron	84
Figure 4.3	Overview of the relation between the network, layers, loss function, and optimizer (adapted from [8])	84
Figure 4.4	The words occurrence for the two samples in Table 4.2	87
Figure 4.5	The TF-IDF for the the two samples in Table 4.2	89
Figure 4.6	The ANN methodology structure.	91
Figure 4.7	ANN structure before and after using the dropout [9]	91
Figure 4.8	The elements of RNNs architecture	92
Figure 4.9	Structure of the LSTM cell (adapted from [10])	95
Figure 4.10	Structure of the GRU cell (adapted from [10])	95
Figure 4.11	Simple RNN without output	96
Figure 4.12	The RNNs methodology structure.	97
Figure 4.13	Example of converting sample into a character representation for the RNNs model	99
Figure 4.14	The general structure of The hybrid model	105

Figure 4.15	The architecture of the CNN model applied by Orabi et al. [6]	106
Figure 4.16	Two examples of confusion matrix for binary and multi-class classification	108
Figure 5.1	The best architecture of the ANN model for OpenITI and APCD datasets	117
Figure 5.2	The confusion matrix for five literature eras (a) and 300-years custom period (b) setups for the OpenITI dataset using the ANN model.	117
Figure 5.3	Confusion matrix for binary literature eras (a) and five literature eras (b) setups for the APCD dataset using the ANN model.	120
Figure 5.4	The best architecture of the RNNs model for all the datasets	123
Figure 5.5	Confusion matrix for five literature eras (a) and 300-years custom period (b) setups for the OpenITI dataset using the RNNs model.	126
Figure 5.6	Confusion matrix for binary literature eras (a) and five literature eras (b) setups for the APCD dataset using the RNNs model.	129
Figure 5.7	Confusion matrix for five literature eras setup using logistic regression (a) and CNN (b) models for the OpenITI dataset.	131
Figure 5.8	Confusion matrix for five literature eras setup using logistic regression (a) and CNN (b) models for the APCD dataset.	133
Figure 5.9	Confusion matrix for five literature eras (a) and 300-years custom period (b) setups from the OpenITI dataset using the ANN model with merging the authors.	135
Figure 5.10	Confusion matrix for five literature eras when we separated the authors (a) and when we merge them (b) setups for the APCD dataset using the ANN model.	140
Figure 5.11	The confusion matrix for the five literature eras class setup for the OpenITI dataset using the ANN model	144
Figure 5.12	The confusion matrix for the five literature eras class setup in the APCD dataset using ANN model	148

Figure A.1	The confusion matrix for the binary literature eras setup in the Open- ITI dataset using ANN model	164
Figure A.2	The confusion matrix for the 3-eras setup proposed by Belinkov et al. [5] in OpenITI dataset using ANN model	164
Figure A.3	The confusion matrix for the 200-years custom period setup in the OpenITI dataset using ANN model	165
Figure A.4	The confusion matrix for the 100-years custom period setup in the OpenITI dataset using ANN model	166
Figure A.5	The confusion matrix for the binary literature eras setup in APCD dataset using ANN model	167
Figure A.6	The confusion matrix for the 11-eras setup in the APCD dataset using ANN model	168
Figure A.7	The confusion matrix for the 12-eras setup in the APCD dataset using ANN model	169
Figure A.8	The confusion matrix for the binary literature eras setup in the Open- ITI dataset using RNN model	169
Figure A.9	The confusion matrix for the 3 class setup proposed by Belinkov et al. [5] in OpenITI dataset using RNN model	169
Figure A.10	The confusion matrix for the 200-years periods setup in the OpenITI dataset using RNN model	170
Figure A.11	The confusion matrix for the 100-years periods setup in the OpenITI dataset using RNN model	171
Figure A.12	The confusion matrix for the 11-eras setup in the APCD dataset using RNN model	172
Figure A.13	The confusion matrix for the 12-eras setup in the APCD dataset using RNN model	173
Figure B.1	Example from the human evaluation test, the sample form the Open- ITI dataset.	175
Figure B.2	Confusion matrix for the human experiment results in the five litera- ture eras setup for OpenITI (a) and APCD (b) datasets.	177

THESIS ABSTRACT

NAME: Zainab Alhathloul
TITLE OF STUDY: Classifying Arabic Text Based on Temporal Periods
MAJOR FIELD: Computer Science
DATE OF DEGREE: December 2021

The Arabic Language has developed over time, new words are being used, old words are stopped from being used, and the usage of words changed. A common example of that development is classical and modern Arabic. Moreover, historians and linguists divide Arabic literature into eras. Few studies were conducted in relation to classifying Arabic text into different time periods, and most of the works are related to poems only. However, to the best of our knowledge, the temporal period classification of Arabic literature has not been reported in the literature. In this thesis, we present classifying Arabic literature into time periods using neural networks, including deep learning approaches. Fully connected ANN and RNNs were explored, including two tokenization techniques. The presented models were evaluated on two datasets extracted from two publicly available corpora OpenITI and Arabic poem comprehensive dataset (APCD). Both corpora covered the time from the Pre-Islamic period until

the modern period. We tried different class setups, including predefined time periods (eras) and custom time periods. Additionally, we utilized traditional machine learning classifiers for some setups, so the results are compared to the neural network results. The best results for the five-class problem in terms of F1-measure are 0.70 score for the poem dataset, 0.66 score for the OpenITI dataset. We also investigated the influence of merging and separating the authors; further, we investigated other influences like lemmatization and removing the stop words. Based on the results of some experiments, we found that dividing the literature into predefined eras is more natural than arbitrary dividing into custom time periods. Lastly, we investigated different predefined eras and custom time periods, reported results, and presented interesting insights.

ملخص الرسالة

الاسم: زينب الهذلول

عنوان الدراسة: تصنيف النص العربي الى فترات زمنية

التخصص: علوم الحاسب

تاريخ الدرجة العلمية: ديسمبر 2021

تطورت اللغة العربية بمرور الوقت ، ويتم استخدام كلمات جديدة ، والتوقف عن استخدام الكلمات القديمة ، وتغير استخدام الكلمات. ومن الأمثلة الشائعة على هذا التطور اللغة العربية الفصحى والحديثة. علاوة على ذلك ، يقسم المؤرخون واللغويون الأدب العربي إلى عصور. أجريت دراسات قليلة على تصنيف النص العربي إلى فترات زمنية مختلفة ، وتتعلق معظم الأعمال بالقصائد فقط. ومع ذلك ، على حد علمنا ، لم يتم ذكر تصنيف الفترة الزمنية للأدب العربي في البحوث السابقة. نقدم في هذه الرسالة تصنيف الأدب العربي إلى فترات زمنية باستخدام الشبكات العصبية ، بما في ذلك مناهج التعلم العميق. تم استكشاف شبكات ANN و RNNs ، بما في ذلك طريقتان من تقنيات الترميز. تم تقييم النماذج المقدمة على مجموعتي بيانات مستخرجة من مجموعتين متاحيتين للجمهور من OpenITI ومجموعة بيانات شاملة للقصائد العربية (APCD). غطت كلا المجموعتين الوقت من فترة ما قبل الإسلام حتى العصر الحديث. لقد جربنا إعدادات الفصل المختلفة ، بما في ذلك الفترات الزمنية المحددة مسبقًا (العصور) والفترات الزمنية المخصصة. بالإضافة إلى ذلك ، استخدمنا مصنفات التعلم الآلي التقليدية لبعض الإعدادات ، بحيث تتم مقارنة النتائج بنتائج الشبكة العصبية. أفضل النتائج لمشكلة الفئات الخمس من حيث مقياس F1 هي 70.0 درجة لمجموعة

بيانات القصيدة ، و 66.0 درجة لمجموعة بيانات OpenITI. لقد بحثنا أيضًا في تأثير دمج المؤلفين وفصلهم ؛ علاوة على ذلك ، قمنا بالتحقيق في التأثيرات الأخرى مثل التجريد وإزالة كلمات التوقف. بناءً على بعض نتائج التجارب ، وجدنا أن تقسيم الأدبيات إلى عصور محددة مسبقًا هو أمر طبيعي أكثر من التقسيم الافتراضي إلى فترات زمنية مخصصة. أخيرًا ، قمنا بالتحقيق في عصور مختلفة محددة مسبقًا وفترات زمنية مخصصة ، وأبلغنا عن النتائج ، وقدمنا رؤى مثيرة للاهتمام.

CHAPTER 1

INTRODUCTION

Languages, in general, developed over time and were affected by diverse types of changes like semantic and lexical change [11]. For example, new words are being used, and old words are stopped being used. Thus, languages evolved and are affected by many factors. Arabic language is not an exception; it developed and changed over time. A common example of that development are classical, and the modern Arabic [5]. Arabic literature, which dates back to about 150 years ago and more is term the classical Arabic [12].

The Arabic language is used as the official language in many countries. Standard Arabic or Modern Standard Arabic is used in written materials nowadays. Moreover, the Arabic language is considered the language of Islam because the Quran and Hadith were written in it. However, the Arabic language was used before Islam, and there exist literature from that time.

Alrabiah et al. [12] showed how semantic of words changed. The term عادي *Aady* in Arabic was used to mean *old* or *aggressor*, in the modern Arabic it means normal. Moreover, the frequency of using the word *Aady* is more in modern Arabic

Table 1.1: The semantic changes of the word أدب *literature* in the Arabic language.

Period	Meaning
Pre-Islamic	Invitation to food
Islamic	The moral discipline
Umayyad	The moral discipline and they add educational meaning For example: teachers called preceptor.
Abbasid	Combined between the Islamic and Umayyad meaning
3th century AH	Arab poems and their news
4th century AH	All the knowledge including religious and non-religious knowledge
13th century AH	Poems snippets and news anecdotes
from the mid of 13th century AH until now	Indicates two meaning: 1) Any written scripts (they followed the French language description of the literature word) 2) Pure literature that expresses a mean like poems and prose.

than classical Arabic [12].

The previous example showed the semantic change for two time periods only. The semantic change through multiple time periods is shown in the following example. The word أدب (literature) in Arabic indicates various meanings that are used differently through time [13]. Table 1.1 presents the changes of using that word in the Arabic language. The author showed that each period used the word for diverse purposes. For example, the distinctive meaning is used in the Pre-Islamic period. Moreover, Al-Fakhoury defined the Arabic literature الأدب as "the group of written monuments in which the human mind is manifested" [14].

The above examples presented the change in the Arabic language. The origination of the Arabic literature and the influences on it are possible factors of that change. Part of human history has no writing and books, but it was probably not without literature [14]. For a long time, literature was narrated orally among people without

writing it down. Linguists in the area mentioned that poetry came before the artistic prose. This means that poetry is one of the oldest literary works.

Furthermore, Al-Fakhoury [14] introduced four general influences on the Arabic literature origination. These influences are the natural environment, the social environment, the mixing of nations, and religions. The natural environment is about the place details like the weather, whether it is hot or cold. The social environment included the political, cultural, and living environment, which the situation of nations living in as the wealth and poverty.

Automatically labeling text into a certain period of time is essential to categorize the literature data as a natural language processing (NLP) task. It can help researchers to identify where approximately the evolution of the language occurs. In addition, detecting the development of the language helps in many fields as linguistic and history fields [15]. They can study the change of the literature according to the periods that show significant differences. Moreover, the periods with no apparent differences in the development of the language could interest the researcher. Last but not least, the research findings can also aid in authenticating literature attributed to a specific era in history [16].

In addition, some NLP tasks are time-sensitive, like question answering and information retrieval. For example, some questions are contextualized based on a specific era, so knowing that era will help the question answering system provide the proper answer. Classifying the time of given text is beneficial for this kind of application. Thus, several studies employ classifying temporal text issue as sub-task not the main task, e.g [17].

Text classification is one of the NLP tasks that acquired an interest in many different fields. Numerous studies over the last decade have been done in Arabic natural language processing (ANLP). Different ANLP applications exist such as data mining, discretization, text classification, and more [18], [19], [20]. Arabic language complexity is one of the major challenges facing researchers in ANLP. However, with the advent of machine learning, many problems can be solved more effectively.

Furthermore, temporal period classification is an example of a text classification task. In general, text classification has been introduced for many problems such as sentiment analysis and text categorization [21], [18]. However, temporal periods classification has not been research as widely as other text classification especially for Arabic, we only found two studies in literature that classify poems into eras. Accordingly our research focus on temporal periods classification for Arabic literature which includes poems and non-poems texts.

To the best of our knowledge, no reported study in the literature attempted to classify Arabic literature texts into time periods which motivated us to investigate the problem of classifying the Arabic text based on temporal periods using different machine learning approaches. In this research, we will present our work for automatically classifying Arabic text based on various temporal periods.

1.1 Motivation

Numerous Arabic eras have existed throughout the history that the Arabic literature developed through it. The eras from a historical perspective are debatable and have

الليلُ شَيْبٌ والنَّهارُ كِلَاهُمَا . . . رَأْسِي بِكَثْرَةِ مَا تَدُورُ رَحَاهُمَا.
وما الشَّيْبُ إِلَّا تَوَامُ الموتِ للفتى . . . وعيشُ امرئٍ بعدَ المشيبِ جهادٌ
كل داءٍ يرجي الدواء له . . . إلا الفظيعين: مَوْتُهُ و مَشْيِبُهُ
قد رابني هَرَبُ الشَّبَابِ وَراعني . . . شَيْبٌ يَدِبُ بَيَاضُهُ في مَفَرِّقِي

Figure 1.1: Examples of Abbasid poems that concern about gray hair, the word in red represent the gray hair in Arabic [7]

some overlapping periods. However, some eras are fixed in most researches, like the Pre-Islamic era and Islamic era. We will discuss in detail the divisions of Arabic literature eras in Section 1.2.

Furthermore, the linguistic studies present some characteristics and features of the Arabic language for a specific era. AlSamurai et al. [7] propose a study of how the words in Abbasid poetry related to a concern or subject in that era [7]. *Gray hair* شيب is an example, it was a severe concern for poets in the Abbasid era. Therefore, they involve it considerably in their poems and treat it as an enemy who attacks them. Figure 1.1 presents some examples of using gray hair in poems, each row from a different poem. The examples show that the poets used the word gray hair in different forms like nouns or adjectives.

Nazem [1] focuses on the fighting words mentioned in the pre-Islamic poetry [1]. We show below some interesting statistics related to words in pre-Islamic poetry. The author showed the fighting word as verbs and nouns in different tenses present, past, and future. The author lists the words with the poet's name, who mentions them, and the appearance frequency. Moreover, he counted the frequency for all the synonyms of fighting words. Some objectives of this research are study the semantic relations

Table 1.2: The frequency for all the synonyms of fighting word by [1].

Rank	Word In Arabic	Word In English	Frequency
1	القتال	the fighting	131
2	الحرب	the war	86
3	الغارة	the raid	36
4	الوغي	the rogue	29
5	النزال	the fight	21
6	الهيحاء	(means the war)	18
7	الكريهة	the unpleasant	15
8	المعركة	the battle	14
9	الغزوة	(the battle which prophet Muhammad peace upon him a partner on it)	12
10	الوقية	the feud	10
11	الرحى	the mill	7
12	المعمعة	the fray	5
Total:			384

between the words, and provide a linguistic dictionary for all the mentioned words.

Furthermore, Table 1.2 presents the synonyms to the fighting word along with its frequency. We show a description between brackets for the words that do not have a direct translation in English. Fighting is the most said word by the poets, with 33% of the total synonyms words of fighting. At the same time, the war word comes in the second rank of the most mentioned words.

The above studies show that Arabic literature has several features in a certain time period from a linguistics perspective. This evidence motivates us to investigate the ability of machine learning techniques to learn such kinds of features from Arabic texts. Therefore, we propose our work of classifying Arabic text into time periods.

1.2 Background

In this section, we will present NLP and ANLP and text classification. Moreover, we will show the different divisions of Arabic literature as eras and an overview of some Arabic literature eras.

1.2.1 NLP and ANLP

NLP is a sub-field of artificial intelligence. It is a set of techniques employed to make the computer understand the natural language. NLP is a wide domain with diverse branches. Moreover, NLP is an interdisciplinary area that is related to computer science, linguistics, mathematics, psychology, and others[2]. The main goal of NLP tasks is to analyze and process the language as humans do. Applications of NLP covered the spoken, and the written language [22].

Phonology, morphology, semantics, and pragmatics are the four language levels presented by Reshamwala et al. [23] which humans need to understand language. NLP benefits from these levels to improve its techniques. Part of speech (POS) is an example of morphology analysis that researchers applied to improve their systems as in [24], [25].

Numerous application for NLP exists in various fields, like sentiment analysis, text categorization, and dialect identification, among others [21], [26], [20]. Moreover, NLP applications are available in different languages other than English such as Dutch, Arabic, and French [27], [28], [29].

On the other hand, the ANLP acquired an interest from researchers in recent years.

ANLP Applications are suffering from challenges related to the nature of the Arabic language and its complexity. Darwish et al. present the main challenges in Arabic that could face the researchers in developing the ANLP system. Five challenges are presented morphological richness, orthographic ambiguity, dialectal variations, orthographic noise, and resource poverty. Table 1.3 present the first three challenges with an example for each one. The orthographic noise challenge is about the spelling inconsistency in Arabic dialects. The authors introduce some solutions for the previous challenges. Several morphological analyzers have been mentioned, such as standard Arabic morphological analyzer (SAMA) from linguistic data consortium (LDC) ¹. However, most of the morphological analyzers are for modern standard Arabic. Syntactic parsing aid to clarify the ambiguity of orthographic. Farasa is one of the most common syntactic parser for Arabic ².

Neural networks and deep learning has been utilized recently to make robust NLP systems. Neural networks relies on a group of algorithms that learn multilevel representations to find the high-level abstraction of the data. Likewise, Neural networks including deep learning has been applied successfully in various ANLP tasks.

Table 1.3: Some of the challenges that face the researchers in the process of developing the ANLP system proposed by Darwish et al. [2].

Challenge	Description	Example
Morphological Richness	Arabic words come in many forms, derived from a rich inflection system.	واسيدرسونها wa+sa+ya-drus-uuna+ha = and they will study it
Orthographic Ambiguity	Diacritics are optional in Arabic script, however it helps to distinguish between identical words with different meaning.	كُتِبْتُ ktḅt (no diacritics), كَتَبْتُ katabtu = I wrote, كَتَبْتُ katabat = she wrote, كَتَبْتُ ka+tibat =such as Tibet The word for ‘car’ is
Dialectal Variations	Arabic has many dialects, like Gulf, Egyptian, Levantine, and Moroccan Arabic. Arabic dialects have their own grammars and lexicons.	سيارة sayyaara in Standard Arabic, عربية arabiyya in Egyptian Arabic, كرهبة karhba in Tunisian Arabic, موتّر motar in Gulf Arabic

1.2.2 Text Classification

Text classification is about classifying text into various categories. Arabic text classification has many applications involve dialect identification which identify an input text into common Arabic dialects like Egyptian, Gulf, Iraqi, Levantine, and more. Moreover, sentiment analysis is an other example of the text classification applications where the given text could be a positive sentiment or a negative sentiment.

Neural networks including deep learning approaches acquired an interest in the most recent studies. Neural networks models have the advantage of the ability to handle large datasets. In general, neural networks including deep learning approaches showed promising results in terms of Arabic text classification. Therefore, we apply neural networks and deep learning approaches to classify text into a time period.

1.2.3 Division of Literature Arabic eras

This section discusses dividing the Arabic literature into eras regarding the historians and literary critics accord. The division of Arabic literature eras is a debatable subject with similarities and differences in the division presented in the literature. Hence we will review the divisions and follow the most reasonable one.

Daif [13] divide the Arabic literature into five eras based on the agreement of the most of the historians [13]. Pre-Islamic era, Islamic era, Abbasid era, *Aldoul wa al-amarat* era, and modern era are the five eras. The previous division was from the first book in the a ten volums series titled "History of Arabic Literature". Each era is illustrated by the author in detail in one book or more. *Aldoul wa al-amarat* era were discussed in five books covering most of the locations of the Arabic regions. Interested readers in Arabic literary history can find more details in the book series [30].

Furthermore, a similar division was introduced by Al-Fakhoury [14] for the Arabic literature eras. The author identifies five eras: the pre-Islamic era, Islamic era, Abbasid era, Turkish era, and modern era based on three revivals in history which are Pre-Islamic and Umayyad renaissance, Abbasid renaissance, and modern renaissance. Comparing eras in this book to Daif [13] book, we can see it is identical except for the fourth era, and the end date of the Abbasid era is different.

Ghoniem [3] focused on discussing the history of partitioning the Arabic literature eras from previous studies. Besides the eras division, the author argues the naming of Arabic literature eras introduced by earlier research. The characteristic of each era and the period represented by that era are the main aspects of the author's argument.

Table 1.4: The periods of six literature eras by Ghoniem [3].

Era	Period
Pre-Islamic	150 years before 1 AH
Islamic	1 AH – 132 AH
Abbasid	132 AH – 334 AH
<i>Aldoul wa al-amarat</i>	334 AH – 923 AH
Ottoman	923 AH – 1335 AH
Modern	1335 AH – till date

As a result, the study proposed six Arabic literature eras: Pre-Islamic era, Islamic era, Abbasid era, *Aldoul wa al-amarat* era, Ottoman era, and modern era. Table 1.4 presents the start and the end of each era according to the Hijri calendar. The most extended period is for the *Aldoul wa al-amarat* era, whereas the shortest one is the modern era.

The preceding researchers agreed on identifying the first three eras as Pre-Islamic, Islamic, and Abbasid eras. Moreover, the Islamic era is composed of Islamic and Umayyad periods in most of the studies. However, the end year of the Abbasid era is varying in all the proposed divisions. Moreover, we found that the fourth era in [13] and in [14] divisions are conflicting and overlapping, which Ghoniem [3] introduced them as separate eras by dividing the Arabic literature into six eras instead of five eras.

Furthermore, Daif [13], and Ghoniem [3] mentioned that the Abbasid era can be divided into three intervals representing the first Abbasid era, second Abbasid era, and third Abbasid era. We believe that Arabic literature eras division is complicated and an open research area for historians and literary critics. Thus, in our research, we will use a recent division by Ghoniem [3] that avoids overlapping between the eras.

1.2.4 Overview of Some Arabic Literature Eras

We show an overview of some of the six Arabic literature eras mentioned previously. Pre-Islamic, Islamic, and Abbasid eras were the most constant in the Arabic literature eras division. Therefore, we select them to present an overview of that time periods.

Pre-Islamic Era: started in the middle of the fifth Gregorian century and ended with the advent of Islam [31]. The Arabs were the inhabitants of the Arabian Peninsula. Many tribes were spread during the pre-Islamic era. Each tribe was fully self-governing and independent. The tribes were not united into a comprehensive political. *Quraysh* is a known tribe in the Pre-Islamic era which lived in Makkah city.

Furthermore, most of the Pre-Islamic literature has been lost except what reached us until now [14]. Their literature reached us through the narrators like *Al-Asma'i*. Linguists were concerned about losing the old literature that belonged to the Pre-Islamic era. Therefore, at the start of the second Hijri century, they wrote it down to save it. Moreover, the Pre-Islamic literature was affected by the desert environment in which the authors lived.

Islamic Era: consisted of three periods: the emergence of Islam and the mission of the Prophet Muhammad (peace and blessings be upon him), the era of the Rightly-Guided Caliphs, and the Umayyad state [14]. Islamic period shows how the new environment impacted Arabic literature. The main factors related to the new environment are the advent of Islam and the Arab interaction with other nations.

Quran is the book of revelation in Islam which helps in unifying the language and preserving it [14]. It is the basis of Islamic religious sciences and Arabic sciences.

Furthermore, the Arabs mixed with the Persians and the Romans, and their customs infiltrated them. As a result, the economics improved throughout their state. However, the region was insatiable at times because of wars and strife. The tendency of the Umayyads to revive the tribalism restricted the development of literature.

Abbasid Era: started at the end of the Umayyad state. The political coup of the Abbasid state impacted the economic environment, the social environment, the religious environment, and the cultural environment [14]. Moreover, the Abbasid culture was affected by foreign cultures like the Greek culture. It is clear from all of the above that the Arab culture had expanded. Therefore, the Arabic literature was affected by these changes.

The movement of translation and copying other languages literatures into Arabic was the main reason that delivered most of the ancient sciences to the Abbasid state. They printed and published these sciences over all the Abbasid state. Therefore, the streams of Arab thought diversified, the public libraries and scientific institutes became wide spread.

1.3 Problem Statement

As discussed, the Arabic language changes over time, as indicated by several studies like Belinkov et al. [5]. In this research, we are investigating the temporal period classification of Arabic texts. For that, we set research questions that need to be addressed through the research. The research questions are as follows:

1. How effectively can neural networks including deep learning approaches classify

Arabic text into its era?

2. Does classifying the Arabic text in custom temporal periods differ from classifying it into predefined literature eras?
3. Does classifying Arabic poems into literature eras differ from classifying Arabic text into literature eras?

1.4 Overview of The Thesis

In our thesis we will review the related studies for the temporal classification task in the survey chapter 2. The dataset chapter 3 will present the data collection details and the data analysis. Moreover, we will explain the models structure and properties in the methodology chapter 4. The experiments and results chapter 5 will present the different experiments setups and report all the results and the best model architecture. Lastly, we will show a brief of our work and the possible future works in the conclusion and future work chapter 6.

CHAPTER 2

SURVEY

2.1 Introduction

The languages develop through time which results in some changes [11]. Several of these changes reflect specific periods of literature or history [3]. Therefore, classifying text into periods helps in detecting changes in the language. Besides that, it is beneficial to identify the period for unclassified texts.

In our survey, we intend to review the studies that classify the date of a text document, whether the exact date or period it belongs to. Most researchers benefit from machine learning approaches to address the problem. On the other hand, some researchers follow statistical methods. Thus, modeling the problem varies between the studies.

Datasets, features, and methods are the aspects that we will focus on for each study. We present diverse datasets from the literature that was reported for this task. Furthermore, we show an overview of the features which were used by the previous studies. Finally, we preview the methods utilized to address the task of classifying

text into periods.

We also draw the attention to the competition that held in 2015 for temporal period classification. Popescu et al. [32] introduced the Diachronic Text Evaluation(DTE) task as competition on Semantic Evaluation 2015 ¹. The task is divided into three subtasks as Task1 (T1), Task2 (T2), and Task3 (T3). T1 is to predict the time for documents with well-known persons or events related to a specific period. T2 is classifying documents that have different use of the language based on the time related to. T3 detects the time expressions in the document. A newspapers dataset posed to be used by the participants for developing the models that will address the problems. Moreover, they define three types of intervals: fine, medium, and coarse. The sizes of the intervals as follows: six years, 12 years interval, and 20 years interval. These intervals overlap each other, and each one has an equal number of years before and after the publish date of the news item. However, the intervals were considered in T1 and T2 but not for T3. They introduced a score that depend on the loss value to evaluate the models. We will refer to it in our survey as DTE score. Lastly, four teams participated in the competition; however, only one team addressed all the tasks.[33]

To the best of our knowledge, no survey for classifying text based on temporal periods was reported in the literature. This chapter is divide into the following sections: Datasets, features, methods, and discussion.

2.2 Dataset

Data, in general, is paramount in the experiments for the research work. Quality, size, and source are some of the requirements for the dataset. Moreover, special requirement differs from one task to another. For this task, the essential requirement is to have enough data spanning through different time periods.

In this section, we will present the datasets used by the covered studies in our survey. English is the most used language in the researches experiments. Thus, we split the datasets into English datasets, and datasets for Arabic and other languages datasets.

2.2.1 English Datasets

GigaWord English Corpus

GigaWord is a famous corpus from Linguistic Data Consortium (LDC)². LDC started in 1992; it is an open association between libraries, universities, government research laboratories, and corporations.

GigaWord English corpus utilized by [34], they used 395.9K news items spanning nine years interval. 10% of the data were used for training the model and the remaining were used for the test purpose. The important information extracted for each sample: date, a day of the week, week number, and quarter which were calculated from the exact date of the document.

Project Gutenberg Corpus

Project Gutenberg corpus is an online library of free electronic books (eBook)³. The library contains over 60K free eBooks.

Kumar et al. [35] used texts in the range between 1798 to 2008 from the Gutenberg project. Each sample is labeled by the published year. They split the dataset into development and test sets with 333 and 345 short stories for each set, respectively. However, the authors did not specify details about the sample length in terms of tokens. Moreover, external data was used besides the Gutenberg corpus to obtain the gold standard. Biographies from Wikipedia in the range 3800 B.C. to 2010 CE. Beside the Gutenberg corpus to obtain what the authors in [35] term as the gold standard, i.e. the mid point between the author birth date and dead date. They excluded the biographies for the individuals whose birth date, and death data are out of the range.[35]

Furthermore, Project Gutenberg corpus, used by Colgrove et al. [36]. They focuses on novels that were published in the four literary movements: Augustan, Romantic, Naturalist, and Modernist. The selected periods are between the 18th century to the mid-20th century, which is quite distinct. In addition, the text chosen by the author, for each period, they select the primary authors who represent the style of that literary periods. They extracted 282 samples and created datasets with various settings to evaluate the model from different aspects. However, the length of the extracted samples are not specified.

Google Books N-Gram

Google books N-Gram includes historical data prepared by Google for different research purposes. The publish dates of the books are in the range of 1500 CE to 2008 CE. The first version of the corpus contains 5M books with a total of about 500 billion words in seven different languages[37]. The second version incorporates 8M books in eight different languages [38]. Moreover, the first version has uni-grams, bi-grams, and up to five n-grams, whereas the second one has up to eight n-grams. However, all the studies that utilized Google books corpus focused only on the English language.

Mihalcea et al. [39] used the first version of Google Books N-Gram. A set of target words has been considered according to the word frequency in a certain period or all the periods. Thus, the top 100 texts snippets being collected for each target word in the corpus and used as samples. Furthermore, the authors in [40], [33] also utilized the first version of Google Books N-Gram corpus.

Diachronic Text Evaluation Datasets

The datasets introduced by Popescu et al. [32] for Diachronic evaluation task semeval2015 competition⁴. They defined three tasks: Task one (T1), task two (T2), and task three (T3). Their data is collected from diverse news sources published between 1700 CE to 2010 CE. However, a snippet of news was used in the competition, not the complete news item. The snippets length is in the range of tens to a couple of hundred words that focus on information relevant to the time period. The dataset size depends on the task number; for instance, T1 used 451 documents. T2 and T3 datasets sizes are 6.5K documents and 115 documents, respectively. The datasets are

labeled by three intervals as follows: Fine, medium, and coarse as will be explained in Section 2.5. The intervals are used only with T1 and T2. Moreover, each text has a tag indicating when it was published. [32]

The authors in [32], [24], [41] are the three teams that participated in this competition and utilized previous datasets. However, competitors address part of the tasks, not all of them. For example, Salaberri et al. [41] solved all the three tasks. In addition, the authors in this [42] used the T3 dataset as a source along with two other sources to composed their corpus.

Corpus of Late Modern English Texts Version 3.0 (CLMET3.0)

CLMET has three versions, but the third one superseded the others. CLMET3.0 is an English dataset collected from the Oxford Text Archive and the Project Gutenberg ⁵. CLMET3.0 is comprised of about 10 million tokens and split into three intervals of 70 years. Niculae et al. [15] used CLMET3.0 for their English experiment [15].

Diachronic Text corpus (DATE)

DATE is a collection of English texts created by [42] from three sources: COHA corpus, T3 dataset from Diachronic Text Evaluation[32], and portion of CLMET3.0 corpus. The document’s dates are between 1700 and 2010. The covered years in the three datasets are diverse. Fremann et al. [42] used their DATE corpus, the sample is a short text with around ten words in length labeled by the publishing year.

Newspaper Corpus

Kanhabua et al. [43] created the newspaper corpus from the Internet Archive ⁶ to enhance a temporal language model. Internet Archive is a free library with millions of books, websites, software, and more. They collected the history of web pages, most of which are newspapers like New York Post and ABC news. For each source, data of eight years period has been collected with a total samples of 9K documents. The total sources are 15, ten of them are used for training purpose and the remaining for test.[43]

2.2.2 Datasets for Arabic and Other Languages

Open Islamicate Texts Initiative (OpenITI)

OpenITI, proposed by Romanov et al. [44] is a sizeable historical corpus of Arabic and other languages, with a total size of 1.5 billion words [44]. The corpus was collected from different periods and freely is available resource of religious and literary texts. Al-Maktaba Al-Shamela (known as Shamela)⁷ is the primary source of these texts (cf. [5]). The documents are labeled by the author’s death date. Lastly, Belinkov et al. [5], used the OpenITI corpus in their experiments.

Arabic Poems Dataset

The dataset collected from the Adab website ⁸, it contains about 58K poems ranging from the 6th to the 21st century. In addition, the dataset includes information related to the poem like poet name, poem title, poem category, and more details explained

in the dataset website ⁹.

Abbas et al. [27] focus on the Arabic poetry to be classified in different eras. Thus, they used only the categories that represent literature eras for their experiment. Thus, the pre-Islamic era, Ummayyad era, Abbasid era, and Andalusian era are the four classes that data belong to.

Arabic Poems2 Dataset

Orabi et al. [6] extracted their dataset from the Adab website as the above dataset. They include five classes as follows: the pre-Islamic era, Ummayyad era, Abbasid era, Andalusian era, and the modern era. The poems are split into snippets of 20 lines and limited to 200 words in each snippet. After splitting the poems, they have about 86,061 poem samples.

