Universidad Politécnica de Valencia

Departamento de Sistemas Informáticos y Computación

Reconocimiento de Formas e Inteligencia Artificial

Arabic Named Entity Recognition

The accepted dissertation
of

Yassine Benajiba

in partial fulfillment of the requirements
to obtain the academic degree of
Doctor en Informática

under the supervision of

Dr. Paolo Rosso
Universidad Politécnica de Valencia

Valencia, Spain
May 2009

to my language,

to my parents, my sisters, my brother and Fatma for their love, care and constant support,

to Oreste (Paolo's son), my nephew Taha and my niece Anissa for reminding me that I should play and smile even in tough times,

with a sincere hope that this will make you proud of me.

Acknowledgments

I wish to thank **Dr. Paolo Rosso**, first for supervising this thesis, second for the innumerable meetings which we have held to discuss the ideas and sketch plannings. Third for his continuous encouragements during all the time that I have been under his priceless supervision. Last but not least, I would also like to thank him for the great travelling tutor and ski instructor that he has been to me.

I am really grateful to **Dr. Imed Zitouni** who showed me the industrial viewpoint of the work that I am doing in my Ph.D. During the 6 months of my internship with him at *IBM T. J. Watson Research Center* he gave me my first opportunity to explore more challenging tasks and do my first steps in the world of "very large data".

I am deeply indebted to **Dr. Mona Diab** whose help and guidance made a lot of unthinkable achievements possible. During my internship with her in the *Center of Computational Learning Systems* at *Columbia University*, she has made each and every minute profitable and enjoyable. After the internship she still has dedicated several hours on *Skype* to share her ideas and opinions on the ongoing work.

I would like to express my gratitude to **Dr. Manuel Montes-y-Gómez** for giving me the opportunity to have an internship under his supervision in the *Language Technologies Lab* at the *National Institute of Astrophysics, Optics and Electronics* (INAOE) in Puebla, Mexico. I had pleasure learning about all the ideas they are exploring and was thrilled to investigate the compatibility of some of these ideas with the Arabic language. I have furthermore to thank **Dr. Luis Villaseñor Pineda** for the precious time and interest that he had dedicated to my research work.

Also **Dr. Vasudeva Varma** from the *International Institute of Information Technology* at Hyderabad. I want to thank him and his Ph.D students for making my stay in India comfortable and for having generously shared a lot of information about their research works and their scientific feedback on ours.

I also thank my colleagues, the nearly Dr., **Davide Buscaldi** and, the already Dr., **David Pinto** for tutoring me at the beginning of my PhD career and for all the discussions that we had at lunch and coffee breaks. **Piedachu Peris García** for being an eye opener to me and for her constant support.

Finally, I would like to thank all my friends, my family and especially my beloved **Fatma**.

This PhD. thesis has been supported by the 3-year grant offered by "Agencia Española de Cooperación Internacional". The research work of this PhD thesis has been developed in the context of the following projects:

- MCyT TIN2006-15265-C06-04: TEXT-MESS, "Intelligent, Interactive and Multilingual Text Mining based on Human Language Technology" ("Minería de Textos Inteligente, Interactiva y Multilingue basada en Tecnología del Lenguaje Humano", http://gplsi.dlsi.ua.es/text-mess/
- PCI-AECI A/7067/06 and PCI-AECI A/010317/07 : two bilateral (Spain-Morocco) AECI funding projects to develop an Arabic Question Answering system.

The committee for this Ph.D. thesis has been composed by:
Dr. Felisa Verdejo (U.N.E.D., Spain)
Dr. Horacio Rodríguez (Universidad Politécnica de Cataluña, Spain)
Dr. Mona Diab (Columbia University, USA)
Dr. Imed Zitouni (IBM T.J. Watson Research Center, USA)
Dr. Encarnación Segarra (Universidad Politécnica de Valencia, Spain)

Abstract

This Ph.D. thesis describes the investigations we carried out in order to determine the appropriate approach to build an efficient and robust *Arabic Named Entity Recognition* system. Such a system would have the ability to identify and classify the Named Entities within an open-domain Arabic text.

The Named Entity Recognition (NER) task helps other Natural Language Processing approaches (e.g. Information Retrieval, Question Answering, Machine Translation, etc.) achieve a higher performance thanks to the significant information added to the text. In the literature, many research works report the adequate approaches which can be used to build an NER system for a specific language or from a language-independent perspective. Yet, very few research works which investigate the task for the Arabic language have been published.

The Arabic language has a special orthography and a complex morphology which bring new challenges to the NER task to be investigated. A complete investigation of Arabic NER would report the technique which helps achieve a high performance, as well as giving a detailed error analysis and results discussion so as to make the study beneficial to the research community. This thesis work aims at satisfying this specific need. In order to achieve that goal we have:

1. Studied the different aspects of the Arabic language which are related to the NER task;

2. Studied the state-of-art of the NER task;

3. Conducted a comparative study among the most successful Machine Learning approaches on the NER task;

4. Carried out a multi-classifier approach where each classifier deals with only one NE class and uses the appropriate Machine Learning approach and feature-set for the concerned class.

We have evaluated our experiments on different nine data-sets of different genres (newswire, broadcast news, Arabic Treebank and weblogs). Our findings point out that the multi-classifier yields the best results.

الملخص

ان الموضوع الَأسَاسي لهذه الرسَالة للّدكتورَاه هو تقديم الَأبحَاث التي قمنَا بهَا لبنَاء نظَام دقيق و متين **لتعريف أسمَاء العلم** . الدور الذي تلعبه هذه الَأنظمة هو تحديد وتصنيف أسمَاء العلم الموجودة بنص عربي من اي نوع.

عملية **تعريف أسمَاء العلم** تسَاعد عمليَات أخرَى للّمعَالجة الَالية للّغة (مثلَا: استخرَاج المعلومَات، أنظمة اجَابة الَاسئِلة، الترجمة الَالية، الخ.) علَى الحصول علَى أدَاء أحسن بفضل المعلومَات الهَامة التي تضيفهَا للّنص. الَأبحَاث العلمية التي تعَالج تعريف أسمَاء العلم للّغة محدّة أو بَاستعمَال طرق تصلح لكل اللّغَات. مع ذلك ، فَان الَابحَاث المنشورة التي تعَالج الموضوع نفسه مستهدفة اللّغة العربية تعد علَى رؤُوس الَأصَابع.

اللّغة العربية لهَا حروفهَا الخَاصة ومورفولوجيتهَا التي تعتبر اكتر تعقيدَا بَالنسبة للّغَات اخرَى لمَّا يخلق لعملية تعريف أسمَاء العلم تحديَات جديدة لم تعرف للّغَات الخَرَى. درَاسة كَاملة لعملية تعريف اسمَاء العلم بَالنصوص العربية تستوجب تحديد التقنية التي تمكّن من الحصول علَى أحسن ادَاء ومنَاقشة النتَائِج المحصل عليهَا و تحليل أخطَاء النظَام حتَى تكون الدرَاسة مفيدة لكَافة البَاحثين. البحث الذي نقدم في هذه الوثيقة يهدف الَى تحقيق هذَا الغرض، ولضلك قمنَا ب:

١. درَاسة جميع جوَانب اللّغة العربية التي تخص عملية تعريف أسمَاء العلم ؛

٢. درَاسة المرَاجع التي تذكر النتَائِج المحصل عليهَا في عملية تعريف أسمَاء العلم ؛

٣. مقَارنة بين تقنيَات **تعليم الَالة** الَأكثر نجَاحَا في عملية تعريف اسمَاء العلم ؛

٤. تجَارب تستعمل مصنفين عدة فيهَا كل مصنف يختص بصنف وَاحد من أسمَاء العلم ويتم مزج مخَارج هؤُلَاء المصنفين لَاستنبَاط مخرج وَاحد.

ولقد قمنَا بتقيّم تجَاربنَا علَى تسعة قوَاعد معطيَات مختلفة ونتَائِجنَا تبيّن أن التقنية التي تستند مزج عدة مصنفين هي التي أعطت أحسن النتَائِج.

Resumen

En esta tesis doctoral se describen las investigaciones realizadas con el objetivo de determinar las mejores técnicas para construir un *Reconocedor de Entidades Nombradas en Árabe*. Tal sistema tendría la habilidad de identificar y clasificar las entidades nombradas que se encuentran en un texto árabe de dominio abierto.

La tarea de Reconocimiento de Entidades Nombradas (REN) ayuda a otras tareas de Procesamiento del Lenguaje Natural (por ejemplo, la Recuperación de Información, la Búsqueda de Respuestas, la Traducción Automática, etc.) a lograr mejores resultados gracias al enriquecimiento que añade al texto. En la literatura existen diversos trabajos que investigan la tarea de REN para un idioma específico o desde una perspectiva independiente del lenguaje. Sin embargo, hasta el momento, se han publicado muy pocos trabajos que estudien dicha tarea para el árabe.

El árabe tiene una ortografía especial y una morfología compleja, estos aspectos aportan nuevos desafíos para la investigación en la tarea de REN. Una investigación completa del REN para el árabe no solo aportaría las técnicas necesarias para conseguir un alto rendimiento, sino que también proporcionaría un análisis de los errores y una discusión sobre los resultados que benefician a la comunidad de investigadores del REN. El objetivo principal de esta tesis es satisfacer esa necesidad. Para ello hemos:

1. Elaborado un estudio de los diferentes aspectos del árabe relacionados con dicha tarea;

2. Analizado el estado del arte del REN;

3. Llevado a cabo una comparativa de los resultados obtenidos por diferentes técnicas de aprendizaje automático;

4. Desarrollado un método basado en la combinación de diferentes clasificadores, donde cada clasificador trata con una sola clase de entidades nombradas y emplea el conjunto de características y la técnica de aprendizaje automático más adecuados para la clase de entidades nombradas en cuestión.

Nuestros experimentos han sido evaluados sobre nueve conjuntos de test de diferentes tipos (artículos de periódico, noticias transcritas, documentos del Arabic Treebank y weblogs). Nuestros resultados muestran que la técnica basada en varios clasificadores ayuda a obtener los mejores resultados en todos estos tipos de documentos.

Resum

En aquesta tesi doctoral es descriuen les investigacions realitzades amb l'objectiu de determinar les millors técniques per a construir un *Reconeixedor d'Entitats Nomenades en Árab*. Tal sistema tindria l'habilitat d'identificar i classificar les entitats nomenades que es troben en un text árab de domini qualsevol.

La tasca de Reconeixement d'Entitats Nomenades (REN) ajuda a altres tasques de Processament del Llenguatge Natural (per exemple, Recuperació d'Informació Recerca de Respostes, Traducció Automática, etc.) a assolir millors resultats grácies a l'enriquiment que afegeix al text. En la literatura existeixen diversos treballs que investiguen la tasca de REN per a un idioma específic o des d'una perspectiva independiente del llenguatge. No obstant aixó, fins al moment, s'han publicat molt pocs treballs que investiguen aquesta tasca per a l'árab.

L'árab té una ortografia especial i una morfologia complexa que aporten nous desafiaments per a investigar en la tasca de REN. Una investigació completa del REN per a l'árab aportaria la técnica necessária per a aconseguir un alt rendiment, ó també proporcionaria una análisi dels errors i una discussió sobre els resultats, per a beneficiar amb tal estudi a la comunitat d'investigadors del REN. L'objectiu principal d'aquesta tesi és satisfer aqueixa necessitat. Per a aixó hem:

1. Elaborat un estudi dels diferents aspectes de l'árab relacionats amb aquesta tasca;

2. Analitzat l'estat de l'art del REN;

3. Portat a terme una comparativa dels resultats obtinguts per diferents técniques d'aprenentatge automátic;

4. Desenvolupat un métode basat en la combinació de diferents classificadors, on cada classificador tracta amb una sola classe d'entitats nomenades i empra el conjunt de característiques i la técnica d'aprenentatge automátic més adequats per a la classe d'entitats nomenades en qestió.

Els nostres experiments han estat avaluats sobre nou conjunts de test de diferents tipus (articles de diari, notícies transcrites, documents del Arabic Treebank i weblogs). Els nostres resultats mostren que la técnica basada en diversos classificadors ajuda a obtenir els millors resultats en tots els tipus de dades.

Contents

Title page i
Acknowledgments iii
Abstract v
Table of contents xiii
List of tables xv
List of figures xvii

1 Introduction **3**

2 Peculiarities and Challenges of the Arabic NLP **9**
2.1 The Arabic Language: Scripture, Encodings and Morphology 12
2.2 Hardness of Information Retrieval and Question Answering 21
2.3 Hardness of Arabic Named Entity Recognition 34
2.4 Concluding Remarks 35

3 The Named Entity Recognition Task **37**
3.1 How Important is the Named Entity Recognition Task? 40
3.2 The Named Entity Recognition Task: Definitions 44
3.3 The Named Entity Recognition Task State-of-Art 50
3.4 Concluding Remarks 58

4 Statistical Modeling Approaches for Named Entity Recognition **61**
4.1 Maximum Entropy 64
4.2 Conditional Random Fields 68
4.3 Support Vector Machines 72
4.4 Concluding Remarks 83

5 A Maximum Entropy based Approach: 1 and 2 steps **85**
5.1 The ANERcorp 87
5.2 1-step ME-based Approach 93
5.3 The 2-step Approach 98
5.4 Concluding Remarks 103

6 ME, CRFs and SVMs with Different Data-sets 107
6.1 Features . . . 109
6.2 Data . . . 114
6.3 Experiments and Results . . . 116
6.4 Results Discussion and Error Analysis . . . 119
6.5 Concluding Remarks . . . 126

7 A Classifiers Combination Approach 129
7.1 Features . . . 131
7.2 Classic and Fuzzy Borda Voting Scheme . . . 133
7.3 Data . . . 138
7.4 Experiments and Results . . . 139
7.5 Results Discussion and Error Analysis . . . 146
7.6 Concluding Remarks . . . 150

8 Using Large External Resources 153
8.1 Introduction . . . 154
8.2 Extracting Feature from Aligned Corpus . . . 156
8.3 Experiments and Results . . . 160
8.4 Results Discussion and Error Analysis . . . 163
8.5 Concluding Remarks . . . 165

9 Conclusions 167
9.1 Findings and Research Directions . . . 169
9.2 Thesis Contributions . . . 171
9.3 Further Challenges . . . 172

Bibliography 175

A Bayesian Networks 187

List of tables

2.1 Ranking of the 5 top living Semitic features according to number of speakers . . . 10
2.2 Corpora description . . . 17
2.3 Results obtained for Complexity, Variety and Kullback-Leibler distance for each corpus . . . 21

3.1 Percentage of questions in CLEF 2004 and 2005 containing NEs per type 41
3.2 Size of corpora used in CoNLL 2002 and 2003 evaluations . . . 48

4.1 ME model probabilities distribution in case of absence of all information for a vegetable v . . . 65
4.2 ME model probability distribution in case we know that the vegetable v is orange . . . 65
4.3 IOB1, IOB2, IOE1 and IOE2 Base Phrase annotation schemes . . . 80
4.4 Illustrating example of the Beginning/End annotation example . . . 82

5.1 Ratio of sources used to build ANERcorp . . . 88
5.2 Ratio of NEs per class . . . 89
5.3 Number of times each class was assigned to the word *union* in ANERcorp . . . 89
5.4 Unigrams and bigrams frequency thresholds for each class . . . 90
5.5 Matrix of probabilities of a word of class c_i (rows) appears after a word of class c_j (columns) . . . 91
5.6 Baseline results . . . 96
5.7 ANERsys results without using external resources . . . 96
5.8 ANERsys results using external resources . . . 96
5.9 ANERsys results using external resources in case a credit is gained when a single token of a multi-word NE is tagged correctly . . . 97
5.10 ANERsys: 2-step approach results . . . 101
5.11 Siraj (Sakhr) results . . . 101
5.12 Evaluation of the first step of the system . . . 102
5.13 Evaluation of the second step of the system . . . 102

6.1 Desciption of MADA morphological features 112
6.2 Desciption of MADA morphological features 113
6.3 Characteristics of ANERcorp 2.0 and ACE 2003, 2004 and 2005 data 116
6.4 Parameter setting experiments: Comparison among different window sizes, and the impact of tokenization on the NER task 117
6.5 Features ranked according to their impact 119
6.6 Best obtained results for each corpus. Best obtained results for each data-set are in bold characters 121
6.7 Overall and per class results obtained for the 2003 BN data-set . . . 125

7.1 Experts rankings 134
7.2 Experts rankings and CBVS result ranking 136
7.3 Experts rankings and weights 136
7.4 Experts rankings and CBVS result ranking 138
7.5 Statistics of ACE 2003, 2004 and 2005 data 139
7.6 Illustrating example of the difference between the 3-way and 4-way classes annotations 141
7.7 F-measure Results using 3-way vs. 4-way class annotations using SVMs 141
7.8 Ranked features according to FBVS using SVMs for each NE class . . 144
7.9 Ranked features according to FBVS using CRFs for each NE class . . 145
7.10 Final Results Obtained with selected features contrasted against all features combined 147
7.11 Number of features and ML approach used to obtain the best results 147

8.1 Number of NEs per class in the Arabic part of the parallel corpus annotated by propagation from English 160
8.2 Individual impact of the features extracted from the hand-aligned parallel data using SVMs. 162
8.3 Final results Obtained with selected features contrasted against all features combined 163
8.4 Number of features and ML approach used to obtain the best results 164

List of figures

2.1 Example of Arabic text 12
2.2 Buckwalter mapping table 14
2.3 An example of Arabic language derivation 14
2.4 An example of Arabic words composition 15
2.5 The word Iraq with different affixes 16
2.6 The words frequency distribution and the Zipf's ideal curve for corpus 1 21
2.7 The words frequency distribution and the Zipf's ideal curve for corpus 2 22
2.8 The words frequency distribution and the Zipf's ideal curve for corpus 3 22
2.9 The words frequency distribution and the the Zipf's ideal curve for corpus 4 23
2.10 Generic Architecture of our Arabic Question answering system, ArabiQA 27
2.11 The JIRS architecture 28
2.12 An example to illustrate the performance of the Density Distance model (an English translation is given in between parenthesis) 29
2.13 Comparison of Coverage and Redundancy of JIRS over both light-stemmed and non-stemmed Arabic corpora 30
2.14 Illustrating example of the input and output of an NER system . . . 31
2.15 Illustrating example of the Answer Extraction module's performance steps 33
2.16 Illustrating example of an NE and non-NE starting with the same characters 34
2.17 Illustrating example of an NE appearing in different contexts because of the complex morphology of the Arabic language 35

3.1 NEs quantity of information in comparison with Part-of-Speech word categories 39
3.2 MUC-6 named entity annotation sample 46
3.3 An extract of an IOB2 tagged corpus 47
3.4 Character-level HMM. c are the character observation and s are the entity types 55

4.1 Illustrating example of a CRFs graph. A clique is circled 69
4.2 Illustrating example of the label-bias problem 69
4.3 Illustrating figure of linear hyperplane 73
4.4 Illustrating figure of linear hyperplane 74
4.5 Two different regions boundaries 75
4.6 Linear separating hyperplane. Support Vectors are circled 77
4.7 Non-linearly separable problem with linear hyperplane. Support Vectors are circled 78
4.8 Illustrating example of Arabic tokenization characters annotation . . 80

5.1 An extract of ANERcorp 88
5.2 Four examples of NEs preceeded by a nationality 92
5.3 The basic architecture of the first version of ANERsys 94
5.4 Two illustrating examples of ANERsys error in tagging multi-word NEs. Translation of Example 1:"pointing out the *Kurdistan Labor Party*". Translation of Example 2:"The Tunisian president *Zine El Abidine Ben Ali*" 98
5.5 The generic architecture of the 2-step ANERsys 100

6.1 Results per approach and number of features for the ACE 2003 (Broadcast News genre) data 120
6.2 Results per approach and number of features for the ACE 2003 (Newswire genre) data 120
6.3 Results per approach and number of features for the ACE 2005 (Weblogs genre) data 121
6.4 Examples of NEs which were missed by the ME-based module and captured by SVMs and CRFs based ones 125
6.5 Per class results when each time the best N features are selected . . 126

7.1 Illustrating example for classifiers conflict 142

8.1 Annotation and projection steps 157

A.1 Illustrating example of a Bayesian Network: the vegetables case study 187

List of Acronyms

ACE	Automatic Content Extraction
AdaBoost	Adaptive Boosting
AE	Answer Extraction
ANERcorp	Arabic Named Entity Recognition corpora
ANERsys	Arabic Named Entity Recognition system
ATB	Arabic TreeBank
BN	Broadcat News
BNs	Bayesian Networks
BPC	Base Phrase Chunking
CG	Conjugate-Gradient
CLEF	Cross Language Evaluation Forum
CoNLL	Conference on Computational Natural Language Learning
Coref	Coreference Resolution
CRFs	Conditional Random Fields
DARPA	Defense Advanced Research Projects Agency
EDR	Entity Detection and Recognition
EDT	Entity Detection and Tracking
EM	Expectation-Maximization
FAC	Facility
FB	Forward-Backward
GIS	Generalized Iteratuve Scaling
GPE	Geo-Political Entity
HMM	Hidden Markov Models
IR	Information Retrieval
LOC	Location
MADA	Morphological Analysis and Dismabiguation for Arabic
MD	Mention Detection

ME	Maximum Entropy
MEMM	Maximum Entropy Markov Models
MISC	Miscellaneous
ML	Machine Learning
MSA	Modern Standard Arabic
MUC-6	The 6th Message Understanding Conference
NE	Named Entity
NER	Named Entity Recognition
NLP	Natural Language Processing
NW	News Wire
OOV	Out Of Vocabulary
ORG	Organization
PER	Person
POS	Part-Of-Speech
PR	Passage Retrieval
QA	Question Answering
QP	Quadratic Programing
RRM	Robust Risk Minimization
Snow	Sparse Network Of Winnows
SV	Support Vector
SVMs	Support Vector Machines
TBL	Transformation-Based Learning
TREC	Text REtrieval Conferences
VEH	Vehicle
WEA	Weapon
WL	WebLogs
WSJ	Wall Street Journal

Chapter 1

Introduction

The Named Entity Recognition (NER) task consists of identifying and classifying the Named Entities (NEs) within an open domain text. For instance, let us consider the following sentence:

> Human Rights Watch accuses in a report the army and police in Nigeria of executing more than 90 people.

An accurate NER system would extract two NEs: (i) "Human Rights Watch" as an organization; and (ii) "Nigeria" as a location.

Some Natural Language Processing (NLP) applications might use the output of the NER system to enhance their performance because it is much richer in information than the raw text. For instance, if we consider an application which attempts to translate the sentence which we have given in the previous example into Arabic, the translation of "Human Rights Watch" without considering that it is the name of an organization would be "مرَاقبة حقوق الَانسَان" (in Buckwalter tranliteration[1] "mrAqbp Hqwq AlAnsan"), whereas the real translation is a character-based transliteration of the words which sounds like English when read in Arabic, i.e. "هيومن رَايتس وَاتش" ("hywmn rAyts wAt$"). Other examples are abound which show that NEs need to be handled differently for a good translation. Question Answering (QA) is a task which aims at giving an accurate answer to a precise question given by the user in

[1] http://www.qamus.org/transliteration.htm

natural language. In this task the type of questions which rely the most on the usage of an NER system to process both the question and document-set are called "factoid questions". These questions ask information about the name of a specific person, location, etc. or a date, e.g. "What is the capital city of the region of Liguria?". After identifying and classifying the NEs in the document-set, the QA system would only consider NEs which were classified as location potential answers. Following, it would proceed to extract the correct answer which is "Genova" (in Chapter 3 we explain with more details how NER helps to improve the performance of other NLP tasks)

For this reason, the investigation of novel approaches which help obtain efficient and accurate NER systems for the English language has been strongly encouraged. Proof is abound in the literature and the reported results by the evaluation campaigns which have been organized for this purpose, e.g. the 6th Message Understanding Conference (MUC-6) and Conference on Computational Natural Language Learning (CoNLL). Nevertheless, very few published research works have attempted to investigate which approaches are adequate for other languages such as Arabic. In this document, we present a large experiment-set which attempts to tackle the Arabic NER problem and we provide all of the necessary details about our experiments and their results.

1.0.1 Arabic Named Entity Recognition Challenges

The Arabic language has some peculiarities which harden the NER task. It has a rich and complex morphology which hardens the NER task significantly. To our knowledge, no published work has shown exactly the error rate induced by the agglutinative characteristic of the Arabic language. Moreover, no published works have attempted to use the rich morphology of the Arabic language so as to enhance the Arabic NER. In order to tackle these problems we have a number of research questions which would need to be answered to be able achieve a set of empirical proofs. These research questions are as follows:

1. Describe with details the peculiarities of the Arabic language and show how they exactly harden the Arabic NER;

2. Give empirical results which show the error rate induced by the morphology complexity of the Arabic language and the lack of capitalization;

3. Explore the possibility of using morphological charactersitics of an Arabic word to help the NER system better determine and classify the NEs within an Arabic text

4. Explore a large feature-set for Arabic NER. Our investigation here should aim also at reporting the impact obtained from each feature. Thereafter, we would investigate the best feature-set which helps to obtain high performance. Also, we would investigate if these features have the same impact, in terms of F-measure, on the different NE classes and a multi-classifier approach where each classifier uses an optimized feature-set for the concerned class.

5. In the literature, the comparison studies which have been carried out to determine the best supervised Machine Learning (ML) technique to be used for the NER task report very shallow results and do not show why exactly one ML approach should be used rather than another. In our research study, we want to conduct our own comparative study which would aim at deeper comparisons and would help decide the appropriate ML approach for the Arabic NER task. Explore whether or not the different ML approaches yield to comparable results for the different classes. Similarly to the features study, we want to investigate the possibility of building a multi-classifier approach where each classifier concerns only one NE class and uses the ML approach which helps to yield the best result for the concerned class.

6. Since the NER task has been more investigated for the English language than for other languages, we want also to attempt importing knowledge about NEs from English to Arabic.

1.0.2 Thesis Overview

Chapter 2 gives a full description of the different aspects of Arabic which are concerned by most of the NLP tasks. This chapter situates the Arabic language linguistically, i.e., it provides information about its origin and its speakers, it gives a description of its orthography and morphology, and at last it emphasizes the peculiarities which have proved to "hurt" the performance of most of the NLP tasks.
Chapter 3 is the NER task state-of-art chapter, i.e., it gives the necessary background which would help later to situate our research work in the NER research community. It starts by showing the different NER official definitions and emphasizes the differences among them. Thereafter, it describes the techniques which have been used by the most successful NER systems. Those systems have either achieved very good performance in official competitions or have achieved the state-of-art for the concerned language. Finally, we focus on the Arabic NER published works.
Chapter 4 is dedicated to remind the theory behind each of the ML approaches which have proved to be efficient for the NER task, namely Maximum Entropy (ME), Support Vector Machines (SVMs) and Conditional Random Fields (CRFs). After a description of the theory, we give an overview of the different research studies which have shown that those approaches are appropriate solutions. We also report some of the NER research works which give further proofs on the efficiency of the mentioned ML approaches. However, the deep insights of these works are given in Chapter 3.
In Chapter 5 we describe our first attempt to build an Arabic NER system. It consists of an ME-based approach which uses a reduced feature-set and two different techniques. The first one uses a 1-step approach which consists of exploring the performance that might be obtained by using an ME-approach and gives an idea about the impact of using external resources (i.e., gazetteers). The second one, uses a 2-step approach. In the first step it marks the boundaries of the NEs within the text. The second step classifies those mentions. Finally, we compare the results obtained for both experiment-sets and we give a further error-analysis.
Chapter 6 shows a comparative study which we have conducted among ME, SVMs and CRFs. In this research work, we have employed a large feature-set which covers

lexical, syntactical and morphological features. In order to carry out a solid study of these features and a deep comparison among the ML approaches which we have mentioned, we have conducted a large experiment-set which shows the impact of each feature with each of the ML approaches on the performance. Thereafter, we have performed an incremental feature selection technique to determine the best feature-set for each ML approach. We have evaluated our experiments on nine different data-sets of different genres (Newswire, Broadcast News, Weblogs, Arabic Treebank).

We also present, in this chapter, a detailed error-analysis which emphasizes the behavior difference amongst the different ML approaches.

Chapter 7 presents a research study which uses a different classifier for each class. Each of these classifiers uses the appropriate ML approach and an optimized feature-set for the concerned class. The final module combines the outcomes of these classifiers in one single output. We report the impact of each feature on each class and describe how we have optimized the feature-sets as well as give a detailed insight of the final results and error-analysis.

Chapter 8 describes a research work which has been carried out during a six-month internship of the candidate at the IBM T. J. Watson Research Center. In this work, the possibility of importing knowledge about NEs from another language (English in our case) has been investigated. In order to do so, an approach which employs a large manually-aligned corpus was used. In this chapter, we describe all the conducted experiments and the obtained results.

In Chapter 9 we draw our conclusions and give our intuition of the possible directions which the research work presented in this document might trigger.

Chapter 2

Peculiarities and Challenges of the Arabic NLP

The importance of Arabic as a language is not going to go away, no matter what happens in the Middle East.
Even if things cool down there – which I think is impossible in the immediate future
– it will be an important language.
– ZoeGriffith–

The word "Semitic" is an adjective derived from *Shem*, the name of one of the three sons of *Noah* in the *Bible*[1]. This word is widely used to refer to a family of languages which are historically related and have very similar grammars. The "Semitic languages" family includes 17 languages[2], the five most widely spoken ones (with the number of speakers) are given in the Table 2.1.

Table 2.1: Ranking of the 5 top living Semitic features according to number of speakers

Languages	Number of speakers
Arabic	206,000,000
Amharic	27,000,000
Hebrew	7,000,000
Tigrinya	6,700,000
Syriac	1,500,000

Among the main grammatical features shared by the Semitic languages are the following:

- They accept three numbers for nouns: Singular, dual and plural;
- All the words derive from a root which is composed of only consonants. These consonants are called *radicals* and are generally of the number of three or four [29];
- Verb tense: *imperative*, *perfect* (for completed actions) and *imperfect* (for uncompleted actions);
- Three cases for nouns and adjectives: *nominative*, *accusative* and *genitive*

In the context of NLP these features introduce new challenges. Thus, different techniques will be needed in order to achieve a performance which is comparable to the

[1]http://en.wikipedia.org/wiki/Semitic

[2]Akkadian, Amharic, Amorite, Arabic, Aramaic, Ge'ez, Gurage, Hebrew, Maltese, Moabite, Nabatean, Phonoecian, Punic, Syrica, Tigrinya, Tigre and Ugaritic.

ones obtained in other languages such as English.
The Arabic language, the target language of the research work presented in this document, is the most widely spoken Semitic language (see Table 2.1). It also ranks the sixth most used language in the world (English is ranked third) and one of the six official language of the United Nations[3].
There are three forms of the Arabic language:

1. Classical Arabic: or Quranic Arabic is the form used in Quran and also in the official documents from the 7th to the 9th century. Nowadays, classical Arabic is only used in special occasions;

2. Modern Standard Arabic (MSA): is the form of Arabic used in television, radio, newspapers, poetry, etc. It is the common language of all the Arabic speakers and the most widely used form of the Arabic language. *In the remainder of this document we will use "Arabic" to refer to "MSA"*;

3. Colloquial Arabic: is an only spoken form, even if most of the words derive from MSA, it is region-specific and might be very different from one area of the Arab world to another.

In this chapter, a detailed description of the different aspects of the Arabic language will be given. We will emphasize each of the obstacles induced by the special features of the Arabic language and give a state-of-art of the task in question. The remainder of this chapter is organized as follows: In Section 2.1 we present the Arabic scripture, codification and morphology. We illustrate the impact of the complex morphology of the language on Arabic corpora investigating their *Complexity, Variety and Harmony*. Further proofs on the impact of the Arabic morphology on the NLP tasks (mainly Information Retrieval and Question Answering) are given in Section 2.2. Finally, we give a special focus on the challenges of Arabic NER in 2.3 because of the importance of the task.

[3]http://en.wikipedia.org/wiki/United_Nations

2.1 The Arabic Language: Scripture, Encodings and Morphology

2.1.1 Scripture

The Arabic language has its own script (written from right to left) which is a 28 letters alphabet (25 consonants and 3 long vowels) with allographic variants and diacritics which are used as short vowels, except one diacritic which is used as a double consonant marker. The Arabic script does not support capitalization. It is the second most widely used script in the world (after the Latin script). It is used by other languages different than Arabic such as Persian[4], Urdu[5], Uyghur[6] among others. Figure 2.1 shows an illustrating example of Arabic text.

أُسّست الجَامعة المتعدِّدة التقنيات لفالنسيا سنة 1971.

The Technical University of Valencia was established in 1971.

Figure 2.1: Example of Arabic text

2.1.2 Encodings

One of the main challenges of the Arabic text editors is the encoding, the two most commonly used encodings are the following:

1. Windows CP-1256: 1-byte characters encoding supports Arabic, French, English and a small group of Arabic extended characters;

[4]http://en.wikipedia.org/wiki/Persian_language
[5]http://en.wikipedia.org/wiki/Urdu_language
[6]http://en.wikipedia.org/wiki/Uyghur_language

2. Unicode: 2-byte characters encoding and supports all the Arabic extended characters.

Both of of these encodings are human compatible because they allow to normal users to write, save and read Arabic text. However, many problems might be faced when a program is processing an Arabic text encoded with one of the above mentioned encodings. For this reason, Arabic NLP researchers would rather use the Buckwalter transliteration[7]. This transliteration is a simple one-to-one mapping from Arabic letters to Roman letters (Figure 2.2 shows the Buckwalter mapping table). Thus, it is more machine compatible because machines are more prepared to work with Roman letters. Nowadays, the Buckwalter transliteration has become the most commonly used encoding in the Arabic NLP research community and many Arabic corpora such as Arabic Treebank and Arabic Semantic Labeling task corpus used in SEMEVAL 2007 [8] use this transliteration.

2.1.3 Morphology

The Arabic language has a very complex morphology because of the two following reasons:

1. It is a *derivational* language: All the Arabic verbs derive from a root of three or four characters root verb. Similarly, all the adjectives derive from a verb and almost all the nouns are derivations as well. Derivations in the Arabic language are almost always templatic, thus we can say that: *Lemma = Root + Pattern*. Moreover, in case of a regular derivation we can deduce the meaning of a *lemma* if we know the *root* and the *pattern* which have been used to derive it. Figure 2.3 shows an example of two Arabic verbs from the same category and their derivation from the same pattern.

2. It is also an *inflectional* language: *Word = prefix(es) + lemma + suffix(es)*. The *prefixes* can be articles, prepositions or conjunctions, whereas the *suffixes* are

[7] http://www.qamus.org/transliteration.htm
[8] http://nlp.cs.swarthmore.edu/semeval/tasks/task18/description.shtml

ء '	ذ *	ل l
ا آ	ر r	م m
أ >	ز z	ن n
ؤ &	س s	ه h
إ <	ش $	و w
ئ }	ص S	ى Y
ا A	ض D	ي y
ب b	ط T	ً F
ة p	ظ Z	ٌ N
ت t	ع E	ٍ K
ث v	غ g	َ a
ج j	ـ _	ُ u
ح H	ف f	ِ i
خ x	ق q	ّ ~
د d	ك k	ْ o

Figure 2.2: Buckwalter mapping table

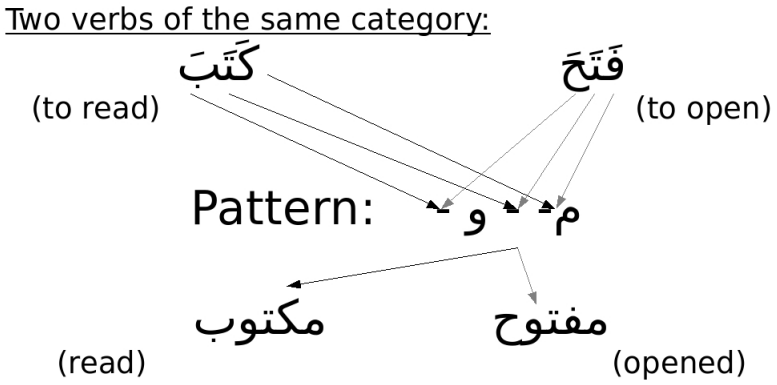


Figure 2.3: An example of Arabic language derivation

generally objects or personal/possessive anaphora. Both prefixes and suffixes are allowed to be combinations, and thus a word can have zero or more affixes (Figure 2.4 shows an example of the composition of an Arabic word).

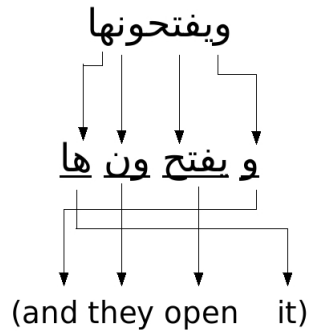


Figure 2.4: An example of Arabic words composition

The Arabic morphology is also very rich. Following we present the morphological features and their possible values for an Arabic verb:

- Aspect : perfective, imperfective, imperative
- Voice : active, passive
- Tense : past, present, future
- Mood : indicative, subjunctive, jussive
- Subject: person, number, gender
- Object : acquire clitics

Moreover, the morphological features for an Arabic noun and their possible values are as follows:

- Number : singular, dual, plural, collective
- Gender : masculine, feminine, neutral
- Definiteness: definite, indefinite
- Case : nominative, accusative, genitive
- Acquire possessive clitics

2.1.4 Sparseness of the Arabic Data

Some of the advantages of working with a morphologically rich language such as Arabic will be reported in Chapters 6, 7 and 8. In this subsection, we aim at giving a clear overview on the difference between Arabic data and the data of other languages with a less complex morphology.

As we have mentioned earlier in Subsection 2.1.3, an Arabic word is formed by a stem plus affixes and clitics. Therefore, as it is shown in Figure 2.4, what can be expressed in one Arabic word requires many words to be expressed in other languages. From an NLP perspective, *highly complex morphology causes "data sparseness" and thus many NLP tasks would require a pre-processing of the data in order to obtain a high performance. Data sparseness* can also be defined as the insufficiency of data. In the NLP context, data are seen as sparse (or insufficient) when the *ratio* of *cardinality of the vocabulary* to *total number of words* is very high. Thus, one can intuitively deduce that this ratio is higher for Arabic data than for other languages with less complex morphology because the same word (such as the word "Iraq" in Figure 2.5) can be attached to different affixes and clitics and hence the vocabulary is much bigger.

والعراق	للعراق	بالعراق	العراق
and Iraq	for Iraq	in Iraq	Iraq

Figure 2.5: The word Iraq with different affixes

In order to tackle the problem of data sparseness, two solutions are possible:

1. *Light-stemming*: consists of simply striping off all the affixes and keeping only the stem. In [66], the authors report a comparative study between different Arabic light-stemming techniques. We briefly mention the technique which has lead to obtain the best results in Subsection 2.2

2. *Tokenization*: differently from *light-stemming* the affixes are not completely removed but separated from each other and from the stem by the space character. In all our experiments, we use Mona Diab's tokenizer [30] which is freely available on her website[9].

In order to prove experimentally the negative effect of the morphology complexity of the Arabic language, we have carried out preliminary experiments with different types of Arabic corpora and using some stylometric measures [15]. Following we present the details about our experiments:

Data: We have selected four corpora of different types for our experiments. Table 2.2 gives details about our corpora:

Table 2.2: Corpora description

Corpus	Corpus 1	Corpus 2	Corpus 3	Corpus 4
Description	Poetry	Newspapers articles	Linux Red Hat tutorial	Religious book
Author	Abu-Taib Al-Moutanabbi	Different authors	Unknown	Ibnu Qayyim Al-Jaweziyya
Number of words	66,000	50,000	55,000	65,000
Size in kB	360	260	126	460

We have intentionally chosen the corpora to have approximately the same number of words (see Table 2.2) in order to be able to compare the results obtained on each one of them. The choice criteria of each corpora are the following ones:

- *Corpus 1*: a collection of poems of one of the greatest poets of the Arabic language[10]. This corpus has no specific topic and it is written in a very high quality writing style. Also, it is very rich in vocabulary because the use of synonyms in poetry is very well appreciated.

[9] http://www1.cs.columbia.edu/∼mdiab/
[10] http://en.wikipedia.org/wiki/Al-Mutanabbi

- *Corpus 2*: a collection of 111 newspaper articles. The texts are of different topics and different domains which makes the vocabulary very rich.
- *Corpus 3*: a scientific book focused on one single topic. Although the vocabulary size is not very big, the reader would require more effort to read this book than other types of books.
- *Corpus 4*: it also contains a restricted vocabulary because it is focused on just one topic. This religious book has been written by Ibnu Qayyim Al-Jaweziyya[11] who has been also well appreciated for his writing style quality.