Colonia Corpus

Colonia is a Portuguese corpus composed of historical text in the range of 16th to the 20th century by Zampieri et al. [45]. The corpus contains 5.1 million tokens, and every single sample includes up to 90K tokens.

Stajner et al. [4] employ Colonia corpus in their conducted experiment to classify the century of a text. However, they used a subset of the corpus that contains 87 samples. The maximum size of the sample is 2K of tokens. Furthermore, Colonia corpus used in [15] along with another two corpora in a different language.

The Romanian Corpus

Corpus was introduced by Ciobanu et al. [46], that represents various phases of the evolution in the Romanian language. Moreover, it contains 26 documents spanning from the 16th CE to the 20th CE century. For more samples, they split the texts in the dataset into chunks of 500 sentences. Each sample is labeled by the century that was written in. Thus, the number of the classes are five which represent the centuries from 16th CE to 20th CE.[46]

Ciobanu et al. [46] proposed a temporal classifier that trained and tested on their introduced corpus. Furthermore, Romanian corpus being utilized in [15] as part of their experiments to train the model in the Romanian language.

Documents of Early England Data Set (DEEDS)

DEEDS ¹⁰ is from the University of Toronto contains charters document written in the Latin language. The dataset text spanning from 1089 to 1438, and among all the documents, half of the words occur once. DEEDS contains about 10K documents; however, the authors in [16] used only 3353 documents for their experiment.

DEFT Corpora

DEFT was proposed by[47] for DEFT 2011 challenge. The corpus consists of seven French newspaper snippets that its created date between 1801 and 1944. Each snippet consists of about 300 to 500 words.[47]

Garcia et al. [29] train and test their model on DEFT corpora. The dataset was divided into a training set, development set, and evaluation set. The training,

development, and evaluation sets contain 2396, 1200, and 2445 items, respectively.

Dutch Corpus

Jong et al. [28] stem the data from Twente News Corpus ¹¹. They collected the data from two famous Dutch newspapers in Twente News Corpus. The news items dates are between 1999 CE and 2005 CE. The size of data is 2GB with about 1.3 million unique words. The dataset was used by Jong et al. [28] to train and test their model.

2.3 Features for Temporal Prediction

Features are used to improve the temporal classification performance. Different features are applied like: Stylistic features, lexical features, grammatical features, and Length meta-features. Length meta-features present the number of sentences or tokens. Considering the structure of the sentences in a grammatical way, as part of speech tags (POS) is an example of the grammatical features. Stylistic features can show how texts are diverse from each other by the lexical density, average sentence length, and more. N-gram of words or characters are types of lexical features.

We divide the features into five categories: Stylistic, lexical, grammatical, length meta, and other features. The feature which is not under predefined types will be grouped in the other features type.

Table 2.1: The formula for the four features proposed by [4].

Feature	Formula
Average sentence length	Total number of words / total number of sentences
Average word length	Total number of characters / total number of words
Lexical density	Total number of unique tokens / total number of all tokens
Lexical richness	Total number of unique lemmas / total number of all tokens

2.3.1 Stylistic Features

Lexical density and lexical richness features are utilized by [4], [24]. Average sentence length and average word length features applied by [36], [4], [46]. Table 2.1 presents the formula for the previous features. Moreover, many stylistic features applied by [36] as follows: ratio of words to punctuation, ratio of non-punctuation characters to punctuation chars, and the average length of dialogue passages.

2.3.2 Lexical Features

Token n-gram used as the lexical feature in [24]. Character n-grams and word n-grams applied by [25], [29]. For the word n-grams, the authors in [29] applied it from one to four grams. However, the best result was achieved by the two grams, whereas 5-grams was the best result for the character n-grams [29]. The lexical feature utilized by [15] was the number of occurrence of all the words that present at least once in a document. Besides the previous feature [15] applied character n-grams feature. Furthermore, a lexical feature used by [39] was the most five words that represent the context of a certain word period and occur at least three times on it. Lastly, an n-gram tokenizer was utilized by [27] to improve the model performance.

2.3.3 Grammatical Features

Part of speech (POS) n-grams utilized by [24], [25], [36], [43]. Syntactic phrase-structure rule occurrences applied by [25]. Moreover, POS of the current word, the context of ambiguous word with a size of three words before and after it, the POS of the surrounding words, the noun to the left and the right of the target word, and the verb to the left and the right of the target word are the features used in [39] experiments. However, they name it as local features.

2.3.4 Length Meta-Features

The number of sentences, types, and tokens are the length meta-features used in [24] beside the other features. The authors in [36] study applied length meta-features. They utilized the following features: vocabulary size (total number of unique words), number of sentences, number of words in dialogue passages, and number of uni-grams.

2.3.5 Other Features

Dalli et al. [34] applied the probability density functions. They calculate it for each word, and then it is weighted according to the inverse document frequency.

Furthermore, Salaberri et al. [41] used linguistic features in the language change approach for T1 and T2 from [32] competition. They study the patterns of language change that appear in the training data. These patterns consist of linguistic factors such as orthography, lexicon, semantics, morphology, and syntax.[41]

Besides the POS Kanhabua et al. [43] applied more than four semantic pre-

processing. Collocation extraction, word sense disambiguation (WSD), concept extraction, and word filtering are the used techniques for semantic pre-processing. Meaning of word could change dramatically according to the co-occurrence with other words, which explains the importance of collocation extraction pre-processing. WSD analyzes the context of a sentence to determine the correct meaning of a particular word. To avoid the issue of low occurrence words, they utilized the feature that called concept extraction. Lastly, in order to decrease the vocabulary size, they select the top-ranked terms according to TF-IDF technique.[43]

In addition, Ciobanu et al. [46] utilized three features along with the stylistic features. Another feature applied is called stopwords which is the frequency of the top 50 words. The third feature is named endings that represent the occurrence of the word suffixes with lengths up to three and occur at least five times. The last feature is a dictionary, which is the number of words that match predefined filters from the dexonline database. The dexonline is a collected data from about 30 Romanian dictionaries covering diverse periods.[46]

Over and above, Garcia et al. [29] refer to the chronological methods like named entities to predefined the following features: birth dates of persons, neologisms, and archaisms, and French spelling reforms [29].

2.4 The Temporal Period Predictions Task

In this section we will present a brief discussion about the temporal period predictions task. The task was modeled differently in each research. Most of the studies introduces

the task as a classification problem. Few studies addressed the task as a regression. Additionally, researchers have been investigating grouping temporal periods using unsupervised learning techniques like clustering. Both machine learning and statistical methods were utilized for the temporal period predictions task. Although machine learning was used more than the statistical methods.

The predictions problem as discussed before either a classification or a regression. If the model predict a certain date, it considered as a regression problem as the years can be continues. But treating time as discrete set of possible categories is a classification problem. Moreover, several setups in terms of class were examined. Time periods were defined variously in the reviewed researches. Decades and century are examples of the used time period.

Time periods were identified according to the the literature development or introduced by the researchers. In general, the literature development periods represent historical periods like the literature eras in Abbas et al.[27] work. Moreover, the custom defined periods by the researchers represent a decade or little more or less.

Furthermore, features engineering was used for temporal periods predictions in some studies. The features were combined into different sets to form different features setups. The effectiveness of individual features was also investigated. Lastly, only one study used regression applied the features engineering, and the remaining were classification problems.

2.5 Methods of Predicting Texts Based on Temporal Periods

Various methods were introduced to model the problem of classifying texts into temporal periods. As mentioned earlier, most of the researchers applied their work to English text. The others studies classified texts from several languages as Arabic. Thus, we will split the studies into subsections as follows: the studies that were conducted to classify the English text only, the studies that classify text from Arabic and other languages. However, the studies that use a set of different languages datasets including English, will consider as another language study.

2.5.1 Works Related to English Texts

Dalli et al. [34] proposed an unsupervised method that detects the periodic component from text to create time series of the language model. Rules based on temporal word associations were created by the temporal language model. These rules are used to predict the publish date of the news articles automatically. The experiments were conducted on a news dataset from GigaWord corpus that spanning nine years range. Five models are being applied to predict a set of information: Day of the week, week, month, quarter, and the year. To evaluate the model, the percentage of the correct prediction calculated, the best result achieved by the year model.[34]

Jong et al. [28] applied statistical language models to explore the development of the Dutch language [28]. This study will be reviewed later in section 2.5.2. Moreover, Kanhabua et al. [43] attempted to improve the language model that was proposed

by Jong et al. [28]. In order to enhance the model performance, they applied four features as semantic-based pre-processing. Three methods were proposed to improve the language model: Word interpolation, temporal entropy, and Google Zeitgeist; they use search statistics. Word interpolation solves the problem of words that have zero probability. At the same time, temporal entropy is about the weight of the term and term selection method. The third method is to examine the effect of using external information like search engine statistics. Furthermore, news data was collected from various resources to train and test the model (see section 2.2). The dataset is split into a training set to build the language model and a test set to predict the timestamp of the text. Data was labeled by five granularity: week, month, 3-months, 6-months, and 12-months. They combined the features and the approaches in various groups to evaluate the effect. Precision and recall are the performance measurement used to assess the model. As a result, the model with all the semantic-based pre-processing and Google Zeitgeist outperformed other models with 0.71 precision. Lastly, the authors created a tool, which is explained in this study [48].[43]

On the other hand, Colgrove et al. [36] considered the problem as genre classification of the four literary movements. They used the Gutenberg corpus for their experiments. Many authors and texts are available in the selected period: Augustan, Romantic, Naturalist, and Modernist. Moreover, they focus on English fictional novels. Four setups for the dataset are being created to test the classifiers. Various features were extracted from the dataset and used to train a maximum entropy classifier. Precision, recall, and F1 are the evaluation metrics used. Their results vary from one setup to another. Moreover, the authors applied various setups but the most

standard one as they mentioned was called the general setup. The results for that setup achieved F1 of score 0.80.[36]

Wijaya et al. [40] attempted to identify when the change of word semantic occur. They applied Topics-Over-Time (TOT) and k-means clustering on Google Books N-gram corpus. Moreover, they focus on words representing entity of interest, then two words before and after that word retrieved. A pre-processing step is utilized for the k-means method. The term frequency-inverse document frequency (TF-IDF) scores were computed for all the words in each document. Whereas for the TOT method, a match count is used. Experiments were conducted on chosen words that present diverse behaviors through the time to be analyzed. As a result, the chosen words were divided into three groups of semantic changes: Change in meaning, getting additional meaning, and unchanged meaning.[40]

Kumar et al. [35] proposed a supervised language model that classifies the text date [35]. Their model is built of histograms based on Kullback Leibler’s divergence between the language models that were linked to the documents. Gutenberg project corpus along with Wikipedia biographies dataset are used to train and test the proposed model. The dataset split it into development and test set equally. A baseline model was constructed to compare their results to it. They considered a constant prediction for all the samples in the baseline model. The year 1903 was the fixed prediction, which was the midpoint of all the publication dates. Their model outperformed the baseline by decreasing the median error from 50 to 23 score.[35].

Classifying the period of word usage proposed by Mihalcea et al. [39]. They used Google Books N-Gram corpus for their experiments. Experiments were conducted on

word usage collected from three years intervals: 1800 CE, 1900 CE, and 2000 CE. They create a word target set contains 165 words covering all the predefined periods. The top 100 snippets were collected for each target word from the Google books corpus. A set of features integrated within the Naive Bayes classifier. To evaluate the model, average accuracy was calculated using 10-fold cross-validations. As a result, their model presents improvement by 18.5% comparing to the baseline model that achieved 43% accuracy.[39]

Popescu et al. [33] focused on a statistical methods to identify the uncommon relationship between the English language usage and the interval of time. They used the Google N-gram corpus to investigate term distribution and its relation to emotion words. Various statistical tests are utilized to analyze the data: Normalization, Welch’s test, Run Test, Least-Squares, Ratio, C1, C2, and Spearman and Kendall Test. The statistical tests were able to detect the language change, but they could not define the period of that change. Thus, they propose an algorithm for period detection using statistical tests. The output of the algorithm is a set of possible years related to the changes. Furthermore, experiments were conducted on a list of political terms using the proposed algorithm. The algorithm identifies six periods in the range between 1899 and 2009 for 761 political terms. Their system achieved 69% accuracy, which is better than the baseline system that achieved only 15% accuracy.[33]

Stajner et al. [4] focus on stylistic changes on old text spanning from the 17th to the 20th century. To automatically extract the stylistic features from the corpus, they apply quantitative analysis. Moreover, different experiments are being employed to classify the text into five set of classes. The athours used Colonia corpus as disucssed

in section 2.2. They create different classes setups: all the centuries from 17th to 20th, between (17th - 18th) and (19th–20th) centuries, between the 17th and 18th centuries, between the 18th and 19th centuries, and between the 19th and 20th centuries. Furthermore, they use the Weka tool to train and test the following classifiers: Naive Bayes, Support Vector Machines using Weka implementation (SMO) JRip, and J48. F-measure utilized to evaluate the model performance. The highest F-measure of 0.92 was achieved by Naive Bayes and SMO.[4]

Salaberri et al. [41] presented their work for addressing all the DTE competition tasks. Temporal text classification is used to address T1 and T2. Year entity detection, Wikipedia entity linking, Google NGrams, and language change approaches were utilized to predict the period in which the text was written. Among the four approaches, the final decision is considered by the highest precision. Year entity detection approach outperformed all the approached and achieved the highest precision. T3 is solved by detecting the relevance of phrases in the sentences to their period. T2 achieved the best result in the conducted experiments with a 0.62 DTE score.[41]

Anachronism Modeling by Ranking (AMBRA) system was proposed by Zampieri et al. [24] which solves the T1 and T2 of DTE competition. They apply a ranking approach depend on pairwise comparisons was proposed by Niculea et al. [15] to classify the news document into intervals. However, the previous method does not deal with interval as Zampieri et al.[24] did. Thus, the authors used a non-parametric memory-based approach to make it better. Furthermore, a set of features were applied as explained in section 2.3. Two features setups were reported for model. They named the model that utilized all the features as AMBRA full. Moreover, they created a ridge

regression as baseline model. Lastly, the AMBRA full system outperformed all the models including the baseline model with DTE score of 0.87.[24]

Szymanski et al. [25] participated in T2 only from DTE tasks by [32]. Three periods were introduced for competition tasks to classify the text according to it. The authors include an extra period besides the three period intervals introduced in advance. The extra period interval size is 50 years. Support vector machine (SVM) is applied to classify the period of a given text. To improve the classifier performance, four features were used. Moreover, along with the stylistic features as mentioned in Section 2.3, they use Google Books Syntactic N-Grams (GSN). The best result was achieved by using the four stylistic features and features extracted from GSN with 73.3% accuracy for the 50-year setup.[25]

Scan system based on Bayesian model proposed by Frermann et al. [42] to identify the change in meaning through time. Their system detects the word representations over time as senses and their prevalence. Several experiments were applied to a large corpus extracted from three resources as explained in section 2.2. Temporal dynamics, novel sense detection, word meaning change, and task-based evaluation are the four experiments. The parameters are different for each experiment; for example, the size of the years interval is 20 years in the first and the second experiments, but for the third experiment, it is ten years only. In the task-based evaluation experiment, they evaluate their model by the tasks proposed by [32]. Moreover, SVM with and without character n-gram is being applied within their model. Consequently, a comparison was made between their results and results from [24], [25], [41]. Their model outperformed all the systems in the task T1. However, for T2, their model outperformed all the

systems in terms of precision only, the best accuracy achieved by [25] model.[42]

2.5.2 Works Related to Arabic Text and Other Languages

Texts

Abbas et al. [27] classified Arabic poems into different eras as follows: Pre-Islamic era, Umayyad era, Abbasid era, and Andalusian era. They used the poem dataset that were shown in section 2.2, it composed of 58K poems from the 6th to the 21st century. For the experiments, they use Weka to examine eight various classifiers: K-Nearest Neighbors, Multinomial Naive Bayes, Bagging classifier, and five more classifiers. Multinomial Naive Bayes outperformed other models, with an accuracy equal to 70.21% and F1-Score of 68.8%.

Belinkov et al. [5] presented a novel automatic periodization algorithm, namely Word-Embedding-based Neighbor Clustering (WENC). WENC helps to study the Arabic language by identifying the period of language development. The ability to divide classical Arabic into different periods based on linguistic evidence is one of the research questions of their research. Furthermore, the OpenITI corpus used in this study and the pre-processed version of the data is available for the researcher. They enhanced the proposed algorithm by Porat et al. [49] to identify the text reuse in the OpenITI corpus. The text reuse includes identifying boilerplate passages, identifying extremely frequent short phrases, and identifying approximate matches. Quranic verses are an example for the boilerplate passages where *peace be upon him* ”عليه السلام” is an example for the extremely frequent short phrases. Moreover, word embeddings is

the base of the WENC to automatically grouping periods of similar language use. They found proof that several distinct periods exist in the Arabic language development, even though words in Arabic literature persist more than words in English literature. As a result, they identify three main periods for Arabic language development which are early period, middle period, and late period.[5]

Orabi et al. [6] attempted to classify Arabic poems into the same eras as in [27] in addition the authors added a new era which they term as the modern era. This work proposed a convolutional neural network (CNN) model to address the problem. They composed an updated version of the Arabic poetry dataset used by [27]. Their dataset extends the [27] dataset by 2k poems. They examined many classes setups in their experiments. Binary class and set of multi-class setups. Each classifier has two models: the baseline model and the deep learning model. FastText was used to build the baseline model, which is a word embedding using unlabeled Arabic poems corpus. Whereas CNN is utilized in the deep learning model. Modern and non-modern are the classes for the binary classifier. A different set of classes is applied in the multi-class classifiers as follows: three classes and five classes classifiers. Accuracy and F1-score were applied to measure the model performance. F1-score of 0.914 was reported for the binary classification task, whereas using CNN with removing the frequent terms (frequent terms that present in the dataset with more than 25% of the poems). Lastly, for the three eras classification problem they report F1-score of 0.875, whereas for the five eras classification F1-score of 0.796 was reported.

As far the other languages, a statistical language models applied by Jong et al. [28] to investigate the evolution of the Dutch language. To classify an undated doc-

ument, they compare it to the constructed language model. Two methods were applied to compare language models: comparison on document level and comparison on partition-level. Experiments were conducted to test the model’s usefulness for the dating task. The data used to test the model consist of articles extracted from two famous Dutch newspapers which discussed in section 2.2. As a result, the comparison on the document-level method outperformed the comparison on the partition-level method.[28]

Automatically identifying the date of a document in the French language presented by Garcia et al.[29]. The DEFT corpus is used to train and evaluate the system, and their dataset is split into training, development, and evaluation sets. The documents are classified based on the following classes setups which are years or decades. Chronological methods and classification methods were utilized in this study. Named entities, *neologisms and archaisms*, and *French spelling reforms* are the chronological methods. For the classification models, they apply cosine similarity-based and SVM. Due to the lack of the DEFT corpus data, the authors used Google Books Ngrams corpus as an additional dataset to train the cosine similarity-based model. The setup where the classes are divide based on decades achieved better results than the year class setup. They used the similarity score is based on a Gaussian function to evaluate their model. Score of 0.42 was the best result achieved by the SVM model with decades setup.[29]

Statistical study on the charters documents written in Latin presented by Tilahun et al. [16]. They used a subset of the DEEDS corpus in the range between 10 to 14 centuries. Three different methods were applied to date the undated documents. K-nearest neighbors (kNN), maximum prevalence (MP), and quantile regression (QR)

are applied with diverse optimal parameters. kNN calculates the distance between the documents then estimates the date based on the similarity. MP assigns a probability of document occurrence in a certain period, then the date predicted according to the highest probability of occurrence. The minimum value in the QR plot is the estimation for the document date. Furthermore, to measure the performance, they applied several metrics like mean absolute error (MAE). The best result achieved by the MP method with shingle size two by 6.0 MAE.[16]

Ciobanu et al. [46] focused on Romanian language temporal classification [46]. Two classifiers proposed to predict the time of Romanian documents written between the 16th to the 20th century. The Romanian corpus used to train and test the models. Support vector machines (SVM) and random forests were applied along with four predefined features. However, the lexical feature is the most beneficial to the model. Several experiments were conducted by utilizing all the possible sets of the four features (see section 2.3). Random forests outperformed SVM by achieving 99.25% accuracy.[46]

Niculae et al. [15] proposed a combined method of text ranking and probability to address temporal text classification [15]. English, Portuguese and Romanian are the covered languages, each language with an independent corpus that discussed earlier in section 2.2. Moreover, they applied a pairwise method for the ranking model, and for the ordering, they obtained probability values using logistic regression. The ranking model learns the directions that keeps the order of the temporal texts as much as possible. Lexical and morphological features are used with the model. Furthermore, each corpus were divided into two equal sets for training and test. Decade and century

are the class setups that were applied in their experiments. To evaluate the model, they follow Pedregosa et al. [50] evaluation by using the percentage of correctly ordered pairs. Ridge regression and two SVMs with century and decade levels were applied to the three corpora. A comparison between the two previous models and the pairwise logistic method presents that their approach outperformed the other models. Overall, the performance of the models in terms of accuracy is between 83% and 93%.[15]

2.6 Discussion

In this survey, we attempt to cover most researches that classify text into time periods. Due to the lack of related studies to our domain, we include studies in different languages. Next, we will discuss the reviewed studies from different aspects: used data, label type features, and Best methods.

More than half of the reviewed studies were conducted in the English language. The remaining studies are in various languages as follows: Arabic, Latin, French, Romanian, Dutch, and Portuguese. Table 2.2 presents data source of data, the language of the data, and total covered years in all the reviewed studies. Only one of the studies examined multiple languages which are: English, Portuguese and Romanian by [15].

From Table 2.2 the most used of the data source are news and historical texts. Historical texts represent the data from books and literature including poems. In contrast, news data are from newspaper articles and news stories. Therefore, from the reviewed studies, we found that we have two main categories. Studies that pre-

Table 2.2: Summary of the covered studies in terms of data source, the language of the data, and the total covered years.

Research	Data Source	Language of The Data	Total Covered Years
Jong et al. [28]	News	Dutch	6
Dalli et al. [34]	News	English	9
Kanhabua et al. [43]	News	English	8
Colgrove et al. [36]	Fictional novels	English	239
Garcia et al. [29]	News	French	143
Wijaya et al. [40]	Historical texts	English	508
Kumar et al. [35]	Historical texts	English	210
Tilahun et al. [16]	Charters	Latin	349
Mihalcea et al. [39]	Historical texts	English	200
Popescu et al. [33]	Historical texts	English	209
Ciobanu et al. [46]	Historical texts	Romanian	400
Stajner et al. [4]	Historical text	Portuguese	300
Niculae et al. [15]	Historical texts	English	400
		Portuguese	
		Romanian	
Zampieri et al. [24]	News	English	310
Szymanski et al. [25]	News	English	310
Salaberri et al. [41]	News	English	310
Frermann et al. [42]	Historical texts	English	310
Belinkov et al. [5]	Historical texts	Arabic	1435
Abbas et al. [27]	Poems	Arabic	1014
Orabi et al. [6]	Poems	Arabic	1542

dict temporal periods for data obtained from news sources and the other category is classifying historical texts. Each category involves challenges that are solved using several techniques.

News, in general, is written to describe events or to present facts. Thus, news text is different than historical texts which contain books that focus on specific subjects. As well, historical texts contain writing styles and diverse vocabulary. Whereas news data focus on events and famous characters related to the publish time. The maximum covered years for the news data is about 310 years as in [32], and the minimum is six years like [28]. Thus, such kinds of studies would be beneficial for future classifying research that involved short ranges of years.

On the other hand, predicting the period of historical texts required more effort as compared to news due to the complexity of the language. Most of the researches in English utilized Google Books N-Gram for the historical texts. The minimum year's range in the category of the historical text is about 200 years. Whereas the maximum is 1542 years used by [6] work for classifying Arabic poems into eras. Therefore, studies that predict historical text involved long years range.

Furthermore, in term of total covered years by the researchers in their studies, Arabic language researches are the longest. As mention earlier, the most extended years range is 1542 years by [6]. Subsequently, 1430 is the years range in another Arabic language research by [5]. On the other hand, the shortest years range, related to the Arabic studies, is 1014 years[27]. However, the shortest year range for the Arabic language is more than all the remaining studies in other languages. Thus, the Arabic language is more persistent than the other languages [5].

Additionally, label type on all the covered researches is either a period or specific numbers such as year or month. Table 2.3 presents the label type, intervals length and years range. in Table 2.3 if the label type is period, the intervals length column clarifies the period's length along with each period name. At the same time, the years' range column shows the ranges of the year covered by certain research if the label type is a specific year. Most studies apply periods as label types with different intervals length. Dalli et al. [34] did not specify the ranges of the covered years in their study, thus instead of years range, we present the total covered years. The authors in [29], [40] applied two label types: Periods and year.

Table 2.3: The label type and interval length or years range for all the covered studies

Study	Label Type	Intervals Length	Years Range
Jong et al. [28]	periods	quarter year down to a granularity of 2-days	-
	day - month		
Dalli et al. [34]	year	-	9 years
	week number		
	year quarter		
		1-week - 1-month	
Kanhabua et al. [43]	periods	3-months - 6-months 12-months	-
Colgrove et al. [36]	periods	Augustan - Romantic Naturalist - Modernist	-

Garcia et al. [29]	year and decade	-	1801 - 1944
Wijaya et al. [40]	year and periods	20 years	1500 - 2008
Kumar et al. [35]	year	-	1798 - 2008
Tilahun et al. [16]	year	-	1089 - 1438
Mihalcea et al. [39]	periods	50 years	-
Popescu et al. [33]	periods	epoch1 (1800-1860)	-
		epoch 2 (1860-1900)	
		epoch 3 (1900-1950)	
		epoch 4 (1950-1975)	
		epoch 5 (1975-1999)	
		epoch 6 (1999-2009)	
Ciobanu et al. [46]	periods	16th century - 17th century	-
		18th century 19th century	
		20th century	
Stajner et al. [4]	periods	17th century - 18th century	-
		19th century - 20th century	
Niculae et al. [15]	periods	decade (10 years)	-
		century (100 years)	

Zampieri et al. [24]	periods	fine (6 years)	-
		medium (12 years)	
		coarse (20 years)	
Szymanski et al. [25]	periods	fine (6 years)	-
		medium (12 years)	
		coarse (20 years)	
		50 years	
Salaberri et al. [41]	periods	fine (6 years)	-
		medium (12 years)	
		coarse (20 years)	
Frermann et al. [42]	periods	20 year	-
Belinkov et al. [5]	periods	50 year - 100 year	-
Abbas et al. [27]	periods	Andalusian (750 - 1492)	-
		Abbasid (750 - 1258)	
		Islamic & Umayyad (625 - 750)	
		Pre-Islamic (478 - 624)	
		Modern (1798 - 2020)	
Orabi et al. [6]	periods	Andalusian (750 -1492)	-
		Abbasid (750 - 1258)	
		Islamic & Umayyad (625 - 750)	
		Pre-Islamic (478 - 624)	

Table 2.4: Number of features used by each study from each feature type and the total number of features per study.

Study	Stylistic Features	Lexical Features	Grammatical Features	Length Meta-Features	Other Features	Total
Dalli et al. [34]	0	0	0	0	1	1
Kanhabua et al. [43]	0	0	1	0	4	5
Colgrove et al. [36]	5	0	1	4	0	10
Garcia et al. [29]	0	2	0	0	1	3
Tilahun et al. [16]	2	0	0	0	0	2
Mihalcea et al. [39]	0	1	5	0	0	6
Ciobanu et al. [46]	2	0	0	0	3	5
Stajner et al. [4]	4	0	0	0	0	4
Niculae et al. [15]	0	2	0	0	0	2
Zampieri et al. [24]	2	1	1	3	0	7
Szymanski et al. [25]	0	1	2	0	0	3
Salaberri et al. [41]	0	1	0	0	3	4
Abbas et al. [27]	0	1	0	0	0	1

In terms of features, Table 2.4 shows the number of applied features from certain feature types and the total of the used features for each study. The most number of features used is by Colgrove et al. [36], they applied ten features. In contrast, only one feature utilized by the authors in [34], [27]. Moreover, in [24], the authors utilized all the different types of features. The most common feature type is the lexical features which are being applied by seven studies. Many studies did not explicitly mention the reasons of using lexical features, but it could be assumed that because of the ease of extracting it.

Table 2.5 presents the summary of the results achieved by the best models. Traditional machine learning techniques are used by most studies to address the issue. Three of the studies reported the best results using the SVM model [24], [25], [29]. While Naive Bayes outperforms other models in two studies. Fewer works have been done using statistical methods. Only one research utilized deep learning (CNN) to solve the problem [6].

Table 2.5: Survey summary in terms of the best models and their results.

Study	Best Model	Result
Jong et al. [28]	Comparison on Document Level	55% (accuracy)
Dalli et al. [34]	Unsupervised method of extracting periodicity information from text (year classifier)	87.78% (accuracy)
Kanhabua et al. [43]	Language model with Google Zeitgeist approach and four semantic features	0.71 (precision)
Colgrove et al. [36]	Maximum entropy classifier	0.80 (F1 measure)
Garcia et al. [29]	SVM model that predict decades as classes	42% (similarity score)
Wijaya et al. [40]	TOT and k-means	They identify three groups of semantic change using their model as follows: Change in meaning, getting additional meaning, and unchanged meaning
Kumar et al. [35]	Supervised language model	23 (median error)
Tilahun et al. [16]	Maximum prevalence	6.0 (MedAE)

Mihalcea et al. [39]	Word epoch disambiguation algorithm	61.55% (average accuracy)
Popescu et al. [33]	Algorithm for period detection using the statistical tests	69% (accuracy)
Ciobanu et al. [46]	Random forests	99.25% (accuracy)
Stajner et al. [4]	Naive Bayes and SMO	0.92 (F-measure)
Niculae et al. [15]	Combined method of text ranking and probability	93% (accuracy)
Zampieri et al. [24]	Ranking approach depend on pairwise comparisons	T2 = 0.87(DTE score)
Szymanski et al. [25]	SVM with all stylistic features and features extracted from GSN	T2 = 73.3% (accuracy)
Salaberri et al. [41]	Year entity detection approach	T2 = 0.62 (DTE score)
Frermann et al. [42]	SVM with SCAN system and n-gram feature	0.569 (precision)
Belinkov et al. [5]	Word-Embedding-based Neighbor Clustering (WENC)	They identify three main periods for Arabic language development
Abbas et al. [27]	Multinomial Naive Bayes	70.21%(accuracy) 0.68 (F1-Score)

Orabi et al. [6]	CNN without 25% of the most frequent terms	0.914 (F1-score)
------------------	---	------------------

Only two studies introduced the problem of classifying Arabic poems into literature eras. They focused on Arabic poems data that acquire an interest in ANLP applications due to its uniqueness and characteristics [51]. Classifying Arabic poems using deep learning showed promising results in [6]. Moreover, the authors in [27] apply n-gram tokenization as a feature; however, the best result is achieved without it. Thus, there is no clear view for using features in Arabic experiments. The Arabic language suffers from the lack of feature extraction tools which is caused by the difficulty of the language, especially in old text as discussed in [5].

Lastly, to the best of our knowledge, there is no study in the literature attempted to predict the era of the Arabic literature which are not poems. Few studies were conducted to address the problem in Arabic, but they focus only on a specific type of text: Poems. Therefore, this motivates us more to present our work to contribute to ANLP applications.

CHAPTER 3

DATASET

The available resources of historical Arabic text data are few compared to other languages, especially the English language [12]. Moreover, most Arabic corpora are for modern Arabic text that does not focus on historical Arabic literature [5]. However, there exist some datasets that offer historical Arabic literature, which is relevant for our work.

The remaining of this chapter is divided into the data collection section and data analysis section. The data collection section presents our datasets for the temporal classification task that we collected from diverse corpora. At the same time, in the data analysis section, we show an insight in terms of common words and unique words for our datasets.

3.1 Data Collection

We utilize two primary datasets for our experiments: open Islamicate texts initiative (OpenITI) corpus [44] and Arabic poem comprehensive dataset (APCD) [52]. OpenITI

is an enormous corpus that contains Arabic text from different authors and books through the years. At the same time, APCD consists of Arabic poem verses over the eras.

We split the data into training, validation, and test sets for both datasets. The training, validation, and test sets sizes are 73%, 12%, and 15% respectively. Moreover, to ensure that the model learns the stylometry of a specific period, we separate the authors in the three sets. Thus, if samples from an author in the training set exist, no samples for the same author in the validation or test sets exist. Therefore, we avoid learning the author’s writing style and focus on the time period style.

In order to prepare the datasets for the experiments we are using two text corpora: the OpenITI corpus and the APCD corpus. A simple pre-processing step will be applied while extracting samples from the text corpora. The pre-processing aims to clean the data from any extra content that is not related to the prepared dataset. For example, some texts contain extra spaces that must be removed.

3.1.1 OpenITI Dataset For Temporal Text Classification

Overview of the OpenITI corpus

The OpenITI corpus presented by Belinkov et al. [5] that is publicly available in [44]. OpenITI is a sizeable historical corpus of Arabic text and other languages. The size in terms of words is approximately 1.5 billion. The corpus was collected from different time periods and is freely available resources of religious and literary texts. Al-Maktaba Al-Shamela (known as Shamela) is the main source of these texts [5].