Measures: In [72], three measures have been proposed in order to determine the writing quality and reading complexity of a certain text. Those measures are:

1. $Complexity = C.log(M)$
   where C is the average number of characters in a word and M is the average number of words in a sentence. The complexity is a factor which is very related to the nature of the corpus.

2. $Variety = n/log(N)$
   where n is the cardinal of the vocabulary and N is the total number of words. This factor gives an idea about the variety of expressions in a document. It is most of all related to the author style and the nature of the document, e.g. a low variety is expected in a scientific document or any document which covers only one topic whereas a high variety is expected in a poem or a collection of newspapers articles of different topics. Other considerations should be taken into consideration for this factor, namely the language, e.g. in Arabic the use of synonyms is appreciated as a good writing style which would significantly raise the variety.

3. *Correctness of the corpus words frequency distribution*
   This measure is based on "the principle of least-effort" of the Harvard linguist

[11] http://en.wikipedia.org/wiki/Ibn_Qayyim_Al-Jawziyya

George Kingsley Zipf [112]. *Zipf's law* is an empirical law based on the observation that the frequency of occurrence of some events is a function of its rank in the frequency table. The simplest equation to express this function is the following:

$$freq(r) = \frac{C}{r^{\alpha}} \tag{2.1}$$

where r is the rank, C is the highest observed frequency and α is a constant usually close to 1. This equation states that the most frequent event will occur twice as often as the second most frequent word. Its graphical representation in a log-log scale is a straight line with a negative slope. The different equations which might express the Zipf's law are widely discussed in the literature and presenting an overview of the above goes beyond the scope of this document and can be found in [76].

In [72], it has been also claimed that one way to measure the *harmony* of a text is to compare its words frequency distribution with the ideal Zipf's curve. The authors explain that a corpus is less or more harmonious when it requires, respectively, more or less effort from the reader. However, generally this comparison is visually done by a human being. This has motivated us to look for an automatic comparison of the words frequency distribution and the Zip's ideal curve.

Kullback and Leibler have suggested in [64] a measure in order to determine the distance between two probability distributions related to the same experiment.

$$D_{KL}(P||Q) = \sum_{i} P(i) log \frac{P(i)}{Q(i)} \tag{2.2}$$

where P is the "true" distribution (in this case, the words frequency distribution) and Q represents a theory (in this case, the Zip's Law). It is also important to note that the Kullback-Leibler distance has three important characteristics:

(a) $D(P||Q) \geqslant 0 \forall P, Q$

(b) $D(P||Q) = 0 \Longleftrightarrow P = Q$

(c) $D(P||Q) \neq D(Q||P)$

The Kullback-Leibler distance is very convenient to measure the correspondence of the words frequency distribution to the Zip's ideal curve. However, in order to be able to use Equation 2.2, all the frequencies should be normalized to $[0-1]$ (see "Experiments and Results").

Experiments and Results: We have organized our experiments as follows:

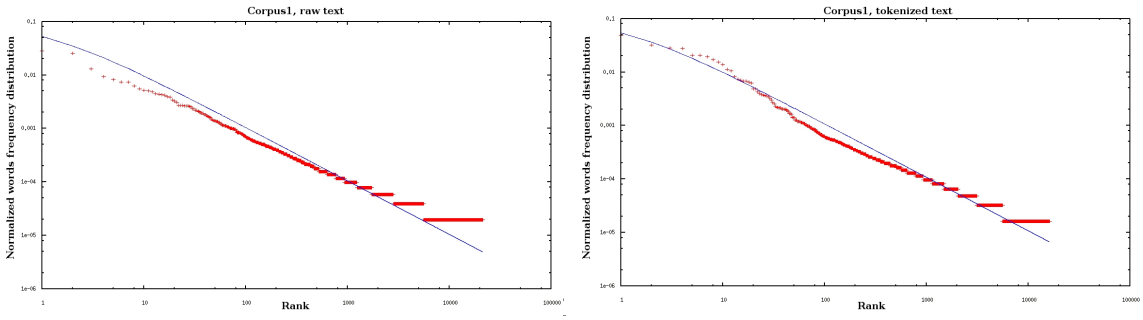
1. *Pre-processing*: eliminate the short vowels from all the corpora;
2. *First experiment*: compute $Complexity$, $Variety$, the words frequency distribution, the Zip's ideal curve and the Kullback-Leibler distance;
3. *Tokenization*: tokenize the texts in order to be able to compare the results obtained with raw text;
4. *Second experiment*: compute the same values of the first experiment on the tokenized corpora.

Table 2.3 shows the results obtained for each of the corpora and measure. Figures 2.6, 2.7, 2.8 and 2.9 show the words frequency distribution and the Zipf's ideal curve for Corpus 1, Corpus 2, Corpus 3 and Corpus 4.

Discussion The results obtained for the complexity and variety measures are not affected by the complex morphology of the Arabic language because we have obtained the expected values for both raw and tokenized corpora. The complexity of Corpus 1 (Poetry) is considerably lower than the other corpora because the average number of words in a poem sentence is approximately 5 words. The results shown in the Kullback-Leibler distance columns prove that in all the cases the distance is signicantly lower when a corpus is tokenized, i.e., we can make better statistical analyses on Arabic corpora after tokenization.

Table 2.3: Results obtained for Complexity, Variety and Kullback-Leibler distance for each corpus

Corpora	*Complexity*		*Variety*		*Kullback-Leibler distance*	
	Raw	*Tokenized*	*Raw*	*Tokenized*	*Raw*	*Tokenized*
Corpus 1	2.14	1.84	1887.35	1547.86	-62486.31	22120.32
Corpus 2	18.55	14.94	1501.76	1033.87	49292.38	32836.98
Corpus 3	19.55	14.78	881.47	508.39	65381.42	41893.44
Corpus 4	23.62	16.52	1042.03	760.56	44473.40	28870.38



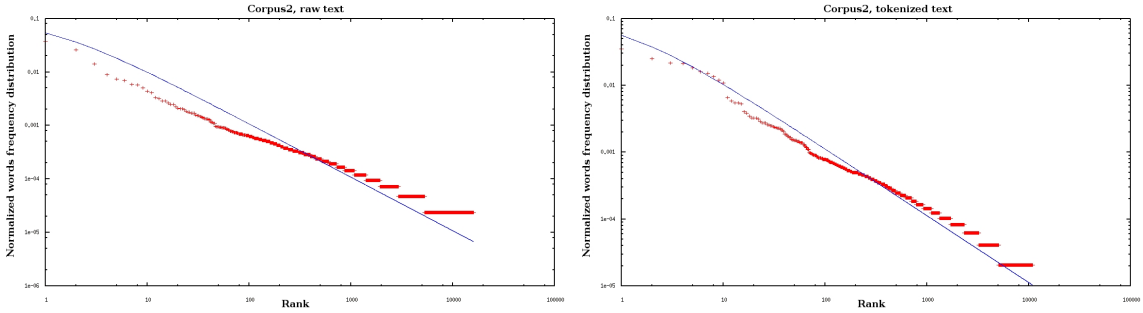
((a)) ((b))

Figure 2.6: The words frequency distribution and the Zipf's ideal curve for corpus 1

2.2 Hardness of Information Retrieval and Question Answering

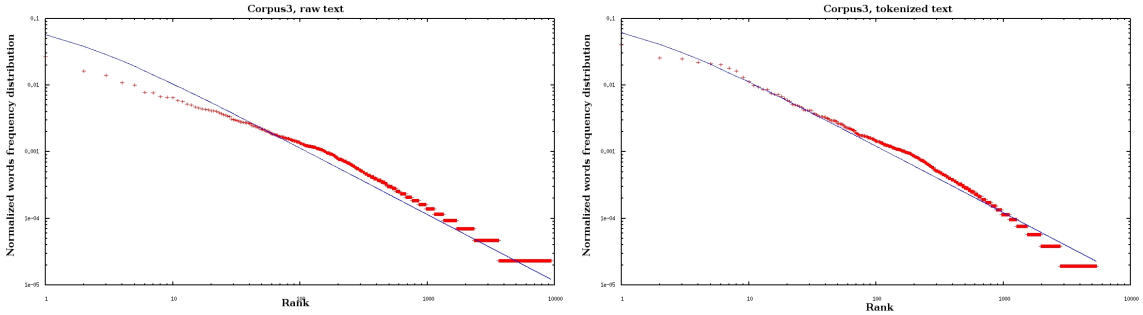
In the previous subsection we have shown that it is not possible to perform the most basic measures on Arabic corpora unless we first perform a pre-processing step where the data is tokenized. This pre-processing step helps to overcome the data sparseness problem which is induced by the complex morphology of the language (see Subsection 2.1.3).

In the present and next subsections, we will show how the agglutinative feature of the



((a)) ((b))

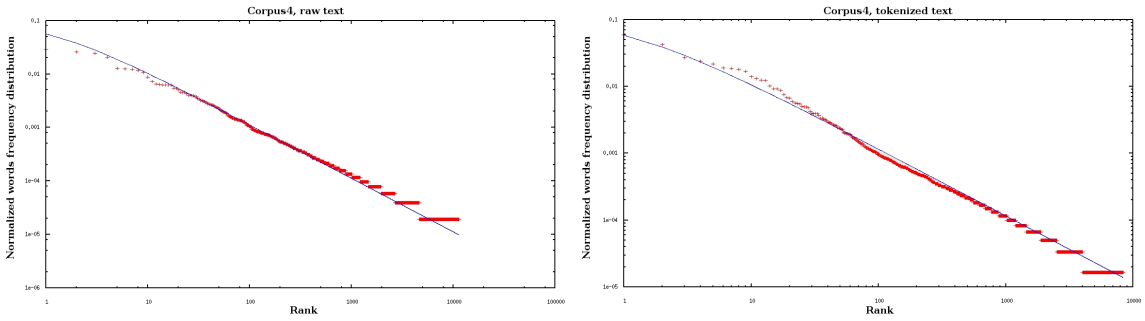
Figure 2.7: The words frequency distribution and the Zipf's ideal curve for corpus 2



((a)) ((b))

Figure 2.8: The words frequency distribution and the Zipf's ideal curve for corpus 3

Arabic language makes both the supervised and unsupervised NLP tasks much more challenging. For this purpose, we have chosen two of the most important tasks of NLP: (i) Information Retrieval (IR); and (ii) Question Answering (QA). Within the QA task we will give also details about the hardness of the Passage Retrieval (PR) and Named Entity Recognition (NER) tasks for the Arabic language. We finish this chapter describing more in detail anout NER because it is: (i) very important for most of the NLP tasks, (ii) approached by small number of researchers for the Arabic language due to its difficulty; and (iii) the research topic of this Ph.D. thesis.



((a)) ((b))

Figure 2.9: The words frequency distribution and the the Zipf's ideal curve for corpus 4

2.2.1 Information Retrieval

IR is a task which aims at providing a set of documents that might contain the information needed by the user [9]. Therefore, an IR system receives at the input a "user query" (written in natural language) and returns at its output the set of the most relevant documents to the query formulated in the input. In the case of Internet search engines, such as Google[12], Yahoo[13] or MSN[14], the IR system extracts the relevant documents from all the texts with all the formats (text, HTML, PDF, etc.) available on the web. On the other hand, in IR competition tasks such as those in the Text REtrieval Conferences[15] (TREC) and Cross Language Evaluation Forum[16] (CLEF), the documents are extracted from a common document-set with the same format. Moreover, in TREC 2001[17] and TREC 2002[18] an Arabic-English IR task (in which the queries were given in English and the data set where to look for relevant documents was in Arabic) was included which has considerably contributed to boost research in Arabic IR. An overview of the obtained results is given in [32]. All the

[12]http://www.google.com
[13]http://www.yahoo.com
[14]http://www.msn.com
[15]http://trec.nist.gov/
[16]http://www.lef-campaign.org
[17]http://trec.nist.gov/pubs/trec10/t10_proceedings.html
[18]http://trec.nist.gov/pubs/trec11/t11_proceedings.html

best systems, [67][24][106], report that they have used light-stemming to improve their results. [67] have re-published in 2007 an improvement of their work in order to give more details and deeper analyses of the best light-stemming technique for the Arabic language [66]. As reported in the paper, the authors have used the TREC 2002 corpus for their experiments. The best results (*average precision* = 0.413 vs. 0.196 for raw text) were obtained when the authors performed an "affixes removal" light-stemming. This technique consists of removing the strings which have been frequently seen as prefixes and suffixes and infrequently seen as part of the stem. The list of these affixes is also published in order to make possible an implementation of the light-stemmer.
In our research work [16], which we present with details in 2.2.2, we have used the same light-stemmer as in [66] for the passage retrieval task (a subtask of IR). Similarly, we have achieved better results when the documents were light-stemmed.

2.2.2 Question Answering

As we have described previously (see "Information Retrieval"), IR systems are designed to retrieve the documents which are estimated to be relevant to the user's query. Therefore, these systems are unable to satisfy the users who are interested in obtaining a simple answer to a specific question. The research line aiming at satisfying this kind of users is Question Answering. The study of the QA task research guidelines [21] reported that there are generally four kinds of questioners where each type represents questions with a certain level of complexity:

1. *Casual questioner*: asking concrete questions about specific facts; for instance: "Who is the king of Morocco?", "In which city will LREC'08 be held?", etc.;

2. *Template questioner*: this type of questioner might ask some questions which require the system to retrieve portions of the answer from different documents and combine them in one answer. For instance: "What are all the countries that border Morocco", "What do we know about the life of Malcom X?", etc.;

3. *Cub reporter*: this other type of questioners would require a QA system to be able to collect many information from different sources about a single fact. This type of questioners was named "Cub Reporter" because reporters need all the available information to write a report about a fact, for instance a "flooding" (place and time, the damaged area, the estimated dollar damage, number of killed and injured citizens, etc.), a "war" (countries involved in the war, impact of the war, political reasons, damaged area, etc.), etc.;

4. *Professional information analyst*: finally, this is the highest level of questioners which need a system able to be able to deduce and decide by itself the answer because the question might be something like "Is there any evidence of the existence of weapons of mass destruction in Iraq?".

The analysis of the CLEF and the TREC results shows that up to now only the two first types of questions (casual and template) have been covered by the QA research community. The most successful system in the monolingual task of the CLEF 2007 used sophisticated NLP tools (embedded ontologies, synonyms dictionaries, coreference resolution tools, an accurate NER system, etc.) [68] and obtained an accuracy of 54%. The second best participation[6] obtained 44.5%. The authors report that the most relevant module of their system is a syntactical question analyzer which performs the extraction of named entities and keywords from the questions.
In TREC 2007 [28], the best participation [79] obtained an overall result of 48.4. The authors report that the most relevant aspect of their system consisted in enhancing their NER system, which they have named "Rose", adding finer grained types. The second best results [52], obtained 35.75 by enhancing the performance of their retrieval system. In order to do so they have adopted a keyword-based approach to retrieve the passages in the documents which might contain the answer; i.e., their retrieval system has been boosted by an NER system, a semantic dependencies parser and a semantic frames recognizer. Finally, in [87] the authors have obtained the third best participation with an overall score of 23.10. They argue that their system is more focused on casual questioner type of questions (also called "factoid questions"). For this purpose, their factoid questions component uses a model to resolve time con-

straints, a query expansion module and an answer ranking module.

The CLEF organizers offered an evaluation platform for many languages but unfortunatly the Arabic language does not figure among them. The non-existence of a test-bed for Arabic language makes QA even more challenging. However, there have been two attempts to build Arabic QA systems oriented to: (i) structured data, a knowledge-based source of information [78]; and (ii) unstructured data [49]. The test-bed of the second QA system was composed of a set of articles of the Raya[19] newspaper. In the evaluation process four Arabic native speakers were asked to give the system 113 questions and judge manually the correctness of its answers. The reported results of precision and recall are of 97,3%. These (possibly biased) results seem to be very high if compared with those obtained before for other languages in the CLEF 2007 and TREC 2007 competitions. Unfortunately, the test-bed which was used by the authors is not publicly available in order to compare the QA system with others. Hence, the QA task for the Arabic language is very few investigated and the research works which were published either they do not tackle QA problems for unstructured data, do not report the real problems encountered and their possible solution or both. For this reason, we have considered that an effort needs to be done in order to explore the challenges and the required techniques in order to build an Arabic QA system, which we named "ArabiQA", that: (i) extracts the answers from natural language texts; and (ii) uses the CLEF and TREC guidelines for the evaluation of the different modules of the system.

In order to build such a system, we have chosen the architecture illustrated in Figure 2.10. From a general viewpoint the system is composed of the following components:

- *Question Analysis* module: determines the type of the given question (in order to inform the Answer Extraction (AE) module about the expected type of answer), the question keywords (used by the passage retrieval module as a query) and the named entities appearing in the question (which are very essential to validate the candidate answers);

[19] http://www.raya.com

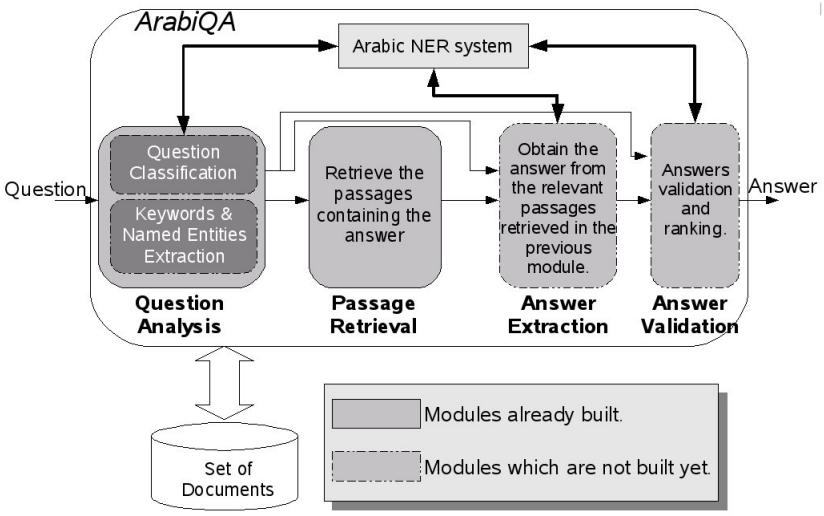


Figure 2.10: Generic Architecture of our Arabic Question answering system, ArabiQA

- *Passage Retrieval* (PR) module: it is the core module of the system. It retrieves the passages estimates relevant to contain the answer;
- *Answer Extraction* (AE) module: it extracts a list of candidate answers from the relevant passages;
- *Answers Validation* module: it estimates for each of the candidate answers the probability of correctness and ranks them from the most to the least probable correct one.
- *Named Entity Recognizer*: a vital component in the QA system which identifies the different named entities within the text (both documents and questions).

Following we give more details about ArabiQA components which have been already developed. This will allow us to clarify other aspects of the errors induced by the complex morphology of the Arabic language.

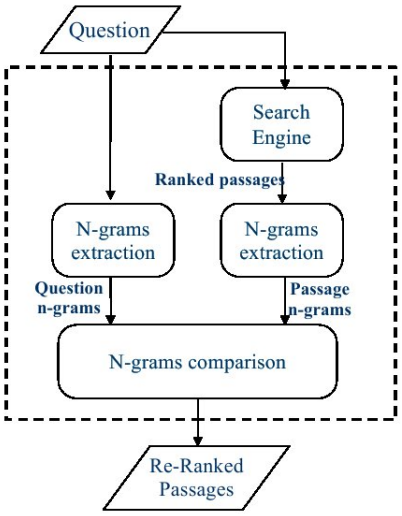


Figure 2.11: The JIRS architecture

Passage Retrieval

In order to develop a PR system for our Arabic QA system, we decided to investigate the possibilty of adapting the language-independent Java Information Retrieval System (JIRS)[20]. Many are the systems participating in different tasks of CLEF 2005 [110][42] and 2006 [19] [85] [37] which have used JIRS for PR. This shows that JIRS can be also employed in other NLP tasks than just QA [37]. JIRS proved to be efficient for the Spanish, Italian, French and English languages. The peculiarity of the Arabic language of being highly inflectional and, therefore, very different with respect to the above languages, made the study of the possibility of using JIRS very interesting and its adaptation very challenging.
Figure 2.11 illustrates the architecture of the JIRS system.