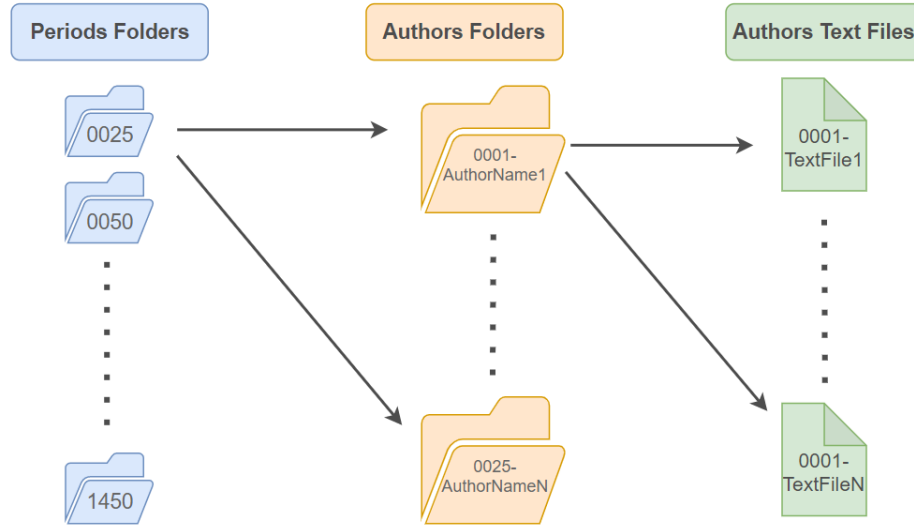


Figure 3.1: OpenITI corpus structure

The structure of OpenITI corpus is shown in Figure 3.1. From the figure we can see the corpus data are split into periods of 25 years that start from 1AH (Hijiri year) until 1450H. Each period contains a list of the author's folders labeled by the author's death year and name. In addition, each author folder includes text files that consist of the writer's literature works. For example, the author's folders labeled with 1450AH means that the author has not died yet. Moreover, the author's text files could be a book or *Diwan* that contains poems.

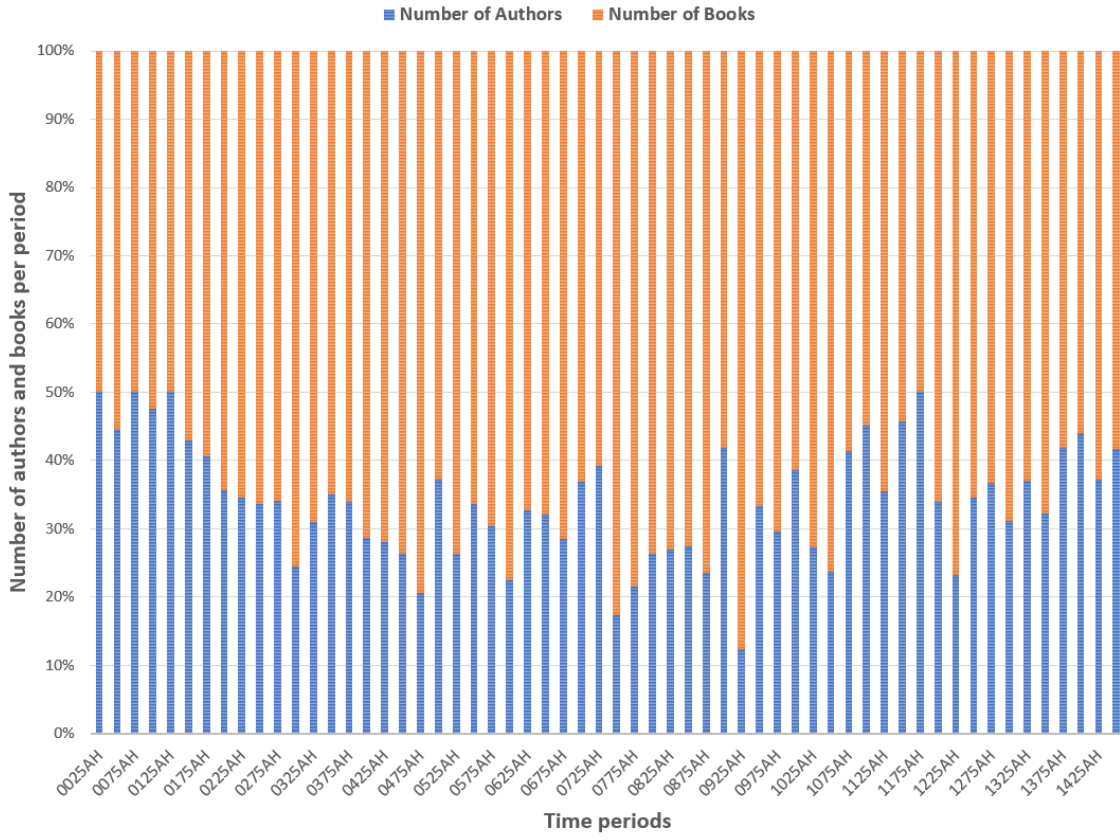


Figure 3.2: OpenITI corpus distribution in term of authors and books

Figure 3.2 illustrates the distribution of authors and books per period in the original OpenITI corpus. Each author has one book in a certain period if the number of authors and number of books are equal. For example, the first column of figure 3.2 which is 0025, presents the case of each author having one book, where 50% for both the number of authors and the number of books.

Temporal Periods Dataset Using OpenITI Corpus

Due to the large size of OpenITI corpus and the exceptionally long samples, we created subsets by extracting a maximum of 100 words long as a sample. Each author has one book or more, as described previously in the corpus structure. Thus, in the dataset creation process, we unify the number of samples from the author books. Thereby we ensure that our dataset generalized by covering various texts from the original corpus. Moreover, each book has a list of versions, we use the first version in that list.

Books in OpenITI corpus are considerably large. The average length of the book is about 176,718 words. We attempt to avoid the possibly worthless content like introduction and table of contents. Therefore, before extracting samples, we skip the first 1000 words. Moreover, we exclude the non-Arabic books, which are only two books.

We follow the Jurafsky et al. [53] recommendation for presenting the details of a dataset by providing a datasheet. The datasheet for our dataset is presented in Table 3.1.

As well, we will examine different setups in terms of classes ending up with two sets

Table 3.1: The datasheet for the OpenITI dataset.

Motivation	We created the OpenITI dataset for Arabic literature eras classification task.
Situation	We extracted the dataset from the OpenITI corpus. The data is from the Pre-Islamic time period until date, representing the literature through the time. Most of the data are from books.
Language variety	The language of the data is Arabic. The data span for long-spanning years, so the writing style could be different.
Writers demographics	The dataset includes writers from different time periods covering more than 1000 years. Thus, the characteristics of the authors are varied.
Collection and annotation process	The original OpenITI corpus consists of books written by different writers. We extracted a maximum of 100 words long as a sample. If a book has multiple versions, we extracted the first version. We extracted approximately the same number of samples from various books if an author has more than one book. Upon manual inspection, we found that the first pages of the books contain introductory pages. However, these pages are not part of the book. For example, it could be the table of contents or publisher notes. Based on that, we skip these early contents. Furthermore, the data was labeled by the death date of the author. Thus for the predefined setups, we extend the periods of each era ends. We also separate the authors in the training, validation, and test sets. There are no samples for the same author in the validation or test sets if an author is selected for the training set. Thus, each set has independent authors. Lastly, we selected random samples and manually inspected them for correctness.
Distribution	The dataset was created from an existing corpus. Accordingly, the copyrights are maintained.

of experiments. The first set of experiments we are dealing with classifying literature as well defined eras in the literature. In that situation, we try the following setups: the binary literature eras, the five literature eras, and three-eras as Belinkov et al. [5] proposed. Additionally, we also investigate some custom periods which are 300 years period, 200 years period, and 100 year period.

The five literature eras are as presented by Ghoniem et al. [3] that discussed the eras based on the previous studies. The binary literature eras are formed by combining the five literature eras into modern era and classical era. The classical era represents the remaining four eras after excluding the modern era. 300-years period lead to five classes, whereas 200-years period gives eight classes and lastly the 100-years period leads to 15 classes.

Table 3.2 shows the periods of the five literature eras setup. However, we ignore the Pre-Islamic era due to the lack of text files as compare to the other eras. The author folders are labeled by the death year of the author. Therefore, we extend the periods of each period ends. Despite excluding the Pre-Islamic era, we extend its period to ensure that the Islamic era are not affected. We extend the eras based on when an era ends and the division of the original corpus. The values for the extended years are between 18 and 50. The highest number is 50 years for the Pre-Islamic era because most of the existed authors in the text corpus belonged to it. The remaining values like 18 is to follow the corpus original division which is 25 years periods.

The values for the periods in Table 3.2 already extended. The original periods for the literature eras are presented previously in Chapter 1. Table 3.3 presents the three-eras and its periods which proposed by [5]. The table shows the original start

Table 3.2: The periods for the five literature eras setup in OpenITI dataset.

Era	Period
Islamic	50 AH – 175 AH
Abbasid	175 AH – 400 AH
<i>Aldoul wa al-amarat</i>	400 AH – 950 AH
Ottoman	950 AH – 1350 AH
Modern	1350 AH – 1450 AH

and end years for each era. The end year for the early era is either 200 AH or 300 AH. Thus we extend it to end at 300 AH. Moreover, the middle period ends around 1300 AH; we extend it to end at 1300 AH.

Table 3.4 presents the sample distribution of all the setups for the OpenITI dataset, besides the total size for each setup. For example, the binary literature eras setup is the largest in terms of the number of samples. In contrast, the 200-years custom periods are the smallest.

Table 3.5 shows the numbers of authors per era and samples per author for different literature eras setups. The ranges of a number of samples per author column in Table 3.5 presents the minimum and the maximum number of samples for all the authors in a certain era. As mentioned earlier we took samples from all the publishes for each author. The number of publishes is differ for each authors. Thus, the number of samples per author are varied, we only presented the minimum and maximum number of samples. Furthermore, the distribution of authors for the modern era in the training, validation, and test sets are the same for the five literature eras and the binary literature eras setups. Nonetheless, the number of samples per author differ on each setup. Because in the five literature eras case the modern era represent 20% of the total samples but in the binary case it is the half.

Table 3.3: The three periods for the Arabic language development proposed by [5] in the OpenITI dataset.

Era	Period
Early period	from 1 AH to 200/300 AH
Middle period	from 200/300 AH to around 1300 AH
Late period	from 1300 AH to modern days

Table 3.4: The number of samples for all the class setups in OpenITI dataset.

Setup	Training	Validation	Test	Totals
Binary literature eras	54,400	9,602	11,296	75,298
3-Eras as Belinkov et al. [5] proposed	24,100	4,242	4,988	33,330
Five literature eras	34,000	6,000	7,055	47,055
300-years custom period (5-class)	34,000	6,000	7,055	47,055
200-years custom period (8-class)	24,000	4,232	4,982	33,214
100-years custom period (15-class)	30,000	5,292	6,226	41,518

3.1.2 APCD Dataset For Temporal Poem Classification

Overview of the APCD corpus

The APCD corpus was collected by Waleed et al. [52] using two primary resources: Poetry Encyclopedia¹ and Diwan². Each sample in the dataset contains the following: one verse, the poet name, the meter, and the era. The dataset includes 3701 poets spanning 12 eras: Pre-Islamic, Mukhadramayn, Islamic, Umayyad, Mamluk, Abbasid, Ayyubid, Ottoman, Andalusian, the era between Umayyad and Abbasid, Fatimid, and the modern era. Furthermore, the corpus contains around 1.8M verses in total. Table 3.7 presents the distribution of verses and poets per era. The modern era has more verses among all the eras. On the other hand, the least number of verses are for the Islamic era.

Table 3.5: The number of authors and samples per author for the binary eras and the five literature eras setups in the OpenITI dataset.

Setup	Era	Number of Authors per Era			Number of Samples per Author		
		Training	Validation	Test	Training	Validation	Test
Binary literature eras	Classical	1255	220	258	1 to 614	1 to 559	2 to 636
	Modern	60	10	12	85 to 508	135 to 543	145 to 510
	Islamic	32	5	6	17 to 614	3 to 641	32 to 532
	Abbasid	304	53	62	4 to 38	1 to 61	3 to 25
Five literature eras	<i>Aldoul wa al-amarat</i>	657	116	136	3 to 94	1 to 36	2 to 88
	Ottoman	262	46	54	2 to 39	18 to 32	22 to 39
	Modern	60	10	12	85 to 114	121 to 126	59 to 120

Temporal Periods Dataset Using APCD Corpus

We combine all the verses for each author in one file. Then we extract the samples to compose a balanced dataset and assure that the authors are separated. Considering the verses for each author could be from different poems due to the lack of the poem name for each verse.

As a result of the large corpus size, we generate subset datasets. We were interesting in investigating the size of the sample and its effect on the classification performance. Therefore, we created one set where the sample only contains one verse and gradually increased it until 16 verses. Similarly to OpenITI we provide a datasheet for the APCD dataset in Table 3.6.

Furthermore, we investigate several setups: the binary literature eras, the five literature eras, and 12-eras. The binary literature eras setup contains the classical and the modern eras. The five literature eras setup consists of Pre-Islamic era, Islamic era, Abbasid era, Ottoman era, and modern era as the division presented by [3]. The five literature eras setup are similar to the OpenITI five literature eras that are mentioned in Section 3.1.1. But unlike OpenITI dataset, we ignore the *Aldoul wa al-amarat* era and include the Pre-Islamic era. The 12-eras setup contains all the eras in the original APCD corpus. The binary literature eras setup pursues the same approach in OpenITI dataset. Thus, we end up with three setups in terms of classes for the APCD datasets.

In the five literature eras setup, the Islamic era combined the Islamic and the

Table 3.6: The datasheet for the APCD dataset.

Motivation	We created the OpenITI dataset for Arabic poems eras classification task.
Situation	We extracted the dataset from the APCD corpus. The data is from the Pre-Islamic time period until date, representing the poems through the time.
Language variety	The language of the data is Arabic. The data span for long-spanning years, so the writing style could be different.
Poets demographics	The data are from different poets since our dataset covers long-spanning years. Thus, some poets are famous and well-known, and some are not.
Collection and annotation process	Since the original corpus provides a single verse for each sample, we group all the verses for each author. So we can separate the authors in the training, validation, and test sets. There are no samples for the same poet in the validation or test sets if a poet is selected for the training set. We created different sets of the APCD dataset regarding the number of verses per sample. We extract only one verse for the first set and gradually increase the number of verses until 16 verses. Furthermore, we ensure that we do not split one verse because it represents the poem’s essential structure and style. Lastly, we selected random samples and manually inspected them for correctness.
Distribution	The dataset was created from an existing corpus. Accordingly, the copyrights are maintained.

Table 3.7: The total number of verses and poets per era in the APCD corpus.

Era	Poets	Verses
Pre-Islamic	552	21907
Mukhadramayn	174	34005
Islamic	169	2942
Umayyad	248	63776
In Between	42	24077
Abbasid	392	234494
Mamluk	138	164780
Ayyubid	102	112350
Fatimid	480	124129
Andalusian	291	111011
Ottoman	182	159576
Modern	799	778723

Umayyad periods according to the literature eras division that we followed. Therefore, the Islamic era in the five literature eras setup is different from the Islamic era in the 12-eras setup.

For each setup, various verses per sample will be examined to observe the model performance. The minimum number of verses per sample is one, and the maximum is 16. The maximum number of verses per sample is limited to 16 due to the lack of samples for some eras. Therefore, we end up having 16 configurations per setup. Because of the few samples in the Islamic era in the original APCD corpus. We exclude the Islamic era from the datasets with more than four verses per sample in the 12 eras setup only. Subsequently, the 12 eras setup turns into 11-eras only.

Table 3.8 presents the sample distribution of all the APCD datasets. The second column in the Table 3.8 shows the ranges of the number of verses per sample for the each setup. For a particular setup, the sizes of the set vary to have balanced dataset. As the number of verses increases per sample, the size of the dataset will

Table 3.8: The samples distribution of all the class setups in APCD dataset.

Setup	Number of Verse per Sample	Training	Validation	Test	Total
Binary literature eras	1 - 8	24,000	4,242	4,988	33,330
	12 - 16	16,000	2,819	3,318	22,137
Five literature eras	1 - 8	15,000	2,645	3,110	20,755
	12 - 16	8,498	1,498	1,761	11,757
11-eras	5 - 10	18,700	3,300	3,882	25,882
	11 - 16	13,200	2,329	2,740	18,269
12-eras	1 - 2	20,400	3,600	4,235	28,235
	3 - 4	9,600	1,694	1,993	13,287

decrease. The number of authors per era and the number of samples per author in the 16 configurations for a particular setup are similar. Thus, we combine the numbers into ranges to present them concisely. The ranges start from the minimum possible number of authors or samples to the maximum number of them.

The ranges for the number of authors per era and number of samples per author for the training, validation, and test sets are presented in Table 3.9 for some setups. The modern era has more authors than the remaining eras. In contrast, the least number of authors was for the Ottoman era.

Table 3.9: The number of authors and number samples per author, for the binary eras and the five literature eras setups in the APCD dataset.

Setup	Era	Number of Authors			Number of Samples per Author		
		Training	Validation	Test	Training	Validation	Test
Binary literature eras	Classical	1081 - 1109	188 - 194	219 - 229	1 - 270	1 - 59	1 - 72
	Modern	552 - 577	9 - 102	113 - 119	1 - 48	1 - 49	1 - 51
	Pre-Islamic	385 - 400	66 - 70	79 - 82	1 - 35	1 - 35	1 - 35
Five literature eras	Islamic	289 - 303	49 - 52	57 - 62	1 - 267	1 - 73	1 - 108
	Abbasid	239 - 280	41 - 49	47 - 57	1 - 40	1 - 33	1 - 36
	Ottoman	125 - 132	22 - 23	25 - 27	1 - 53	1 - 54	1 - 54
	Modern	441 - 568	77 - 101	87 - 118	1 - 19	1 - 15	1 - 16

Table 3.10: The number of the unique words and vocabulary size for each era in OpenITI and APCD datasets.

Era	OpenITI		APCD	
	Unique Words	Vocabulary Size	Unique Words	Vocabulary Size
Pre-Islamic	-	-	15,561	44,602
Islamic	40,966	99,496	26,912	66,528
Abbasid	26,420	83,735	23,017	64,757
<i>Aldoul wa al-amarat</i>	29,905	97,522	-	-
Ottoman	27,778	91,741	24,496	65,704
Modern	26,271	83,015	26,007	67,580

3.2 Data Analysis

Our OpenITI and APCD datasets have various setups as explained in Section 3.1.

However, for the data analysis, we will focus on the five literature eras setup only.

The selected setup represents the actual literature eras through time from a linguistic perspective. Thus, we will consider the five literature eras as the standard setup in our work. Lastly, all the analysis will be done using the training set for the standard setup.

3.2.1 Unique Words

The unique words per era are words that occur only in a particular era. Table 3.10 presents the number of unique words and vocabulary sizes for each era. Islamic era in both datasets has the highest number of unique words. Whereas the Pre-Islamic era in APCD dataset has the least number of unique words.

From the Table 3.10, the Islamic era vocabulary size is 99,496 words in the OpenITI

dataset. Out of the 99,496 words, 40,966 words are unique only for the Islamic era. As a result, the vocabulary size for the entire OpenITI dataset is 248,872, and 151,340 is the total of unique words for each era. So 60% of OpenITI vocabulary is unique for specific eras. The vocabulary size of the whole APCD dataset is about 180,924 words, and the total of the unique words per era is 115,993. That means about 63% of the vocabulary are unique words specific to different eras. Previously percentages show that unique words per era are more than half of the vocabulary size, indicating each era has its own specific words.

Furthermore, Table 3.11 shows selected unique words per era. The selected unique words are from the top 100 frequent unique words for each era. In the first column we list the datasets from which the unique word was extracted. An example using that word is shown the last column.

We attempt to search how some words are related to that specific era. Abbasid poets are concerned about grey hair, as mentioned earlier in Chapter 1. From the Table 3.11 the word **للمشيب** is a unique word in Abbasid era which is composed of a word **المشيب** and a preposition **حرف اللام**. The root of the word is **شيب** which means *grey hair*. The Abbasid poets show great concern for the grey hair as was claimed by Kamal et al. [7]. Moreover, names are common in most eras for both datasets. **معاوي**, **مخنف** and **جهاركس** are names that were related to each era that appeared in. Each name mentioned previously has an incident or story behind it. For instance, **جهاركس** is a Mamluki prince who was famous and belong to the *Aldoul wa al-amarat* era.

Table 3.11: Example of the unique words per era for OpenITI and APCD datasets.

Dataset	Era	Word	Description	Example
APCD	Pre-Islamic	نهدة	المرتفعة الخلق	ولو شئت نجتني من الخيل نهدة
			High morals	ترى خلفها الحو الجياد تواليا
		العدي	جماعة القوم يعدون راجلين للقتال People are walking on foot to fight	لها وفضة فيها ثلاثون سيحفاً إذا آنست أولى العدي اقشعرت
	Islamic	معاوي	معاوية بن ابي سفيان Muawiyah bin Abi-Sufyan (a person's name)	أبوك وعمي يا معاوي أورثا تراثاً فيحتار التراث أقاربه
		مخنف	عبدالرحمن بن مخنف Abdul Rahman bin Mukanf (a person's name)	فيا عين بكى مخنفا وابن مخنف وفرسان قومي قصرة وأقاربي
	Abbasid	للمشيب	اللام حرف جر المشيب هو الكبير في السن وتغير لون الشعر الى الابيض the first letter is preposition the word without the preposition means old	وآباء الصباية أم بنوها يروضون الشيبة للمشيب

الديوك	جمع ديك plural of rooster	حتى بديك كنت ألف قربه حلو الشمائل في الديوك رشيق
Ottoman	جمع دوحة وهي الشجرة العظيمة كثيرة الفروع plural of 'Doha' which is the great tree has many branches	تظلمها الادواح كالاعلام تزورها الأرواح للسلام
يوسف	النبي يوسف عليه السلام Prophet Yusuf, peace be upon him	وعلمت ما فعل الزمان بيوسف بالسجن عاش بذلة وقود
العدالة	Justice	القي اليه مقاليد العدالة من قد ساس بالعدل لبناناً واهليه
مدينتي	my city	رفقا بدمع الناس قبر مدينتي رفقا بدمع الناس ما بين أحياء المقابر
هوجاء	ريحٌ مختلطة الهبوب Mixed wind blowing	أربت عليها كل هوجاء رادة زجولٍ بجولان الحصى حين تسحق
Islamic		

مساميح	مساميح جمع مساح من سمح	مساميح بالمعروف وسط رحالنا وشبائنا بالفحش أخل باخل
	إذا جاد وأعطى عن كرم وسخاء	
	Serious and giving for	
	generosity and generosity	

هرکند	اسم بحر	وأما بحر هرکند فله ريح غير
Abbasid	sea name	هذه ما بين المغرب الى بنات نعش فيغلي لها البحر كغليان القدور

نولك	يَنْبَغِي لَكَ	يقال ما نولك أن تفعل ذاك
	you should	أي لا ينبغي لك أن تناله من نال ينال

Aldoul wa al-amarat	جهازكس الخليلي - أمير مملوكي name of a Mamluk prince	فاتنق جهازكس وزين الدين قراجا على مفارقة ديار مصر فسارا عنها وتبعهما سراسنقر
الفلاكة	الفلاكة هو اسم فارسي يعني الفقر والفقراء Persian name meaning poverty and poor	أن الفلاكة والإهمال ألصق بأهل العلم والزم لهم من غيرهم

Ottoman	الخطابات	جمع خطاب ولها معاني كثيرة منها الكلام والرسالة plural of speech and it has many meanings including speech and message	وإنما تكشف عنها عند العالم بالوضع على سبيل البداة والضرورة ففيه أن الخطابات اللفظية لم تعتبر أدلة على الخطابات النفسية
	السببية	العلاقة بين السبب والمسبب The relationship between cause and effect	وحرف عن يدل على السببية كقوله أدوا عمن تمونون
Modern	عشائر	جمع عشيرة Plural of clan	فذكرت عشائر تهامة بأسماء بلادها مثل بيش
	وهو عن	حرف الواو للوصل والضمير هو وعن حرف الجر which is about	يروي أيضا عن عبد الرحمن ابن عمر القباي وهو عن أبيه عمر بن عبد الرحمن القباي المذكور والله أعلم

3.2.2 Common Words

The words that appear in all the eras are considered as common words. Therefore, we count the frequency of the word that appears in both datasets. The top three common words and the frequencies of the top 10 common words are illustrated below.

Table 3.12 illustrates the top three common words for all the eras in OpenITI dataset. Each column represents a certain era with the rank number of the common words according to its frequency. Islamic, *Aldoul wa al-Emarat*, Ottoman and modern eras have the exact top three common words but different frequencies. The top one word is the same in Abbasid and *Aldoul wa al-Emarat* eras. Abbasid era is the only era that the words *في* and *من* are not from its top three words as the other eras.

Table 3.13 presents the top three common words in APCD dataset. The common word's rank for each era is shown in the table in terms of the occurrences in the dataset. In contrast to the OpenITI dataset, the top three common words for the

Table 3.12: The Top three common words in OpenITI dataset.

Word	Islamic	Abbasid	<i>Aldoul wa al-amarat</i>	Ottoman	Modern
من	1	5	3	2	2
في	2	6	2	1	1
الله	3	2	4	4	4
بن	9	1	1	5	3
قال	5	3	7	13	18
على	4	11	6	3	5

Table 3.13: The Top three common words in APCD dataset

Word	Pre-Islamic	Islamic	Abbasid	Ottoman	Modern
من	1	1	1	1	1
في	2	2	2	2	2
إذا	3	4	5	17	17
ما	4	3	3	3	3

APCD are very similar. Islamic, Abbasid, Ottoman, and Modern eras are identical in the top three common words. Whereas the Pre-Islamic era has the same top two common words with remaining eras, the third one is different.

The top three common words for both datasets have mutual words which are: من and في. These words are first or second rank for most eras. The mutual words represent two prepositions from the Arabic language. The Arabic language contains 20 prepositions, highlighting that two of them are considered the top common in two different datasets.

In addition to the top three common words, Figure 3.3 shows the frequency of the top 10 common words in each era for OpenITI dataset. The top 10 common words in the Abbasid era are more frequent than the remaining eras. Figure 3.4 presents the frequency of the ten most common words in each era for APCD. However, unlike OpenITI, the Ottoman era had the maximum frequencies of the most common words.

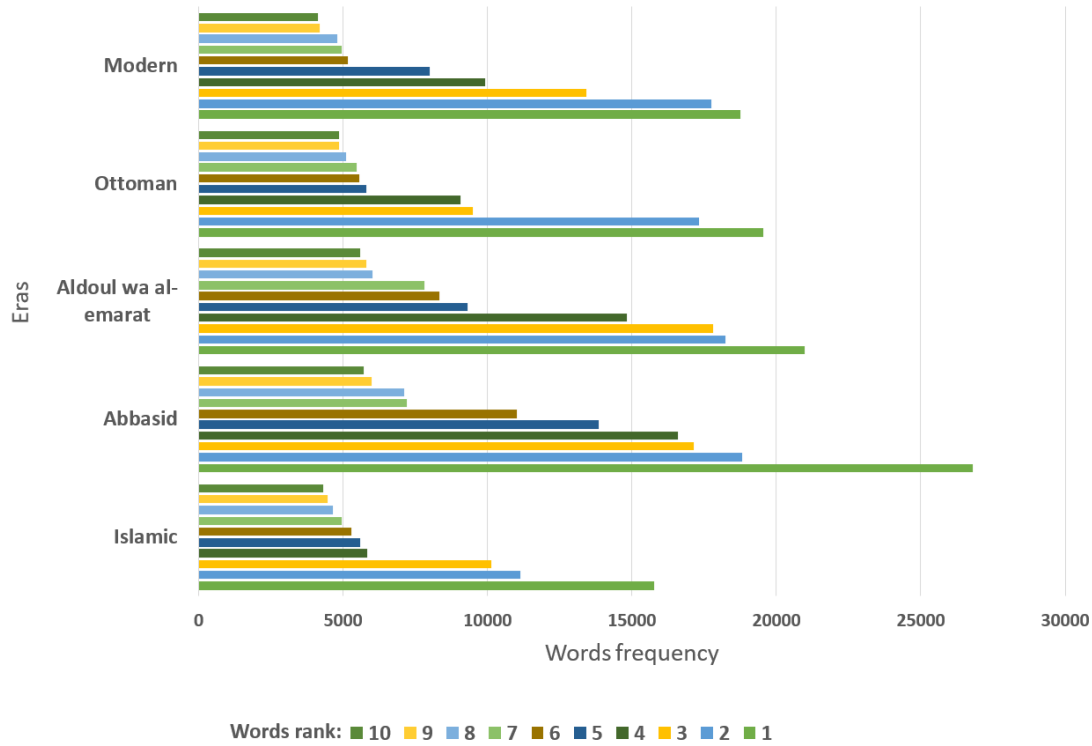


Figure 3.3: Frequency of the top 10 common words in OpenITI dataset

The top 10 common words occurrences in the Pre-Islamic era are remarkably less than the other eras in APCD. Which may refer to the unique characteristics of famous poems in that era known as Mu'allaqat (المعلقات) [54].

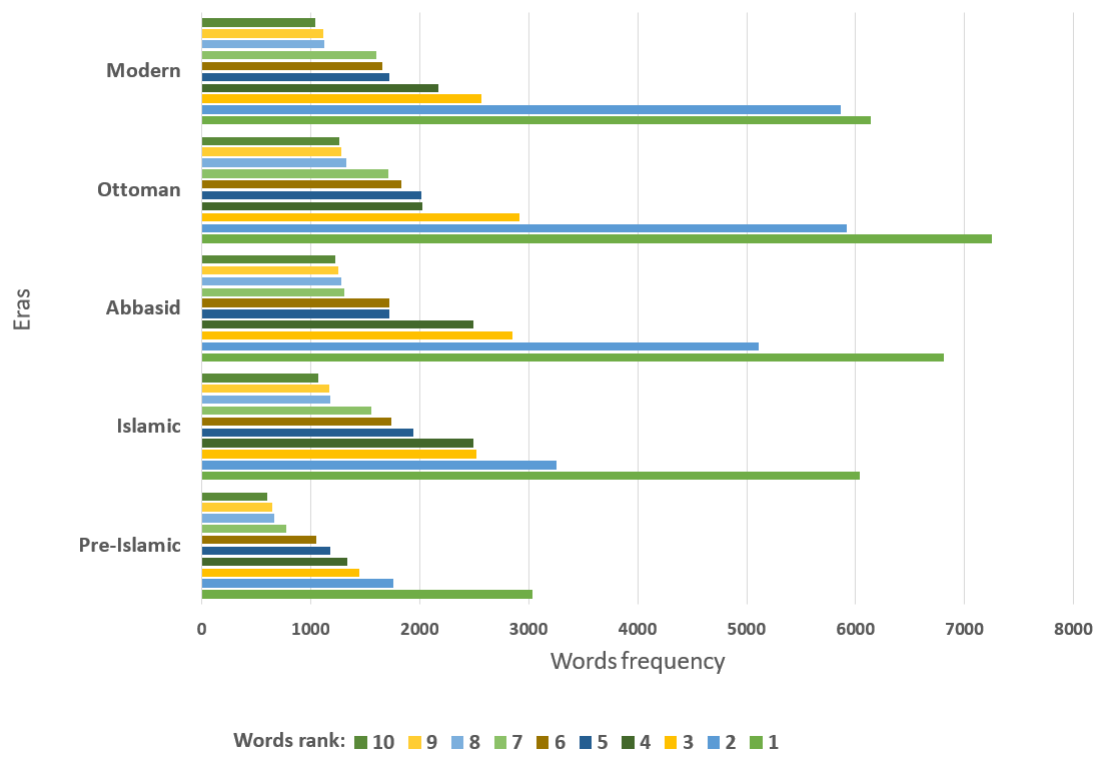


Figure 3.4: Frequency of the top 10 common words in APCD dataset

CHAPTER 4

METHODOLOGY

This chapter describe our methodology for classifying Arabic text based on temporal periods using neural network including deep learning approaches and traditional machine learning models. We apply several neural networks types. Different approaches with various architectures will be examined to achieve better results. Moreover, data will be pre-processed before feeding it to the model.

We cover in this chapter the following sections: dataset preparation, model design, and evaluation metrics. First, the Dataset section illustrates the process of cleaning data and preparing it as input for our models. Then, in the model design section, we focus on two types of neural networks in detail, the artificial neural networks (ANN) and the recurrent neural networks (RNNs). Lastly, different evaluation metrics will be discussed in the evaluation metrics section.

4.1 Dataset Preparation

The data collection was explained in detail in Chapter 3. In this section, we illustrate the process of preparing our dataset in the following subsections.

4.1.1 Data Pre-Processing

Preparing the data is essential to ensure that the model learning process is done correctly. Typically, data have extra information which is not related to the context and can lead to over-fitting. Therefore, we remove some of it and reduce the others, like in lemmatization step. The pre-processing steps employed are explained below:

1. **Data cleaning:** The datasets contain non-Arabic digits, hypertext markup language (HTML) tags, symbols, and punctuation marks. RegEx matching was employed to remove all the extra data not related to our problem. Therefore, we removed symbols, punctuation marks, non-Arabic digits, and HTML tags. For example, we will remove the symbols like ”|” and ”#”.
2. **Normalization:** The different forms are treated as separate words, which increases the vocabulary size. We avoid overfitting by reducing and converting them into a more unified sequence as follows:

- Diacritics like ”[ؔ]” were removed.
- *Kashida* ”.” characters were removed. *Kashida* is used for aesthetic reasons.

It is used to make the words look longer without affecting the meaning.

For example, using *Kashida* in the word ”الرسول” can render the word as

”الرسول”.

Table 4.1 presents an example before and after normalization. We can see that most of the words have diacritics in the original sample. Some words have one diacritic only and and the some others have more than one.

3. **Remove stop words:** There is a group of commonly used words like prepositions in any language. For example, the word **في** is an Arabic preposition used for place and other purposes. Moreover, stop words are shared and used in most samples, which means they appear in all the eras. These words may not be useful for our problem and could increase the complexity of the model and lead to overfitting. Therefore, we remove Arabic stop words using the list of Arabic stop words by the Natural Language Toolkit (NLTK) ¹.

Table 4.1 presents an example before and after removing the stop words. The first row shows the original sample without applying pre-processing techniques. The second row shows the sample after removing the stop words. The two words **فيه** and **في** have been removed.

4. **Lemmatization:** Is based on the morphological analysis of words. It aims to return the base form of a word, which is known as the lemma. Thus, converting words into lemmas decreases the vocabulary size and could avoid the overfitting problem. Darwish et al. [55] proposed Farasa, which is a well-known tool for Arabic text processing, Belinkov et al. [5] applied it in their study. We utilize it to apply lemmatization to our data. The tool is freely available in this link ².

Table 4.1: Example of pre-processed sample using remove stop words and lemmatization.

The Applied Pre-processing	Example
The original sample	تلوذُ به الأَكابرُ في صغارٍ وترجو فيه مَقبولَ السَّؤالِ
Normalization	تلوذ به الأَكابر في صغار وترجو فيه مقبول السؤال
Remove stop words	تلوذ الأَكابر صغار وترجو مقبول السؤال
Lemmatization	تلوذ ب أكبر في صغير رجا في مقبول سؤال

Table 4.1 presents an example of a sample before and after lemmatization. The thrid row shows the sample after applying the lemmatization. The word السؤال (the question) and it converted to its lemma which is سؤال (question).

Lemmatization will not be applied in all the experiments. However, we will conduct some experiments with and without lemmatization to observe its effect.

4.1.2 Tokenization

The input form for the models is numerical representation. Thus, we must tokenize the text data. We will apply two features representation bags of words (BOW) and term frequency-inverse document frequency (TF-IDF) for the ANN model. In the RNN model, we will utilize two schemes, word as token and characters as tokens. Finally, we will present the tokenization process in detail for each model.

4.2 Temporal Text Classification Model Design

Our proposed problem consider as a text classification problem as discussed earlier.

To classify text into time periods, we focus on neural network including deep learning

approaches. Neural network showed promising results in ANLP applications in the literature. Moreover, automating feature engineering is an advantage of using neural network models. Feature engineering is one of the most critical steps in machine learning [8]. Therefore, neural network systems are beneficial for our temporal classification task due to the complexity of the Arabic language.

Furthermore, as mentioned in Chapter 3, we split our data into three sets for both datasets. Since our datasets are significant, we will follow the hold-out mechanism to train and test our system, where we will use the training and validation sets to train and validate the model and test the model using the test set.

In our experiments, we will focus on two types of neural networks models: ANN and RNN. Also, we will use different properties for each model. Besides the ANN and RNN models, we will apply other models as secondary experiments like convolutional neural network (CNN) and logistic regression.

4.2.1 ANN Base System For The Temporal Classification

Neural Networks

The main scenario of the neural networks learning process and the data representation are shown below. To train a neural network system, four components are required to complete the process as follows:

- **Layers:** Which used to build the neural network model. Figure 4.1 illustrates the elements of a neural networks architecture. The hidden layer could be one layer or more.

The layer consists of individual units called neurons. A neuron used to transfer the data through the network. It receives an input and process it to produces an output. The input processing consist of two steps the linear transformation and applying the activation function. Figure 4.2 presents the structure of the neuron showing its elements in details. A constant bias input used to allows shifting the activation function. The formula for one neuron is

$$f(b + \sum_{i=1}^n x_i w_i). \quad (4.1)$$

Several options available for the activation function according to the usage purpose. *Sigmodi*, rectified linear unit (ReLU), and softmax are some of the activation function. For example, *sigmoid* activation function work better in the classification problems. Moreover, *ReLU* used only in the hidden layers. The formulas for the previous activation functions are:

$$\text{Sigmoid}(\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{x})} \quad (4.2)$$

$$\text{ReLU}(\mathbf{x}) = \max(\mathbf{x}, 0) \quad (4.3)$$

$$\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \quad (4.4)$$

We will apply *sigmoid* activation in the output layer for all the binary models. Softmac will be used in the output layer in the mlti-class models. Lastly, ReLU activation will be utilized in the hidden layers in all the models.

- **The input and output data:** There are many options of input and output for the model. The most common example is mapping a one input to a one output which is linear stack of layers. Moreover, a hybrid model is an example, it can take more than one input and output could be more than one.
- **The loss function:** That provides the feedback signal to the optimizer. The loss function is an essential component that calculates the gradients to be used to update the weights. Various loss functions used according to the required objectives. The two common loss functions are mean squared error (MSE) and cross entropy. MSE is normally used for the regression problems. Cross entropy has different types of functions according to the problem type in terms of classes number. Moreover, it mainly used with the classification tasks. Binary cross entropy and sparse multi-class cross entropy are the popular loss functions, which we will apply in our experiments.
- **The optimizer:** That defines how to proceed with the learning process. Optimizer decides how to change the weights to reduce the losses. Various Optimization algorithms available and each algorithm having its advantages and disadvantages. Some examples of the optimizer are gradient descent (GD), stochastic gradient descent (SGD), Root mean squared propagation (RMSProp), and adaptive moment estimation (Adam). Gradient descent is the basic optimization algorithm that rely on first-order derivative of a loss function. Adam calculates the adaptive learning rates for each parameter [56]. Adam is the used optimizer in the recently studies [6]. Moreover, RMSProp and Adam considered

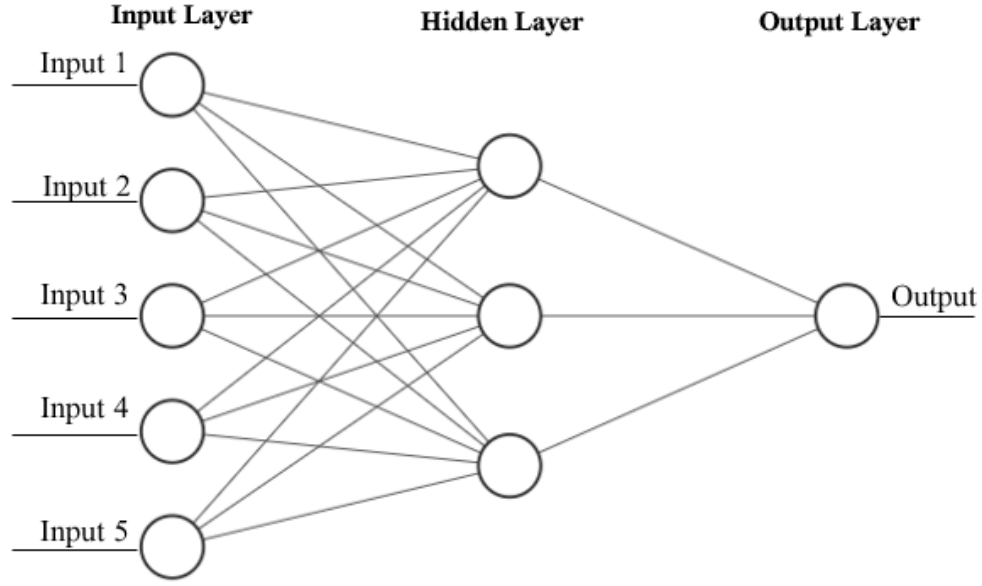


Figure 4.1: The elements of a neural networks architecture

better than the traditional optimizer according to Chollet et al. [8]. Therefore, we will utilize them in our experiments.

Figure 4.3 illustrates the visualization of the interactions between the four components. In Figure 4.3, the network consists of linked layers, the input data is mapped to the output targets. Then, the loss function produces the value of loss by comparing the output predictions to the targets [8]. The loss score is used to measure the performance of the model prediction to the expected targets. Lastly, the optimizer employs the loss value to update the weights of the model.

Data Representations

Different data representations are used with the neural networks. The tensor is the basic structure of the data representation in neural networks [8]. Tensor is the con-

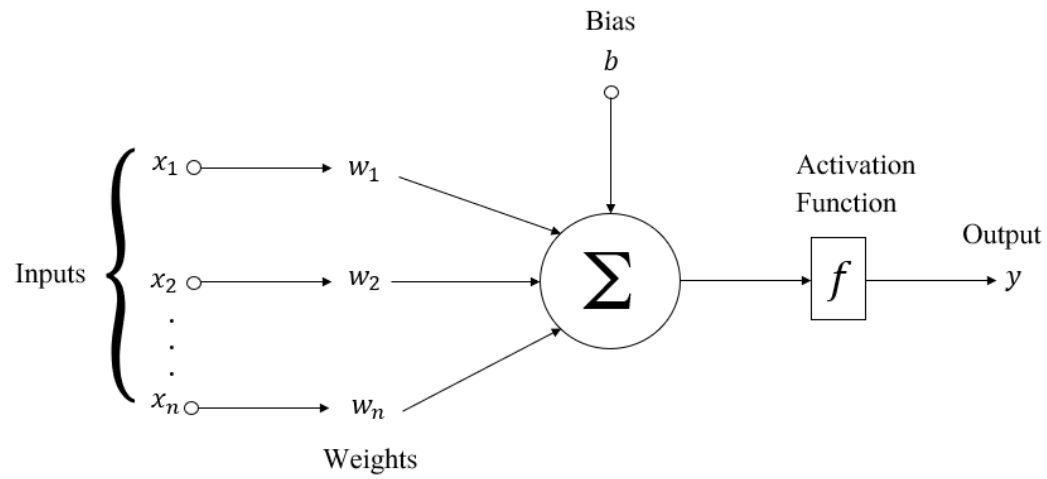


Figure 4.2: The structure of the neural networks neuron

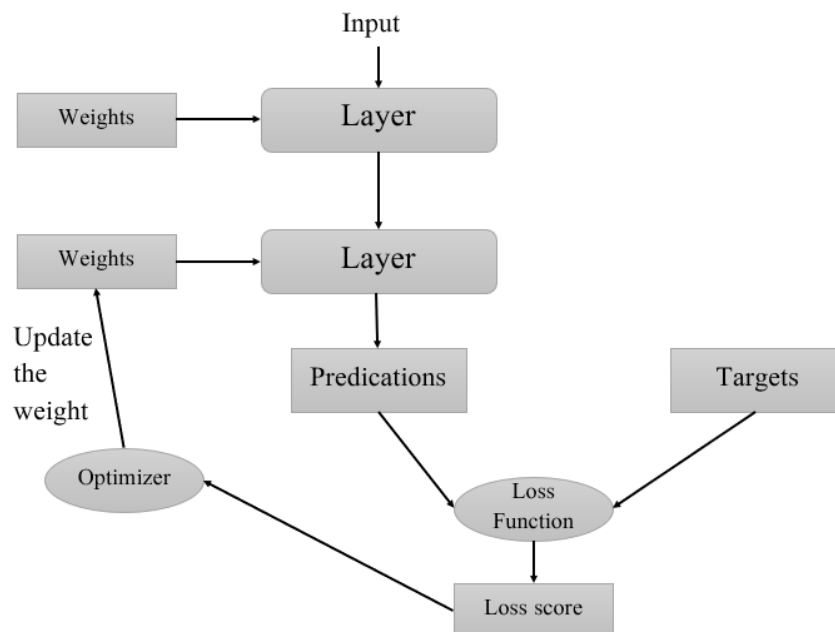


Figure 4.3: Overview of the relation between the network, layers, loss function, and optimizer (adapted from [8])

tainer for the data, which is usually numerical data. Vector data(2D tensors) and sequence data(3D tensors) will be utilized for our models. Vector data are of the shape (samples, features), whereas sequence data are of the shape (samples, time-steps, features). Sequence data are composed of vector data. Each sample is encoded as a series of vectors. Then, a group of data will be encoded as a 3D tensor which is the sequence data. Lastly, sequence data is a set of tokens that are either word, character, or sub-word.

ANN Methodology

We will investigate the network of fully connected layers. It is the base model for the temporal classification task. Figure 4.6 illustrates the methodology for the ANN model to classify Arabic text into time period. First, we prepare the input data, the pre-processing discussed previously in Section 4.1. For the tokenization step, we will utilize the following features representation:

1. **Bag-of-Words (BoW):** Is converting a set of texts into vectors whose dimensions are equal to the vocabulary size of the dataset. Each vector contains token counts ending up with a sparse representation of that counts. The disadvantage of this technique is its inability to identify the more important and less important words for identifying specific class.

We show a representation for a simple dataset of two samples only in Table 4.2. The samples in the table are verses from the poem dataset. Figure 4.4 shows two samples converted into count frequency using word occurrence. Sample-2 does not contain the word "أُو"; hence its frequency is calculated as zero. The

same word appears twice in sample-1; hence its frequency is equal to two.

2. **Term frequency-inverse document frequency (TF-IDF):** Is similar to word occurrence except that it considers overall samples of words' weight instead of counts only. The weight is calculated using the TF-IDF technique, considering how important a word is to a sample in the dataset. TF-IDF is better than the word occurrence because it provides the importance of the words rather than only focusing on the frequency. Thus, we can remove the less important words to make the model less complex by reducing the vocabulary size.

The TF IDF formula [57] for a word i in sample j as following:

$$w_{i,j} = tf_{i,j} \times \log \frac{N}{df_i} \quad (4.5)$$

Where $tf_{i,j}$ is the number of occurrence of word i in sample j , df_i is the number of samples containing word i , and N is the total number of samples.

We apply Tf-IDF for the two samples in Table 4.2. Figure 4.5 illustrates the dataset samples after applying TF-IDF technique.

We can see from the table that the weights calculated depend in We can see from the table that the weights for the words are more complex than the word occurrence. However, in this example, the dataset size is small that does not reflect the real use of the TF-IDF technique.

The ANN model architecture contains different layers. We explain the layers of the ANN model below:

Table 4.2: Two samples from the poem dataset to show the representation using word occurrence and TF-IDF.

Sample ID	Sample																	
Sample-1	يمثل قوما أو يصور حالة فيبعث حزنا أو يضيء سرورا																	
Sample-2	فجاءوا كموج البحر والبحر مزبد إلى غاية ما خلفها قط ملمح																	
	يمثل يضيء يصور والبحر ملمح مزبد ما كموج قوما قط فيبعث فجاءوا غاية سرورا خلفها حزنا حالة البحر إلى أو																	
Sample-1	0	1	1	0	0	1	0	1	1	0	1	0	1	1	1	1	0	0
Sample-2	2	0	0	1	1	0	1	0	0	1	0	1	0	0	0	0	1	1

Figure 4.4: The words occurrence for the two samples in Table 4.2

- **Input layer:** Take the input and transfer it to the next layer.
- **Dense layer:** All the neurons in the are connected to all the neurons in the next layer, as presented previously in Figure 4.1. The computation for the dense layer cell is as in the Equation 4.1. Since the dense layer in the ANN architecture are consider as hidden layer it use the ReLU activation function.
- **Dropout layer:** We utilize it to avoid overfitting. A dropout is a form of regularization that randomly drops some cells. Dropping a cell turns its value in the corresponding activation function into zero. Since the dropped cell dose did not affect the activation function, the gradients turned to zero. Figure 4.7 presents ANN structure before and after applying the dropout [9]. The number of cells decreased in the right figure when the dropout applied.
- **Output layer:** Which give the predictions as results. The activation function for the output layer depends on the number of classes. If we have binary problem we use *sigmoid* activation function, otherwise we use *softmax* activation function.

For the ANN model we will try different number of layers. ANN could have one or more of the combination of fully connected layer followed by a dropout layer. The previous combination are represented as yellow box in Figure 4.6.

	أو	إلى	البحر	حلبة	حزنا	خلفها	سرورا	غلبة	فجاءوا	قيمت	قط	قوما	كسوح	ما	مزبد	ملج	والبحر	بصور	بشيء	بطل
Sample- 1	0.00000	1.405465	1.405465	0.000000	0.000000	1.405465	0.000000	1.405465	0.000000	0.000000	1.405465	0.000000	1.405465	1.405465	1.405465	1.405465	1.405465	0.000000	0.000000	0.000000
Sample- 2	2.81093	0.000000	0.000000	1.405465	1.405465	0.000000	1.405465	0.000000	0.000000	1.405465	0.000000	1.405465	0.000000	0.000000	0.000000	0.000000	0.000000	1.405465	1.405465	1.405465

Figure 4.5: The TF-IDF for the two samples in Table 4.2

4.2.2 RNNs

RNNs are a type of ANN that utilizes sequence data. RNNs has a sense of time context because it has a recurring connection between its neurons. Thus, RNNs have an internal state that is updated as a sequence is processed. Figure 4.8 presents the elements of RNNs architecture in general. RNNs architecture is similar to the ANN, but the addition is the recurring connection in the hidden layer.

The basic formula for the RNNs is

$$h^{(t)} = f_w(h^{(t-1)}, x^{(t)}) \quad (4.6)$$

Where $h^{(t)}$ the current hidden state $h(t)$, $h^{(t-1)}$ is the previous hidden state, $x^{(t)}$ the current input, and f_w is some function with parameter w . The network learns to use $h^{(t)}$ as a kind of summary from the past sequence of inputs up to t . Figure 4.11 shows a simple RNN without output where the black square means that an interaction done with a delay of 1 time step.

RNNs are commonly used in NLP tasks because of their ability to process long sequences of data. RNN has various process sequences. We use many to one process sequence where the input is a sequence of tokens, and the output is a class. Vanilla RNN is the first model of RNNs proposed by Pineda et al. [58]. However, vanilla RNN's main problem is vanishing gradient [59]. Thus, other types of RNNs will be employed to build our model as follows:

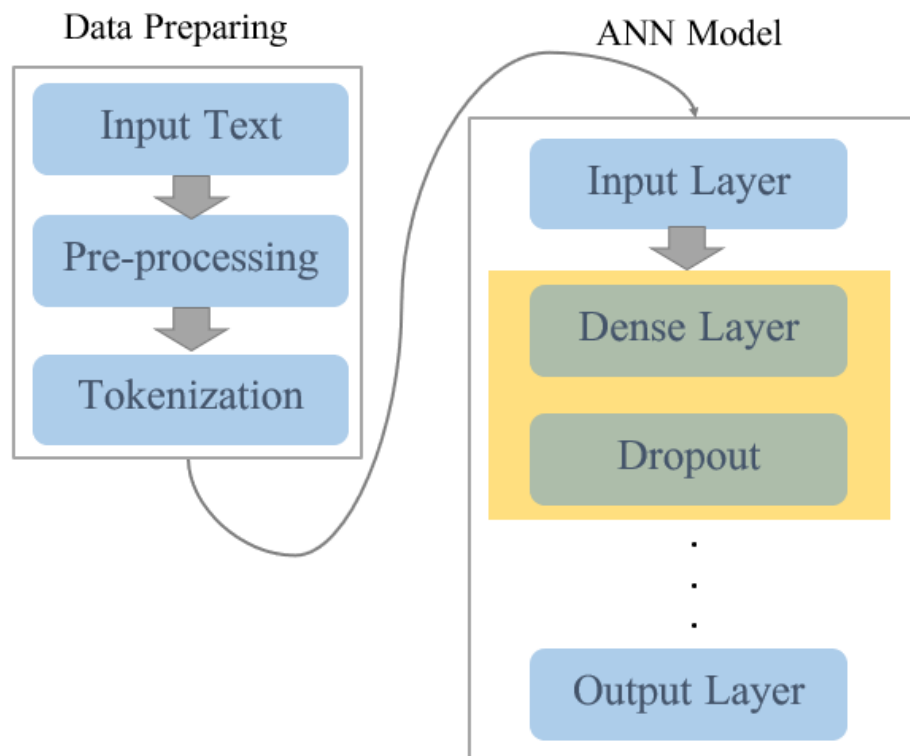


Figure 4.6: The ANN methodology structure.

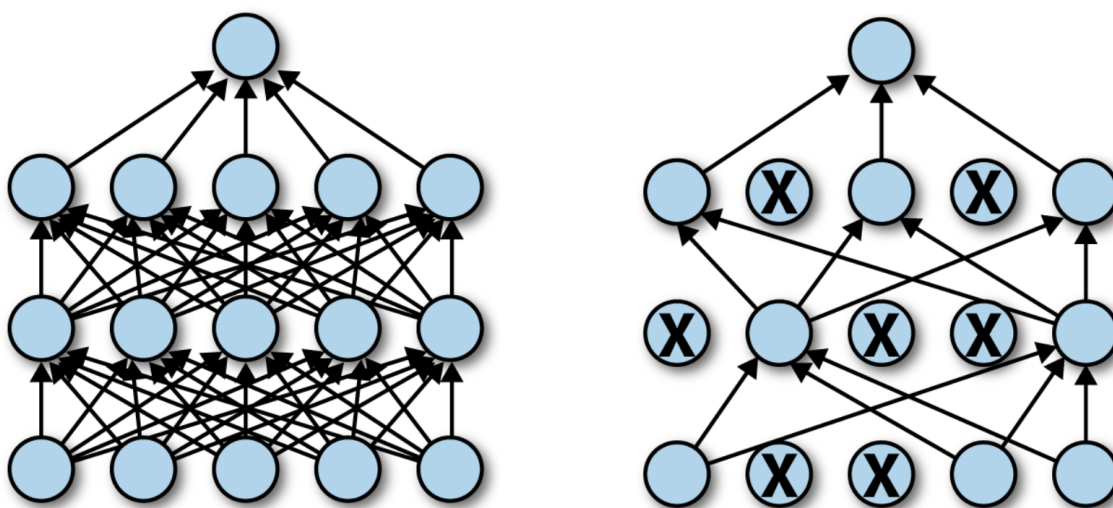


Figure 4.7: ANN structure before and after using the dropout [9]

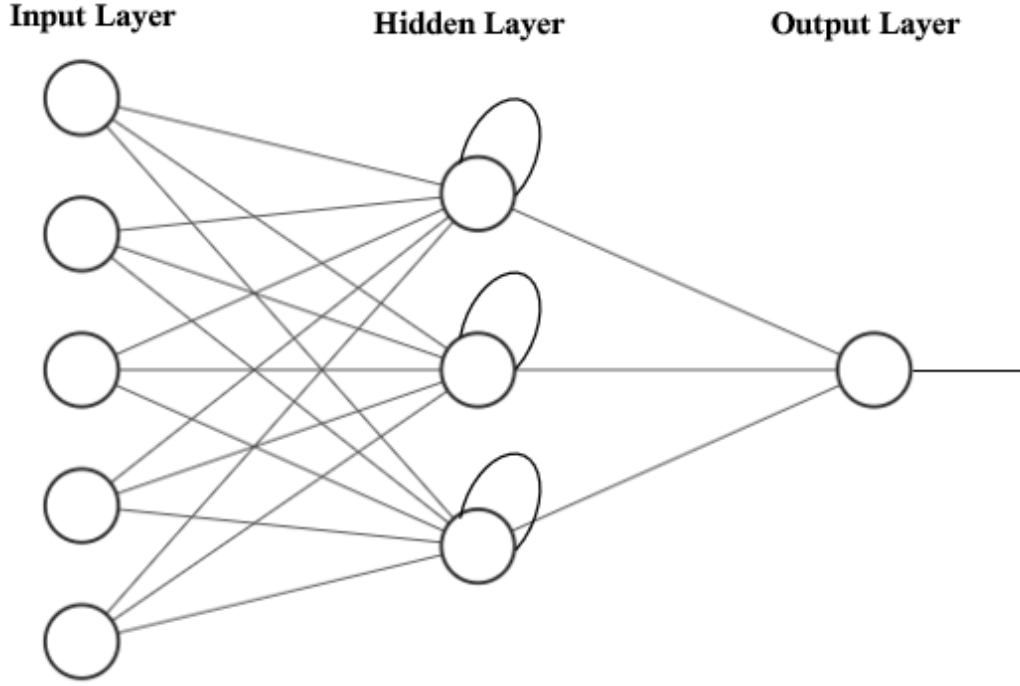


Figure 4.8: The elements of RNNs architecture

- **Long short-term memory (LSTM):** is the solution for vanishing gradient problem in vanilla RNNs which was proposed by Hochreiter et al. [60]. LSTM work better with the tasks that require memories of events that occurred in the earlier time steps. Moreover, LSTM cell has three gates as follows: input gate, forget gate, and output gate. Figure 4.9 illustrates the LSTM cell including the three gates. The gates in the figure as follows: $i_{(t)}$ the input, $f_{(t)}$ the forget gate, and $o_{(t)}$ the output gate. The gate $g_{(t)}$ controlled by the input gate to decide which parts of it should be added to the long-term state ($C_{(t)}$) [10]. Below we list the equations for the LSTM cell states:

$$i_{(t)} = \sigma(W_{xi}^T \cdot x_{(t)} + W_{hi}^T \cdot h_{(t-1)} + b_i) \quad (4.7)$$

$$f_{(t)} = \sigma(W_{xf}^T \cdot x_{(t)} + W_{hf}^T \cdot h_{(t-1)} + b_f) \quad (4.8)$$

$$o_{(t)} = \sigma(W_{xo}^T \cdot x_{(t)} + W_{ho}^T \cdot h_{(t-1)} + b_o) \quad (4.9)$$

$$g_{(t)} = \tanh(W_{xg}^T \cdot x_{(t)} + W_{hg}^T \cdot h_{(t-1)} + b_g) \quad (4.10)$$

$$C_{(t)} = f_{(t)} \oplus C_{(t-1)} + i_{(t)} \oplus g_{(t)} \quad (4.11)$$

$$y_{(t)} = h_{(t)} = o_{(t)} \oplus \tanh(C_{(t)}) \quad (4.12)$$

where W_{xi} , W_{xf} , W_{xo} , and W_{xg} are the weights to be connected to input ($x_{(t)}$). W_{hi} , W_{hf} , W_{ho} , and W_{hg} are the weights to be linked to the previous short-term state ($h_{(t-1)}$). The bias applied to prevent forgetting all the states at the beginning of training. Lastly, due to a large number of parameters, LSTM suffers from slowness issue [51].

- **Gated recurrent units (GRU):** Is a simplified version of the LSTM which was introduced in 2014 by Cho et al. [61]. GRU is faster because it has fewer parameters than LSTM, it has only two gates: the update gate and the reset gate. Moreover, GRU cell omits the output gate and merge the input and forget gate into the update gate [10]. The reset gate decide the amount of the past information to forget. Figure 4.10 presents the structure of the GRU cell. The formulas for the GRU cell states as follows:

$$z_{(t)} = \sigma(W_{xz}^T \cdot x_{(t)} + W_{hz}^T \cdot h_{(t-1)} + b_z) \quad (4.13)$$

$$r_{(t)} = \sigma(W_{xr}^T \cdot x_{(t)} + W_{hr}^T \cdot h_{(t-1)} + b_r) \quad (4.14)$$

$$g_{(t)} = \tanh(W_{xg}^T \cdot x_{(t)} + W_{hg}^T \cdot (r_{(t)} \oplus h_{(t-1)}) + b_g) \quad (4.15)$$

$$y_{(t)} = h_{(t)} = z_{(t)} \oplus h_{(t-1)} + (1 - z_{(t)}) \oplus g_{(t)} \quad (4.16)$$

where W_{xz} , W_{xr} , and W_{xg} are the weights related to the input ($x_{(t)}$). W_{hz} , W_{hr} , and W_{hg} are the weights to be connected to the previous short-term state ($h_{(t-1)}$). Lastly, we can see that both states are merged into a single one which is $h_{(t)}$, unlike LSTM that contains $h_{(t)}$ and $c_{(t)}$.

In some situations it is important to explore the context from the both the directions for such problem bidirectional recurrent neural network (BRNN) was proposed by Schuster et al. [62]. BRNN trained in both directions for a given text to gain all the information about the data. Bidirectional is done by linking the outputs of two RNNs, one to process the sequence from left to right and the other to do the reverse. In addition to the BRNN, we can combine it with GRU to achieve better results. BiGRU is the name of the layer that is composed of bidirectional GRU units. Promising results achieved by the BiGRU in Arabic text classification as in [20], [51].

RNNs Methodology

We will investigate RNNs model to classify Arabic text into time period. Figure 4.12 presents our RNNs methodology for the temporal classification task. The first step is preparing the input data before feeding it to the model. The pre-processing step explained in advance in Section 4.1. In the tokenization step we will attempt

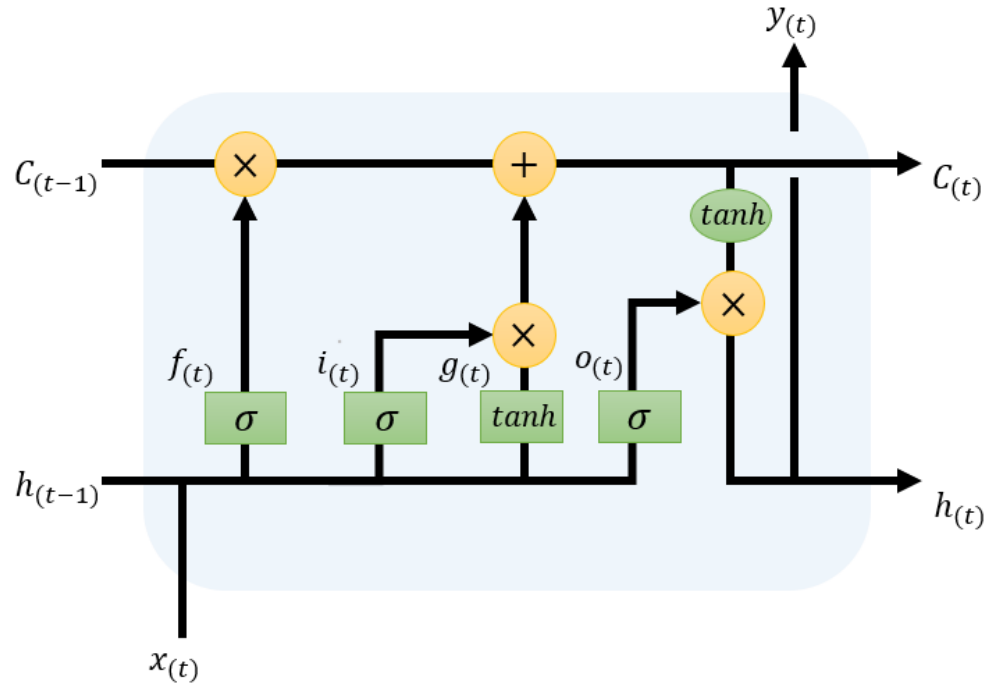


Figure 4.9: Structure of the LSTM cell (adapted from [10])

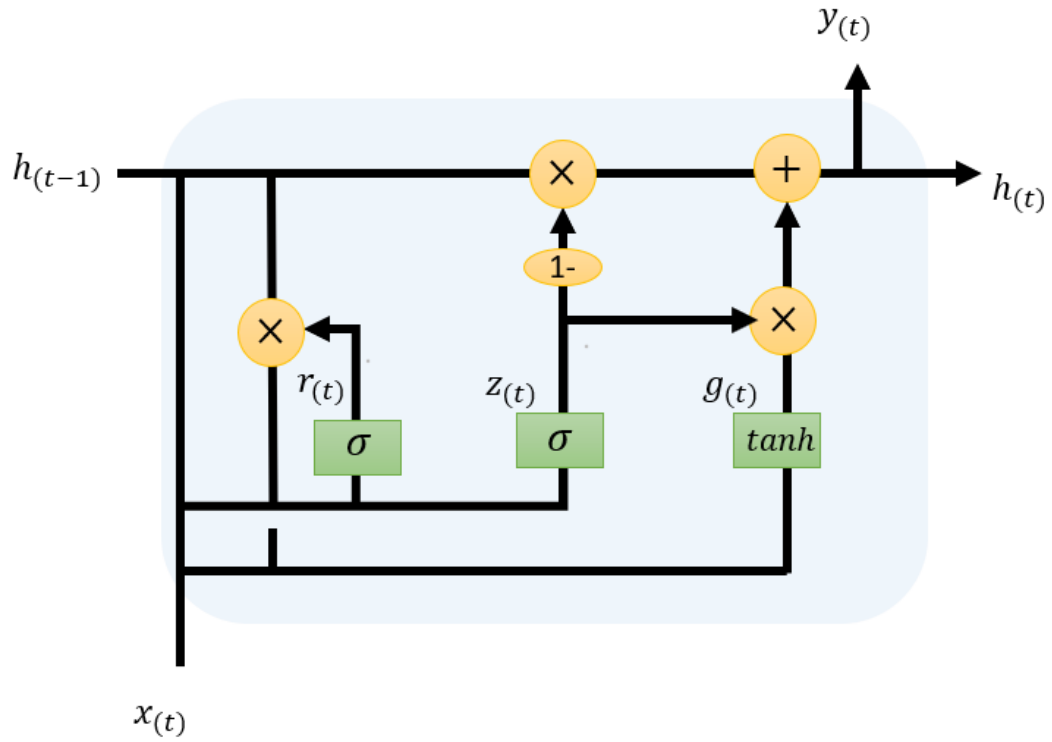


Figure 4.10: Structure of the GRU cell (adapted from [10])

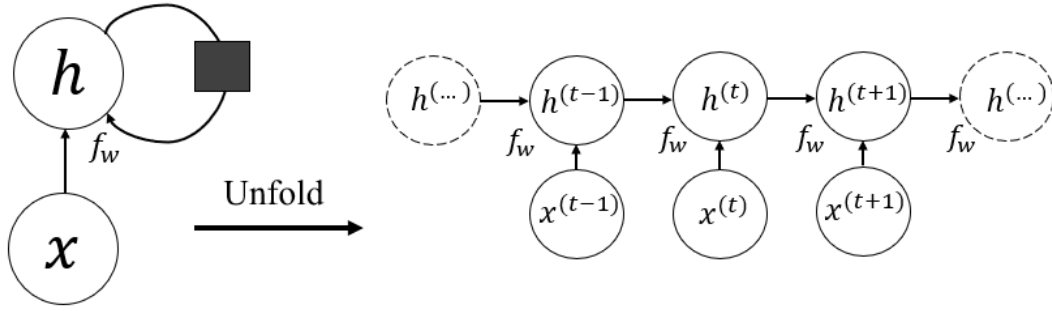


Figure 4.11: Simple RNN without output

two approaches: the word-level model and the character-level model. Therefore, two schemes will be utilized as follows:

1. **Words as tokens:** Where the sequence of words is input as tokens. We represent each word as a unique integer. Thus, we end up with a dictionary of all the unique words in the dataset after pre-processing. The size of the dictionary varies according to the vocabulary size of the used dataset.
2. **Characters as tokens:** Where the sequence of characters used as input tokens. We apply the same approach as in words as tokens scheme, but we represent characters instead of words. Figure 4.13 shows an example of characters as tokens scheme. The upper part of the figure presents the dictionary where the characters are converted to unique numbers. The lower part shows the example converted to characters tokens and represented as numbers.