JIRS is a QA-oriented PR system which in order to index the documents it relies on an n-gram model. The PR systems retrieve the relevant passages in two main steps [43]. In the first step it searches the relevant passages and assigns a weight to each ine of them. The weight of a passage depends mainly on the relevant question terms appearing in the passage. The second step performs only on the top *"m"*

[20]freely available at http://sourceforge.net/projects/jirs

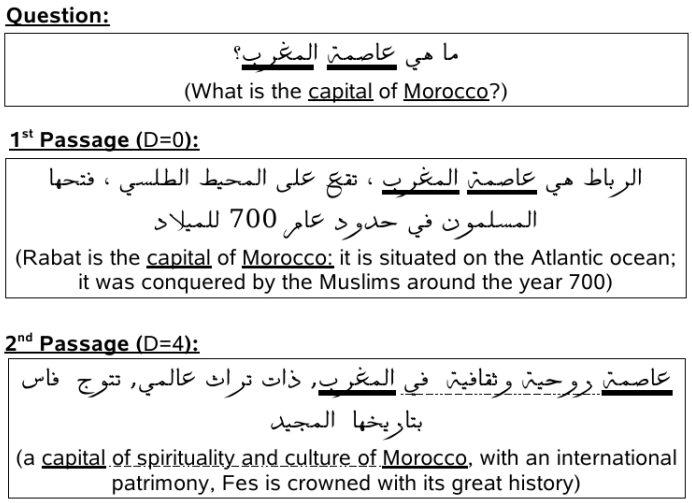


Figure 2.12: An example to illustrate the performance of the Density Distance model (an English translation is given in between parenthesis)

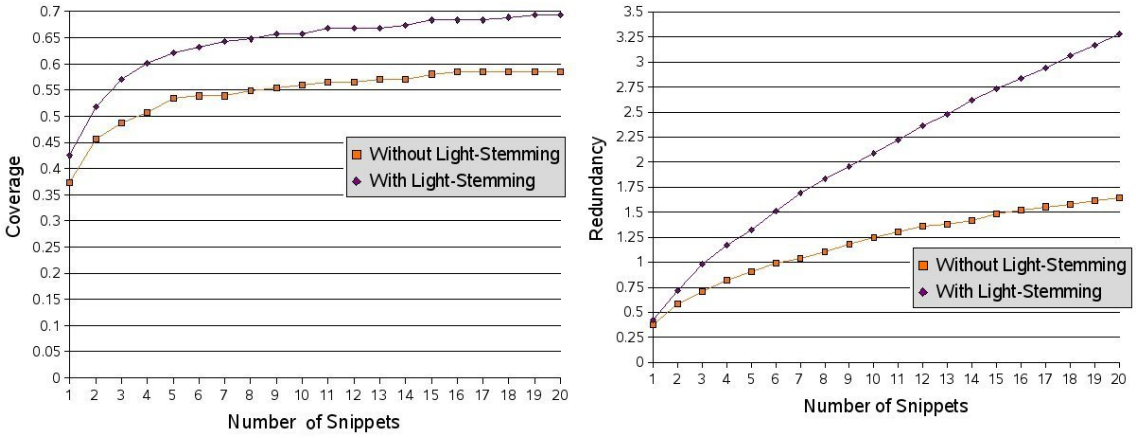
passages of the relevant passages returned by the first step (generally m=1,000). In this step, JIRS extracts the necessary n-grams from each passage. Finally, using the question and the passage n-grams, it compares them using the *Density Distance* model. The idea of this model is to give more weight to the passages where the most relevant question structures appear nearer to each other. For example, let us suppose the question and the two passages shown in Figure 2.12. The correct answer to the question is "Rabat". The Density Distance model would give more weight to the first passage because the distance between the words *capital* and *Morocco* is smaller than the distance between these same words in the second passage.

The Arabic-JIRS version of the passage retrieval system [16] relied on the same architecture of Figure 2.11. The main modifications were made on the Arabic language-related files (text encoding, stop-words, list of characters for text normalization, Arabic special characters, question words, etc.). The Arabic-JIRS is also available at the main web page[21].

Finally, in order to evaluate the efficiency of our PR system on Arabic text, we have

[21]http://sourceforge.net/projects/jirs

used a snapshot of the articles of the Arabic Wikipedia[22] (11,638 documents). We have manually built a set of 200 questions (considering the different classes that were reported in the CLEF 2006 competition) and a list of the correct answers of these questions. Our experiment-set consisted of evaluating JIRS on both raw documents and light-stemmed documents. In order to obtain an efficient light-stemming we have used the same light-stemmer which has helped to obatin the best results in [66][23]. For each experiment we have computed the *Coverage* (ratio of the number of the correct retrieved passages to the number of the correct passages) and *Redundancy* (average of the number of passages returned for a question) measures to evaluate the system. Figure 2.13 shows the coverage (a) and the redundancy (b) measures for both experiments.



((a)) ((b))

Figure 2.13: Comparison of Coverage and Redundancy of JIRS over both light-stemmed and non-stemmed Arabic corpora

Figure 2.13 shows that the complex morphology of the Arabic language decreases the coverage of the PR system up to 10%. Moreover, in redundancy the system has raised from 1.65 to 3.28 (an increase of 100%) when we have performed the light-

[22]http://ar.wikipedia.org

[23]At the moment of performing these experiments the light-stemmer was freely available at http://www.glue.umd.edu/~kareem/research/

stemming on the documents.

The PR system is an essential component of a QA system because it is charged of retrieving the passages where the answer is located. Thus, the other components cannot extract the answer if the PR does not retrieve the relevant passages. Our experiments show that the same error rate induced by the complex morphology of the Arabic language in the IR task can be experimented in the PR task and consequently in the QA task. Similarly to the IR, light-stemming has been proved to be a convenient solution in order to enhance the performance of the PR module.

Named Entity Recognizer

An accurate NER system is of great importance for a QA system and especially for answering factoid questions. Its role is to identify and classify all the NEs within the passages retrieved by the PR module (Figure 2.14 gives an example of the output of the NER system). To our knowledge, there are no freely available Arabic NER

Input Text:

واكد الرئيس مبارك ان مصر علي اتم استعداد لدعم الاشقاء العراقي

(President Mubarak stressed that Egypt stands ready to support Iraqi brothers)

Output Text:

واكد الرئيس **<Person>**مبارك**</Person>** ان **<Location>**مصر**</Location>**
علي اتم استعداد لدعم الاشقاء العراقي

(President **<Person>**Mubarak**</Person>** stressed that
<Location>Egypt**</Location>** stands ready to support Iraqi brothers)

Figure 2.14: Illustrating example of the input and output of an NER system

systems. The existing ones were built for a commercial purpose: Siraj[24] (by Sakhr), ClearTags[25] (by ClearForest), NetOwlExtractor[26] (by NetOwl) and InxightSmartDis-

[24]http://siraj.sakhr.com/

[25]http://www.clearforest.com/index.asp

[26]http://www.netowl.com/products/extractor.html

coveryEntityExtractor[27] (by Inxight). Unfortunately, no performance accuracy nor technical details have been provided and a comparative study of these systems is not possible.

The NER system that we have integrated in ArabiQA, *ANERsys* [18], is described with details in 4 and the details about its performance are given in 7. This system is based on a Maximum Entropy approach and reaches an overall F-measure of 55.23. As we explain in 2.2.2, a good performance of the global QA system is not possible without enhancing the NER system. Moreover, the NER task becomes much more complicated (see Section 2.3) because of the complex morphology of Arabic which we have described in 2.1.3. For this reason, we have decided to make further investigations to obtain an accurate and efficient NER system oriented to the Arabic language. The different approaches which were take into considerations in these investigations, together with the obtained results, are the main subject of this Ph.D. thesis.

Answer Extraction

The AE task is defined as the search for candidate answers within the relevant passages. The task has to take into consideration the type of answers expected by the user [25], and this means that the AE module should perform differently for each type of question. Using an *accurate* NER system together with patterns seems to be a successful approach to extract answers for factoid questions [1][80]. The same Approach has been reported to give promising results for the Arabic language [49]. However, for difficult questions it is needed a semantic parsing to extract the correct answer [50][75]. In our research work [17], we have built an AE module for only Arabic *factoid* questions. Our module performs in three main steps (Figure 2.15 gives an illustrating example):

1. The NER system tags all the NE's within the relevant passages;

2. The AE module makes a preselection of the candidate answers eliminating NEs which do not correspond to the expected type of answer;

[27] http://www.inxight.com/products/smartdiscovery/ee/index.php

3. A final decision on the list of candidate answers is taken by means of a set of patterns which we have prepared manually.

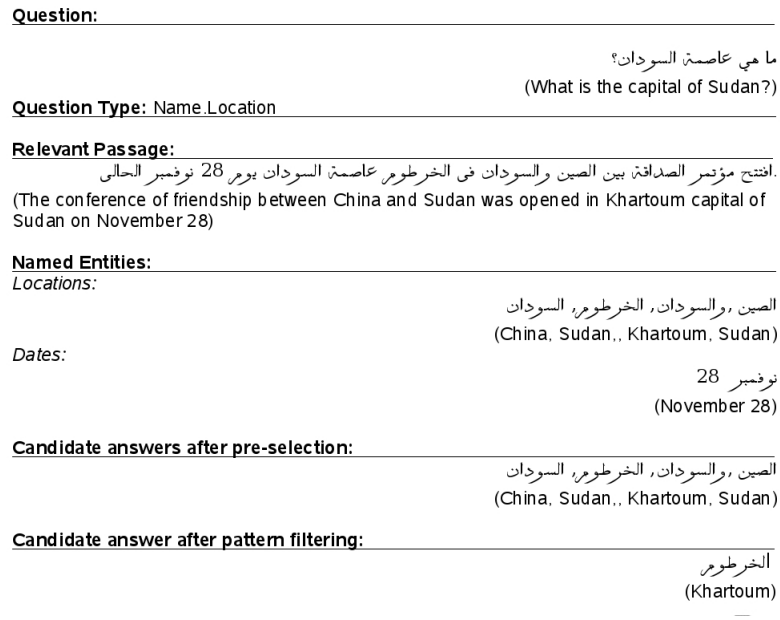


Figure 2.15: Illustrating example of the Answer Extraction module's performance steps

In order to make an automatic evaluation of the AE module we have prepared a test-set composed of the following elements:

1. List of questions of different types;
2. List of questions types which contains the type of each of the test questions;
3. List of relevant passages (we have manually built a file containing a passage which contains the correct answer for each question);
4. List of correct answers (containing the correct answer for each question).

Note that the idea behind doing a manual selection of the relevant passages is to be able to estimate the exact error rate of the AE module.

The results showed that with this method we can obatain a *precision* (number of

correct answers / number of questions) of 83.3%. Our error analysis showed that the precision can be increased only if the NER system is enhanced (the NER system used in these experiments is ANERsys 1.0, see 5 for a detailed description of the experiments).

2.3 Hardness of Arabic Named Entity Recognition

As we have introduced in the Subsection 2.2.2, the lack of accurate NER systems oriented to the Arabic language and the hardness of the Arabic NER task itself have lead us to conduct further investigations in this research field. The illustrating example given in 2.14 shows the input and output strings of an NER system.
The NER task is considerably more challenging when it is targeting a morphologically complex language such as Arabic for two main reasons:

1. Lack of capitalization; English, like many other Latin script based languages, has a specific signal in the orthography, namely capitalization of the initial letter, indicating that a word or sequence of words is an NE. Arabic has no such special signal (see Subsection 2.1.1) making the detection of NEs more challenging. Thus, from a lexical viewpoint, there is no difference between an NE (see the word "Mexico" in Figure 2.16) and the other word (see the word "life" in Figure 2.16).

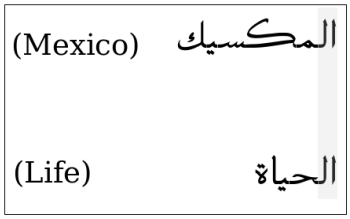


Figure 2.16: Illustrating example of an NE and non-NE starting with the same characters

2. Arabic is an agglutinative language. As we have argued in 2.1.3 the Arabic language has a complex morphology. We have also shown (see Section 2.2) that

this characteristic renders most of the NLP tasks more challenging. Similarly, in the NER task it is very important to take into consideration that a pre-processing step is needed to tackle the data sparseness. Otherwise, a very large amount of data would be required for a good training because both the NEs (see Figure 2.5) and the context in which they occur (see Figure 2.17) might appear in different surface forms. However, in the context of NER, light-stemming is not a convenient solution because the affixes should be kept in the text otherwise many important words of the context are lost. For this reason, tokenization is a more appropriate solution because it only separates the clitics from the stem word without removing any tokens. In the next chapter, we will describe in detail the NER task.

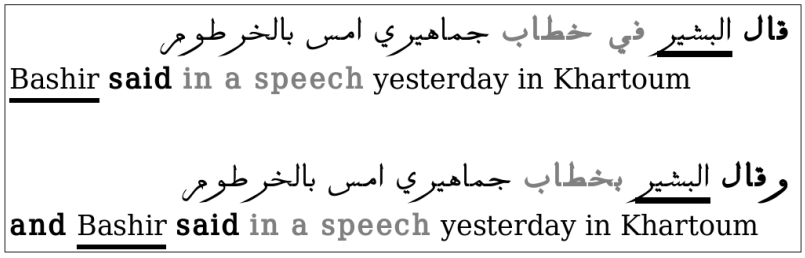


Figure 2.17: Illustrating example of an NE appearing in different contexts because of the complex morphology of the Arabic language

2.4 Concluding Remarks

The Arabic language is a Semitic language and thus it has a templatic morphology where words are made up of roots and affixes (clitics agglutinate to words). This causes a great variety of surface forms. In the context of NLP, this phenomenon is called "data sparseness" (or "data insufficiency"). For this particular reason, both the NLP tasks , supervised and unsupervised, become more challenging and need to be approached differently.

For IR and PR (a subtask of IR and the core component of a QA system), the agglutinative feature of the Arabic language makes the documents which are relevant

to the user's query invisible to the IR system because a word has a considerable probability to appear with different affixes. Thus, the research studies which we have presented prove empirically that an IR system performs much better on light-stemmed Arabic texts. Moreover, the QA task also becomes considerably harder because each of its components has to overcome the problems induced by the complex morphology of the Arabic language. The NER system, whose accuracy has direct impact on the overall QA system performance (see Section 3, required major efforts in Arabic. The NER task is harder for the Arabic language than for other languages because of: (i) lack of capital letters in the Arabic scripture; and (ii) the great sparseness of the Arabic data decreases the efficiency of the training. In order to tackle this problem we have conducted several experiment-sets which we present with the necessary details throughout this document.

Chapter 3

The Named Entity Recognition Task

In Western languages the names of the four seasons became complete only a few centuries ago. Words for "winter" and "summer" appear quite early but in English "spring" came to be used as the name of the season as late as the sixteenth century, and in German "fruhjahr", "spring" appeared about the same time. Similarly, in India "hemanta (winter)" and "vasanta (spring)" appear in Sanskrit literature very early, while other seasonal terms come much later.
–Anonymous Japanese author–

From a human viewpoint, the contribution of NEs to make the communication easier and more precise is obvious. However, a scientific proof is needed in order to be able to make the same statement about NEs in the statistical NLP context. In [100], Shannon defines the "self-information" (also designed as the "quantity of information") contained in an event x as the measure of the amount of surprise. For instance, if a person "A" informs a person "B" on *Friday* that the day after will be a *Saturday* then the amount of surprise of "B" is zero. Whereas if "A" informs "B" that the Pope Benedict XVI has converted to Islam then the amount of surprise will be huge. Thus, Shannon explains that self-information of an event x is inversely proportional to its occurrence probability. Hence, it can be expressed by the following formula:

$$I(x) = -log_2(p(x)) \tag{3.1}$$

In order to calculate the self-information of NEs we have carried out an experiment which consists of the following steps:

- Tokenize and Part-Of-Speech (POS) tag a corpus annotated for the NER task: for this purpose we have used the corpus which we have developed ourselves[1] (see Chapter 5) and a POS tagger which is freely available[2].

- Compute the probability of occurence of each of the following word categories:

    1. *NEs*: the proper nouns of people, places and organizations (tagged as NEs in the NER annotation).

    2. *Verbs*: the actions tagged as VBD (Verb, past tense), VBP (Verb, present tense, not 3rd person singular) and VBN (Verb, past participle) by the POS-tagger;

    3. *Common nouns*: the classes and categories of entities (tagged as NN and NNS by the POS-tagger).

[1] http://www.dsic.upv.es/grupos/nle/downloads.html
[2] http://www1.cs.columbia.edu/∼mdiab/software/AMIRA-1.0.tar.gz

4. *Stop words*: the rest of the words which are adjectives (JJ), prepositions (PRP), punctuations (PUNC), conjunctions (CC), determiner (DT), etc.

- Finally, using Equation 3.1, compute the self-information of each of the four categories for different number of words.

Figure 3.1 illustrates the obtained results. It shows that the only word category which contains more information than NEs (NEs represent around 11% of the corpus) refers to verbs.

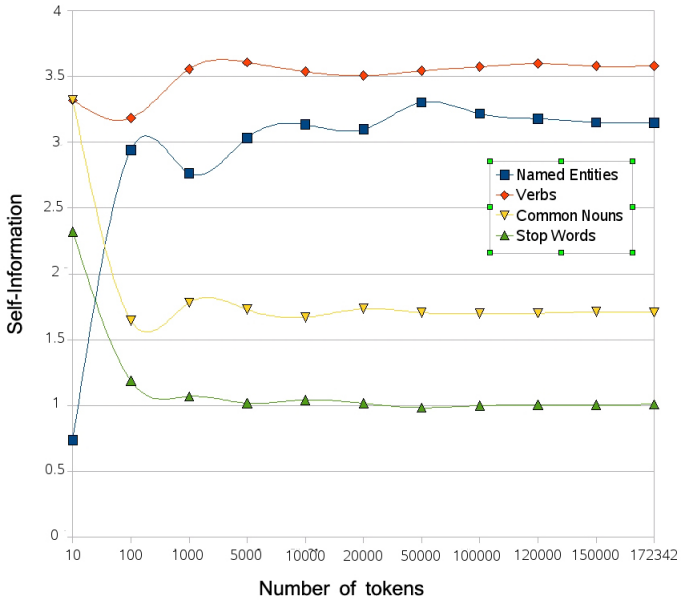


Figure 3.1: NEs quantity of information in comparison with Part-of-Speech word categories

This measure highlights the importance of NEs from a statistical viewpoint. From an NLP viewpoint, we can find abundant research works in the literature which show the necessity of an accurate NER system for many tasks. In Section 3.1, we describe some of the most interesting research works which remark the importance of NER for NLP tasks such as IR, monolingual and cross-lingual QA, text clustering and machine translation. Whereas in Section 3.2 we present all the formal definitions of the NER task. We report the state-of-art of this task in Section 3.3 and we draw some

conclusions of this chapter in Section 3.4.

3.1 How Important is the Named Entity Recognition Task?

In the previous chapter, we have described the different peculiarities of the Arabic language and have given an overview of their impact on the NER and other NLP tasks. In this section, we will show the importance of the NER task for other NLP tasks from a language-independent perspective. Thus, differently from the previous chapter, we will present several research works from different NLP research areas which have proved the necessity of accurate and efficient NER systems.

Information Retrieval In [102], the authors carried out a statistical study about the ratio of user queries containing NEs, over a period of several days, to different news databases. They report that 67.83%, 83.4% and 38.8% of the queries contained an NE according to Wall St. Journal, Los Angeles Times and Washington Post, respectively. Furthermore, the authors present the study which they have conducted on the impact of using an NER system to boost the IR task. Their experiments consisted in using a set of 38 queries containing person NEs and a document-set of 410,883 documents. Thereafter, both the queries and the text are tagged by an NER system. During the indexing phase of the IR system, the words tagged by the NER system are the only words which are not stemmed. This is because the idea of stemming words in order to group those words which have the same stem cannot be applied to NEs. Another important aspect of their NER-based approach is that each term in the query is treated as a different concept except the NEs terms which are always kept together. In order to illustrate this approach, the authors give the example of the query "*Cases involving jailhouse lawyer Joe Woods*". In this case the NE-based approach would compute the *tf-idf* [95] (term frequency-inverse document frequency) for *Joe Woods* (0.78) as a single concept instead of computing it for *Joe* (0.31) and *Woods* (0.24) a

part. The results showed that the new approach outperforms the baseline precision (a probabilistic retrieval engine [108]) on all the levels of recall.

Monolingual and Cross-lingual Question Answering The impact of the NER task on the monolingual and CLQA has been investigated further than for IR. On one hand, in [36] the authors argue that a study of the percentage of the questions containing one or more NEs in the CLEF 2004 and 2005 competitions, showed that the majority. more precisely 87.7%, of questions contain an NE (see Table 3.1).