Because of the disparity of samples lengths for the both schemes, we count the maximum number of tokens by sample. Then we apply padding to the end of each sample in the dataset to unify the lengths of the samples.

The RNNs model architecture in Figure 4.12 contains different layers. The input,

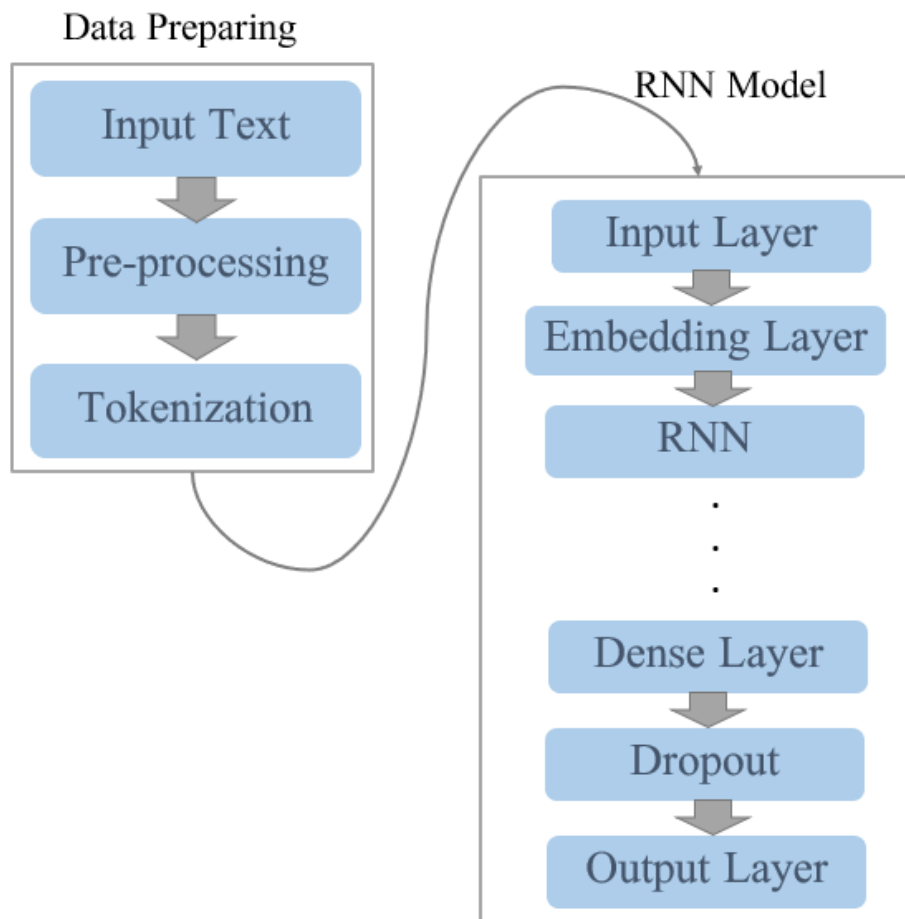


Figure 4.12: The RNNs methodology structure.

dropout, dense, and output layers are similar to what being explained in the ANN methodology. The embedding layer Is initialized with random weights. Then it will learn the embedding for all the words in the training set. Moreover, we use the embedding layer as part of the RNN model, where the embedding is learned during the model training. The RNN layer in the Figure 4.12 could be one of the three layers types: LSTM, GRU, and BiGRU. The activation function for the previous layers is ReLU.

The Dictionary																																
ي	و	ه	ن	م	ل	ك	ق	ف	غ	ع	ظ	ط	ض	ص	ش	س	ز	ر	ذ	د	خ	ح	ج	ث	ت	ب	ا	إ	أ			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1		
Example																																
وإني أراك غدا ماضيا في الكلام																																
	م	ا	ل	ك	ل	ا			ي	ف		ا	ي	ض	ا	م		ا	د	غ		ك	ا	ر	أ		ي	ن	إ	و		
	27	4	26	25	26	4	1	31	23	1	4	31	18	4	27	1	4	11	22	1	25	4	13	2	1	31	28	3	30			

Figure 4.13: Example of converting sample into a character representation for the RNNs model

For the RNNs model, we investigate two different setups architectures word-level model and character-level model. Each approach has its advantages and disadvantages. The word-level model take the input samples sequence of words. The main advantage of it the short input length. However, the word-level model suffers from out of vocabulary (OOV) problem, where the model cannot predict unseen words [63]. In contrast, the previous issue does not affect the character-level model. The major advantages of the character-level model are the small vocabulary and flexibility in handling any words that formed from the vocabulary. Despite that, the character-level model suffers from an long input sequence that needs more deep architecture to capture the context.

4.2.3 Other Models

In addition to the previous models, we will investigate some of traditional machine learning algorithms and other architectures of neural networks types. Moreover, we

will investigate the logistic regression, naive Bayes, and random forest models because it was reported in the literature to compare its performance to the neural network systems. However, since the five literature eras setup is our standard setup, we will only examine these models in that setup for both datasets.

Traditional Machine Learning Models

- **Logistic Regression Model:** is used mostly with classification problems. Logistic regression uses *sigmoid* function to calculate the probability of the distance between the input and decision boundary. Thus, the output will a value between zero and one. The equation of the *sigmoid* function is mentioned earlier in Equation 4.2. The logistic regression equation is

$$\hat{y} = \frac{1}{1 + \exp(-(w_0 + w_1x))} = \begin{cases} 1, & \text{if } \hat{y} > 0.5 \\ 0, & \text{otherwise} \end{cases} \quad (4.17)$$

where y is the output of the logistic regression model that represent the predicted probability. Moreover, $w_0 + w_1x$ is the linear model within logistic regression which is the input to the *sigmoid* function. Where x is the input variable, w_0 is the bias, and w_1 is the weight for the input.

The cost function for logistic regression is Log Loss. Log Loss is the classification metric based on probabilities, a lower cost value indicates better predictions.

The formula for the Log loss is defined as follows:

$$\text{Log Loss} = \sum_{(x,y) \in D} -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y}) \quad (4.18)$$

where $(x, y) \in D$ is the dataset which contains x as sample and y as label, y is the label, and $\hat{y}$ is the predicted value. Furthermore, regularization is important for logistic regression model to avoid overfitting issue. We will use the ridge regularization method and add it to the cost function. The ridge penalty is equal to $\lambda \|w\|_2^2$. The hyperparameter λ is used to control the penalty on the weights. The higher value of λ will reduce the overfitting of the model. Therefore, we will tune the hyperparameter λ and the best value of it will be reported. Lastly, the optimization method that will be used is the limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm (*lbfgs*).

- **Random Forest Model:** is an ensemble learning method for classification, and other tasks, it constructs many decision trees at training time. For each decision tree the model calculates the nodes importance using *Gini* importance as following:

$$\text{Gini} = 1 - \sum_{i=1}^C p_i^2 \quad (4.19)$$

where C is the number of classes and p_i is the probability of class i . The output of the random forest is the class which was selected by most trees.

- **Naive Bayes Model:** is based on Bayes' Theorem, which calculates the probability of an event based on the probability of an already occurred event, called

conditional probability. The conditional probability calculated as follows:

$$P(y|X) = \frac{P(y|X)P(y)}{P(X)} \quad (4.20)$$

where, y is the class and X is a feature vector:

$$X = \{x_0, x_1, \dots, x_n\} \quad (4.21)$$

Naive Bayes has many classifiers, we will use Multinomial Naive Bayes that used for text classification. The feature vectors represent the frequencies with which a multinomial distribution has generated certain events.

Other Neural Networks Models

The additional neural networks models are:

- **The hybrid model:** Combines the RNN model and ANN model. For the ANN model we are going to use word average length and lexical density features as input instead of bag of words. These features represent part of the information of the sample. Therefore, we combine the ANN model with the RNN model to provide more information for the model.

Hybrid model unlike ANN and RNN, which consider as a sequential model. Sequential built the model layer by layer in sequence. Therefore, shared layers are not allowed, and no multiple inputs or outputs. On the other hand, the hybrid model allowed shared layers and multiple inputs or outputs, allowing

more flexibility to create complex models. Figure 4.14 present the structure of our hybrid model. The concatenate layer takes two inputs which are the outputs of the ANN and RNN models. The output of the layer feed to a dense layer to show the prediction results.

Furthermore, the best architecture of the RNNs and ANN will be combined to build the hybrid model. Thus, the model is considered a multiple input model. The input for the RNN could be a sequence of words or characters according to the best performance achieved. Nevertheless, the output for the hybrid model is the same as the other models, the time period for a given text.

- **CNN:** Is mostly used with images and objects classification, recognition, and detection. It achieved exceptional classification performance in the literature [64]. Recently CNN has acquired an interest in text classification problems. Orabi et al. [6] applied it to classify Arabic poems into eras [6]. Therefore, we will follow the same architecture they used in their experiments. Figure 4.15 shows the architecture for the CNN model that been applied in [6]. They used words as tokens as input for the model that feed into the embedding layer. Moreover, they used a convolution layer of one dimension (Conv1D). Also, they applied a max-pooling layer of one dimension.

4.2.4 Models Tuning Process

The models tuning process will be done based on the performance of the validation set which is separate from the test set. For the best configuration we will tune the

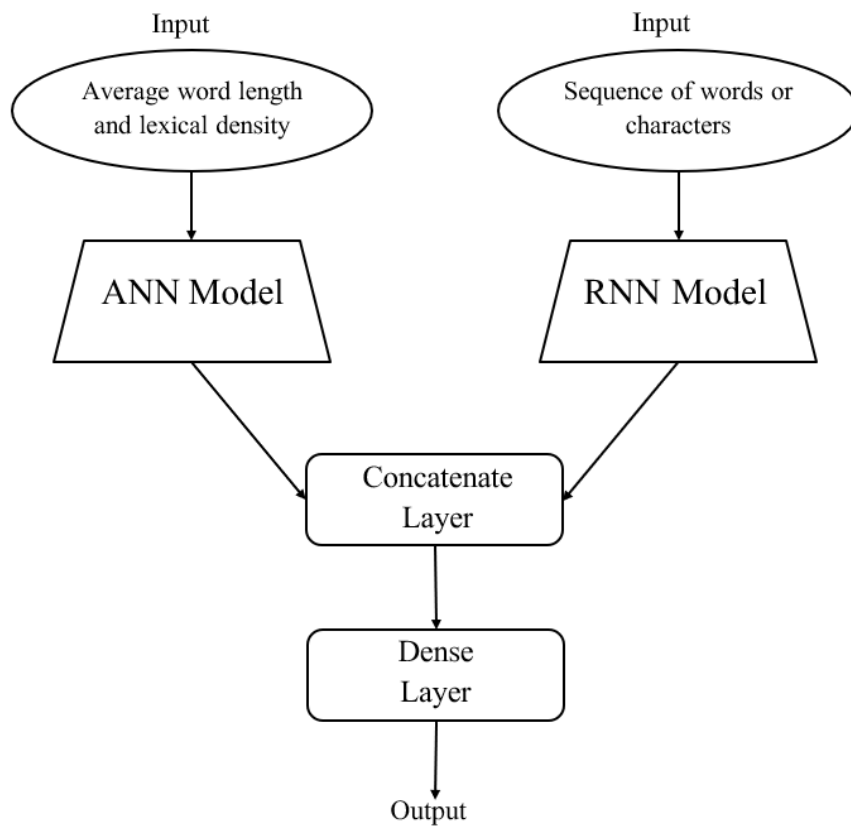


Figure 4.14: The general structure of The hybrid model

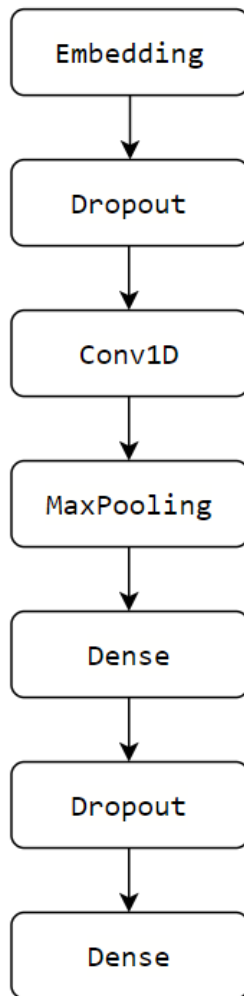


Figure 4.15: The architecture of the CNN model applied by Orabi et al. [6]

model for both RNN and ANN models. However, the other models are not included in this process. Models tuning process includes the following:

- **Vocabulary size:** We will try different sizes of vocabulary. Moreover, we will try all the vocabulary. Then, we will report the best vocabulary size for all the models with each dataset.
- **Tokenization techniques:** We will tune them for the both models. For the RNN model we will employ two approaches: word-level model and character-level model for the RNN model. Initial results will determine which approach to continue with and improve. At the same time, we are going to use word occurrences and TF-IDF with the ANN model. According the initial results the best one will be utilized for the rest of the experiments.
- **The model architecture alteration:** For the ANN model we will alter it in terms of number of layers and neuron. Since we only used dense layer in the ANN. In contrast, we will investigate three different architectures for the RNN model which are LSTM, GRU, and BiGRU architectures. For each architecture we will first start with one layer then increase them until the performance decreases, or we reach a plateau. Subsequently, we alter the number of neurons for the best architecture obtained in advance.
- **The optimizer:** As mentioned in Section 4.2.1 that we will use RMSProp and Adam optimizer. We will investigate these two optimizers in our experiments. Moreover, we are going to try different batch sizes.

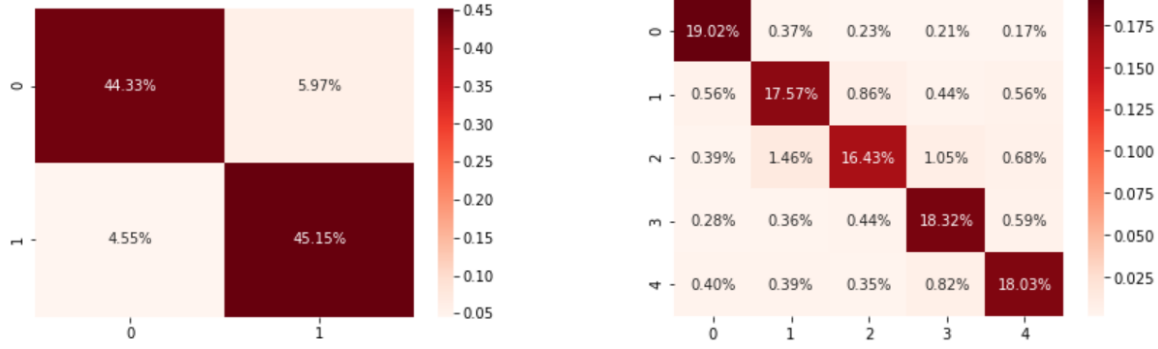


Figure 4.16: Two examples of confusion matrix for binary and multi-class classification

Furthermore, instead of altering the learning rate value manually, we will use the callbacks from Keras that alter it automatically. Another useful callback that save the best model according to its best accuracy. Thus, if the best result was achieved in the middle of the training process, and the performance started to decrease. In this case, the best result for the model is already saved.

4.3 Evaluation Metrics

In this section, we present the measures we used to evaluate the performance of the temporal text classification model. Accuracy, precision, recall, and F1-measure metrics will be used for our models. In addition to the previous metrics, the confusion matrix will be visualized as a heat-map to overview the wrongly predicted data. Figure 4.16 shows examples of heat-map confusion matrix for the binary and multi-class classification. The most used metric is accuracy for binary and multi-classes problems [65]. Therefore, in the evaluation of the experiments, we focus on accuracy because our datasets are balanced. The best results of the models will be reported using all the metrics.

Furthermore, the confusion matrix for binary classification will be illustrated to point some critical values. In any binary classification problem one of the classes treated as positive class and the other as negative class. Accordingly, we have four different numbers which are true positive (tp), false negative (fn), false positive (fp), and true negative (tn). The tp and the tn indicate the count of positive and negative samples which are predicted correctly. Whereas the fn and the fp indicate the count of mispredicted samples for the positive and negative classes, respectively. Moreover, the confusion matrix is utilized with multi-classes, and most of the metrics can be generalized accordingly.

Based on the confusion matrix we explain the evaluation measurement used. The definition and the formula for each evaluation metrics are listed below:

- Accuracy: Is the rate of the correct predictions over the total of all the evaluated samples.

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fn + fp}$$

- Precision (p): Is the portion of the correct predicted values over the total predictions for a particular class.

$$p = \frac{tp}{tp + fp}$$

- Recall (r): Is the proportion of the samples that have been predicted correctly

over the total number of samples in a specific class.

$$r = \frac{tp}{tp + fn}$$

- F1-Measure: Depends on the precision and recall values which is the harmonic mean between their values.

$$F1 = \frac{2 \times p \times r}{p + r}$$

The previous formulas for the precision, recall, and F1-Measure are for binary classification only. Therefore, we will compute the average for them. A macro-average will be applied for each metric because it treats all the classes equally.

Furthermore, to compare the performance of two different classifiers that use the same dataset. We will use the difference of two proportions proposed by Dietterich et al. [66] as our measure to see if the differences in the results are statistically significance or not.

CHAPTER 5

EXPERIMENTS AND RESULTS

This chapter presents the conducted experiments to classify Arabic text into a time period. We utilized ANN, RNNs, and other models for the temporal classification task. Moreover, we used OpenITI, and APCD datasets to train the models. The OpenITI dataset contains Arabic literature texts like book scripts. APCD dataset include only Arabic poems.

For each dataset, we will investigate several class setups. The class setups are either predefined periods or custom periods. Five literature eras setup is an example of the predefined periods, whereas the 300-years custom period represent custom periods. Moreover, to find the best design of ANN and RNNs models in each dataset. First, we will alter the models using only the five literature eras class setup. Then we will use the best design for the rest of the class setups.

Remove stop words and lemmatization are common in ANLP tasks and could improve the model performance. Therefore, we will explore its effectiveness in our experiments. In Chapter 3, we mentioned that we separate the authors in the training set, validation set, and test set. We will investigate the effectiveness of merging the

authors between these sets.

5.1 Temporal Classification with Authors Separated

Our experiments were implemented using Colab from Google. Keras along with Tensorflow as backend were used to train the models. Besides the previous frameworks, we utilize *scikit-learn* libraries for the tokenization techniques and traditional machine learning models.

Earlier in Chapter 4 we introduced some cases to be investigated in the experiments like remove the stop words, lemmatization, and merging the authors. In this section, all the results are with separating the authors in the training, validation, and test sets. Moreover, we show the results with and without removing the stop words. The remaining cases will be discuss in the next sections. Lastly, we report validation accuracy for all the results in this section.

We will utilize the binary cross-entropy loss for the binary class models and sparse categorical cross-entropy loss for the multi-class models. Moreover, the number of training epochs is 10 for all the models. However, we will apply Early-Stopping callback as mentioned in Chapter 4, which saves the best model during the training. Lastly, The batch size varies for each model.

5.1.1 Temporal Classification Using ANN

ANN model is our baseline model for classifying Arabic text into time periods. We assess the performance of the model based on the validation set. Moreover, we tune the model hyperparameter for each dataset. We alter the model using the five literature eras setup since it is our standard setup. The tuning process includes vocabulary size, tokenization techniques, model architecture, and the optimizer. Due to memory limitation, we try the following vocabulary sizes: 10000, 15000, and 20000 words. Moreover, for tokenization, we investigate BOW and TF-IDF techniques. Lastly, we alter the model architecture in terms of the number of layers and number of neurons.

Table 5.1 shows the initial results for the OpenITI dataset in terms of vocabulary size, and tokenizer. Overall the presented results are similar. However, BOW outperformed TF-IDF tokenizers in most of the experiments. The lowest result was by using the 10000 words as vocabulary size and TF-IDF tokenizer. ANN model achieved the best results with a vocabulary size of 15000 words.

Table 5.2 shows the initial results for the APCD dataset in terms of vocabulary size, and tokenizer. Since each setup in the APCD dataset has many configurations regarding the number of verses per sample, the presented results use one verse per sample. The ANN's APCD dataset achieved the best results with 15000 words as vocabulary size like the OpenITI dataset. Similar to the OpenITI dataset, BOW is better than TF-IDF tokenizers in most of the experiments.

Table 5.3 presents the initial results for the OpenITI dataset of the alteration in terms of the number of layers and tokenizer using the ANN model. The 1-dense layer

architecture achieved better results in both tokenizers. However, the BOW tokenizer outperformed the TF-IDF tokenizer.

Table 5.4 shows the initial results for the APCD dataset of the tuning process in terms of the number of layers and tokenizer. The results are for the five literature era setup and one verse per sample configuration. BOW tokenizer outperformed the TF-IDF tokenizer in both model architectures. However, the ANN model best result was achieved by the 1-dense layers architecture and BOW tokenizer.

Furthermore, we tune the model in terms of the number of neurons. We tried 512, 252, 128, 64, and 32. The best results were achieved by using 32-units for the model layers. Moreover, in terms of the optimizer, we tried RMSPROP and adam optimizer. We will report the best optimizer for each dataset.

From the previous tables, the 15000 words vocabulary size, the architecture of the 1-dense layer, and the BOW tokenizer achieved the best results in both datasets. The architecture of the ANN model is illustrated in Figure 5.1. The first layer is dense with 32 units. The dropout rate is 0.7. Moreover, the last layer units size depends on the class setup that is being applied. Lastly, the batch size used for the ANN model is 512 samples.

OpenITI Dataset

We use the five literature era setup to tune the ANN model. We train the model with 34000 samples, and we measure its performance based on the validation set which contains 6000 samples. Each sample contains 100 words maximum. The total number of words in the training set is 339K words, and after removing the stop words it

Table 5.1: The initial results of the five literature eras setup from the OpenITI dataset using the ANN model in terms of tokenizer and optimizer.

Tokenizer	Vocabulary Size	Validation Accuracy
BOW	10000 words	0.4808
	15000 words	0.4853
	20000 words	0.4843
TF-IDF	10000 words	0.4780
	15000 words	0.4820
	20000 words	0.4815

Table 5.2: The initial results for the five literature eras setup (one-verse per sample) from the APCD dataset using the ANN model in terms of tokenizer and optimizer.

Tokenizer	Vocabulary Size	Validation Accuracy
BOW	10000 words	0.3302
	15000 words	0.3461
	20000 words	0.3298
TF-IDF	10000 words	0.3241
	15000 words	0.3279
	20000 words	0.3343

Table 5.3: The initial results for the five literature eras setup from the OpenITI dataset using the ANN model in terms of number of layers and tokenizer.

Tokenizer	Model Architecture	Validation Accuracy
BOW	1-Dense layers	0.4853
BOW	2-Dense layers	0.4705
TF-IDF	1-Dense layers	0.4820
TF-IDF	2-Dense layers	0.4708

Table 5.4: The initial results for the five literature eras setup (one-verse per sample) from the APCD dataset using the ANN model in terms of number of layers and tokenizer.

Tokenizer	Model Architecture	Validation Accuracy
BOW	1-Dense layers	0.3461
BOW	2-Dense layers	0.3241
TF-IDF	1-Dense layers	0.3279
TF-IDF	2-Dense layers	0.3200

decreases to 256K. The average number of words per sample is 99 words, and after removing the stop words is 75 words.

We investigate two optimizers for the OpenITI dataset. The results are similar in the ANN model for both RMSPROP and adam optimizer. However, the OpenITI dataset results were better when we utilized the RMSPROP.

Table 5.5 presents the validation and test accuracy for all the classes setups using the ANN architecture. The second column explains stop words removal is applied or not. We show the result with and without the stop words. Most of the results are similar or have slight improvement without removing the stop words, which could be because stop words affect the context of the Arabic literature. Moreover, we can see that the accuracy decreases when the number of classes increases. For example, the accuracy for the binary class setup is 0.68, whereas it is 0.21 for the 15-class setup. The lowest baseline result for the 15-class problem is 0.06 accuracy which can achieve by randomly selecting a class. Thus, although the temporal classification task is difficult, our model results show significant improvement more than the baseline model results.

Figure 5.2a shows the five literature eras confusion matrix using ANN. More than half of the Islamic era (class-0) was classified incorrectly as Abbasid era (class-1), *Aldoul wa al-amarat* era (class-2), or Ottoman era (class-3). At the same time, Abbasid era (class-1), *Aldoul wa al-amarat* era (class-2), and Ottoman era (class-3) are the eras that the model can identify correctly the most. Moreover, most eras overlap with consecutive eras because there is no clear distinction between their periods.

The confusion matrix for 300-years custom period setup is presented in Figure 5.2b. Class-0 (represents the pre-Islamic, Islamic, and first half of the Abbasid periods)

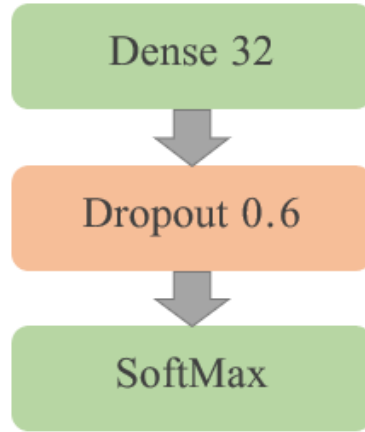


Figure 5.1: The best architecture of the ANN model for OpenITI and APCD datasets samples were identify correctly the most by the ANN model. More than half of the class-1 (represents the second half of the Abbasid, and 40% of the *Aldoul wa al-amarat* periods) samples were misclassified. Most of the wrongly predicted samples were either labeled as the next or the previous era. The confusion matrix for the remaining setups are illustrated in the Appendix A.1.

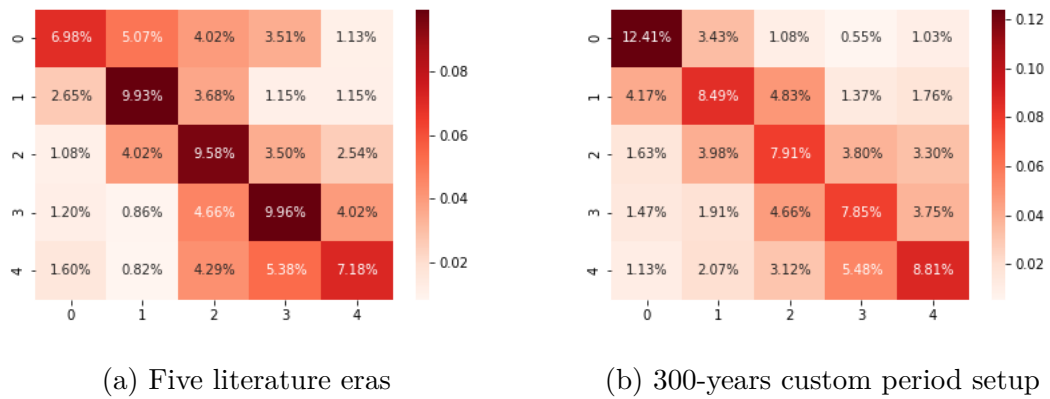


Figure 5.2: The confusion matrix for five literature eras (a) and 300-years custom period (b) setups for the OpenITI dataset using the ANN model.

Table 5.5: The results for OpenITI dataset using ANN model.

Class Setup	Remove Stop-words	Validation Accuracy	Test Accuracy
Binary literature eras	No	0.68	0.67
	Yes	0.68	0.67
3-eras	No	0.67	0.65
	Yes	0.65	0.65
Five literature eras	No	0.49	0.44
	Yes	0.48	0.43
300-years custom period (5-class)	No	0.46	0.45
	Yes	0.45	0.45
200-years custom period (8-class)	No	0.35	0.33
	Yes	0.32	0.32
100-years custom period (15-class)	No	0.21	0.20
	Yes	0.21	0.20

APCD

In the APCD dataset, we have many configurations in terms of verses per sample for each class setup. We use the five literature era setup with one-verse-per-sample configuration to tune the ANN model. We train the model with 15000 samples, and each sample contains one poem verse. The average word length for the sample is nine words, and after removing the stop words, it is around seven words.

In terms of the optimizer, we tried different optimizers for the APCD dataset. Although results are similar to the OpenITI dataset, the results were better when we used the adam optimizer for the APCD dataset.

Moreover, in APCD for each class setup, we investigate different configurations of the number of verses per sample. Table 5.6 shows the results for different configurations for the five literature eras setup. We first start with one-verse-per-sample then we jump to eight-verses-per-sample. The results for eight verses is better than one

Table 5.6: The results of several configurations for the five literature eras setup from the APCD dataset using the ANN model.

Number of Verses per Sample	Validation Accuracy
1 verse	0.3275
8 verses	0.5361
12 verses	0.6046
14 verses	0.6053
15 verses	0.6520
16 verses	0.5907

verse, thus we jump to 16 verses per sample. If the performance decreases we go back to the middle point between the 16 and the eight. We continue on the same process going back and forward until we find the best configuration. Lastly, the 15 verses per sample achieved better results.

The results for the configurations of the number of verses per sample for the remaining setups are similar. That indicates more verses increase the accuracy considering that we are limited to 16 verses due to the lack of samples. However, the 12-eras setup is an exception. Because we convert it into an 11-eras setup due to the lack of samples for a certain era. More details about 12-eras setup is explained in Chapter 3.

The best results for APCD using ANN are shown in Table 5.7. Similar to OpenITI results, we present the results with and without removing the stop words. Most of the results were similar when we included the stop words. However, the 11-eras setup achieved better test accuracy without removing the stop words.

Figure 5.3a shows the confusion matrix for binary literature eras setup. A quarter of the classical era (class-0) samples were predicted as the modern era (class-1).

Table 5.7: The results for APCD dataset using ANN model.

Class Setup	Remove Stop-words	Validation Accuracy	Test Accuracy
Binary literature eras	No	0.81	0.78
	Yes	0.78	0.78
Five literature eras	No	0.65	0.62
	Yes	0.63	0.61
11-eras	No	0.33	0.36
	Yes	0.33	0.33
12-eras	No	0.20	0.19
	Yes	0.19	0.18

However, about 18% of the modern era (class-1) samples was predicted as classical era (class-0). Furthermore, Figure 5.3b presents the confusion matrix for five literature eras setup. The Pre-Islamic (class-0) is the era which has most misclassification. In contrast, 80% of the Islamic era (class-1) samples were predicted correctly. The Ottoman and modern eras are consecutive eras. About 30% of Ottoman era (class-3) samples were mispredicted as the modern era (class-4) and vice versa. The rest of the confusion matrices for the other setups are presented in the Appendix A.2.

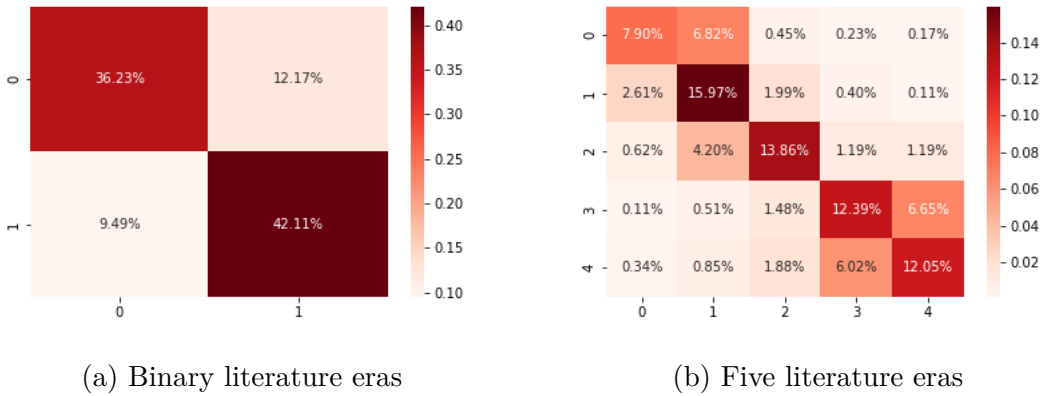


Figure 5.3: Confusion matrix for binary literature eras (a) and five literature eras (b) setups for the APCD dataset using the ANN model.