Table 3.1: Percentage of questions in CLEF 2004 and 2005 containing NEs per type

Type of NE	*Percentage*
Person	29.5%
Location	26%
Organization	20.5%
Miscellaneous	19.3%

The authors argue that it is also possible to find more than one NE in a question. For instance, question 106 in CLEF 2004 was:
What institution did Brazil, Portugal, Angola, Mozambique, Saint Tome and Principe, Guinea-Bissau and Cape Verde form in Brasilia?
where we can find 8 NEs (all of them of the same type, i.e. Location) which are: *Brazil, Portugal, Angola, Mozambique, Saint Tome and Principe, Guinea-Bissau, Cape Verde and Brasilia.*
Question 130 in CLEF 2006 is an example of a question containing different types of NEs: *Which organization did Primo Nebiolo chair during the Athletics World Championships in Gothenburg?*
where the NEs are the following: *Primo Nebiolo* (Person), *Athletics World Championships* (Organization) and *Gothenburg* (Location). Therefore, an accurate NER system able to recognize the necessary NEs in a question and within the documents is of significant relevance in order to build an efficient QA system.

On the other hand, in [44] the authors state that the performance of a QA system can be considerably improved if an NER system is used to preprocess questions which expect an NE as an answer. For this type of questions, it is typically more adequate to use a technique based on automatically creating a list of patterns which help to extract the answers. However, for some questions it is not possible to find good patterns. For instance, for the question *"When did X die?"*, two of the best patterns found to extract the answer were:

- *X died in ANSWER*
- *X was killed in ANSWER*

Such patterns cannot be used to extract the answer because in the place of *ANSWER* both a date and a location could fit perfectly. Thus, in order to avoid such ambiguity in the patterns, the authors have carried out some experiments using an NER system to replace the ANSWER occurrences by a more specific type of NE such as LOCATION, DATE, PERSON, etc. The experiment-set consisted of extracting answers for two sets of questions: (i) questions with the pattern *"When was X born"*; and (ii) with the pattern *"When did X die"*. The ratio of correctly answered questions for the former set of questions was increased from 52% to 53%. Whereas for the latter one it has been increased from 0% to 53%.

Furthermore, in [81] a QA system (named "AnswerFinder") which relies heavily on the quality of the embedded NER system is described. In this research work, the authors experimentally prove that a QA system can obtain a higher benefit from the NER system if the levels of granularity of the NER system and the types of questions match. The reason behind this hypothesis can be explained by, briefly, reviewing the approach used by *AnswerFinder* to extract the answers. Similarly to many other QA system, after analyzing and classifying each question, AnswerFinder proceeds to retrieve the passages where the answer can be found. Once all the candidate passages are extracted, it makes a first filtering of the passages which do not contain the type of answers expected by the user. For instance, let consider the first question of the 217th group of questions in TREC 2007: "*What company produces Jay-Z records?*",

and we presume that the question classification phase correctly classifies the type of answer expected by the user: *name of an organization*. After retrieving the candidate passages, AnswerFinder would tag all the passages and perform the filtering step which would consist of removing all the passages which do not contain any NE of type *Organization*. The authors argue and prove that the efficiency of this method depends highly on two main factors: (i) the first one is that the levels of granularity of both the NEs types considered by the NER system and the filtering module should match as much as possible; and (ii) the recall of the NER module should be considerably high. For instance, if the NER system is not designed to classify the NE type *Organization* then the filtering module might remove relevant passages to extract the answer. In their experiments, the authors have used the TREC 2005 corpus and different types of NER systems. Their results show that when an NER system with the adequate tags-set is used the results are enhances up to 1.3%.

Additionally, on the CLQA side the NER systems are necessary also to avoid a translation of the NEs as common nouns. In [36] and [94] the authors show that a considerable error rate is experimented when the automatic translator does not manage to transliterate properly the NEs.

[8] report a study of the possibility of improving the performance of a **Machine Translation** (MT) by embedding an NER system. The authors have tagged all the text which has to be translated by an NER system as a pre-processing step. Thereafter, the words tagged by the NER system were translated using the methods which are specific for NEs translation. The results showed that this technique outperforms the methods which do not consider tagging the NEs before the translation.

Text Clustering Search result clustering (a sub-task of Text Clustering) is an NLP task focused on clustering in groups the results returned by a search engine. For instance, if we have the documents returned by a search engine for the query *"Michael Jordan"*, in order to make these results easier to explore for the user, a result clustering system would cluster the documents concerning Michael Jordan the

basketball player[3] in one cluster, and the ones concerning the Berkeley professor[4] in another cluster. In [105], the authors report that they have outperformed the existing search results clustering by including an NER system in their global system in order to give a special weight to the NEs in their clustering approach.

3.2 The Named Entity Recognition Task: Definitions

Section 3.1 shows that many research works from different NLP fields report that the use of an NER system to pre-process the documents might help to improve the performance of the global system. It also shows that the classes of NEs handled by the embedded NER system should correspond to the needs of the global system. In this section, we describe the standard definitions of the NER task which have been adopted in the key evaluation campaigns. We also aim at emphasizing the exact differences between those definitions.

3.2.1 The 6th Message Understanding Conference (MUC-6)

The 6th Message Understanding Conference (MUC-6) is a conference sponsored by the Defense Advanced Research Projects Agency (DARPA). In 1995, an NER task has been organized with the goal of encouraging research in the Information Extraction (IE) field. In the NER task, which was held within MUC-6, the organizers defined three sub-tasks [45]:

1. **ENAMEX**: Detection and classification of proper names and acronyms. The classes considered in this sub-task are:

    - *ORGANIZATION*: named corporate, governmental, or other organizational entity such as "Bridgestone", "Mips" or "Language Computer Corporation";

[3] http://en.wikipedia.org/wiki/Michael_Jordan
[4] http://www.cs.berkeley.edu/∼jordan/

- *PERSON*: named person or family such as "Mahatma Ghandi", "Marie Curie"or "Bill Clinton";
- *LOCATION*: name of politically or geographically defined location (cities, provinces, countries, international regions, bodies of water, mountains, etc.) such "Morocco", "Italy" or "Spain".

2. **TIMEX**: Detection and classification of temporal expressions. The classes considered in this sub-task are:
   - *DATE*: complete or partial date expression such as "January 2008", "summer" or "first quarter of 2007".
   - *TIME*: complete or partial expression of time of day such as "5 p.m.", "eleven o'clock" or "12h45 a.m.".
3. **NUMEX**: Detection and classification of numeric expressions monetary expressions and percentages. The classes considered in this sub-task are:
   - *MONEY*: monetary expression such as "9,000 Euros", "million-dollar" or "$16,000".
   - *PERCENT*: percentage such "5%", "20 pct" or "20.3%".

378 "*English*" documents have been used in order to train and evaluate the participating systems [40]: 84.12% of the documents for training, 7.93% as a development-set and 7.93% for test. An on-line annotation was used, Figure 3.2 shows a sample of MUC-6 NE annotation.

The evaluation measure which was used is the $F_{\beta=1}$-measure which can be expressed as:

$$F_{\beta=1} = \frac{(\beta^2 + 1) * precision * recall}{\beta^2 * (precision + recall)} \tag{3.2}$$

Where *precision* is the percentage of correct NEs found by the system. It can be expressed as:

$$precision = \frac{Number\ of\ correct\ named\ entities\ found\ by\ the\ system}{Number\ of\ named\ entities\ found\ by\ the\ system} \tag{3.3}$$

```
Mr.  <ENAMEX TYPE="PERSON">Dooner</ENAMEX> met with <ENAMEX TYPE="PERSON">Martin
Puris</ENAMEX>, president and chief executive officer of <ENAMEX
TYPE="ORGANIZATION">Ammirati & Puris</ENAMEX>, about <ENAMEX
TYPE="ORGANIZATION">McCann</ENAMEX>'s acquiring the agency with billings of <NUMEX
TYPE="MONEY">$400 million</NUMEX>, but nothing has materialized.
```

Figure 3.2: MUC-6 named entity annotation sample

and *recall* is the percentage of NEs existing in the corpus and which were found by the system. It can be expressed as:

$$recall = \frac{Number\ of\ named\ entities\ found\ by\ the\ system}{Total\ number\ of\ NEs} \tag{3.4}$$

3.2.2 Conference on Computational Natural Language Learning (CoNLL)

In CoNLL 2002[5] and CoNLL 2003[6] the shared task concerned language-independent NER. In CoNLL 2002 the participants evaluated their systems on Spanish and Dutch corpora [103] and on English and German data in 2003 [104]. In both evaluations the classes which were taken into consideration are the following:

1. **PER** : Person class, same as *PERSON* class in MUC-6 (see Subsection 3.2.1);
2. **LOC** : Location class, same as *LOCATION* class in MUC-6 (see Subsection 3.2.1);
3. **ORG** : Organization class, same as *ORGANIZATION* class in MUC-6 (see Subsection 3.2.1);
4. **MISC**: NEs which do not fit in the other classes such as "Professional League", "Article 4102" or "Law of Criminal Procedure".

Also, in both evaluations the data were annotated using the IOB2 scheme which is a variant of the IOB scheme introduced by [88]. This tagging scheme rules are as following:

[5]http://www.cnts.ua.ac.be/conll2002/
[6]http://www.cnts.ua.ac.be/conll2003/

- The words which are Outside NEs are tagged as **"O"**.
- The tag **"B-TYPE"** is used for the first word (Beginning) of an NE of class *TYPE*.
- Words which are part of an NE of class *TYPE* but are not the first word are tagged as **"I-TYPE"** (Inside)

Figure 3.3 shows an extract of a IOB2 tagged corpus (from the Spanish corpus used in CoNLL 2002[7]) and Table 3.2 shows details about the size of the different corpora.

el O
ministro O
de O
Comunicaciones B-MISC
de I-MISC
Brasil I-MISC
, O
Joao B-PER
Pimienta I-PER
da I-PER
Veiga I-PER

Figure 3.3: An extract of an IOB2 tagged corpus

The $F_{\beta=1}$-measure, F-measure for short, has been used as a measure for evaluation (see Equations 3.2, 3.3 and 3.4).

3.2.3 Automatic Content Extraction (ACE)

In the ACE evaluations there are two main tasks: Entity Detection and Recognition (EDR) (Entity Detection and Tracking, EDT, in ACE 2003) and Relations

[7]http://www.cnts.ua.ac.be/conll2002/ner/data/

Table 3.2: Size of corpora used in CoNLL 2002 and 2003 evaluations

Corpus		Total number of Tokens	Training	Development	Test
CoNLL 2002	Spanish	380,923	71.67%	14.4%	13.93%
	Dutch	333,582	65.57%	12.18%	22.25%
CoNLL 2003	English	301,418	67.55%	11.51%	20.94%
	German	310,318	66.68%	11.05%	22.27%

Detection and Characterization. Within both tracks the considered languages are English, Chinese and *Arabic*. The EDR task is an extension of the NER task and it requires much more than the identification and classification of the NEs within a text. However, we find it very important to give a brief description of the task in order to emphasize the importance of NER for EDR and to give an overview of the tag-sets used in the ACE evaluations because we are using ACE data in our experiments.
The EDR task consists of two sub-tasks:

- Mention Detection (MD): which consists of the recognition and classification of all the "pronominal", "nominal" and "*named*" mentions of entities in the text. For instance, if we consider the sentence "John has resigned from his position as a manager of Walmart", the MD system would have to capture the following mentions: (i) "John", a named mention of a person; (ii) "his", a pronominal mention of a person; and (iii) "manager", a nominal mention of a person; and (iv) "Walmart", a named mention of an organization.

- Coreference Resolution (Coref): in this subtask, the aim is to link the mentions which refer to the same entity. For instance, in the example which we have given to illustrate the goal of an MD system, the Coref system should be able to determine that the extracted mentions refer to two entities, namely: {John, his, manage} and {Walmart}.

The entity types which have been taken into consideration in ACE 2003 [3] are the following:

1. **Person**: Person class, same as *PERSON* class in MUC-6 (see Subsection 3.2.1);

2. **Organization**: Organization class, same as *ORGANIZATION* class in MUC-6 (see Subsection 3.2.1);

3. **GPE (Geo-Political Entity)**: Politically defined geographical regions. It includes nations, states and cities. For instance, "Cubans" or "Iraq" in a context such as "Iraq has been attacked ...";

4. **Location**: Geographical entities with physical extent. It includes landmasses, bodies of water and geological formations. Which is equivalent to the *LOCATION* class in MUC-6 (see Subsection 3.2.1);.

5. **Facility**: Human-made buildings. It includes houses, factories, stadiums, etc. For instance, "Empire State building" or "Santa Monica Hospital".

In the ACE 2004 (and the evaluations which came after [4][5]) two new entity types were introduced which are the following:

1. **Vehicle**: All types of vehicles (land, air, subarea-vehicle, etc.) such as "F-16" or "Hummer H3x".

2. **Weapon**: All types of weapons (biological, chemical, nuclear, etc.) such as "Anthrax" (a biological weapon) or "Nodong-1" (a missile).

The detection and classification of *named* mentions, in the *MD* sub-task, is equivalent to the NER tasks of the CoNLL and MUC competitions. In our research work we have focused on this type of evaluation and we have conducted experiments with data which have been used in ACE evaluations and data which we have produced ourselves taking into consideration the CoNLL corpora as guidelines. In Section 6 we will describe more details about the data which we have used in our experiments (details about the size of the ACE 2003, 2004 and 2005 can also be found).

In order to measure the efficiency of the learning systems participating in the evaluation, the organizers have chosen a measure, named *"ACE-value"*. This measure assigns different weights to different types and levels of mentions, and different

penalities to the types of errors of the MD and Coref systems. The named mentions are assigned the most important weight which is a direct proof that an improvement of the NER task would have as a direct consequence an the improvement of the global EDR system.

In our experiments we will use the F-measure because the "ACE-value" is only applicable for the EDR task.

3.3 The Named Entity Recognition Task State-of-Art

In order to present the most important research works which have been carried out in the NER task, we will present first some characteristics of the systems which have participated in MUC-6, CoNLL 2002 and 2003. Unfortunately, the results and proceedings of the ACE evaluations are not available. However, we will describe other research works which have not participated in competitions but are worth to be presented because of the richness of their investigation.

3.3.1 MUC-6

According to [82], research works in NER have started in 1991. However, the publication rate has really accelerated only since 1996 with the first major event dedicated to the task, i.e. MUC-6. Particularly, in this event more than half of the participations have obtained, in the NER task, an F-measure above 90. The best three participations are the following:

1. *Systems Research and Applications (SRA)* has participated with SRA system [62] which is composed of two main parts. Unfortunately, the part which was oriented to the NER task is the commercial system NameTag^{TM}(which was integrated in the global system). Only the commercial features of "NameTag^{TM}" are presented in the paper. The four official runs of SRA system have obtained the three first and ninth best results in the MUC-6 participations. Those runs are:

(i) "BASE" (96.42) which makes the maximum analysis, it obtained the best results; (ii) "FAST" (95.66) makes less analysis than the previous run but takes less time; (iii) "FASTEST" (92.61) is the fastest mode but obtains the lowest performance in comparison with the first two modes; and (iv) "NO-NAMES" (94.92) is the same as "BASE" but decreasing the tags-set of "NameTag™", because it is prepared to tag more NEs than those required in the competition.

2. *SRI international* has participated with the FASTUS system [7]. This system is based on a series of transducers which transforms text from sequences of characters to domain templates. The authors report that the FASTUS system has obtained good results in many IE sub-tasks (extraction of information on: terrorist incidents, joint ventures, etc.), for different types of texts (open-domain, military texts, hypertext, etc.) and for two languages: English and Japanese. In its participation in the MUC-6 competition, FASTUS has obtained F-measure=94 which makes it rank as second best participant and forth best run.

3. *BBN Systems and Technologies* has participated with the PLUM System [10]. This system obtained F-measure=93.65 and was situated in the third rank as participant and fifth rank as a run. For the NER task, the PLUM system combines a statistical approach (based on Hidden Markov Models) which uses morphological information (extracted by a morphological analyzer), together with lexical patterns (extracted automatically).

3.3.2 CoNLL 2002 and 2003

Differently from the MUC-6 competition (focused only on the English language), in the CoNLL 2002 and 2003 the goal was to encourage people to investigate language-independent approaches for the NER task. For this reason, in both years two corpora of different languages (Spanish and Dutch in 2002 and English and German in 2003) were used. In this subsection we will emphasize the best participations of the CoNLL competitions:

CoNLL 2002 The *baseline* model consisted in assigning to each word the class which it was mostly assigned in the training corpus. The words which have not been seen in the training phase are tagged as outside words ("O"). For the Spanish language the performance obtained with the baseline model (35.86) was lower than for the Dutch language (58.28). However, the overall results obtained for the Spanish language (between 60.97 and 81.39) were higher than the ones obtained for the Dutch language (between 49.75 and 77.05) [103].

The best system [22] was based on a 2-step approach where the two steps are performed sequentially and independently. The first step is fully dedicated to the detection of the NEs existing within the text, and the second step deals with the classification of each of the NEs detected in the first step. In both steps the authors have used an Adaptive Boosting (AdaBoost) [41] approach. The first step is a combination of three modules:

1. **BIO**: it tags as "B" the first word of an NE, as "I" a word inside an NE and the words which are outside the NEs are tagged as "O".

2. Open-Close&I: it detects the word which opens an NE and the word which closes it. This approach uses three classifiers: One classifier for the NE openers, a second classifier for the NE closers and a third one for the words inside the NE.

3. Global Open-Close: it also uses the classifiers used in "Open-Close&I" to detect the the words which open an NE and the ones which close it. Thereafter, an inference mechanism is used to infer the inside word of the NEs

The authors also report that they have used different types of features which include: POS-tags, lexical features, a context window, trigger words, a gazetteer as an external resource, etc. Thanks to the combinations of these techniques the system has achieved the best performance for both the Spanish (81.39) and Dutch (77.05) languages.

The second best participation in CoNLL 2002 [38] is based on a stacking-based approach which uses three modules using three different algorithms (the output of each module is passed as input to the next one). The three modules are:

1. **Transformation-Based Learning (TBL)**: it is an error-driven approach which starts by assigning a class to each word. Then makes the *transformations* that most decrease the number of errors.

2. **Sparse Network Of Winnows (Snow)**: it is also an error-driven approach but differs from TBL in that the changes to make decrease the errors are made on the features weights. This second module of the system improves the classification made by the previous module on the words which present a strong feature interaction.

3. **Forward-Backward algorithm (FB)**: it is used to compute a global-best entity type assignment

[38] has obtained the second best results in the Spanish language, also ranked second in the overall results and third in the Dutch language.
The third best participation was obtained by [27]: the system obtained 77.15 (third best results) for the Spanish language and 72.31 (ranked fifth) for the Dutch language. The authors have used a bootstrapping approach which consists of extracting all the NEs and the contexts ("seeds") from the training data first, thereafter these seeds are stored in character-based tries. An Expectation-Maximization (EM) algorithm is used then to iteratively learn the contextual patterns which are strongly associated to the seeds. Therefore, more members are added to the tries. This EM-based bootstrapping algorithm aims at building a large character-based tries for both NEs (internal evidences) and contexts (external evidences) which would allow a fast identification and classification of the NEs in the test-set.

CoNLL 2003 **The best participation** in CoNLL 2003 was the one described in [39]. The system was an improved version of the system that the authors have used in the participation of 2002 [38]. The major change in their system was combining the old system with other classifiers such as HMM, Maximum Entropy (ME) and Robust Risk Minimization (RRM) [111]. The authors report that they have used a bigger set of features. In order to combine the classifiers, the authors have used a linear interpolation of the classifiers which can be expressed by the following equation:

$$P(C|w, C_1^n) = \sum_{i=1}^{n} P_i(C|w, i, C_i).\lambda_i(w) \tag{3.5}$$

where $P_i(C|w, i, C_i)$ is the estimation the the correct class is C, given that the output of the classifier i on a word w is C_i. Whereas $\lambda_i(w)$ is the weight of the classifier i for the context of the word w. The authors argue that the results for the English language could benefit (17-21%) more than German. Their final results were 88.76 for English and 72.41 for German, best results in both languages.

In [48] **the second best** system is described. Their system is fully ME-based (in Chapter 4 we will describe in detail ME-based NER). Their feature-set contained a considerable variety of types of features; some of these features are:

- Words occuring in more than 5 documents;
- Unigram and bigram contexts of the NEs;
- Lexical features of both NEs and contexts;
- Words between quotes or brackets;
- Capitalization.

In addition, they have run two experiments: the first one without using any external resources and the second one using a lexicon as an external resource. For the English the results using the lexicon (88.12) were almost two points higher than those obtained without using an external resource (86.83). However, the results were higher when no external resource was used for German (77.05 vs. 76.83).