5.1.2 Temporal Classification Using RNNs

We utilized the RNNs model to address the temporal classification task. We evaluated the model according to the validation set accuracy. Moreover, we investigated different RNNs architectures and altered the model hyperparameters for each dataset. For OpenITI and APCD datasets, we tune the model using the five literature eras setup. For the APCD dataset, we use the configuration of one-verse-per-sample.

The hyperparameters alteration involves vocabulary size, tokenization approaches, number of layers, and the optimizer. Regarding the vocabulary size, we start with 20000 and increase it until no sufficient improvement to the performance. We also include all the vocabulary and compare it to the best vocabulary size. The best vocabulary size is different for each dataset.

Furthermore, we utilized tokens as words or characters. In terms of model design, we applied LSTM, GRU, and BiGRU architectures. For each architecture, we tried a different number of layers. Table 5.8 shows the preliminary results for the RNNs model in terms of the type of tokens, layer type, and the number of layers for the OpenITI dataset. The word-level approach performed better than the character-level approach. However, the character-level model achieved better results with a deeper model design. The 2-BiGRU layers architecture achieved the best results with two layers for both datasets.

Table 5.9 presents the initial results for the APCD dataset in terms of the type of tokens, layer type, and the number of layers using the RNNs model. The word-level approach outperformed the character-level approach. Similar to the OpenITI dataset,

Table 5.8: The initial results for the five literature eras setup form the OpenITI dataset using RNNs model.

Model Type	Model Design	Validation Accuracy
Char-level	1-LSTM layers	0.2050
	2-GRU layers	0.2057
	3-GRU layers	0.2050
	1-BiGRU layers	0.4070
	2-BiGRU layers	0.3997
Word-level	1-LSTM layers	0.4212
	2-LSTM layers	0.4182
	1-GRU layers	0.4243
	2-GRU layers	0.4290
	3-GRU layers	0.4170
	1-BiGRU layers	0.4275
	2-BiGRU layers	0.4382
	3-BiGRU layers	0.4257

the word-level approach with 2-BiGRU layers model architecture achieved the best results.

We investigated a different number of neurons as we did for the ANN model. 32-units achieved the best results for each layer. Moreover, in terms of the optimizer, we tried RMSPROP and adam optimizer. The RMSPROP optimizer outperformed the adam optimizer for all the datasets.

Figure 5.4 illustrates the best architecture of the RNNs model for the OpenITI and APCD datasets. The number of units of each layer is shown next to the layer name. For example, the rate for the dropout layer is 0.7. The units of the last layer depend on the class setup that is applied. Moreover, the RMSPROP optimizer outperformed the Adam optimizer in the RNNs model for all the datasets. The batch size used for the RNNs model is 128 samples.

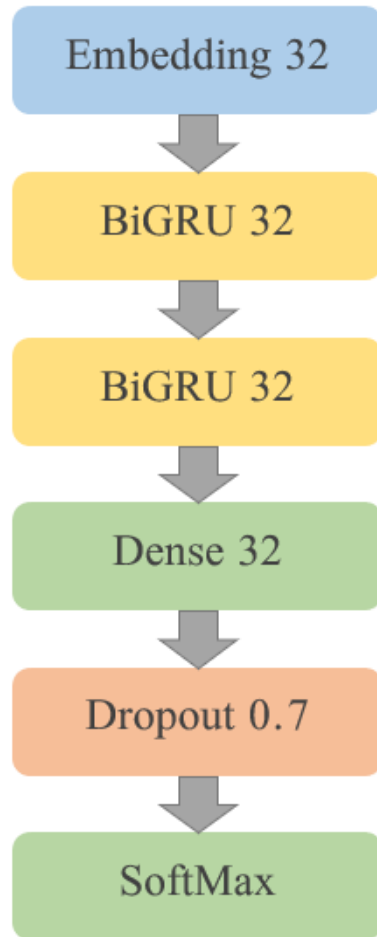


Figure 5.4: The best architecture of the RNNs model for all the datasets

Table 5.9: The initial results for the five literature eras setup (one-verse per sample) from the APCD dataset using RNNs model.

Model Type	Model Design	Validation Accuracy
Char-level	1-LSTM layers	0.2675
	2-LSTM layers	0.2524
	1-GRU layers	0.2418
	2-GRU layers	0.2221
	1-BiGRU layers	0.2792
	2-BiGRU layers	0.2573
Word-level	1-LSTM layers	0.3007
	2-LSTM layers	0.3053
	3-LSTM layers	0.2901
	1-GRU layers	0.3015
	2-GRU layers	0.2966
	1-BiGRU layers	0.3049
	2-BiGRU layers	0.3105
	3-BiGRU layers	0.3049

OpenITI Dataset

We tuned the RNNs model using the five literature eras setup to find the vocabulary size that achieved the highest result. Table 5.10 shows the results for different vocabulary sizes. The accuracy decreases when the vocabulary size is more than 20000 words. However, 20000 words achieved a better result than using all the vocabulary. We use the best vocabulary size with the remaining class setups.

Table 5.11 shows the results for the word-level RNNs model. Binary is an easy problem than a multi-class problem. Thus as the number of classes increases, the accuracy decreases. Like the ANN model, most of the results without removing the stop words are similar or a little better.

Figure 5.5a shows the confusion matrices for five literature eras setup using the RNNs model. More than half of the Islamic era (class-0) samples was classified as the

Table 5.10: The results for the RNNs model using different vocabulary sizes for the five literature eras setup from OpenITI dataset.

Vocabulary Size	Validation Accuracy
All the vocabulary	0.44
10000 words	0.44
20000 words	0.45
40000 words	0.44
60000 words	0.43
80000 words	0.43

Table 5.11: The results for OpenITI dataset using word-level RNNs model.

Class Setup	Remove Stop-words	Validation Accuracy	Test Accuracy
Binary literature eras	No	0.67	0.67
	Yes	0.68	0.67
3-eras	No	0.60	0.61
	Yes	0.59	0.60
Five literature eras	No	0.47	0.40
	Yes	0.44	0.40
300-years custom period (5-class)	No	0.40	0.41
	Yes	0.36	0.39
200-years custom period (8-class)	No	0.31	0.29
	Yes	0.35	0.27
100-years custom period (15-class)	No	0.19	0.18
	Yes	0.17	0.16

Abbasid era (class-1). However, the Islamic era (class-0) was never predicted as the modern era (class-4).

The confusion matrix for the 300-years custom period setup is shown in Figure 5.5b. Most of class-3 samples were mispredicted, either as class-2 or class-4. Moreover, class-0 and class-4 is the most era that the model can distinguish from the other classes. Class-0 cover the Pre-Islamic and Islamic periods, which could be a possible reason for its uniqueness. Lastly, Although both setups are five classes problem in Figure 5.5, but the results are different. The confusion matrices for the rest setups are presented in the Appendix A.3.

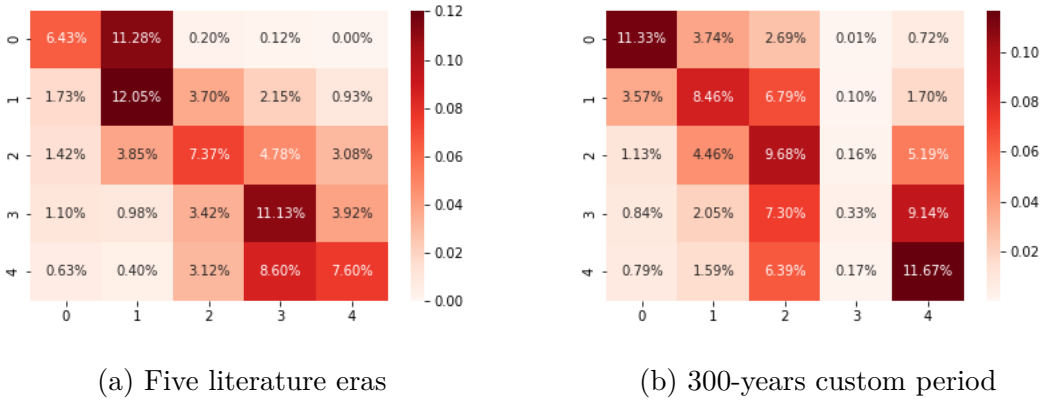


Figure 5.5: Confusion matrix for five literature eras (a) and 300-years custom period (b) setups for the OpenITI dataset using the RNNs model.

APCD

We tuned the word-level RNNs model using the five literature eras setup with one verse per sample to find the vocabulary size that achieves the best performance. Table 5.12 presents the results for the RNNs model using different vocabulary sizes. The performance is stable for most of the sizes. However, it rises to 80000 words, and this

vocabulary size achieves the best results. Therefore, for the remaining setups, we use a vocabulary size of 80000 words.

We followed the same process with the ANN model to find the best configuration regarding the number of verses per sample. Table 5.13 shows the results for the APCD dataset. The binary class setup achieved better results using the RNN model than using the ANN model. Moreover, removing the stop words shows slight improvement in the five literature eras setup; otherwise, it is better to keep it.

Figure 5.6a illustrates the confusion matrix for binary literature eras setup using RNNs model. About 29% of the classical era (class-0) samples were predicted as the modern era (class-1). Although one-third of the modern (class-1) samples were misclassified, the RNNs model correctly identified the modern era more than the classical one (class-0).

Furthermore, Figure 5.6b shows the confusion matrix for the five literature eras setup results using the RNNs model. The *Aldoul wa al-amarat* era (class-2) is the most era that the model classified correctly. In contrast, most pre-Islamic era (class-0) samples were misclassified due to the lack of pre-Islamic period literature. More than half of the modern era (class-4) was predicted as the Ottoman era (class-3). Therefore, an overlap appeared between these eras. Lastly, the confusion matrices for the remaining setups are showed in the Appendix A.4.

5.1.3 Temporal Classification Using Other Models

For the other model, we applied only the standard in terms of number of classes (i.e, five literature eras setup) in the OpenITI and APCD datasets. Moreover, for

Table 5.12: The results for the RNNs model using different vocabulary sizes for the five literature eras setup from APCD dataset.

Vocabulary Size	Validation Accuracy
All the vocabulary	0.31
20000 words	0.31
40000 words	0.31
60000 words	0.31
80000 words	0.32
100000 words	0.31

Table 5.13: The results for APCD dataset using the word-level RNNs model.

Class Setup	Remove Stop-words	Validation Accuracy	Test Accuracy
Binary literature eras	No	0.82	0.76
	Yes	0.81	0.76
Five literature eras	No	0.48	0.42
	Yes	0.45	0.44
11-eras	No	0.22	0.21
	Yes	0.20	0.19
12-eras	No	0.16	0.14
	Yes	0.14	0.13

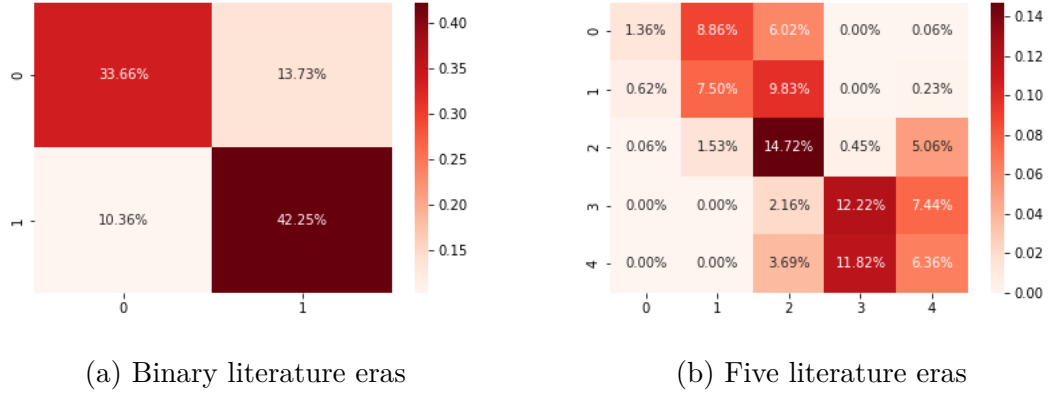


Figure 5.6: Confusion matrix for binary literature eras (a) and five literature eras (b) setups for the APCD dataset using the RNNs model.

the APCD dataset, we used the configuration where each sample includes 15 verses because the best results were achieved using it. The other models involve logistic regression, Naive Bayes, random forest, CNN, and the hybrid-ANN-RNNs models. The hybrid-ANN-RNNs model is explained in other models section in Chapter 4.

We use *scikit-learn* to implement the traditional machine learning models. For the logistic regression, the hyperparameter C represents the inverse of the hyperparameter λ , which we explained in Chapter 4. Therefore, we alter and report the best value of the hyperparameter C for each dataset. Moreover, we tune the number of decision trees in the random forest model.

Furthermore, for the CNN, we follow the same architecture applied by Orabi et al. [6]. However, they split the data into training and validation sets differently from our mechanism. In our CNN experiments, we split the data into three sets as discussed in Chapter 3. Moreover, we try the the adam and RMSProp optimizer for the CNN model. The best optimizer will be reported for each dataset. The batch size used for the CNN model is 128 samples.

The hybrid-ANN-RNNs model is composed of ANN and RNNs models. The ANN and RNNs models, in terms of layers and neurons, are the best model in the corresponding dataset. Thus, the design for each dataset is different according to its results in the previous experiments. Moreover, The input for the ANN in the hybrid model is two extracted features, where the RNNs utilized words as tokens.

OpenITI Dataset

For the logistic regression model, we tune the hyperparameter C to reduce the overfitting. As a result, the model achieved the best results using the hyperparameter C of 0.001. The best number of decision trees for the random forest model is 100. Moreover, for the CNN model we investigate two optimizer. The RMSPROP optimizer outperformed the adam optimizer.

The results for all the other models are illustrated in Table 5.14. The results are similar after stop words removal except on the hybrid model and Naive Bayes. For example, the accuracy for the CNN model before and after removing the stop words is 0.46. Moreover, the accuracy of the hybrid model decreased from 0.45 to 0.43 after removing the stop words. For Naive Bayes the accuracy increased after removing the stop words.

Figure 5.7 presents the confusion matrices for five literature eras setup for the logistic regression and CNN models. Figure 5.7a shows the confusion matrix for five literature eras setup using logistic regression model. About half of the Islamic era (class-0) samples were misclassified. Although the Ottoman era (class-3) is not a neighbor era to the Islamic era (class-0), a quarter of the Islamic era (class-0) samples

Table 5.14: The results for OpenITI in the other models experiments

Model	Remove Stop-words	Validation Accuracy	Test Accuracy
Hybrid Model	No	0.45	0.42
	Yes	0.43	0.40
CNN	No	0.46	0.46
	Yes	0.46	0.42
Logistic Regression	No	0.46	0.42
	Yes	0.45	0.41
Naive Bayes	No	0.43	0.38
	Yes	0.44	0.39
Random Forest	No	0.44	0.40
	Yes	0.44	0.40

were misclassified as the Ottoman era (class-3).

Furthermore, Figure 5.7b illustrates the confusion matrix for five literature eras setup using the CNN model. Half of the Abbasid (class-1) and the Ottoman (class-3) samples were classified correctly. The model identified only a quarter of the Islamic era (class-0).

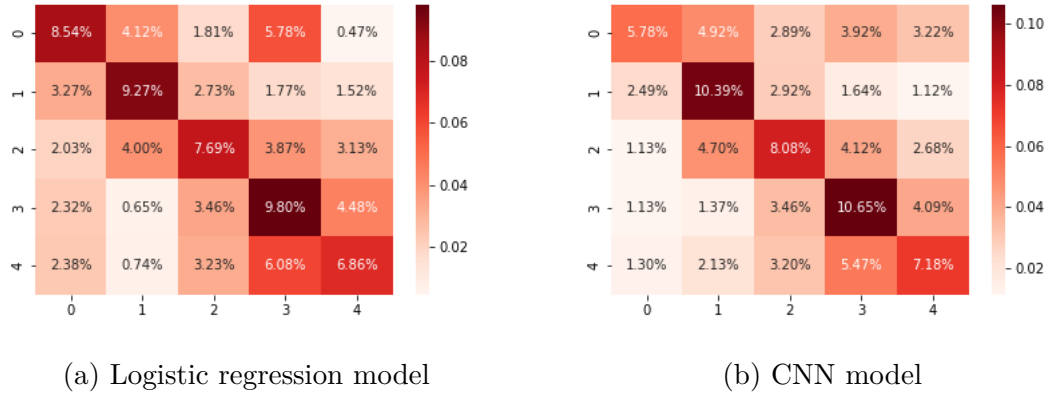


Figure 5.7: Confusion matrix for five literature eras setup using logistic regression (a) and CNN (b) models for the OpenITI dataset.

APCD

We alter hyperparameter C in the logistic regression. The best result was achieved when the value of the C hyperparameter was equal to 0.01. For the random forest model the best number of decision trees is 300. Moreover, we try different optimizer for the CNN model. Unlike OpenITI, the adam optimizer overcome the RMSPROP optimizer. Moreover,

Table 5.15 presents the results for the other models using the APCD dataset. The results for most models are similar before and after removing the stop words. However, the CNN model the results are better without removing the stop words. For example, the accuracy for the CNN model before removing the stop words is 0.51 and 0.49 after removing it.

Figure 5.8a shows the confusion matrix for five literature eras setup using the logistic regression. The model correctly predicted most of the Islamic era (class-1) samples. In contrast, half of the pre-Islamic era (class-0) samples were mispredicted.

Figure 5.8b presents the confusion matrix for five literature eras setup using the CNN models. About half of the Ottoman era (class-3) samples were classified as the modern era (class-4) and vice versa. Moreover, 15% of the modern era (class-4) samples were predicted as Abbasid era (class-2).

Although the pre-Islamic era (class-0) has few samples, logistic regression model can predicted better than CNN model. However, CNN model classified the Abbasid era (class-2) better than the logistic regression model. CNN model missclassified the last two eras with each others.

Table 5.15: The results for APCD in the other models experiments.

Model	Remove Stop-words	Validation Accuracy	Test Accuracy
Hybrid Model	No	0.49	0.44
	Yes	0.48	0.44
CNN	No	0.50	0.51
	Yes	0.48	0.46
Logistic Regression	No	0.60	0.57
	Yes	0.61	0.57
Naive Bayes	No	0.65	0.60
	Yes	0.65	0.59
Random Forest	No	0.58	0.52
	Yes	0.55	0.52

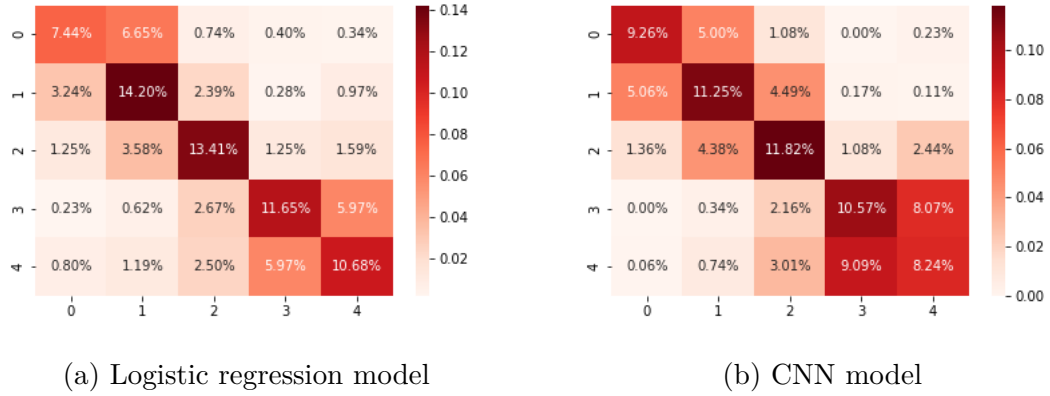


Figure 5.8: Confusion matrix for five literature eras setup using logistic regression (a) and CNN (b) models for the APCD dataset.

5.2 Temporal Classification with Authors Merged

In section 5.1, we showed the results for our experiments where the authors are separated in all the three sets. In this section we present our results where we merge the authors and show the results for all the models and the datasets. Removing stop words showed similar results and, in some cases, decreased the accuracy results in the previous experiments. Therefore, we will keep the stop words the coming experiments.

Table 5.16 presents the results for the best models with merging the authors in

the OpenITI dataset. The performance improved after merging the authors for all the setups using the ANN model. However, the RNNs model shows better improvement than the ANN only in the binary setup. For example, in the 100-years periods' setup, the ANN model achieved an accuracy of 0.40 score, where the RNNs model accuracy is 0.21.

Figure 5.9 presents the confusion matrix for five literature eras and 300-years custom period setups using the ANN model. Although both setups are five classes problem, the five literature eras setup Figure 5.9a outperformed the 300-years custom period setup Figure 5.9b.

Figure 5.9a presents confusion matrix for five literature eras setup. The model identified most of the Pre-Islamic era (class-0) and Ottoman era (class-1) samples correctly. In contrast, about half of the *Aldoul wa al-amarat* era (class-2) was misclassified as the Ottoman era (class-1).

The confusion matrix for the 300-years custom period setup without separating the authors is shown in Figure 5.9b. Class-3 was the most time period that the model mispredicted. Class-0 and class-4 were classified correctly more than the other classes. Therefore, it indicates the difference between the classical and the modern in the Arabic language.

We utilized the same number of verses per author for the APCD dataset that reported the best results earlier when we separated the authors. The results for the APCD are shown in Table 5.17. Similar to the OpenITI dataset, the binary literature eras setup shows better results using RNNs. However, some results were affected negatively after merging the authors. For instance, the binary setup using the ANN

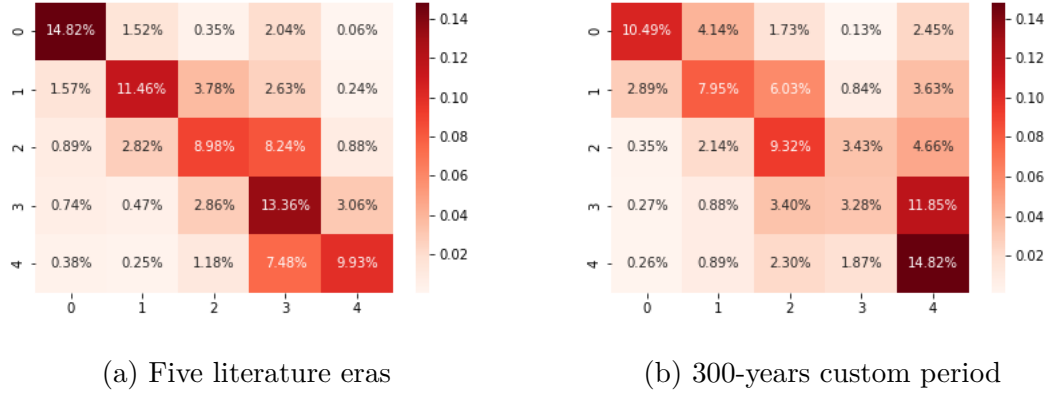


Figure 5.9: Confusion matrix for five literature eras (a) and 300-years custom period (b) setups from the OpenITI dataset using the ANN model with merging the authors.

achieved an accuracy of 0.78, but without merging the authors, it gained an accuracy of 0.81.

From the above experiments, we can see that the results of OpenITI improve after merging the authors. However, in the APCD dataset, most of the results are similar with and without separating the authors.

5.3 The Effect of Lemmatization

Lemmatization is the process of converting each word in the dataset to its raw word. However, the process consumes time and resources. Therefore, we will limit this case to the five literature eras setup in OpenITI and APCD. In the APCD, the configuration of 15 verses per sample is utilized because it achieves the best result in the previous experiments.

Table 5.18 presents the results for both datasets in all the presented models. We show the results with and without separating the authors. The results before and

Table 5.16: The results for all the models using OpenITI dataset with merging the authors.

Class Setup	Model	Validation Accuracy	Test Accuracy
Binary literature eras	ANN	0.83	0.82
	RNNs	0.84	0.83
3-eras	ANN	0.75	0.75
	RNNs	0.73	0.72
Five literature eras	ANN	0.67	0.66
	RNNs	0.60	0.57
	Hybrid Model	0.63	0.63
	CNN	0.64	0.63
	Logistic Regression	0.57	0.57
	Naive Bayes	0.58	0.58
	Random Forest	0.58	0.57
300-years custom period (5-class)	ANN	0.56	0.55
	RNNs	0.47	0.47
200-years custom period (8-class)	ANN	0.49	0.51
	RNNs	0.38	0.38
100-years custom period (15-class)	ANN	0.40	0.40
	RNNs	0.21	0.20

Table 5.17: The results for all the models using APCD dataset without separating the authors

Class Setup	Model	Validation Accuracy	Test Accuracy
Binary literature eras	ANN	0.81	0.79
	RNNs	0.80	0.78
Five literature eras	ANN	0.66	0.67
	RNNs	0.46	0.47
	Hybrid Model	0.50	0.49
	CNN	0.55	0.53
	Logistic Regression	0.59	0.62
	Naive Bayes	0.65	0.65
	Random Forest	0.53	0.55
11-eras	ANN	0.42	0.41
	RNNs	0.23	0.22
12-eras	ANN	0.20	0.22
	RNNs	0.16	0.17

after the lemmatization are similar in all the experiments using the OpenITI dataset experiments. However, CNN model show a slight improvement in the results. With regards to APCD dataset, using lemmas improves some results. ANN model results are impacted positively more than the other models results.

Figure 5.10a presents the confusion matrix for the five literature era setup for the APCD dataset using the ANN model. The model identified the Pre-Islamic (class-0), Abbasid (class-1), and modern (class-2) eras better compared to the results before applying lemmatization. However, the Islamic era (class-1) predicted correctly the most after and before using lemmas. The Ottoman era (class-3) was never classified as the Pre-Islamic era (class-0).

Figure 5.10b presents the confusion matrix for the five literature era setup for the APCD dataset using the ANN model after merging the authors. The Abbasid era

(class-3) samples predicted correctly the most. A quarter of the modern era (class-4) samples were classified as Ottoman era (class-3).

A possible reason for the improved results in the APCD dataset that poems use complex and varied vocabulary, unlike OpenITI dataset that frequently utilized the common vocabulary. We already presented the previous evidence in the data analysis section in Chapter 3, where the APCD dataset is only 25% of the OpenITI dataset, which has 115K unique words compared to 151K in OpenITI. Hence the APCD dataset has a richer vocabulary. Furthermore, lemmatization decreases the number of vocabulary in the APCD model, which reduce the overfitting problem.

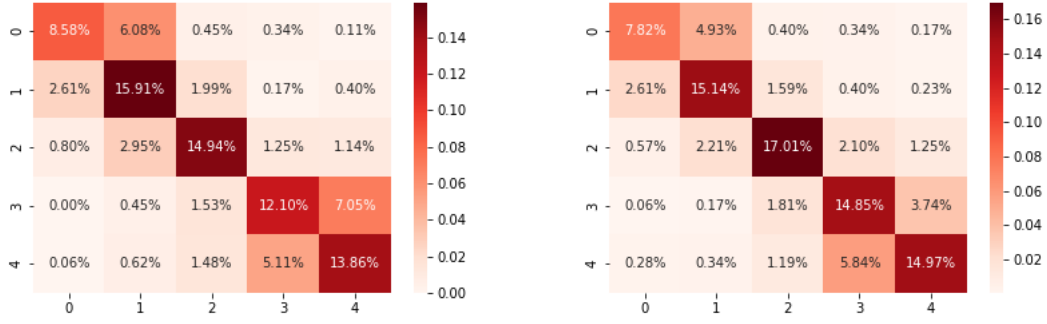
Table 5.19 shows two samples from the OpenITI dataset before and after lemmatization. We presented part of the original samples because it is long. The first sample is a verse from a poem, whereas the other sample is a text from a book. We can see that the tool could not lemmatize all the words for both samples, which is from old eras because most lemmatization tools are used with standard Arabic [5]. For example, the tool should convert the word **وتقتادني** in the first sample to **اقتاد**. That indicates a possible reason for not improving the OpenITI results after using lemmas.

5.4 Summary of The Best Results

In this section, we will report the best model that achieve the highest results among all the models for a given task on a specific dataset. The results for the different datasets reported in terms of accuracy, precision, recall, and F1-measure for the test set. We show the results with and without merging the authors. Moreover, if the

Table 5.18: The results for all the models using five literature eras setup in OpenITI and APCD datasets after applying lemmatization.

Dataset	Model	Separated Authors	Validation Accuracy	Test Accuracy
OpenITI	ANN	Yes	0.49	0.44
		No	0.66	0.65
	RNNs	Yes	0.44	0.39
		No	0.57	0.55
	Hybrid Model	Yes	0.48	0.42
		No	0.59	0.59
	CNN	Yes	0.47	0.42
		No	0.64	0.64
	Logistic Regression	Yes	0.38	0.41
		No	0.58	0.58
	Naive Bayes	Yes	0.42	0.39
		No	0.53	0.54
	Random Forest	Yes	0.45	0.43
		No	0.58	0.58
APCD	ANN	Yes	0.67	0.65
		No	0.69	0.70
	RNNs	Yes	0.50	0.48
		No	0.48	0.47
	Hybrid Model	Yes	0.50	0.49
		No	0.50	0.49
	CNN	Yes	0.55	0.55
		No	0.61	0.59
	Logistic Regression	Yes	0.64	0.61
		No	0.64	0.64
	Naive Bayes	Yes	0.66	0.62
		No	0.67	0.66
	Random Forest	Yes	0.57	0.54
		No	0.56	0.57



(a) Five literature eras (Authors Separated) (b) Five literature eras (Authors Merged)

Figure 5.10: Confusion matrix for five literature eras when we separated the authors (a) and when we merge them (b) setups for the APCD dataset using the ANN model.

Table 5.19: Part of two samples from the OpenITI dataset before and after applying lemmatization.

Original sample	Sample After Lemmatization	Era
وقد يكون الصبا مني بمنزلة يوما وتقتادني الهيـف الرعايد	قد كان صبا من منزلة يوم وتقتادني هيـف رعايد	Islamic
ويروى مشعب أي لم تدر ما قول من يشعبك عنها فيصرفك ويواعدك	أروى مشعب أي لم أدر ما قول من يشعبك عن فيصرفك ويواعدك	Abbasid

results for lemmatization are better, we will report it.

We start with summarizing the best results where the authors are separated in the training, validation, and test sets. ANN model outperformed the other models in all the class setups for the OpenITI dataset. The results for the ANN model using OpenITI are shown in Table 5.20. Binary literature eras and 3-eras setups have similar results for the test set results, but the performance decreases when classes are more than five. Since we have balanced datasets, most of the results are similar using the various metrics.

Table 5.20: The best achieved results in OpenITI dataset with separating the authors using the ANN model.

Class Setup	Accuracy	Precision	Recall	F1-Measure
Binary literature eras	0.674	0.690	0.675	0.668
3-eras	0.664	0.661	0.664	0.662
Five literature eras	0.436	0.447	0.438	0.436
300-years custom period (5-class)	0.455	0.455	0.459	0.456
200-years custom period (8-class)	0.329	0.328	0.334	0.329
100-years custom period (15-class)	0.203	0.202	0.208	0.202

Table 5.21: The best results for the APCD dataset with separating the authors using the ANN model.

Class Setup	Accuracy	Precision	Recall	F1-Measure
Binary literature eras	0.783	0.784	0.782	0.783
Five literature eras (with lemmatization)	0.654	0.661	0.647	0.651
11-Eras	0.358	0.346	0.352	0.346
12-Eras	0.191	0.189	0.184	0.180

Table 5.21 presents the results of the APCD dataset. ANN is the best performing model for most of the APCD setups, except the binary class setup where the RNNs model achieved. Moreover, lemmatization shows a good impact on the APCD result, unlike the Openiti dataset.

In addition to the separated authors' results, we present the best results when the authors are not separated. The ANN model outperforms the other models in most of the class setups in both datasets. The best results for the OpenITI dataset are presented in Table 5.22. RNNs outperformed ANN only in the binary literature eras setup. Five literature eras setup overcome the 300-years custom period setup significantly. Moreover, the 200-years custom period setup has similar results to the

Table 5.22: The best achieved results in OpenITI dataset with merging the authors.

Class Setup	Model	Accuracy	Precision	Recall	F1-Measure
Binary literature eras	RNNs	0.830	0.830	0.830	0.830
3-Eras	ANN	0.747	0.747	0.748	0.747
Five literature eras	ANN	0.658	0.663	0.663	0.663
300-years custom period (5-class)	ANN	0.548	0.549	0.550	0.549
200-years custom period (8-class)	ANN	0.511	0.516	0.516	0.515
100-years custom period (15-class)	ANN	0.399	0.397	0.402	0.399

Table 5.23: The best results for APCD with merging the authors using ANN model.

Class Setup	Accuracy	Precision	Recall	F1-Measure
Binary literature eras	0.792	0.792	0.791	0.792
Five literature eras (with lemmatization)	0.698	0.699	0.690	0.691
11-eras	0.421	0.417	0.403	0.401
12-eras	0.258	0.235	0.241	0.236

300-years custom period setup.

Furthermore, Table 5.23 shows the best results for the APCD dataset. ANN is the best model for most APCD experiments, except the binary literature era setup. The best result for the five literature era setup is achieved while applying lemmatization.

5.5 Result Analysis and Discussion

We illustrate the investigation of the possible error causes in this section. We apply an analysis only for the five literature eras setup for OpenITI and APCD datasets. Moreover, we conducted several experiments for the five literature eras setup. Thus, we will analyze only the best results for the separated author’s case.