Finally, the system of [57] was ranked as **the third best participation** in CoNLL 2003. Their approach represents data on a character-level HMM as illustrated in Figure 3.4.

This approach relies heavily on internal evidences of the NEs. For this reason, the authors made a second run where they used more contextual information. They report that using context helped to increase the performance more than one point higher. Their final results for English were 86.07 (third best results) and for German

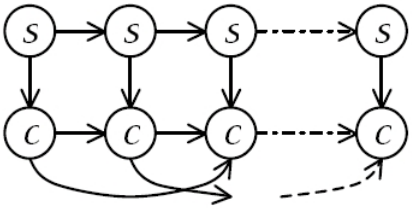


Figure 3.4: Character-level HMM. c are the character observation and s are the entity types

71.9 (second best results) which proved that the approch is highly independent from the language.
Finally, we would like to remark that 5 of the 16 participations in CoNLL 2003 have used the ME approach. Additionally, in the same year, the system described in [77] was one of the first applications using Conditional Random Fields (CRFs) for the NER task (details about CRFs are given in Chapter 4).

3.3.3 Other Important Research Works

Even though ranked ninth in CoNLL 2002, [74] presents an interesting comparison between HMM and ME. The author shows that the F-measure obtained with ME (59.50) is 16 points higher than HMM (43.50). The F-measure increased more than 11 points (72.88) when more features were added. The author argues that when he used a list of 13,821 first names collected from the web as an external resource the results slightly decreased (72.44).

More recently, in her thesis dissertation [59], the author has also carried out experiments on NER from a language-independent perspective. The NER system discussed in this thesis employs a two-step approach where the first one detects the NEs within the text and the second one classifies them (using the same class-set used in CoNLL). For each of these two steps, the system uses three classifiers and combines their outputs using a voting approach. This voting approach assigns the most voted class to the token. In case each of the three classifiers has assigned a different class, the approach selects randomly one of the three outputs as the final class. The three clas-

sifiers are based on both supervised and unsupervised ML approaches. They use the same feature-set, however, which consists of:

- Contextual;
- Orthographic: including capitalization, token position in the sentence, etc.;
- Gazetteers;
- Trigger word, e.g. president, capital, etc.
- Morphological: aims mainly at marking the common prefixes and suffixes of the words.

The gazetteers were extracted automatically using an algorithm based on pattern validation and graph exploration techniques. When the approach was evaluated on CoNLL 2002 data-set, it has ranked third in comparison with the participating systems (F-measure=78.59). However, it has yielded an F-measure of 85 when it was evaluated on other data-sets (EVALITA-2007).

Finally, in [107] the authors have conducted a comparative study between CRFs and Support Vector Machines (SVMs) for the Vietnamese NER task. In their study the authors have used different types of features and internal and external evidences were taken into consideration. They also report that one of the major problems in the Vietnamese NER task is the spelling variations of the NEs. For the evaluation they have manually annotated 500 newspaper articles (156,031 tokens) from different fields (society, health, sport, politics, etc.). The results showed that SVMs (87.75) performs better than CRFs (86.48). The authors also report results for SVMs where they have used different context window sizes. However, they have not used the same sizes for CRFs. For the Hindi language, in [70] the authors have used CRFs training and feature induction. They have achieved F-score=82.55. For the Chinese language, [56] have used CRFs models for both word-segmentation and NER. Even though a restricted number of features were used, the reported results show that an F-measure of 83.10 was achieved. [51] report a research work aiming at morphologically rich

languages in general. They have used the Bengali language as a case-study. In their work, they have used an affixes induction algorithm and an automatic technique to extract a Bengali lexicon. They contrast in their evaluation the performance of a baseline system which uses only lexical features with a system which performs affixes separation and uses the wikipedia-based lexicon as a feature. Their results show that they have achieved an F-measure (71.99) is 6.42 points higher than the baseline system (65.57).

3.3.4 State-of-art of Arabic Named Entity Recognition

Apart from the Arabic NER systems which participated in the ACE evaluations (of which we do not have any information because the results and the proceedings are not published), there are few research works which are oriented to the Arabic NER task. Those works are the following:

1. [73] presents an Arabic NER system which first makes a morphological analysis and POS-tagging of the text and then a pattern-matching engine is used to identify and classify the NEs within the text. The list of patterns used for this purpose is composed of both NE structure patterns and contextual ones. The authors have considered the same tag-set which has been used in the MUC-6 competition (see 3.2.1). In their paper, the authors report both the performance of the POS-tagger (F-measure=82) and the NE recognizer (F-measure=85). It is important to point out that they state that the F-measure obtained for the categories "NUMBER" and "TIME", respectively, 97.8 and 85.5.

2. [2] uses an approach which is fully based on "heuristic" rules. However, as the authors explain, it is not possible to detect the boundaries of the NEs in Arabic text because of the lack of capital letters. Hence, it is not possible to extract the NEs with only rules. In order to tackle this problem the following heuristic was used: if two words appear next to each other in the corpus more than n times then those words have a "strong relationship" and belong to the same name phrase. The evaluation was performed on a set of 500 articles of *Raya*

newspaper and the authors report an overall precision of 91.9%. The results details show that the authors aimed at person, organization and location, NEs. However, no details were given about the corpus annotation.

3. [114] addresses the EDR task as it was defined in the ACE competitions (see 3.2.3). The authors report that their system performs a first step for MD. Thereafter, it performs a second step for Coref in order to cluster the entities mentions which refer to the same object. The system employs an ME-based approach in both steps. The main subject of the reported work was to show the impact of using the stem of each word as a feature and measure its impact on the MD and Coref subtasks for the Arabic language. For this purpose, the authors have used two different baseline systems: (i) an MD system which uses only lexical features; and (ii) an MD system which uses lexical and syntactical features, as well gazetteer information. The authors have used the ACE 2004 corpus for evaluation, and they report an improvement of 1.1 F-measure points was achieved for the first baseline system (65.8 vs. 64.7), whereas only 0.4 points of improvement have been obtained for the second one (69.2 vs. 68.8).
   This research work clarifies by emperical results that languages with a complex morphology, such as Arabic, need an additional effort to tackle the problems induced by their morphlogy in comparison with other languages such as English.

3.4 Concluding Remarks

The aim of this Ph.D. document is investigating how to approach Arabic NER in a reliable and efficient way using standard data and comparing the results with the achieved state-of-art. For this purpose, in this chapter we have investigated the importance of NER from the perspective of other NLP tasks such as IR and QA, the different standardizations and the state-of-art of the task. We have described different aspects showing that:

- From a statistical view point, the NEs represent a huge quantity of information. In order to provide an empricial proof of this statement, we have conducted an

experiment which shows that in a text of more than 170k words, NEs ranked the second most informative words category. Right after the verbs category which represent the actions in a text.

- NER plays a key role for many NLP tasks. We have reported research works which have proved that in order to improve the state-of-art of IR and MT, and mainly to be able to build a QA system, an efficient NER module is necessary. It is important to have an NER system that considers a tag-set which matches the needs of the of the global system.

- The definition of the NER task has been consistent in all the NER competitions and it has been defined as the detection and classification of the NEs existing with an open-domain text. However, the tag-set taken into consideration and the measure employed to compare systems performance have varied from competition to competition.

- The state-of-art of the NER task, shows that the MUC-6 and CoNLL 2002 and 2003 competitions are a precious mine of research works oriented to NER. In the proceedings of these competitions, a considerable variety of techniques have been explored. They show that the most efficient Machine Learning (ML) techniques for the NER task are ME, SVMs and CRFs. Moreover, some of the most efficient system have either: (i) employed approaches which are based either on a 2-step approach in which the first step only detects the boundaries of the NEs and the second one classifies them; or (ii) combined different ML techniques in order to benefit from the advantages of different modeling approaches.

- Finally, with respect to the Arabic NER task, it needs to be remarked that it is very few investigated. The published works have confirmed that due to its complex morphology, the Arabic language requires a further investigation of the adequate techniques to tackle the problems induced by its morphology. The only evaluation campaign which allows the Arabic NLP research community to compare their approaches on the same Arabic corpus (ACE) does not publish neither the results nor the proceedings.

Chapter 4

Statistical Modeling Approaches for Named Entity Recognition

Consider for example the problem of classifying a vegetable as a member of 5 possible classes: tomatoes, carrots, zucchinis, onions and pumpkins. If we represent these classes using the color representation, tomatoes will be identified without ambiguities, but carrots and pumpkins will be mixed. If we choose instead a shape representation for all classes, tomatoes and onions could be confused, while pumpkins would be unambiguously identified, and so on. More generally, we can say that for each class there is at least one representation that captures the best of the "essence" of that class, and thus makes that class easily recognizable between others.

-Caputo and Niemann (2002)-

In the 17th century, one of the most famous "games of chance" in Paris was a game called "the points". It consisted of throwing the dice and counting the number of points obtained: the winner would be the one who has obtained a certain number of points first. When, the French writer and philosopher , "Antoine Gombaud"[1] had to leave one of his "points" matches unfinished, he found it unfair to receive the same amount of money as his adversary because he was claiming that he had more "probabilities" to win[2]. In order to prove his rightness he asked for the help of one of the greatest mathematicians at the time, Blaise Pascal[3]. Gombaud never suspected that he was giving Pascal the chance to establish the formalism of what will be one of the greatest theories in the next years and even in the next century, i.e., the *probability theory*. In the literature, it is cited that Pascal exchanged a considerable correspondence with his contemporary mathematician "Pierre de Fermat"[4] about this theory. Both Pascal and Fermat could solve the "problem of points" individually but more interesting questions were discussed in their correspondence in order to give a complete mathematical proof to all the concepts involved in the probability theory, which had been up to that date "intuitive".

Thomas Bayes[5], one of the most famous mathematicians of the 18th century, added a new concept, called *Bayes' theorem* or *Bayes' Law*, in the framework of the probability theory. Bayes' theorem stands that:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{4.1}$$

Which can be read as: The probability of occurrence of an event "A", given the occurrence of some event "B" ($P(A|B)$), equals the product of the prior probability of A ($P(A)$) and the probability of B given A ($P(B|A)$), divided by the prior of B ($P(B)$). That is to say, that studying statistically the probability of appearance of the event A and B separately and, finally, the probability of appearance of the event A when given that B already appeared, with Bayes' theorem it is possible to

[1] http://en.wikipedia.org/wiki/Antoine_Gombaud

[2] In another version of this story, the name of the gambler is "Gerolamo Cardano" instead of "Antoine Gombaud".

[3] http://en.wikipedia.org/wiki/Blaise_Pascal

[4] http://en.wikipedia.org/wiki/Fermat

[5] http://en.wikipedia.org/wiki/Thomas_Bayes

compute the probability of that event A "will" happen when B is "observed". In other words, one is able to estimate the probability of the occurrence of an event A based on its historical "data" and its "dependency" to another event B. For instance, let consider the problem of predicting whether tomorrow will be *rainy* or not given that today is a *cloudy* day. Let also suppose that we have the weather statistics of the previous year (a 365 days year) which show that last year there have been 130 rainy days, 200 cloudy ones, and 120 rainy days had a cloudy weather the day before. Intuition would suggest to calculate the probability of having a rainy day tomorrow as $120/130 \simeq 92\%$, whereas the right answer, which can be computed by the Bayes' Law, is as following:

$$\begin{aligned} P(rainy_{tomorrow}|cloudy_{today}) &= \frac{P(cloudy_{today}|rainy_{tomorrow})P(rainy_{day}))}{P(cloudy_{day})} \\ &= \frac{(115/130).(130/365)}{(200/365)} \\ &\simeq \frac{0.92 * 0.35}{0.54} \\ &\simeq 0.6 \end{aligned}$$

In the framework of *pattern recognition*, the main issue is to determine the probabilities of an object x to belong to each class of a set of n classes c_i, where $i = 1..n$, given the historical data of x and its features f. The approaches which suggest to compute this probability by making statistical observations on a number of objects x_i and their features are called "supervised" (or "classification" methods). One of them is the "Naive Bayes" approach which is based on the Bayes theorem described above. The Naive Bayes approach considers that the event $P(A|B)$ (see Equation 4.1) can be read as the probability that x belongs to the class c (event A), given the observation of a set of features (event B). Hence, in order to be able to use the Bayes theorem (or any other supervised technique) it is necessary to have a set of objects for whom we know the class (called "annotated data") in order to compute the different elements of the equation ($P(A|B)$, $P(A)$ and $P(B)$). This step is called in the pattern recognition framework "training". On the other hand, in order to estimate the accuracy of our system we need another set of *annotated data*. The characteristics of the objects of this set together with the Bayes formula will be used to classify each of the objects,

whereas the classes annotation will be used to estimate the accuracy of the classifier. This step is called the "test" step.

The supervised algorithms differ from each other in that they attempt to tackle the classification problem from different viewpoints. Even though most of these approaches are claimed to be task-independent, proof is abound in the literature that the choice of the ML approach, for any task, is a basic and essential step. For instance, if we consider the case of Naive Bayes classifiers, [92] describes a research study on using this classifier on different types of problems. In order to carry out this study, the author has simulated each type of problems with the type of features which tend to be available. The study shows that a Naive Bayes classifier performs best when the features are either *completely independent* or *functionally dependent*.

We have used the example of Naive Bayes to introduce this chapter because it is simple and shows better the task of a classifier in general. The rest of the chapter will be more focused on classification approaches which have been reported in the literature to be efficient for sequence modeling and NLP tasks in general and for the NER task in particular. These approaches are: Maximum Entropy (ME), Conditional Random Fields (CRFs) and Support Vector Machines (SVMs) whose descriptions and state-of-art in NLP tasks, with a special focus on the NER task, are presented in Sections 4.1, 4.2 and 4.3, respectively.

4.1 Maximum Entropy

4.1.1 The Approach

The ME approach has been introduced by E. Jaynes in 1957 in [71]. The author argues that the priniciple of this approach is an extension of Laplace's "Principle of Insufficient Reason" [54]. The aim of this approach is to provide the *least biased* possible statistical inference model. For instance, if we have a vegetable v and we want to classify it as: tomato (T), carrot (C), zucchini (Z), onion (O) or pumpkin (P) and we do not have any information about v then we assign the same probability to all the classes and thus our probability distribution will be as the one shown in

Table 4.1.

Table 4.1: ME model probabilities distribution in case of absence of all information for a vegetable v

Class	Probability
T	0.2
C	0.2
Z	0.2
O	0.2
P	0.2

This probability distribution (see Table 4.1) is the *unique* distribution for which the entropy is *maximal*. Let suppose that we have to re-compute the probability distribution because a statistical study showed that 60% of the vegetables having the colour *orange* are either pumpkins or carrots. And let suppose that v has the orange colour. In order to keep the maximum entropy we should adopt the following distribution (see Table 4.2): Hence, every time we have a new information about v

Table 4.2: ME model probability distribution in case we know that the vegetable v is orange

Class	Probability
T	0.13
C	0.3
Z	0.13
O	0.2
P	0.13

we have to look for the, unique, probability distribution which introduces the less

biases possible and thus provides the maximum entropy. It has been very simple to re-compute the probability distribution in the example which we have presented because the example was simple and we have studied the cases of "no information" and "one information". When the number of classes (information) is bigger, the manual computation of the probability distribution becomes much more complex and an automatic algorithm is necessary. In order to present the solution proposed by Jaynes in [54] we have to first reformulate our problem as a classification problem. We have to classify an object x_i with $i = 1, ..., N$ as belonging to one of the classes c_j with $j = 1, .., M$. We know that x_i presents the features $f_k(x_i, c_j)$ with $k = 1, ..., F$. These "features functions" (called "features" for short) are the information which we know about x_i. However, even if these information might have not appeared in the training phase, we still have to rely only on the probability distribution which provides a maximum entropy *to what have been seen in the training phase*. In our research work, we are most of all interested in binary features. Following, we give an illustrating example of a binary feature, which relies on the example which we have given in Tables 4.2 and 4.1:

$$f_k(x_i, c_j) = \begin{cases} 1 & \text{if } c_j = P \text{ and } colour(x_i) = orange \\ 0 & Otherwise \end{cases}$$

In [54], it is shown the final formula which describes $p(c_j|x_i)$ is as follows:

$$p(c_j|x_i) = Z(x).exp(\sum_{k=1}^{F} \lambda_k f_k(x_i)) \tag{4.2}$$

where Z(x) is a normalization factor which can be formulated as:

$$Z(x) = \sum_{j=1}^{M} exp(\sum_{k=1}^{F} \lambda_k f_k(x_i)) \tag{4.3}$$

where λ_k are weights for the features. Therefore, each information has a certain weight for a certain class and thus the information $colour(x_i) = orange$ will have more weight for the classes P and C than for the other classes. It is possible to compute the λ_k using the Generalized Iterative Scaling (GIS) or the limited memory BFGS (L-BFGS) among other algorithms.

4.1.2 Maximum Entropy in the framework of Natural Language Processing

In this subsection, we present some research works which have used ME models for NLP tasks. The most important works which use ME models for the NER task have already been presented in Chapter 3.
To our knowledge, [93] is one of the first applications of ME to the NLP tasks. This thesis work aimed at building language models for the English language. Three years later, [89] built a POS-tagger based on the ME approach. The authors used the *Wall Street Journal* (WSJ) data for training and testing. Some of the features which have been employed for the ME model are:

- The current word;
- Prefix and suffix;
- Contains number or a hyphen;
- Contains upper case;
- etc.

The authors report that they have achieved an accuracy of 96.6% which outperforms the accuracy of the existing POS-taggers. The ME approach was employed also for building a Base Phrase Chunker [58], the author has used:

- The current word;
- Context (surrounding words);
- POS-tags;
- POS-tags of surrounding words.

as features. The WSJ corpus has been used for both training and evaluation and the results achieved an F-meausre of 91.97. In [90] the authors use an ME approach to detect sentence boundaries. Two experiments have been carried out: the former

one using domain-specific keyword (English financial newspaper) and the second one is a more generic model oriented to all types of text. In order to train their systems the authors have used 39,441 sentences from the WSJ articles. For the evaluation, they have used other WSJ articles for the first experiment and the Brown corpus for the second one. The authors report that they have used *binary* features of the tokens (words) which mark the boundaries of a sentence, some of these sentences are:

- Prefix and suffix;
- Title (Mr., Dr., etc.) or corporation (Corp., S.p.A., etc.) designator;
- Context (left and right word);
- etc.

The domain specific experiment results achieved a 98.8% of accuracy, whereas in the open-domain texts experiment the obtained accuracy was 98.0%. These results show that the approach is highly portable and the authors argue that it almost reaches the state-of-art (99.8%) even though they have not used as much training data as the system they compare their results with [91].

4.2 Conditional Random Fields

4.2.1 The Approach

CRFs are a generalization of Bayesian Networks (see Appendix A). They are *undirected* graphs whose mathematical properties are exploited for probabilistic inference. In [65], they are defined as follows:

Definition. *Let $G = (V, E)$ be a graph, where V is the set of vertices and E is the set of edges, such that $Y = (Y_v)_{v \in V}$, so that Y is indexed by the vertices of G. Then (X, Y) is a conditional random field in case, when conditioned on X, the random variables Y_v obey the Markov property with respect to the graph:*
$p(Y_v|X, Y_w, w \neq v) = p(Y_v|X, Y_w, w \sim v)$, *where $w \sim v$ means that w and v are*

neighbors in G.

As a logic consequence of a representation based on vertices which are connected by edges, there will be a composition of *cliques*. In graph theory, a clique is defined as a complete sub-graph whose vertices are all adjacent (see Figure 4.1). CRFs define

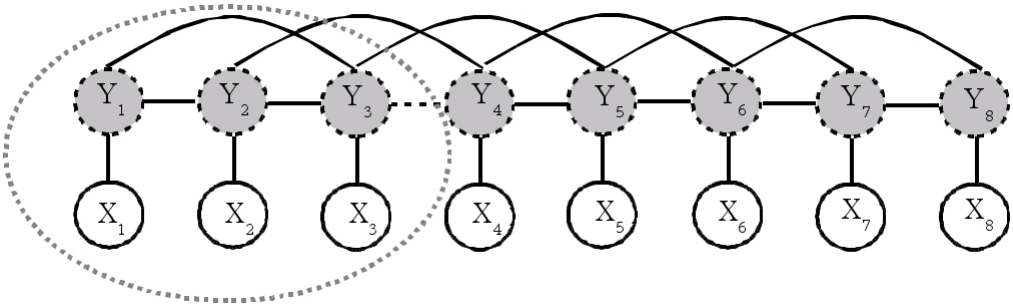


Figure 4.1: Illustrating example of a CRFs graph. A clique is circled

features across cliques and assign a joint distribution over a *label sequence* which can be expressed as following:

$$p_\theta(y|x) \propto exp(\sum_{e \in E,k} \lambda_k f_k(e, y|_e, x) + \sum_{v \in V,k} \mu_k g_k(v, y|_v, x)) \tag{4.4}$$

where x is a data sequence, y is a label sequence, and $y|_s$ is the set of components of y associated to the vertices in a clique S; f_k are the edges features and g_k are the vertices features (both f_k and g_k are given and fixed); λ_k and μ_k are the weights for f_k and g_k, respectively, and they are computed in the training phase.