Furthermore, we want to investigate the ability of humans for the temporal periods' classification task since there is no available benchmark. Therefore, we conducted an experiment where participants classified some texts into five literature eras for OpenITI and APCD datasets. The human evaluation is to look for the problem from another perspective. We were only limited to two participants where each participant labeled 40 samples. Thus, our human evaluation is just for a subjective discussion. The human evaluation setup and results are presented in the Appendix B.

5.5.1 OpenITI dataset

First, we discuss the results for the OpenITI dataset. The classes in Figure 5.11 represent the following eras from 0 to 4: Islamic, Abbasid, *Aldoul wa al-amarat*, Ottoman, and modern. From the previous figure, we found that about half of the Islamic era samples were mispredicted as Abbasid or *Aldoul wa al-amarat* eras. Similarly, for the modern era, about half of its samples were classified incorrectly as *Aldoul wa al-amarat* eras or Ottoman. Hence, there are overlapping and similarity between the consecutive eras. One possible cause for the overlapping is how we collected our dataset because there is no clear boundary between the eras that separate them.

Table 5.24 shows two samples with its actual era that classified incorrectly as Islamic era. For the first sample, we can see it about the biography of the prophet Muhammad's peace upon him, which could be a reason for the misclassification. The second sample contains many Quranic verses as quotes; thus, quotes may lead to the wrong prediction. Both samples include part of the text that makes it look like another era; removing quotes could solve it and improve the results.

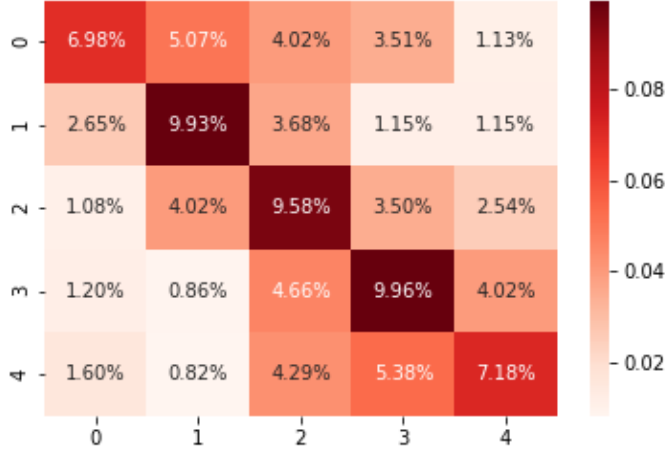


Figure 5.11: The confusion matrix for the five literature eras class setup for the OpenITI dataset using the ANN model

Furthermore, we will compare the sample words frequency in each era. However, relying only on this comparison is insufficient because it is a pure statistic calculation, and the context of the words is essential. Nevertheless, we apply it to get some insight into the problem.

We calculate the word frequency in each era for some samples to compare between the totals. Each sample has a set of n words that the model will classify as a specific era from the eras set of size m as follows:

$$S = \{w_0, w_1, \dots, w_n\}$$

$$E = \{e_0, e_1, \dots, e_m\}$$

where S is the sample and w_0 is the first word in the sample. E is the set of the eras.

We count the frequency of each word in the sample for each era as follows:

Table 5.24: Two samples from the OpenITI dataset for different eras being wrongly predicted as Islamic era.

Era	Sample
Ottoman	المكر به - - وأمر رسول الله صلى الله عليه وآله حذيفة بن اليمان وعمار بن ياسر - معه وأمر - أن يأخذ - الناقة وأمر حذيفة أن - فبينما هم - إذ سمعوا - القوم من - قد - فغضب رسول الله وأمر حذيفة أن - - - - الله حين - حذيفة - أن - قد ظهر - حتى - الناس وأقبل حذيفة حتى أدرك رسول الله صلى الله عليه وآله فلما أدركه قال له - - يا حذيفة - أنت يا عمار - - من العقبة - فقال النبي صلى الله عليه وآله يا حذيفة هل عرفت
Modern	ولو - - عنهم ما كانوا يعملون أولئك الذين - الكتاب والحكم والنبوة فإن يكفر بها هؤلاء فقد - بها قوما ليسوا بها - أولئك الذين هدى الله - - قل لا - عليه أجرا ان هو إلا ذكرى للعالمين ويقول عز من قائل في سورة - قد كانت لكم أسوة حسنة في إبراهيم والذين معه وقد نقل عن مجمع البيان أن المراد من الذين معه هم سائر الأنبياء ويقول تعالى في سورة آل عمران ان أولى الناس - للذين - وهذا النبي والذين آمنوا والله ولي المؤمنين إلى غيرها من الآيات وروى الطبرسي رحمه الله في مكارم الأخلاق والشريف

$$\text{Word frequency} = \sum_{i=0}^n w_{ie_j} \quad (5.1)$$

where n is the number of words in a sample, and e_j is a specific era from the eras set.

We show four cases in Table 5.25, the first one is the wrongly predicted era that has the highest total words occurrences. In the second case, the sample was predicted correctly, and the frequency for the total words is the highest for that era. The rest of the cases show the opposite of the previous cases. The third sample is classified incorrectly, but the total word occurrences are achieved by another era which is neither the actual nor the predicted era. The last case is when the model correctly classified the sample, and the highest word frequency is for a different era.

5.5.2 APCD

The classes in Figure 5.12 represent the following eras from 0 to 4: Pre-Islamic, Islamic, Abbasid, Ottoman, and modern. From the figure, the most correctly predicted era is the Islamic era. Conversely, the Pre-Islamic era is the most incorrectly classified era. A possible reason for the previous case is that Pre-Islamic era samples are the fewest comparing the other eras. Moreover, more than a quarter of the samples in the Ottoman era were predicted as modern and vice versa. Consequently, Ottoman and modern eras are overlapping and have a similarity. The Ottoman and modern are neighboring eras that could cause the overlapping between them, where the literature did not change significantly.

Table 5.25: The actual era, predicted era, and the words frequency per era for some samples from OpenITI datasets.

Sample ID	Actual Era	Predicted Era	Era	Word Frequency
1	Modern	<i>Aldoul wa al-emarkat</i>	Islamic	58,455
			Abbasid	113,925
			<i>Aldoul wa al-emarkat</i>	117,467
			Ottoman	75,741
			Modern	93,003
2	Abbasid	Abbasid	Islamic	99,317
			Abbasid	150,765
			<i>Aldoul wa al-emarkat</i>	147,992
			Ottoman	115,974
			Modern	119,782
3	Ottoman	<i>Aldoul wa al-emarkat</i>	Islamic	71,406
			Abbasid	132,326
			<i>Aldoul wa al-emarkat</i>	126,299
			Ottoman	85,586
			Modern	95,307
4	<i>Aldoul wa al-emarkat</i>	<i>Aldoul wa al-emarkat</i>	Islamic	54,052
			Abbasid	65,135
			<i>Aldoul wa al-emarkat</i>	65,056
			Ottoman	56,480
			Modern	52,854

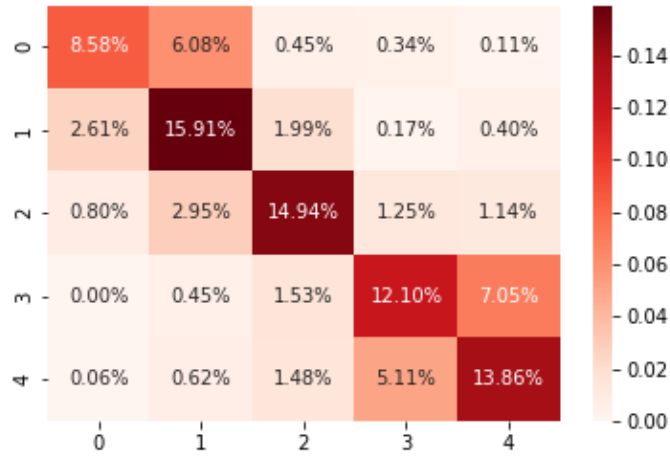


Figure 5.12: The confusion matrix for the five literature eras class setup in the APCD dataset using ANN model

We count the word occurrences in each era, as mentioned earlier in the analysis for the OpenITI dataset. Similarly, we show four cases for the APCD in Table 5.26. For the first sample, the highest total words frequency is for the incorrectly predicted era. The highest total words occurrences are for the actual era in the second sample, which was classified correctly. The remaining cases present the opposite of the already mentioned cases. The third sample was predicted wrongly, but the total word occurrences are for the actual era. The last sample misclassified, and the highest total words frequency is for a different era.

From Table 5.26 we can see that in some cases, the count of words is not important of the certain era. For example, in the last example where the actual era is Pre-Islamic, the number of words in that era is fewer than the others that means the context matter more than the word frequency.

From the above clarification for both datasets, we found that the total word oc-

Table 5.26: The actual era, predicted era, and the words frequency per era for some samples from APCD dataset

Sample ID	Actual Era	Predicted Era	Era	Word Frequency
1	Modern	Ottoman	Pre-Islamic	24,296
			Islamic	45,832
			Abbasid	51,443
			Ottoman	53,749
			Modern	49,004
2	Abbasid	Abbasid	Pre-Islamic	21,783
			Islamic	40,559
			Abbasid	43,822
			Ottoman	43,527
			Modern	40,465
3	Ottoman	Modern	Pre-Islamic	25,253
			Islamic	46,818
			Abbasid	51,990
			Ottoman	53,766K
			Modern	49,434
4	Pre-Islamic	Islamic	Pre-Islamic	18,847
			Islamic	34,724
			Abbasid	35,922
			Ottoman	34,143
			Modern	31,457

currences have a possible impact on the prediction process. However, some cases show the opposite where the total word frequency is not related to the predicted results. Therefore, the classification process does not only depend on the total word occurrences.

5.6 Comparison to Other Works in The Literature

Due to the lack of works in the Arabic language domain, we present some related works in other languages in Table 5.27. The presented studies are not for a comparison purpose; it is for a subjective discussion to show the work of other studies in this

Table 5.27: Other works from the literature similar to our work.

Study	Dataset	Classes	Model	Result
Ciobanu et al. [46]	Romanian literature (91 samples - sample size is 500 sentences)	5 classes (16th century - 17th century - 18th century - 19th century - 20th century)	Random Forest	0.99 (accuracy)
Stajner et al. [4]	Portuguese literature (86 samples - sample size up to 2000 tokens)	4 classes (17th century - 18th century - 19th century - 20th century)	Naive Bayes	0.59 (F1-measure)
our work	Arabic literature (47,055 samples - sample size up to 100 words)	5 classes (Islamic - Abbasid - - Ottoman - modern)	ANN	0.66 (accuracy) 0.66 (F1-measure)
our work	Arabic poems (11,757 samples - sample size up to 169 words)	5 classes (Pre-Islamic - Islamic - Abbasid - Ottoman - modern)	ANN	0.70 (accuracy) 0.70 (F1-measure)

area for similar types of class problems. Ciobanu et al. [46] classified Romanian literature into five centuries from 16th CE to 20th CE, which is similar to our work in terms of number classes. Moreover, Stajner et al. [4] applied Naive Bayes to classify Portuguese literature into four centuries. However, both studies used cross-validation in their experiments, where we presented our test results for our work.

As we did not find other works in the Arabic domain except Orabi et al.[6] work, replicate their binary problem to have an objective comparison. We compare their model to our model using their dataset. Since the RNNs model performs better than ANN in the poem dataset’s binary problem, we will utilize it in the comparison. Table

Table 5.28: Comparison between our best model results and the CNN model proposed by Orabi et al. [6].

Model	Experiment Configuration	Accuracy	F1-Measure	Statistical Significance
Our RNN	Training (73%) - Validation (12%) -	0.92	0.917	± 0.0045
	Test (15%)	0.91	0.911	± 0.0042
CNN by Orabi et al. [6]	Training (90%) - Test (10%)	0.91	0.910	± 0.0052

5.28 shows the results of our RNNs model and the CNN model by [6]. Moreover, our data setup is more restricted than them, as shown in advance in Chapter 3. Orabi et al.[6] split the data into training and validation sets, they directly evaluating their experiments on the validation set. Our RNNs word-level model achieves a better result. Moreover, we calculate the statistical significance for all the accuracy results, showing that our validation results are better than theirs. However, our test set results are similar to their validation results with no significant difference.

5.7 Summary

In this chapter, we presented the experiments on classifying Arabic text into different temporal periods. We evaluated on three different dataset OpenITI, APCD, and the second poem datasets. Several setups in terms of classes are being examined for each dataset. Subsequently, the results were reported for different settings like removing stop words and lemmatization. In this section, we will discuss the finding of our experiments. Moreover, we will address the research questions that are being

introduced in advance in Chapter 1.

The conducted experiments found that the ANN model outperformed the other models for most setups in all the datasets. Furthermore, removing stop words did not affect the results; it achieved similar results, and for some, it showed slight improvement. At the same time, lemmatization improved the performance on the APCD dataset only. On the other hand, merging the authors achieved better results, especially in the OpenITI dataset.

Furthermore, we found many error types that cause the misprediction. The first one is no rigid boundaries between the eras, resulting in overlapping between the neighboring eras. Thus, most misclassification is in the overlapping eras. Moreover, using an old text like Quran or Hadeeth confuse the model and leads to misclassification.

We introduced three research questions for our research. The first question is about the effectiveness of neural networks for addressing the task. Our neural network models exceed the baseline accuracy in all the datasets as presented in Table 5.29. Moreover, the relative improvement is shown in the table. We calculate the relative improvement using the accuracy results for the test set reported in Section 5.4. Significant improvements to the baseline accuracy were shown in the table.

The examined setups in terms of classes are either predefined from previous researches or custom periods proposed for our work. In the OpenITI dataset, we have two setups with five classes the predefined five literature eras and the custom periods, which represent 300-years custom period. Most of the custom time period setup results showed that some era lost its identity and predicted as the next era. In contrast, the five literature eras setup results showed an overlapping between the neighboring

eras without losing the identity of any eras. Moreover, the five literature eras setup outperformed the 300-years custom period setup in the merged authors experiments. Therefore, literature eras show a better representation of the change in the Arabic language than the equal time periods in some cases. The above clarifies that the literature eras differ from custom temporal periods, which is the answer to the second research question.

Table 5.29: The baseline results and the relative improvement to the baseline accuracy for each class setup in all the datasets.

Dataset	Class Setup	Baseline Accuracy	Accuracy (Relative Improvement)	
OpenITI	Binary literature eras	0.50	Separate Authors	Merge Authors
	3-era	0.33	0.67 (0.34)	0.83 (0.66)
	Five literature eras	0.20	0.62 (0.48)	0.74 (0.61)
	300-years custom period (5-class)	0.20	0.44 (0.30)	0.66 (0.58)
	200-years custom period (8-class)	0.13	0.46 (0.33)	0.55 (0.44)
	100-years custom period (15-class)	0.07	0.33 (0.24)	0.51 (0.45)
APCD	Binary literature eras	0.50	0.20 (0.14)	0.40 (0.35)
	Five literature eras	0.20	0.66 (0.52)	0.69 (0.58)
	11-eras	0.09	0.63 (0.54)	0.69 (0.61)
	12-eras	0.08	0.36 (0.30)	0.44 (0.38)
			0.24 (0.17)	0.26 (0.20)

The last research question is about whether the different types of literature have different impacts on classifying the era. Arabic poems are generally different from the literature text from books and others. As mentioned previously, lemmatization impacts the APCD dataset results. At the same time, the results for the OpenITI dataset are similar with and without using lemmas. Thus, classifying poems is different and affected by utilizing additional settings like lemmatization. Based on our experiments, we found that classifying poems into different eras is relatively easier than classifying the other texts.

Earlier in the results analysis and discussions section, we count the occurrences of the words in each era. We found that some words never occur in specific eras. The following examples are from the OpenITI dataset. For example, the word **التي** (means "which" but used for the feminine singular only) never occurred in the Islamic era. Another example, the word **حيثما** (wherever), which is a conditional noun in the Arabic language, never appears in the modern era. The previous examples showed how using Arabic words changes over time. Moreover, some names like **الخفاجي** never occur in

multiple eras which are Islamic, Abbasid, and *Aldoul wa al-amarat* eras. Therefore, the use of some names and words starts from a particular period of time in the Arabic language.

CHAPTER 6

CONCLUSIONS AND FUTURE WORKS

Languages developed over time and were affected by diverse types of changes like semantic and lexical change. Several linguistic studies show the evolution of the Arabic language through time. Historians and linguists divide Arabic literature into eras. These eras represent different time periods. Automatically labeling text into a certain period of time helps in many fields as linguistic and history fields [15]. We benefit from the advent of machine learning techniques to address the problem. In this thesis, we presented machine learning approaches to classify Arabic texts various eras and time periods.

We found from the conducted survey that the time period classification is not widespread in the literature for any language. Moreover, only two studies were presented to classify the Arabic poems into eras. To the best of our knowledge, no works were found for non-poem literature to classify it into a time period. In our research,

we classify both Arabic text and poems into temporal periods.

For the temporal periods' classification, we utilized two publicly available corpora, the OpenITI and APCD corpus, for literature and poems, respectively. Both corpora covered the time from the Pre-Islamic period until the modern period. We tried different class setups including binary and five literature eras setups for each dataset. The binary setup has the classical and modern eras. The five literature eras for the OpenITI dataset consist of Islamic, Abbasid, *Aldoul wa al-amarat*, Ottoman, and modern. The poem dataset's (APCD) five literature eras are Pre-Islamic, Islamic, Abbasid, Ottoman, and modern. We also utilized a second poem dataset partially in the applied experiments.

We employed ANN and RNN as the main models for the temporal period classification experiments. ANN achieved better results for OpenITI and poem datasets (APCD) in most class setups. Whereas in terms of binary classification, our RNN model outperformed some of the models, including Orabi et al. [6] model. For example, the lowest accuracy score for the five-class problem in the OpenITI dataset is 0.40. The highest is 0.66 accuracy, which is better than random guessing but not perfect, which shows that the problem is complex and needs more research.

Furthermore, besides the many class setups, we examined the effect of various settings in our experiments. Removing the stop words did not improve the results of the experiments. At the same time, lemmatization increased the performance in the poem (APCD) results only, which indicates poems use complex and different vocabulary. Better results were achieved in most of the experiments after merging the authors.

The efficiency of the machine learning approaches to classify Arabic text into eras overcame the random guessing model. Moreover, the authors in [6] reported their best results using CNN classifier. However, our RNN outperformed their results. In addition to the evidence above, neural networks based models achieved better results than the traditional machine learning models.

Furthermore, the literature eras setup achieved better results than the custom time periods for the OpenITI dataset when we merged the authors. The five literature eras setup represents the literature's change better than the custom time period. Moreover, the custom time period setup results showed a gap between some time periods that consider one period the next, unlike the literature eras with overlapping but without gaps. Therefore, there is a difference between the two setups, which address the second research question.

The last question was about whether the type of literature impacts the temporal classification results. We tested text and poems as two types of literature; lemmatization affects the poems' results. At the same time, it did not show any impact on the text data type results. This shows that classifying poems has a different level of difficulty than classifying texts into eras.

6.1 Contribution

The following are the major contribution we did:

- Created datasets for temporal periods classification for literature and poems using existing corpora.

- According to our best knowledge this is the first work for non-poems temporal classification in Arabic.
- We presented a thorough investigation of temporal periods classification in both poems and non-poems literature. We investigated several temporal periods setups using different machine learning approaches including deep learning. For the time periods, we applied predefined and custom periods setups. Moreover, we utilized many machine learning and deep learning approaches to address the task and reported the best model.
- We investigated the effect of separating and merging the authors, removing the stop words, and lemmatization.

6.2 Limitations

The limitation of our work are as follows:

- Dataset: we only use two datasets; for the OpenITI dataset, it is the most relevant available corpus to use but not the best for our task. Also, it is large enough to address the task. In addition, the corpus has some issues; for example, the data were labeled by the author’s death date. Thus, the absence of the publish date for the data is one of the issues. However, we add some buffers between the intervals to restrict the effect of this issue. Moreover, we only conducted predefined setups for the APCD dataset due to the lack of information that allows us to create a custom period.

- The sample size for the literature data is a maximum of 100 words because of the limitation of memory and computation speed. For the poems, we are limited to a maximum of 16 verses per sample. Although these samples sizes are good for our experiments, but with a possibility of more memory and computation, we can try a larger sample size to investigate the effect of it.
- There are no existing published works in the area, to benchmark our results. Moreover, the temporal period classification is not widely researched; thus, we can not compare our work objectively with other studies. However, we compare our result to a baseline model which is based on random guessing. Also, we discussed the results of the other similar works to ours subjectively. There is single published work on Arabic poems classification using different dataset, we compared our work on the same dataset.

6.3 Future Work

In this section, we discuss the possible future works for the presented research. The experiments show that lemmatization does not work well for the old classical Arabic. Therefore, improving the lemmatization tools to work better with classical Arabic is possible future work, potentially improving era classification results.

Furthermore, external knowledge about the person names mentioned in the text may help improve the temporal periods' classification. The named entity recognition will help to extract person names, then we get the external knowledge form the background of the person for example his biography.

Colgrove et al. [36] focused on specific authors who led the literary movement on each time period in their dataset. The same idea could be applied to the Arabic literature to show if the performance will improve or not. However, defining the authors who lead the movement for Arabic literature is another research area that historians and literary critics debate.

Belinkov et al. [5] used an efficient algorithm to identify the reused words. Extremely frequent short phrases like *peace be upon him* "عليه السلام" is an example of the reused words. They found that about 20% of the corpus were reused words. They generated a modified version of the OpenITI corpus by removing the reused words. In future work, we can investigate the effect of removing the reused words by using the modified version for the temporal classification task.

APPENDIX A

THE RESULTS CONFUSION MATRICES

A.1 The confusion matrices for the OpenITI dataset results using ANN model

Figures A.1, A.2, A.3, and A.4 present the confusion matrices for the ANN model results in OpenITI dataset.

A.2 The confusion matrices for the APCD dataset results using ANN model

Figures A.6, and A.7 show the confusion matrices for the APCD dataset results using ANN model.

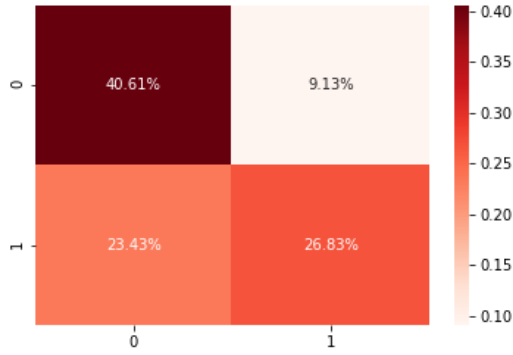


Figure A.1: The confusion matrix for the binary literature eras setup in the OpenITI dataset using ANN model

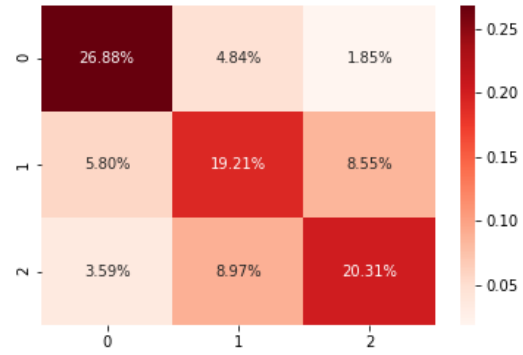


Figure A.2: The confusion matrix for the 3-eras setup proposed by Belinkov et al. [5] in OpenITI dataset using ANN model

A.3 The confusion matrices for the OpenITI dataset results using RNN model

Figures A.8, A.9, A.10, and A.11 present the confusion matrices for the RNN model results in OpenITI dataset.

A.4 The confusion matrices for the APCD dataset results using RNN model

Figures A.12, and A.13 show the confusion matrices for the APCD dataset results using RNN model.

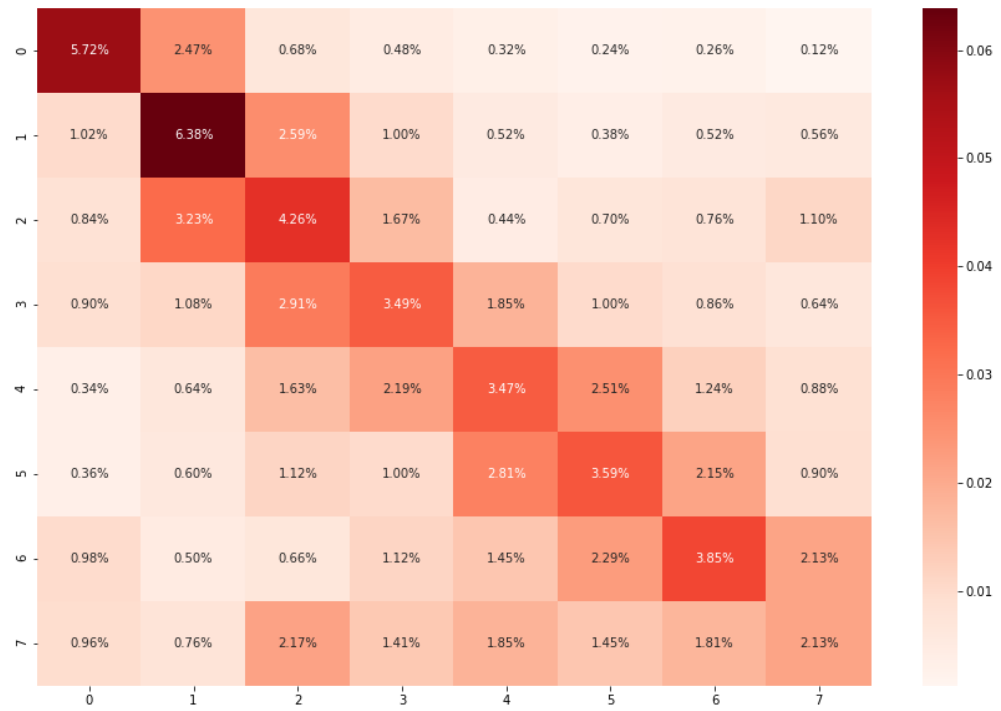


Figure A.3: The confusion matrix for the 200-years custom period setup in the Open-ITI dataset using ANN model

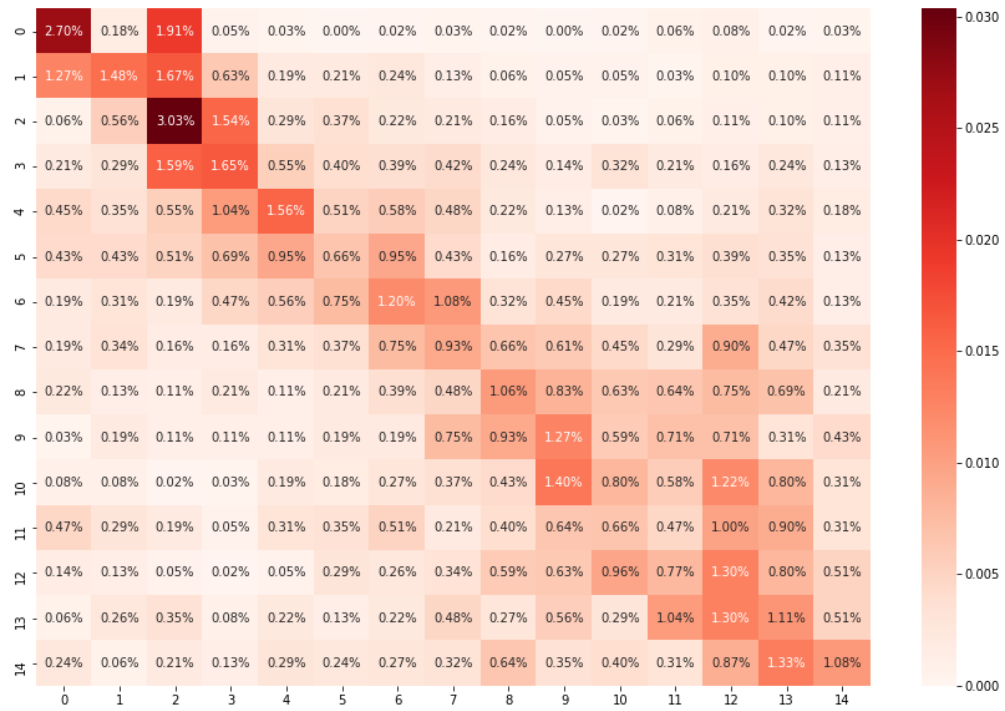


Figure A.4: The confusion matrix for the 100-years custom period setup in the Open-ITI dataset using ANN model

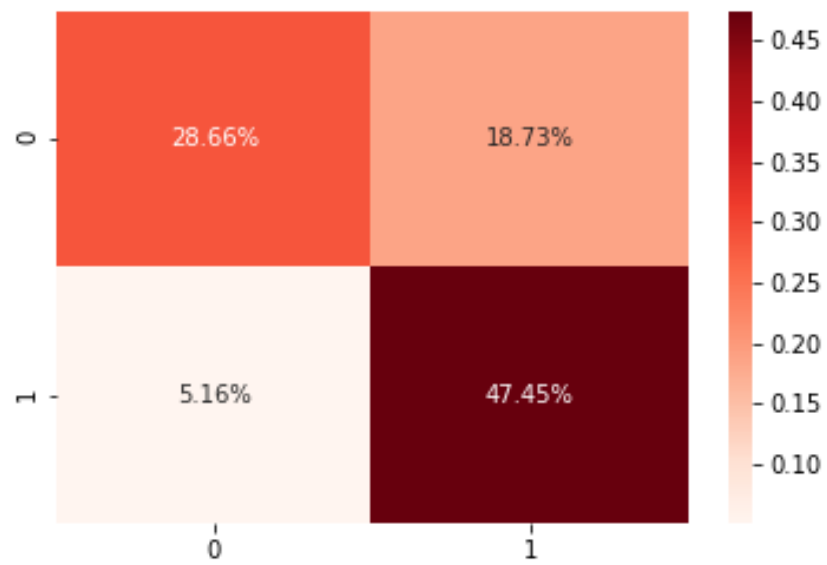


Figure A.5: The confusion matrix for the binary literature eras setup in APCD dataset using ANN model

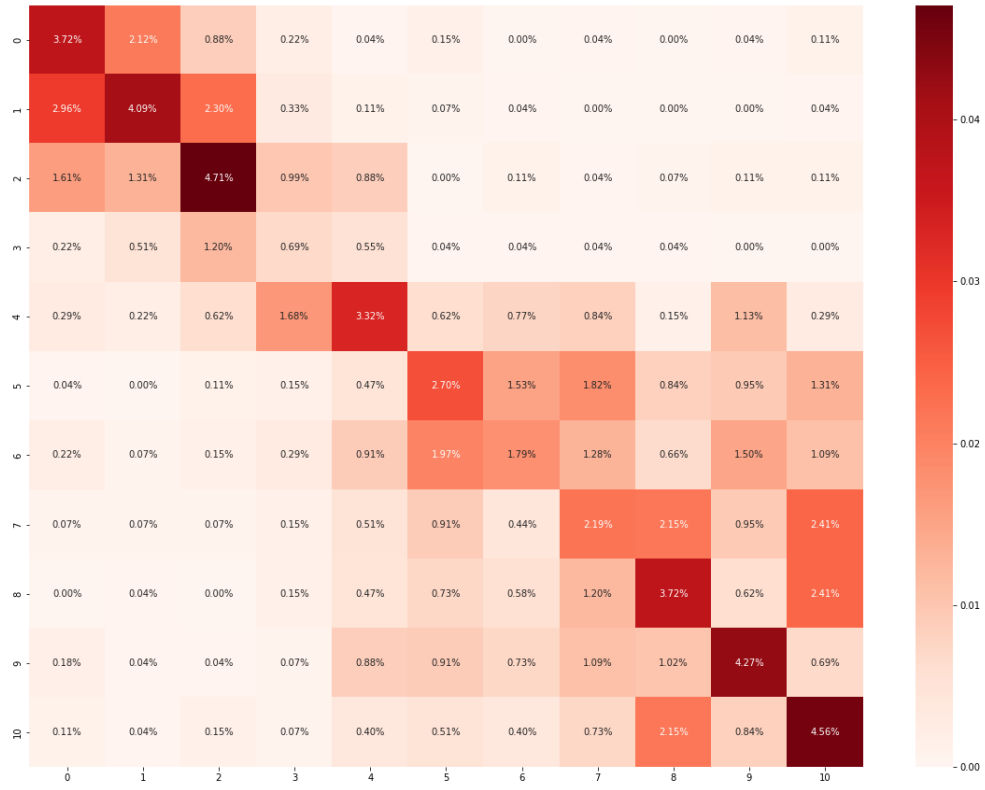


Figure A.6: The confusion matrix for the 11-eras setup in the APCD dataset using ANN model