The major advantage of CRFs over BNs is that CRFs do not need to consider

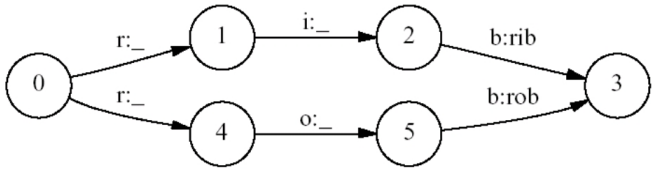


Figure 4.2: Illustrating example of the label-bias problem

the conditional independencies: thus instead of taking into consideration only the

near past to predict the future, the statistical model takes the whole sequence into consideration. CRFs also allow to handle multiple interacting features which is not possible with Hidden Markov Models (HMMs). CRFs also present a major advantage for solving the *label-bias problem* with respect to the other approaches. We use the same example given in [65] to explain the label-bias problem. As shown in Figure 4.2 both states 1 and 4 have only one output. Therefore, if we suppose that the observed sequence is *r o b*, the model will assign the same probability to the states 1 and 4 when the character *r* will be observed. Thereafter, when the second character (i.e. the character *o*) will be observed, even though it has been seen in the training by state 4 and not by 1, both states have to pass all their masses to their only outgoing transition. Thus, the top and the bottom paths shown in Figure 4.2 are equal (or slightly different) independently of the observations. CRFs solve the above problem because the whole sequence is taken into consideration; additionally a sequence is labeled taking into consideration its neighbour labels.

4.2.2 Conditional Random Fields in the framework of Natural Language Processing

[98] is a research work which describes how a shallow parser (BP chunker) is built using a CRFs model. In their research work, the authors have used two methods for training:

1. The iterative scaling method.

2. The Conjugate-Gradient (CG) method which is an iterative method used to solve linear and non-linear equations optimization. It relies on the hypothesis of combining the gradient with the previous direction instead of following the gradient [101].

The authors have used the CoNLL 2000 corpora (for both training and evaluation), the annotation scheme used for this corpus was IOB2. In addition to the tokens and the tags, the corpus contained also the POS-tags. In [86], CRFs have been used for extracting information from tables. This task is very peculiar because it requires to

solve both problems of layout and language. The authors define the table extraction task as the overlap of six sub-tasks:

1. Locate the table;
2. Identify the row positions and types;
3. Identify the column positions and types;
4. Segment the table into cells;
5. Tag the cells as data and headers;
6. Associate data cells with their corresponding headers.

For the first and the second sub-tasks the authors have used CRFs, HMMs and ME models in order to be able to make a comparison. For the CRFs model the authors have used two different training approaches: (i) the GIS approach (CRF baseline); and (ii) CG which we have mentioned previously. Documents were extracted from *http://www.FedStates.gov* for both training (5,764 table lines) and test (3,607 table lines). The obtained results were F-measure=91.8 when the CRFs models were used whereas F-measure=88.7 was obtained for the ME model and F-measure=64.8 for the HMMs model.

Another important research work which employs CRFs for IE is described in [61]. The aim of this research was to build a system which may assist a user to fill a database fields from raw text or Web files. The authors introduce the *Expected Number of User Actions* (ENUA) measure in order to evaluate their system. This measure consists in counting the number of the user's actions (e.g. clicks) in order to fill the database correctly. The results showed that using a CRFs model helps to reduce the ENUA up to 13.9% whereas a ME model caused an increase of 29.0% of the ENUA.

CRFs models have proved to be very efficient in the biomedical NER task as well [97]. The author has used two categories of features:

1. Lexical: alphanumeric, roman numeral and dash characters;
2. Semantic: A list of semantic domain knowledge rules.

In the first experiment the author has used only the lexical features and achieved an F-measure of 69.8. When both lexical and semantic features were used, the author reports that an F-measure of 69.5 was *incomprehensively* obtained. However, it is also reported that using semantic features, helped to capture a special type of entities (*RNA* and *CELL-Line*) which less appear in the corpus.
As we have mentioned in Chapter 2, the NER task is the research field which most benefited from the CRFs statistical model. [56], [107] and [70] (see Chapter 2) are all research works which prove the efficiency of the CRFs model for the NER task. In all of them, the authors have used the iterative scaling training and report promising results.

4.3 Support Vector Machines

4.3.1 The Approach

In order to make a correct description of the SVMs approach, a review of the *"Linear Discriminant Functions"* (LDFs) is necessary [33]. Let remind that we want to tackle the following problem.
We know that:

- The objects to classify can only belong to one of M classes;
- Each object is represented by its features vector which belongs to $\mathbb{R}^d$;
- The classification rule is: $\alpha : \mathbb{R}^d \rightarrow \{1, ..., M\}$
- The discriminant functions are: $\vec{g} : \mathbb{R}^d \rightarrow \mathbb{R}^M$

LDFs propose the following solution:

$$\begin{aligned} g_y(\vec{x}) &= w_{y_0} + <\vec{x}.\vec{w_y}> \\ &= w_{y_0} + \sum_{i=1}^{d} w_{y_i}.x_i \ ; \ 1 \leq \ y \ \leq \ M \end{aligned}$$

where $\vec{w_y}$ is the weights vector and w_{y_0} is the *threshold weight*. Thus, in the simplest case, when $M = 2$ we obtain two functions $g_1, g_2 : \mathbb{R}^d \rightarrow \mathbb{R}$ and the classification rule would be as follows:

$$\alpha(\vec{x}) = \begin{cases} 1 & \text{if } g_1(\vec{x}) > g_2(\vec{x}) \\ 2 & \text{if } g_2(\vec{x}) > g_1(\vec{x}) \end{cases} \tag{4.5}$$

To make it simpler, we define:

$$g(\vec{x}) = g_1(\vec{x}) - g_2(\vec{x}) \tag{4.6}$$

and thus we can say that the classification rule is:

$$\alpha(\vec{x}) = \begin{cases} 1 & \text{if } g(\vec{x}) > 0 \\ 2 & \text{if } g(\vec{x}) < 0 \end{cases} \tag{4.7}$$

Since $g_y(x)$ is linear, the decision boundary is a *hyperplane* $H : g_y(x) = 0$. The

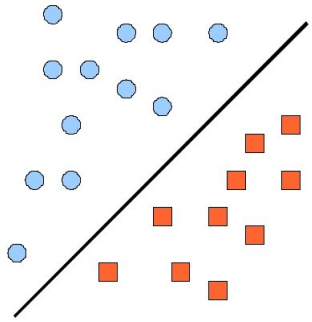


Figure 4.3: Illustrating figure of linear hyperplane

orientation of H is determined by $\vec{w_y}$ and its location by w_0. Figure 4.3 shows an illustrating example of objects and a boundary decision. Hence, the distance of an object from the H may be defined as:

$$r = \frac{g(\vec{x})}{||\vec{w}||} \tag{4.8}$$

Therefore, in case of a two classes linearly separable objects, it is easy to tackle the classification problem by looking, in the training phase, for the the hyperplane which

best separates the objects of the two classes. The problems becomes more complicated when $M > 2$. Instead of having one hyperplane H which separates the objects of two classes, we have a hyperplane $H_{i,j}$ for each two classes C_i and C_j (Figure 4.4 gives an illustrating example). These hyperplanes can be defined as $H_{i,j} : g_i(x) = g_j(x)$ and the distance of x from a hyperplane $H_{i,j}$ is:

$$r(x, H_{i,j}) \ = \ \frac{g_i(\vec{x}) - g_j(\vec{x})}{||\vec{w_i} - \vec{w_j}||} \tag{4.9}$$

Hence, the generalized equation of $g(x)$ for LDFs is the following:

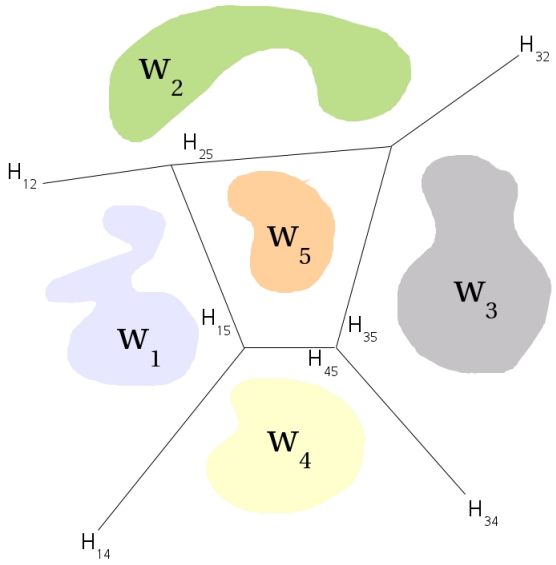


Figure 4.4: Illustrating figure of linear hyperplane

$$g(\vec{x}) = \sum_{i=1}^{d} a_i.y_i(\vec{x}) \tag{4.10}$$

where a is a d dimensional weight vector and $y_i(x)$ are mapping functions from the d-dimensional x-space to the $\widehat{d}$-dimensional y-space (usually $\widehat{d} >> d$). The aim is to map $g(\vec{x})$ to a transformed space where it will be *linear*. Hence, using LDFs, we are able to tackle all the classification problems which accept as a solution any of the functions which are possible to be transformed to a linear one in the y-space. Some of these functions are:

- Lineal discriminant functions:
  $g_(\vec{x}) = w_0 + \sum_{i=1}^{d} \; w_i.x_i$
- Quadratic discriminant functions:
  $g_(\vec{x}) = w_0 + \sum_{i=1}^{d} \; w_i.x_i + \sum_{i=1}^{d} \; \sum_{j=1}^{d} \; w_{ij}.x_i.x_j$
- Polynomial discriminant functions:
  $g_(\vec{x}) = w_0 + \sum_{i=1}^{d} \; w_i.x_i + \sum_{i=1}^{d} \; \sum_{j=1}^{d} \; w_{ij}.x_i.x_j + \sum_{i=1}^{d} \; \sum_{j=1}^{d} \; \sum_{k=1}^{d} \; w_{ijk}.x_i.x_j.x_k$

The correct choice of the type of the discriminant function for a certain classification is predominant. Another important criterion which makes the difference between two classifiers is the choice of the boundaries between regions. For instance, in Figure 4.5, even though both H_1 and H_2 are able to separate the two classes, the former one has a very small margin with the nearest data point. Consequently, if we use H_1 for classification, the error-rate will be much bigger than using H_2. Therefore, it is very important to look for the hyperplane which maximizes this margin in order to have a good generalization of the classification problem and minimize the error-rate of the classifier. The goal for *SVMs* is to find the optimal hyperplane [109] and for this

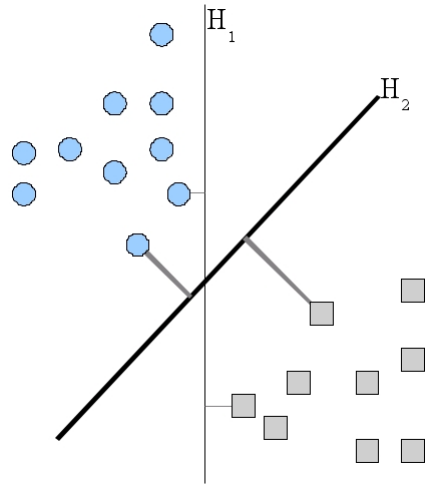


Figure 4.5: Two different regions boundaries

reason SVMs are also known as *maximum margin classifiers*. As we have mentioned

earlier the hyperplane is $H : g(\vec{x}) = 0$ which also means $H : w_{y_0} + <\vec{x}.\vec{w_y}> = 0$. Therefore, the canonical hyperplane is:

$$\min_{1 \leq i \leq n} |<\vec{w}.\vec{x_i}> + w_0| = 1 \tag{4.11}$$

and thus the distance between the hyperplane and the nearest data point is:

$$r(x, H_{i,j}) = \frac{1}{||\vec{w}||} \tag{4.12}$$

Therefore, during the training phase the algorithm should look for $\vec{w}$ which allows to maximize the distance given in Equation 4.12. A maximization of this distance is a problem of optimization aiming at finding the minimum $||\vec{w}||$. If we change this problem by looking for an optimum $\vec{w}$ to minimize $\frac{1}{2}.||\vec{w}||^2$ we do not alter the solution and thus the formal description of the optimization problem is:

$$\text{minimize } \frac{1}{2}.||\vec{w}||^2 \text{ subject to: } t_i(w_0 + <\vec{x}.\vec{w}>) \geq 1,\ 1 \leq i \leq n \tag{4.13}$$

where t_i is the classification function which can be decribed as:

$$t_i = \begin{cases} +1 & \text{if } w_0 + <\vec{x}.\vec{w}> > 0 \\ -1 & \text{if } w_0 + <\vec{x}.\vec{w}> < 0 \end{cases} \tag{4.14}$$

a more compact formulation of these inequalities is:

$$t_i(w_0 + <\vec{x}.\vec{w}>) \geq 1 \tag{4.15}$$

The change in the equation from $||\vec{w}||$ to $\frac{1}{2}.||\vec{w}||^2$ has been made in order to be able to use standard Quadratic Programing (QP) optimization algorithm [60]. In order to find out the final equation for the optimal hyperplane, it is primordial to use *Lagrange multipliers*. The general form obtained is:

$$g_{svm}(\vec{x}) = \sum_{\vec{x_i} \in SV} \lambda_i^*.t_i<\vec{x_i}.\vec{x}> + w_0^* \tag{4.16}$$

where λ_i^* are the Lagrange multipliers and represent the solutions of the optimization problem; $\vec{x_i}$ are the data points which satisfy $t_i<\vec{x_i}.\vec{x}> + w_0^* = 1$ and allow $\lambda_i^* \neq 0$ solutions. The points for which $\lambda_i^* \neq 0$ are called "*Support Vectors*" (SV). Figure 4.6

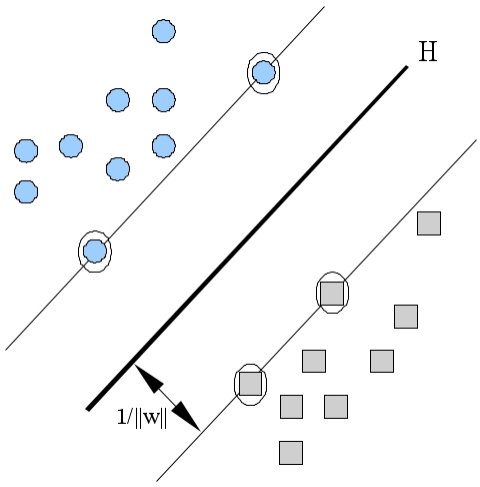


Figure 4.6: Linear separating hyperplane. Support Vectors are circled

shows an illustrating example of a linear separable data points: SVs are circled.

SVMs provide also a solution for non-separable cases, i.e., data points of the different classes which are not linearly separable. In [26], the authors suggest a modification in the conditions described in the Formula 4.15. The new conditions would introduce a positive slack ζ_i, $i = 1, ..., n$, Figure 4.7 shows an illustrating example. The new formulation of the modified conditions are as follows:

$$t_i(w_0 + <\vec{x}.\vec{w}>) \geq 1 - \zeta_i \ ; \ \zeta_i > 0 \ \forall i \tag{4.17}$$

The mapping from data space to features space in SVMs is done by *kernels*. These same kernels compute the distance between two data points. For instance, one of the simplest and most used kernels is the *polynomial* one. The following example illustrates the case of a polynomial kernel:

Suppose we have $\vec{x} = (<x_1, x_2, x_3>)$ and $\vec{y} = (<y_1, y_2, y_3>)$;

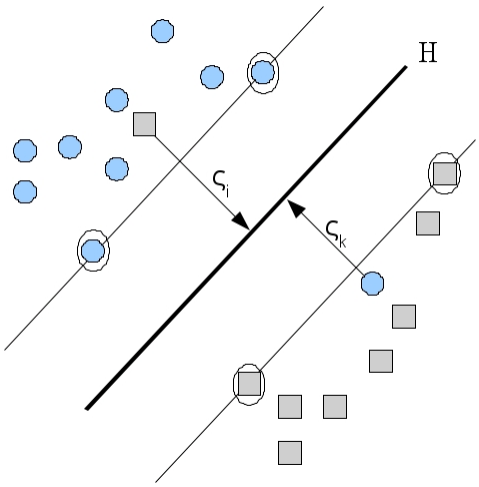


Figure 4.7: Non-linearly separable problem with linear hyperplane. Support Vectors are circled

$$\begin{aligned} K(\vec{x},\vec{y}) &= <\vec{x}.\vec{w_y}>^2 \\ &= (x_1y_1 + x_2y_2 + x_3y_3)^2 \\ &= x_1^2y_1^2 + x_2^2y_2^2 + x_3^2y_3^2 + 2x_1y_1x_2y_2 + 2x_1y_1x_3y_3 + 2x_2y_2x_3y_3 \\ &= <\Phi(\vec{x}).\Phi(\vec{y})> \end{aligned}$$

with

$$\begin{aligned} \Phi(\vec{x}) &= (x_1^2, x_2^2, x_3^2, \sqrt{2}x_1x_2, \sqrt{2}x_1x_3, \sqrt{2}x_2x_3) \\ \Phi(\vec{y}) &= (y_1^2, y_2^2, y_3^2, \sqrt{2}y_1y_2, \sqrt{2}y_1y_3, \sqrt{2}y_2y_3) \end{aligned}$$

Basically, SVMs can handle only two classes. We can use one of two methods in order to employ SVMs in a multi-class ($M > 2$) problem:

1. *One class vs. all others*: Build M classifiers, where each classifier looks for separating one class from the rest.

2. *Pairwise*: Build $M * (M-1)/2$ classifiers, where each classifier considers only two classes. The final decision is made by their weighted voting.

4.3.2 Support Vector Machines in the framework of Natural Language Processing

In the literature, several research studies show that SVMs are an adequate solution for many NLP tasks. For instance, in [63] the authors have built a system to identify BP chunks in English texts. A *chunk* is a sequence of words which has a proper meaning and all its constituent words belong to the same category. The BP chunking task consists of two sub-tasks:

1. Determining the chunks existing within the text.

2. Classifying the identified chunks as one of the possible grammatical categories such as noun phrases, adjectival phrases, verb phrases, etc.

The reported approach is based on SVMs and performs in two main steps:

1. It performs a SVMs-based parsing of the test data using both backward and forward directions and 4 different annotation schemes which are IOB1, IOB2, IOE1 and IOE2. Table 4.3 shows the difference between these annotation schemes.

2. It combines the results obtained from the different annotation schemes using a weighted voting method.

In order to evaluate their approach, the authors evaluated their system over standard data (baseNP-S and baseNP-L data sets[6]). The results show that their approach outperform the existing ones by almost 1 point of F-measure.
In [30], a tokenizer, POS-tagger and a BP chunker were built for the Arabic language employing an SVMs approach. The authors report that they have used the *Arabic Treebank*[7] for both training and evaluating their system. For the tokenization task they have tagged the characters of each word using the IOB2 annotation scheme (Figure 4.8 shows an illustrating example for the annotation used for tokenization). A tag-set of twenty four classes has been used for the POS-tagging task, whereas for the BP chunking the authors used an IOB2 annotation scheme over nine types of phrases.

[6] ftp://ftp.cis.upenn.edu/pub/chunker
[7] http://www.ldc.upenn.edu

Table 4.3: IOB1, IOB2, IOE1 and IOE2 Base Phrase annotation schemes

	IOB1	IOB2	IOE1	IOE2
In	O	O	O	O
early	I	B	I	I
trading	I	I	I	E
in	O	O	O	O
busy	I	B	I	I
Hong	I	I	I	I
Kong	I	I	I	E
Monday	B	B	I	E
,	O	O	O	O
gold	I	B	I	E
was	O	O	O	O

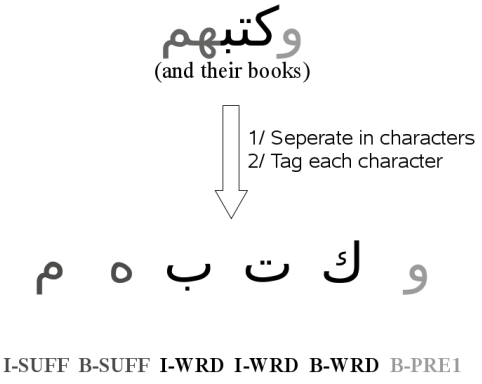


Figure 4.8: Illustrating example of Arabic tokenization characters annotation

The authors report that they have obtained an F-measure of 99.12 for tokenization, 95.49% accuracy for POS-tag and F-measure=92.08 for BP chunking. In both [63] and [30], the authors have used the polynomial kernel (see subsection 4.3.1).