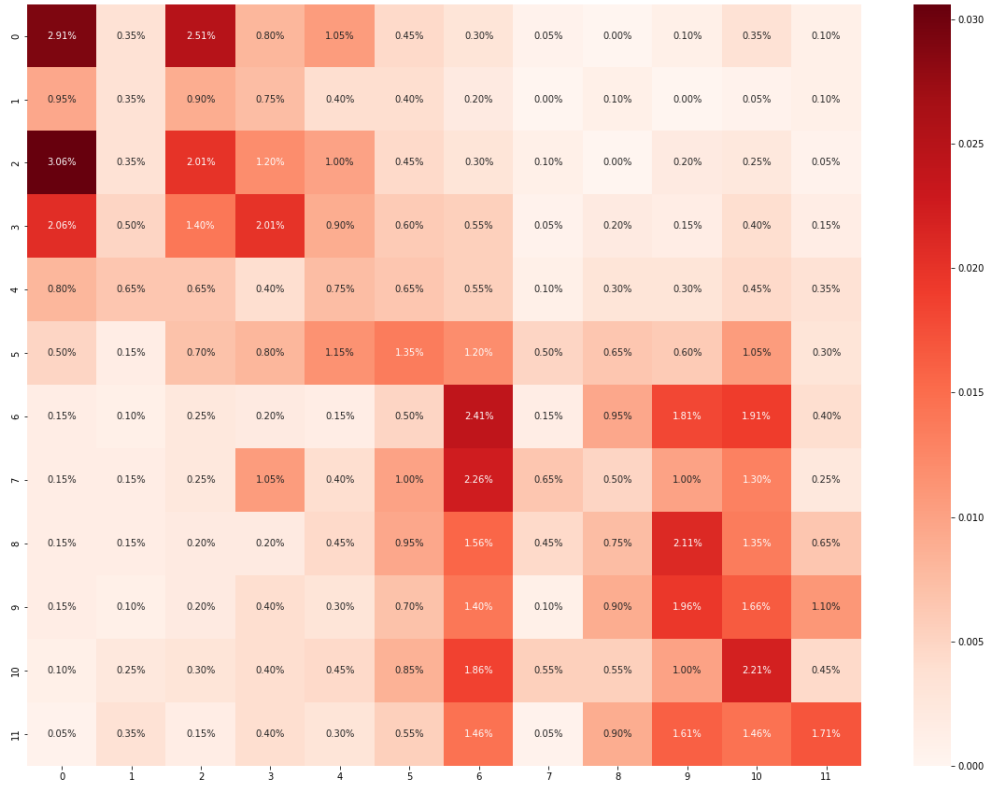


Figure A.7: The confusion matrix for the 12-eras setup in the APCD dataset using ANN model

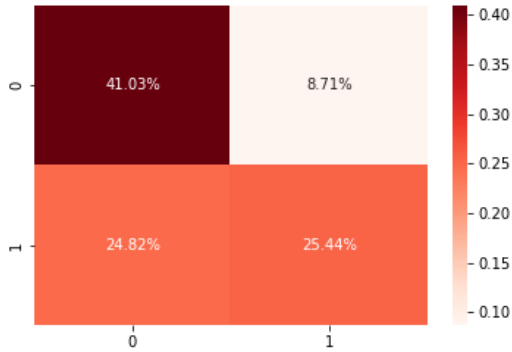


Figure A.8: The confusion matrix for the binary literature eras setup in the OpenITI dataset using RNN model

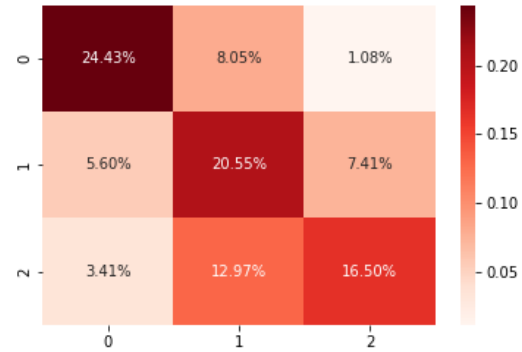


Figure A.9: The confusion matrix for the 3 class setup proposed by Belinkov et al. [5] in OpenITI dataset using RNN model

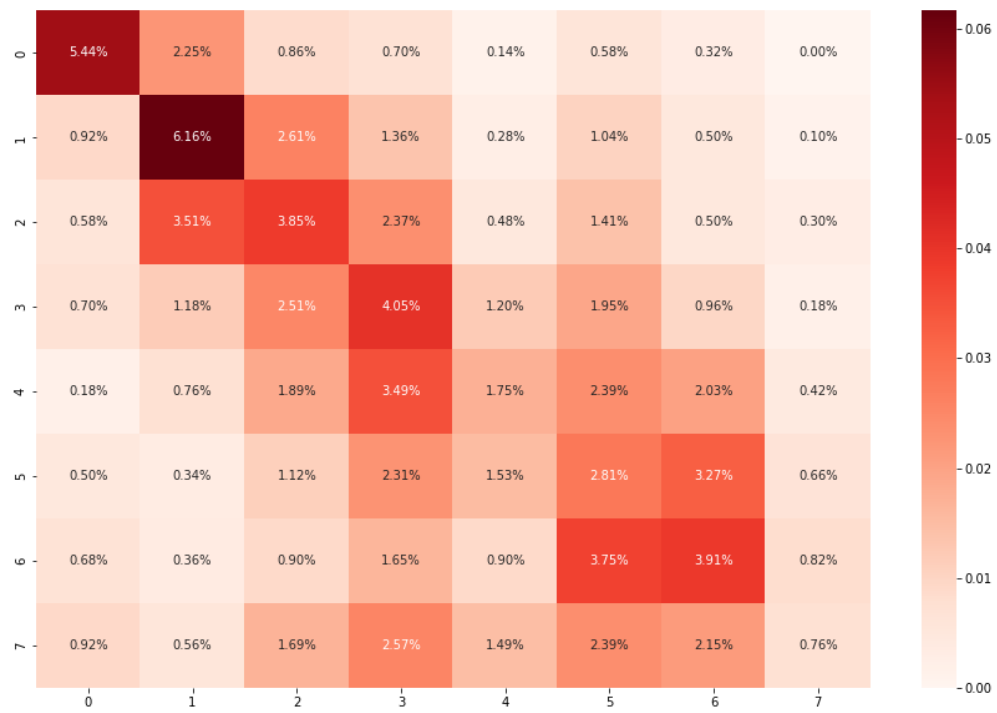


Figure A.10: The confusion matrix for the 200-years periods setup in the OpenITI dataset using RNN model

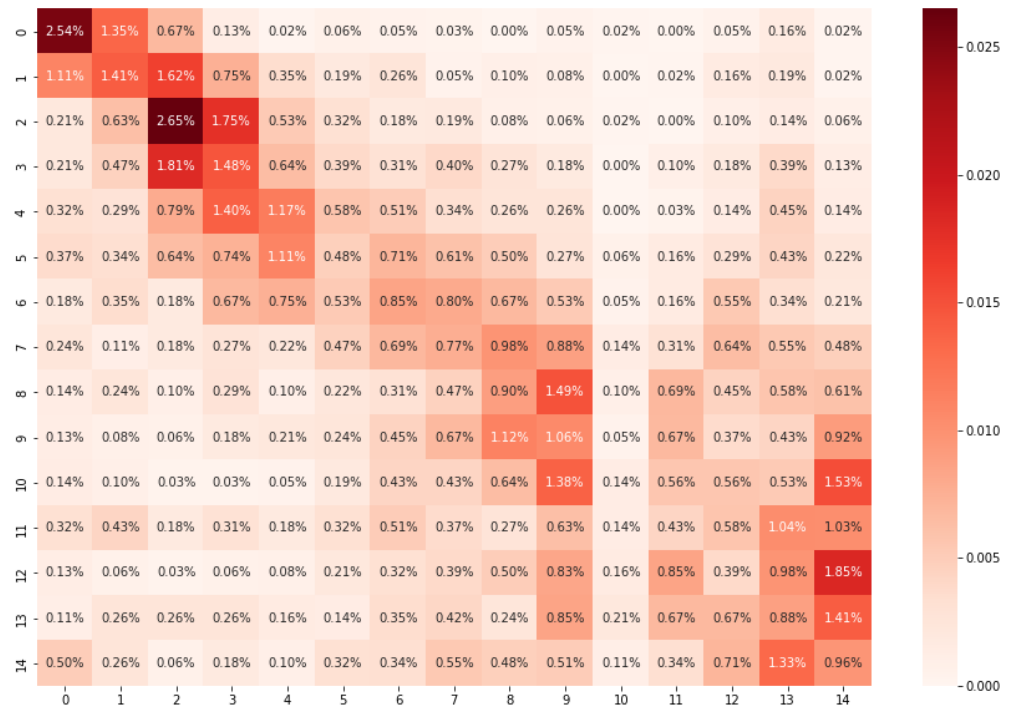


Figure A.11: The confusion matrix for the 100-years periods setup in the OpenITI dataset using RNN model

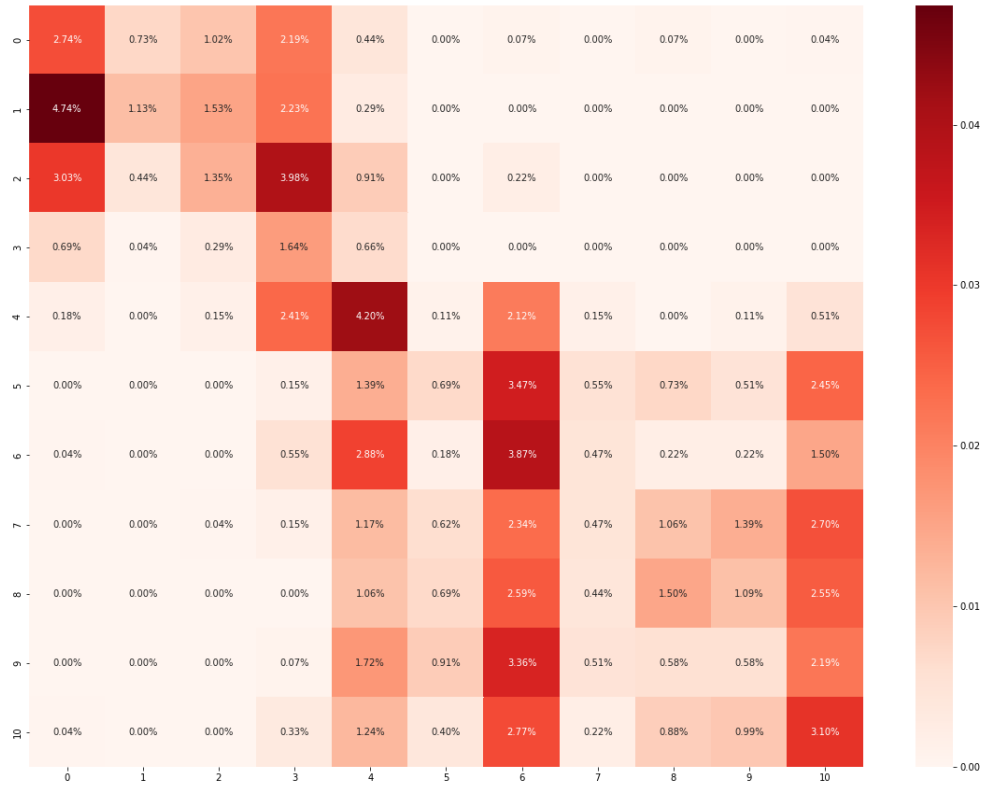


Figure A.12: The confusion matrix for the 11-eras setup in the APCD dataset using RNN model

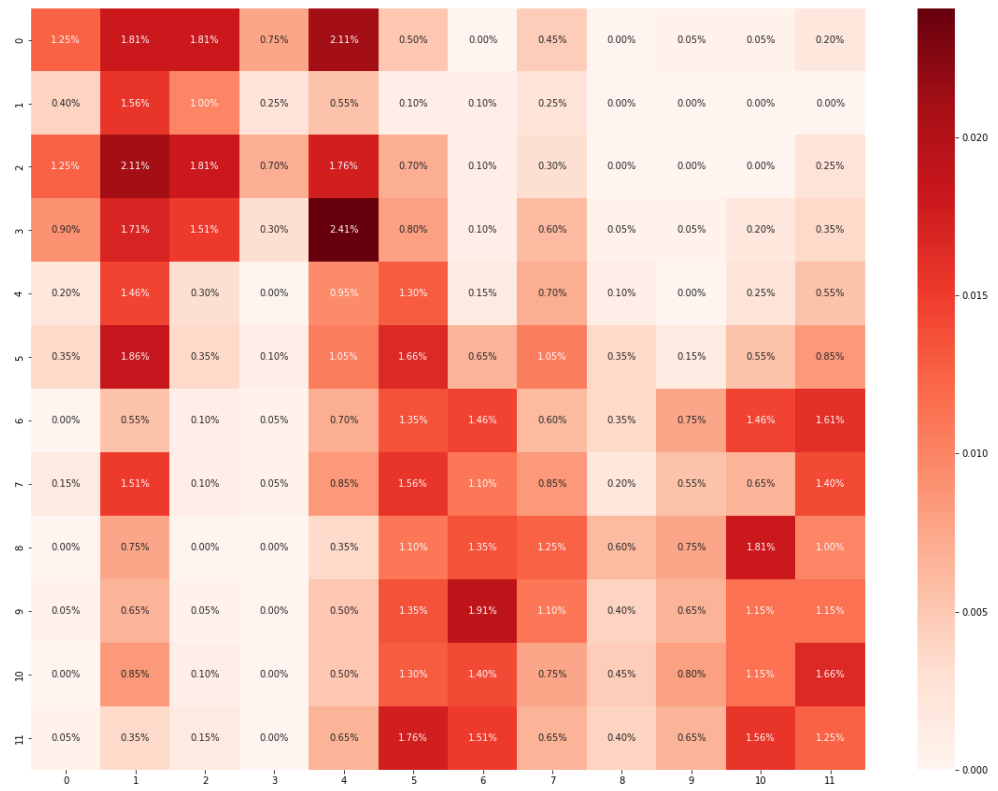


Figure A.13: The confusion matrix for the 12-eras setup in the APCD dataset using RNN model

APPENDIX B

HUMAN EVALUATION

B.1 Human Evaluation Setup

We want to investigate the ability of humans for the temporal periods' classification task. Therefore, we conducted an experiment where a participant classified some texts into five literature eras for OpenITI and APCD datasets. The participants are native Arabic speakers interested in Arab history.

First, we let the participant read samples from different eras for each dataset, considering this step as training. Then the participant classified 20 samples for each dataset and gave feedback for some samples explaining the reason for selecting a specific era. Figure B.1 shows an example of the human evaluation test. The presented sample is from the OpenITI dataset five literature eras setup.

Furthermore, our human evaluation is to look for the problem from another aspect. We were only limited to two participants and 40 samples for the test. Thus, our human evaluation is just for a subjective discussion. We could have an initial impression about the problem to improve it in future work.

من الضأن والمعز والظباء والبقر والنعام وحمر الوحش الشعب الطريق في الجبل أو الانفراج بين الجبلين كرشى أراد أنهم بطانته وموضع سره وأمانته عييتي جماعتي وصحابتي الذين أطلعهم على سري واثق بهم واعتمد عليهم حدثنا حميد عن أنس قال أقبل علينا رسول الله بوجهه حين قام إلى الصلاة قبل أن يكبر فقال أقيموا صفوفكم وتراصوا فإني أراكم من وراء ظهري حدثنا حميد عن أنس أن رسول الله قال انصر أخاك ظالما أو مظلوما فقالوا يا رسول الله هذا ننصره مظلوما فكيف ننصره ظالما قال تكفه عن الظلم حدثنا علي بن حجر قال حدثنا يزيد عن حميد عن أنس بهذا حدثنا

☐ العصر الاسلام

☐ العصر العباسي

☐ عصر الدول والامارات

☐ العصر العثماني

☐ العصر الحديث

سبب اختيار العصر

Figure B.1: Example from the human evaluation test, the sample form the OpenITI dataset.

Table B.1: The human experiment results for five literature eras setup in OpenITI and APCD datasets.

Participants	Dataset	Accuracy	Precision	Recall	F1-Measure
Participant-1	OpenITI	0.300	0.303	0.300	0.290
	APCD	0.700	0.710	0.700	0.686
Participant-2	OpenITI	0.500	0.570	0.500	0.463
	APCD	0.550	0.547	0.550	0.535

B.2 Human Evaluation Results

The first participant has a master degree and he has a personal interested in the Arabic poets and authors biographies. Moreover, he is more curious about authors culture and the environment where they lived. The second participant is a master student whose interested in the poems. The participants was trained on some samples for the five literature eras setup in the OpenITI and APCD datasets.

Table B.1 presents the human test results for both datasets. We can see that participant-1 do well in the APCD dataset. Although the OpenITI results are not good as APCD, it is not random guessing. Participant-2 achieved better result than participant-1 in the OpenITI dataset. Moreover, we found that some of the misclassified samples by the participant are identical to our model. For example, if the era of a given sample is Islamic and the model predicted as Abbasid, the participant does the same.

Figure B.2a shows the confusion matrix for participant-2 result in the OpenITI dataset. The participant correctly predicted all the Islamic era (class-0) samples and most of the modern era (class-4) samples. In addition, the participant was confused between the Islamic era (class-0) and the Abbasid era (class-1).

Furthermore, Figure B.2b presents the confusion matrix for participant-1 result in the APCD dataset. The participant was able to identified all the Pre-Islamic era (class-0) samples. However, unlike OpenITI most of the modern era (class-4) were misclassified as Ottoman (class-3) and Pre-Islamic (class-0) eras.

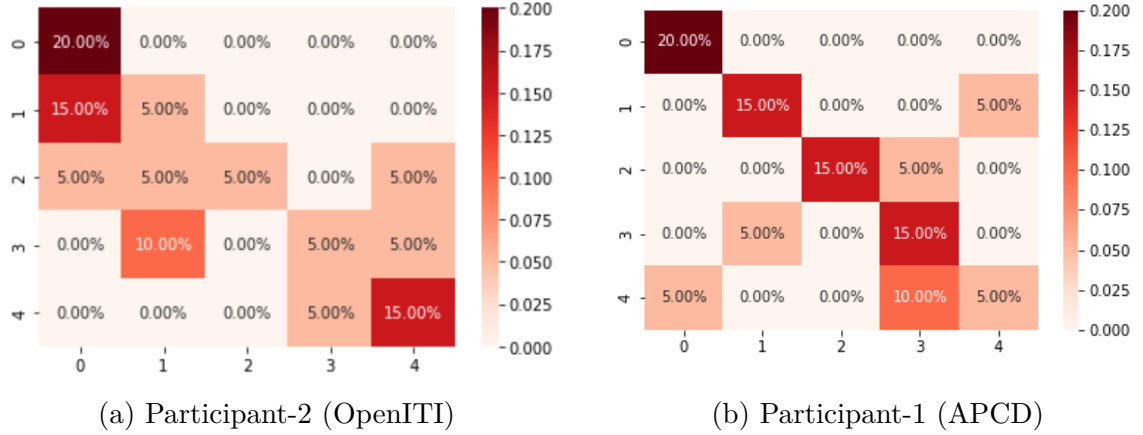


Figure B.2: Confusion matrix for the human experiment results in the five literature eras setup for OpenITI (a) and APCD (b) datasets.

B.2.1 Conclusions Based on Human Evaluation

Table B.2 shows participant-1 feedback for two samples from both datasets. For the OpenITI dataset, the first sample, the participant classified it correctly by finding a name related to that era which is the modern era. However, the second sample was wrongly predicted as *Aldoul wa al-emarar* era. The reason is that *Allmam al-Bukhari* (الامام البخاري) was mentioned; thus the era possibility will be after his death.

Furthermore, in the APCD dataset, the first sample was misclassified as the Ottoman era. The participant mentioned some names, which indicated that he is most likely sure about it. Therefore, we investigated that sample and found it belongs to

Table B.2: Participant-1 feedback for two predicted samples for the OpenITI and APCD datasets.

Dataset	Actual Era	Predicted Era	Feedback
OpenITI	Modern	Modern	ورد في النص ان الرافعي من أهل هذا الزمان اظنه مصطفى صادق الرافعي
	Abbasid	<i>Aldoul wa al-emarāt</i>	هذا بعد الإمام البخاري ممكن يكون في أي عصر بعد البخاري
APCD	Modern	Ottoman	ورد الأزهر والغضنفر في الايات
	Islamic	Islamic	مطلع القصيدة مدح لبني امية

the Ottoman era but is mislabeled in the APCD dataset as the modern era. The second sample was predicted correctly as the Islamic era because the participant noticed that the sample starts with praise for *Bani Umayyah* (part of the Islamic era).

We can see that the person names help to identify some samples. Most of the participants feedback was about some person’s name mentioned in the sample, which help him to predict it correctly. Moreover, even a well-known group or a tribe like *Bani Umayyah* could help to identify the era of the text.

REFERENCES

- [1] N. Alloqah, “الفاظ القتال في الشعر الجاهلي دراسة دلالية,” MS dissertation, Alazhar University, 2011.
- [2] K. Darwish, N. Habash, M. Abbas, H. Al-Khalifa, H. T. Al-Natsheh, H. Bouamor, K. Bouzoubaa, V. Cavalli-Sforza, S. R. El-Beltagy, W. El-Hajj *et al.*, “A panoramic survey of natural language processing in the arab world,” *Communications of the ACM*, vol. 64, no. 4, pp. 72–81, 2021.
- [3] K. Ghoniem, “الأدب العربي الحديث والمعاصر: دراسة في التاريخ والمصطلح,” *Hebron University Research Journal-B (Humanities)-العلوم* (مجلة جامعة الخليل للبحوث-ب), vol. 12, no. 1, pp. 18–34, 2017.
- [4] S. Štajner and M. Zampieri, “Stylistic changes for temporal text classification,” in *International Conference on Text, Speech and Dialogue*. Springer, 2013, pp. 519–526.
- [5] Y. Belinkov, A. Magidow, A. Barrón-Cedeño, A. Shmidman, and M. Romanov, “Studying the history of the arabic language: language technology and a large-scale historical corpus,” *Language Resources and Evaluation*, vol. 53, no. 4, pp. 771–805, 2019.

- [6] M. Orabi, H. El Rifai, and A. Elnagar, “Classical arabic poetry: Classification based on era,” in *Proceedings of IEEE/ACS International Conference on Computer Systems and Applications, AICCSA*. IEEE, 2020, pp. 1–6.
- [7] K. AlSamurai and T. AlShammari, “إحياء الكلمات في الشعر العباسي دراسة تأويلية في,” *Journal of Babylon Center for Humanities Studies*, vol. 4, no. 2, pp. 138–179, 2014.
- [8] F. Chollet, *Deep learning with Python*. Manning, 2018.
- [9] B. Ramsundar and R. B. Zadeh, *TensorFlow for deep learning: from linear regression to reinforcement learning*. ” O’Reilly Media, Inc.”, 2018.
- [10] A. Géron, *Neural networks and deep learning*. O’Reilly, 2018.
- [11] A. M. McMahon and M. April, *Understanding language change*. Cambridge university press, 1994.
- [12] M. Alrabiah, A. Al-Salman, E. Atwell, and N. Alhelewh, “Ksucca a key to exploring arabic historical linguistics,” *International Journal of Computational Linguistics (IJCL)*, vol. 5, no. 2, pp. 27–36, 2014.
- [13] A. S. Daif, *العصر الجاهلي*, ser. تاريخ الادب العربي. Islamic Books, 1982.
- [14] H. Al-Fakhoury, *History of Arabic Literature*. DMC, 1951.
- [15] V. Niculae, M. Zampieri, L. P. Dinu, and A. M. Ciobanu, “Temporal text ranking and automatic dating of texts,” in *Proceedings of the 14th Conference of the*

- European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*, 2014, pp. 17–21.
- [16] G. Tilahun, A. Feuerverger, M. Gervers *et al.*, “Dating medieval english charters,” *The Annals of applied statistics*, vol. 6, no. 4, pp. 1615–1640, 2012.
- [17] Y. Li, D. Genzel, Y. Fujii, and A. C. Papat, “Publication date estimation for printed historical documents using convolutional neural networks,” in *Proceedings of the 3rd international workshop on historical document imaging and processing*, 2015, pp. 99–106.
- [18] M. Alhawarat, M. Hegazi, and A. Hilal, “Processing the text of the holy quran: a text mining study,” *Int. J. Adv. Comput. Sci. Appl*, vol. 6, no. 2, pp. 262–267, 2015.
- [19] G. A. Abandah, A. Graves, B. Al-Shagoor, A. Arabiyat, F. Jamour, and M. Al-Tae, “Automatic diacritization of arabic text using recurrent neural networks,” *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 18, no. 2, pp. 183–197, 2015.
- [20] L. Lulu and A. Elnagar, “Automatic arabic dialect classification using deep learning models,” *Procedia computer science*, vol. 142, pp. 262–269, 2018.
- [21] I. A. Farha and W. Magdy, “Mazajak: An online arabic sentiment analyser,” in *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, 2019, pp. 192–198.

- [22] S. R. Joseph, H. Hlomani, K. Letsholo, F. Kaniwa, and K. Sedimo, “Natural language processing: A review,” *Natural Language Processing: A Review*, vol. 6, pp. 207–210, 2016.
- [23] A. Reshamwala, D. Mishra, and P. Pawar, “Review on natural language processing,” *IRACST Engineering Science and Technology: An International Journal (ESTIJ)*, vol. 3, no. 1, pp. 113–116, 2013.
- [24] M. Zampieri, A. M. Ciobanu, V. Niculae, and L. P. Dinu, “Ambra: A ranking approach to temporal text classification,” in *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, 2015, pp. 851–855.
- [25] T. Szymanski and G. Lynch, “Ucd: Diachronic text classification with character, word, and syntactic n-grams,” in *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015). United States*, 2015.
- [26] M. Alhawarat and A. O. Aseeri, “A superior arabic text categorization deep model (satcdm),” *IEEE Access*, vol. 8, pp. 24 653–24 661, 2020.
- [27] M. Abbas, M. Lichouri, and A. Zeggada, “Classification of arabic poems: from the 5th to the 15th century,” in *International Conference on Image Analysis and Processing*. Springer, 2019, pp. 179–186.
- [28] F. Jong, H. Rode, and D. Hiemstra, “Temporal language models for the disclosure of historical text,” *Cognitive Science*, 2005.

- [29] A. Garcia-Fernandez, A.-L. Ligozat, M. Dinarelli, and D. Bernhard, “When was it written? automatically determining publication dates,” in *International symposium on string processing and information retrieval*. Springer, 2011, pp. 221–236.
- [30] A. S. Daif, *History of Arabic Literature*. Islamic Books, 1982, vol. 1-10.
- [31] دار المعرفة للطباعة والنشر , 2000. تاريخ الادب العربي , أحمد حسن الزيات
- [32] O. Popescu and C. Strapparava, “Semeval 2015, task 7: Diachronic text evaluation,” in *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, 2015, pp. 870–878.
- [33] —, “Behind the times: Detecting epoch changes using large corpora,” in *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, 2013, pp. 347–355.
- [34] A. Dalli and Y. Wilks, “Automatic dating of documents and temporal text classification,” in *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, 2006, pp. 17–22.
- [35] A. Kumar, M. Lease, and J. Baldridge, “Supervised language modeling for temporal resolution of texts,” in *Proceedings of the 20th ACM international conference on Information and knowledge management*, 2011, pp. 2069–2072.
- [36] C. Colgrove, S. Chang, and P. Y. Watanaprakornkul, “Literary period classification,” 2010.

- [37] J.-B. Michel, Y. K. Shen, A. P. Aiden, A. Veres, M. K. Gray, J. P. Pickett, D. Hoiberg, D. Clancy, P. Norvig, J. Orwant *et al.*, “Quantitative analysis of culture using millions of digitized books,” *science*, vol. 331, no. 6014, pp. 176–182, 2011.
- [38] Y. Lin, J.-B. Michel, E. A. Lieberman, J. Orwant, W. Brockman, and S. Petrov, “Syntactic annotations for the google books ngram corpus,” in *Proceedings of the ACL 2012 system demonstrations*, 2012, pp. 169–174.
- [39] R. Mihalcea and V. Nastase, “Word epoch disambiguation: Finding how words change over time,” in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, vol. 2, 2012, pp. 259–263.
- [40] D. T. Wijaya and R. Yeniterzi, “Understanding semantic change of words over centuries,” in *Proceedings of the 2011 international workshop on DETecting and Exploiting Cultural diversiTy on the social web*, 2011, pp. 35–40.
- [41] H. Salaberri, I. Salaberri, O. Arregi, and B. Zapiain, “Ixagroupehudiatic: A multiple approach system towards the diachronic evaluation of texts,” in *Proceedings of the 9th international workshop on semantic evaluation (SEMEVAL 2015)*, 2015, pp. 840–845.
- [42] L. Frermann and M. Lapata, “A bayesian model of diachronic meaning change,” *Transactions of the Association for Computational Linguistics*, vol. 4, pp. 31–45, 2016.

- [43] N. Kanhabua and K. Nørvåg, “Improving temporal language models for determining time of non-timestamped documents,” in *International conference on theory and practice of digital libraries*. Springer, 2008, pp. 358–370.
- [44] M. Romanov and M. Seydi, “Openiti: a machine-readable corpus of islamicate texts,” 2019.
- [45] M. Zampieri and M. Becker, “Colonia: Corpus of historical portuguese,” *ZSM Studien, Special Volume on Non-Standard Data Sources in Corpus-Based Research*, vol. 5, pp. 69–76, 2013.
- [46] A. M. Ciobanu, A. Dinu, L. P. Dinu, V. Niculae, and O.-M. Şulea, “Temporal classification for historical romanian texts,” in *Proceedings of the 7th workshop on language technology for cultural heritage, social sciences, and humanities*, 2013, pp. 102–106.
- [47] C. Grouin, D. Forest, P. Paroubek, and P. Zweigenbaum, “Présentation et résultats du défi fouille de texte deft2011 quand un article de presse a-t-il été écrit? à quel article scientifique correspond ce résumé?” *Actes du septième DÉfi Fouille de Textes*, 2011.
- [48] N. Kanhabua and K. Nørvåg, “Using temporal language models for document dating,” in *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 2009, pp. 738–741.

- [49] E. Porat, M. Koppel, and A. Shmidman, “Identification of parallel passages across a large hebrew/aramaic corpus,” *Journal of Data Mining & Digital Humanities*, 2018.
- [50] F. Pedregosa, E. Cauvet, G. Varoquaux, C. Pallier, B. Thirion, and A. Gramfort, “Learning to rank from medical imaging data,” in *International Workshop on Machine Learning in Medical Imaging*. Springer, 2012, pp. 234–241.
- [51] M. S. Al-shaibani, Z. Alyafeai, and I. Ahmad, “Meter classification of arabic poems using deep bidirectional recurrent neural networks,” *Pattern Recognition Letters*, vol. 136, pp. 1–7, 2020.
- [52] W. A. Yousef, O. M. Ibrahime, T. M. Madbouly, M. A. Mahmoud, A. H. El-Kassas, A. O. Hassan, and A. R. Albohy, “Poem comprehensive dataset (pcd),” July 2018. [Online]. Available: <https://hci-lab.github.io/ArabicPoetry-1-Private/#PCD>
- [53] D. Jurafsky and H. Martin, *Speech and Language Processing (3rd edition draft)*. Stanford University, 2020.
- [54] N. Alnasri, “المعلقات,” *مِيقَاتُ الْحَجِّ*, vol. 5, no. 10, pp. 234–263, 1998.
- [55] K. Darwish and H. Mubarak, “Farasa: A new fast and accurate arabic word segmenter,” in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC’16)*, 2016, pp. 1070–1074.

- [56] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, vol. 9, 2015.
- [57] R. Gopalakrishnan and A. Venkateswarlu, *Machine Learning for Mobile: Practical guide to building intelligent mobile applications powered by machine learning*. Packt Publishing Ltd, 2018.
- [58] F. J. Pineda, “Generalization of back propagation to recurrent neural networks,” *Physical review letters*, vol. 59, no. 19, p. 2229, 1987.
- [59] T. Stérin, N. Farrugia, and V. Gripon, “An intrinsic difference between vanilla rnns and gru models,” *COGNITIVE 2017*, p. 84, 2017.
- [60] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [61] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, “On the properties of neural machine translation: Encoder-decoder approaches,” *arXiv preprint arXiv:1409.1259*, 2014.
- [62] M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks,” *IEEE transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.
- [63] J. Lee, K. Cho, and T. Hofmann, “Fully character-level neural machine translation without explicit segmentation,” *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 365–378, 2017.

- [64] F. Özyurt, E. Sert, and D. Avci, “An expert system for brain tumor detection: Fuzzy c-means with super resolution and convolutional neural network with extreme learning machine,” *Medical hypotheses*, vol. 134, p. 109433, 2020.
- [65] M. Hossin and M. N. Sulaiman, “A review on evaluation metrics for data classification evaluations,” *International journal of data mining & knowledge management process*, vol. 5, no. 2, p. 1, 2015.
- [66] T. G. Dietterich, “Approximate statistical tests for comparing supervised classification learning algorithms,” *Neural computation*, vol. 10, no. 7, pp. 1895–1923, 1998.

VITAE

- Name: Zainab Hathloul Alhathloul
- Nationality: Saudi Arabian
- Date of Birth: 16-05-1990
- Email: *zainab-hathloul@hotmail.com*
- Permenant Address: 7419 Jaber Ibn Abdullah Al Qalbt Street Al-Khobar, 34716 - 3094, Saudi Arabia

Publications

- Alazba, A., Alturayef, N. S., Alturaief, N., & Alhathloul, Z. (2020). Saudi Stock Market Sentiment Analysis using Twitter Data. In KDIR (pp. 36-47).
- Alturayef, N., Alturaief, N., & Alhathloul, Z. (2020). DeepScratch: scratch programming language extension for deep learning education. International Journal of Advanced Computer Science and Applications, 11(7), 642-650.

Experience

- Android mobile application trainer Codes for Girls | 23 Jun 2019 - 27 Jun 2019.

- Mobile application team leader Ministry of Civil Service| 2016 - 2018.
- Mobile application developer Ministry of Civil Service| 2014 - 2016.

Education

- Bachelor's in Computer Science Imam Muhammad ibn Saud University, Riyadh
| 2008 - 2014 second class honor.