In [96] the authors have used a SVMs based approach to learn and extract pairs of bilingual word sequence correspondences from a non-aligned parallel (Japanese-

English) corpora. The authors report that they have:

1. Made both positive and negative samples for SVMs model training;
2. Converted the training samples into features vectors;
3. Annotated a set of candidate translation pairs for evaluation.

The authors report that they have used different types of features, e.g. neighbour words, POS-tag, dictionaries, etc. and have chosen a Gaussian kernel for the SVMs classifier. The Gaussian kernel can be expressed by the following equation:

$$K(\vec{x}, \vec{y}) = exp(-\frac{||\vec{x} - \vec{y}||^2}{2\delta^2}) \tag{4.18}$$

The results show that a recall of 81.1% and a precision of 69.0% have been achieved and the authors argue that their approach is very efficient for translation pairs and could be employed to significantly reduce the cost for making translation dictionaries. In [53], a SVMs-based approach has been employed for the NER task. The authors have used a polynomial kernel in their SVMs classifier and used the IREX[8] data-set for training and evaluation. IREX is a collection of 1,174 articles containing 19,000 NEs where the formal test-set consists of 71 articles containing 1,510 NEs. Data are tagged using the Start/End annotation scheme (see Table 4.4). Three types of features have been employed in the experiments:

1. The word itself;
2. POS-tag;
3. Character type.

The authors also conduct a comparative study between SVMs and ME. The results show that using SVMs (90.03) has lead to obtained results 4 points above ME (86.00). The authors argue that the results obtained with SVMs are the best results achieved for the Japanese NER task. In [107] authors have conducted a comparative study between SVMs and CRFs (see Subsection 4.2.1) for Vietnamese NER. The authors

[8] http://nlp.cs.nyu.edu/irex/

Table 4.4: Illustrating example of the Beginning/End annotation example

	Start/End
Wolff	S-PER
,	O
currently	O
a	O
journalist	O
in	O
Argentina	S-LOC
,	O
played	O
with	O
Del	B-PER
Bosque	E-PER
in	O

report that they have considered the same classes defined in MUC-6 (see Chapter 3) and have used context, external gazetteer and lexical features. They have used a polynomial kernel for the SVMs classifier and the IOB2 annotation scheme. The results showed that the performance obtained with SVMs (F-measure=87.45) was almost one point higher than CRFs (F-measure=86.48).

4.4 Concluding Remarks

The main aim of the classification approaches is to infere the class of an object x having only part of the information about it. In the literature, many research works have proved that the best classification approach for the NER tasks are: ME, CRFs and SVMs. These three supervised and discriminative approaches attempt to solve the classification problem from very different viewpoints:

1. The **ME** statistical model can be described, generally, as a *real world replication* approach. During the training phase, this approach fetches the adequate weight for each feature. Adequate features weights from an ME perspective means a weight-set which will help to get a distribution as close as possible to the distribution observed in the training data. Such a distribution would allow to perform the classification without any biases or any other reasoning algorithm.

2. **CRFs** on the other hand, are graphical models and are more oriented to sequence labeling: i.e., during the training phase the CRFs take into consideration both the features of the current object to classify and the neighbour objects. For this purpose, the CRFs define the *cliques* within the graph and optimize the features weights considering all the elements of the clique. This approach has proved to solve the problem of *label-bias* which has been reported unsolvable for other approached such as HMMs and MEMM.

3. **SVMs**, finally, can be seen as *territory markers*. SVMs work only for two-classes problems. When they are given the training data, of the two classes, and the features, they transform feature vectors to another space and then find

the best hyperplane which defines the boundary between the two classes. This hyperplane is the one which has the maximum margin with the nearest data point. The transformation that we have mentioned before is performed by the so-called *kernels*, and hence there are different types of kernels (e.g. quadratic, polynomial, Gaussian, etc.) and the choice of the right kernel for a specific task is of predominant importance.

All these approaches have proved to be efficient for NLP tasks such as POS-tagging, BP chunking, IE and others. For the NER task, there are many research works which show that all these approaches present an adequate solution. Some research works such as [107] and [61] present comparative studies among these approaches. However, to our knowledge, there are no research works which report a comparative study between these three approaches for the NER task for different feature-sets, different corpora and type of NEs.

In the next chapters, we present the experiments which we have carried out in order to define the right statistical modeling approach and the right technique in order to build an efficient NER system for the Arabic language. Even though, we have used only Arabic data and some of the features are language-dependent, our experiments can be very beneficial for the NER research community because we employed mostly language-independent features. The details about the impact of the different features and the performance of the different approaches are given.

Chapter 5

A Maximum Entropy based Approach: 1 and 2 steps

"The distance is nothing; it is only the first step that is difficult."
- Madame Marie du Deffand -

In the previous chapters, we have presented the study of the needed background in order to ease the comprehension of our research works which we present in this chapter, and in the coming three chapters (i.e., Chapter 6, 7 and 8). We have shown that in order to understand the problem of Arabic NER, it is necessary to study its three main axes:

1. **The Arabic language**: The peculiarities of this language show that for any NLP task, a special attention needs to be paid to its complex morphology and the lack of capital letters in its scripture. At the same time, a special curiosity is aroused on how the rich morphology of this language can be used to enhance the performance for the task in question.

2. **The task definition and state-of-art**: Some of the research studies (i.e. [81][44]) that we have presented when we have described the importance of the NER task for the other NLP tasks, show that the types of NEs taken into consideration by the NER system should be adapted to the needs of the global system in order to obtain the optimal performance. Therefore, in order to deal with the NER task from a general perspective (i.e. not oriented to any NLP task in special) we had to study the different standard definitions of this task. In all our experiments, we will use the CoNLL NER task definition (see Section 3.2) because:

   - it encloses the classes defined previously in the MUC-6 (see Section 3.2), and adds the "Miscellaneous" class for NEs which neither belong to the class "Person", "Location" nor to "Organization".

   - the ACE definition EDT task considers all the types of mentions (name, nominal, pronominal, etc.) of the entities, whereas in our case we are interested only in the *name* mentions.

   Finally, the state-of art of the NER task in general and the Arabic NER task in particular (few works have been published treating the latter one) was necessary to show the statistical modeling approaches and the techniques which have proved to be efficient to obtain a high performance.

3. **The statistical modeling approaches**: The state-of-art of the NER task has shown that three methods help to model the NER problem: i.e., ME, CRFs and SVMs. Those three approaches have been employed to obtain high performance NER systems, even though they tackle the problem from very different perspectives. Some comparative studies have been carried out among the three approaches, even if they do not report enough details to satisfy NER researchers curiosities.

In the current chapter we will describe our first set of experiments which consists of two ME-based approaches. In the first section, we present the data and features which we have employed. The obtained results for our first experiment using the 1-step approach are presented in Section 5.2. Section 5.3 will focus on the 2-step approach which we have adopted in order to improve the performance. Finally, in Section 5.4 we draw our final conclusions about both experiments.

5.1 The ANERcorp

5.1.1 Data

In order to carry out this first experiment we have built the *ANERcorp* corpus [18]. We have considered the same classes which have been used in the CoNLL competitions (see Chapter 3) and the same annotation scheme, i.e., the IOB2 scheme (see Chapter 4). Figure 5.1 shows an extract of ANERcorp, more exactly the first ten tokens of the corpus, which can be translated as: "*Frankfurt* , the *Cars Manufacturing Union* has declared yesterday in *Germany* that". All the tokens of ANERcorp were tagged manually. It consists of a collection of 316 articles which have been manually retrieved from different web sources (see Table 5.1).

ANERcorp contains 150,286 tokens, the size of vocabulary is 32,114 types, which makes a ratio of tokens to types of 4.67. The NEs represent 11% of the corpus and their distribution along the different types is given in Table 5.2.

Figure 5.1: An extract of ANERcorp

Table 5.1: Ratio of sources used to build ANERcorp

Source	Ratio
http://www.aljazeera.net	34.8%
Other newspapers and magazines	17.8%
http://www.raya.com	15.5%
http://ar.wikipedia.org	6.6%
http://www.alalam.ma	5.4%
http://www.ahram.eg.org	5.4%
http://www.alittihad.ae	3.5%
http://www.bbc.co.uk/arabic/	3.5%
http://arabic.cnn.com	2.8%
http://www.addustour.com	2.8%
http://kassioun.org	1.9%

5.1.2 Features

The features which have been used in this experiment are the following:

Table 5.2: Ratio of NEs per class

Class	Ratio
PERSon	39%
LOCation	30.4%
ORGanization	20.6%
MISCellaneous class	10%

- *The word itself*: in the pre-training phase (see Figure 5.3), we compute for each word the number of times that it was assigned each of the classes. Therefore, for each word we have a frequency table such as the one given in Table 5.3 for the word *union*. However, as a feature we use only the class which has the highest

Table 5.3: Number of times each class was assigned to the word *union* in ANERcorp

Class	Frequency
O	35
Person	0
Location	8
Organization	58
Miscellaneous	2
Total	103

frequency, i.e., in the case of the example given in Table 5.3 we would indicate to the classifier that *the current word was mostly assigned in the training phase the class "Organization"*.

- *Context unigrams and bigrams*: in the pre-training phase, see Figure 5.3, we compile a list of the words (unigrams) and word-pairs (bigrams) which fre-

quently appear before NEs of a certain class. Therefore, at the end of the pre-training process we have a list of a unigrams and bigrams for each class, with a frequency assigned to each unigram and bigram. For instance the word "qAl"[1] which can be translated as "said", appears 609 times in the training corpus. 146 times it appears before an NE of class "Person". Thereafter, we have set a frequency threshold for each class in order to transform this feature to a binary one: i.e., instead of passing the exact frequency of a unigram or a bigram to the training module, we pass an information specifying whether this unigram or bigram is above the threshold or not. Table 5.4 shows the threshold which we have set for each class.

Table 5.4: Unigrams and bigrams frequency thresholds for each class

Class	Freq. Threshold
Person	75
Location	140
Organization	50
Miscellaneous	10

- *Previous word's class*: Table 5.5 shows the probability of a word of a class c_i (rows) to appear after a word of class c_j (columns). Those probabilities are computed taking into consideration the frequencies of occurence in the training corpus of ANERcorp: This matrix contains information which would of high significance for a classifier. For instance, a word of a class $I - X_i$ never comes after a word of a class $I - X_j$ because the table suggests that it either comes after the class O, $B - X_i$ or $I - X_i$. Thus, in the training phase we will provide each word with the previous class as a feature, whereas in the test phase we will provide the last predicted class.

[1] Written in Buckwalter tranliteration

Table 5.5: Matrix of probabilities of a word of class c_i (rows) appears after a word of class c_j (columns)

	B-PER	B-LOC	B-ORG	B-MISC	I-PERS	I-LOC	I-ORG	I-MISC	O
B-PER	0.01	0.01	0	0	0.01	0	0.01	0	**0.96**
B-LOC	0	0.02	0	0	0	0	0	0	**0.98**
B-ORG	0	0	0.02	0	0	0	0	0	**0.98**
B-MISC	0	0	0.015	0.015	0	0	0	0	**0.97**
I-PERS	**0.71**	0	0	0	**0.29**	0	0	0	0
I-LOC	0	**0.9**	0	0	0	**0.1**	0	0	0
I-ORG	0	0	**0.81**	0	0	0	**0.19**	0	0
I-MISC	0	0	0	**0.69**	0	0	0	**0.31**	0
O	0.01	0.02	0.01	0	0.01	0	0.01	0	**0.94**

- *Nationality*: this feature is both a contextual and a lexical feature. We mark nationalities in the input text. Such information is useful for detecting NEs because they work as precursors to recognizing NEs. For instance, Figure 5.2 shows four examples where an NE is preceeded by a nationality extracted from ANERcorp.

- *Gazetteers*: in order to measure the impact of using an external resource we have *manually* built *NERgazet*[2] which consists of a collection of three gazetteers:

    1. *Location Gazetteer*: this gazetteer consists of 1,950 names of continents, countries, cities, rivers and mountains found in the Arabic version of wikipedia[3];

    2. *Person Gazetteer*: this was originally a list of 1,920 complete names of people found in Wikipedia and other websites. Splitting the names into

[2]http://www.dsic.upv.es/grupos/nle/downloads.html
[3]http://ar.wikipedia.org

ممثل السياسة الخارجية الأوروبي **خافيير سولانا**
The European External political Affairs Representative **Javier Solana**

وزير الخارجية السوري **وليد المعلم**
The Syrian Minister of External Affairs **Walid al-Mouallem**

العاصمة الألمانية **برلين**
The German capital **Berlin**

عالم الآثار الصيني **لن هوي شيانغ**
The Chinese archaeologist **Lin Hui Xiang**

Figure 5.2: Four examples of NEs preceeded by a nationality

first names and last names and omitting the repeated names, the list contains finally 2,309 names;

3. *Organizations Gazetteer*: the last gazetteer consists of a list of 262 names of companies, football teams and other organizations.

The following step, consists of tagging in the training and test data the NEs according to our gazetteers. In order to do so, we go through our gazetteers and for each element found in the training/test data we tag all its tokens. For the location and organization gazetteers we tagged only the NEs whose tokens are found in the training/test data. For instance, if we suppose that "United States of America" is an element of the location gazetteer and we have the following two sentences in the training data:

(i) The President of United States of America, declared today ...
(ii) The United Nations World Food Programme (WFP) today began an ...

However, for the person gazetteer elements we tagged the tokens on the training/test data even if only part of the gazetteer element is found. For instance, if we consider that "Michael Jackson" is an element of the person gazetteer and we have the following sentence in the training data:

... he was reading George Michael latest news ...

The word "Michael" would be tagged as an element of the person gazetteer.

5.2 1-step ME-based Approach

5.2.1 Approach and Tools

Figure 5.3 shows the basic architecture of the first version of our NER system ANERsys. It consists of two main parts, i.e., training and testing. The training phase is composed of the following modules:

1. *Pre-training*: this module starts by making a *characters normalization* which consists of converting all the "Alif with Hamza" (all kinds of Hamza's) to a simple "Alif". It is necessary to perform this first step because we have noticed that in two different documents we can find the same word written in two different forms, i.e., with two different forms of "Alif"[4]. Thereafter, a *features preparation* is started in order to compute all the necessary uni/bi-gram frequencies, check the words existing within ANERgazet, etc. (see Subsection 5.1.2).

2. *GIS-based model parameters estimations*: in other words it is the module which computes the features weights λ_i (see Section 4.1). In order to do so, we have used the *YASMET* toolkit, which is freely available[5]. The name of this toolkit stands for "Yet Another Small MaxEnt Toolkit"; it is developed with the C++ programming language and it requires a special input file format (fully described

[4]A discussion of the correct form to be used in each case goes out of the scope of our research.
[5]http://www.fjoch.com/YASMET.html

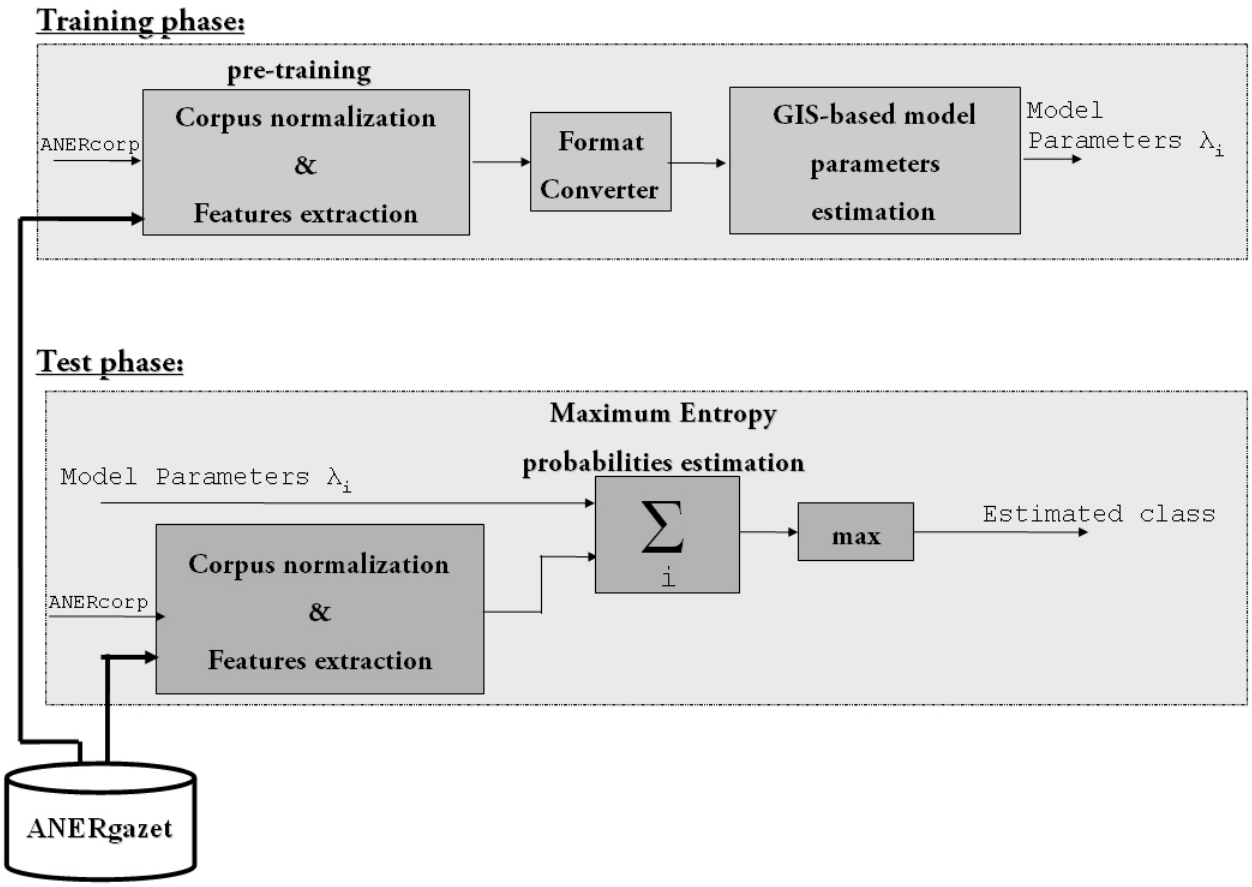


Figure 5.3: The basic architecture of the first version of ANERsys

in the documentation webpage[6]). YASMET is easy to use and efficient to perform the training of ME models.

3. *Format converter*: we have included an additional module to perform a conversion in order to present the features which have been prepared in the first module in the format required by YASMET.

In the *test phase*, the aim is to use the parameters obtained in the training phase to compute the probability of each word to belong to each of the 9 classes c_j. Hence, the following modules are needed:

[6]http://www-i6.informatik.rwth-aachen.de/Colleagues/och/software/YASMET1.txt

1. *Corpus normalization and features preparation*: for the same goal as in the training phase, we need to collect the features of each word w_i.

2. *ME parameters estimation*: this module uses the ME formula which we have presented in Section 4.1 to compute the probability of w_i to each of the classes c_j as a function of f_i and λ_i.

3. We have named *max* the last module which is a short script which looks for the class which has obtained the *max*imal probability and thus assign it to w_i.

In order to evaluate the F-measure we have used the same evaluation standard metrics of precision, recall and F-measure presented in [104]. The CoNLL evaluation metric is a strict metric that does not assign partial credit. An NE has to be identified as a whole and correctly classified in order to gain credit. The script which performs the evaluation is freely available on the downloads webpage[7].

5.2.2 Experiments and Results

First of all, we have computed the baseline with the model used in CoNLL (see Subsection 3.3.2). Thereafter, we have performed two experiments with the features presented in the Subsection 5.1.2, and the approach presented in 5.2.1. In the first experiment we have not used any external resources (i.e., ANERgazet was not used). Whereas in the second one we have included ANERgazet. Table 5.6, 5.7 and 5.8 present the results obtained for the baseline, first and second experiment, respectively.

The baseline model, which is an indicator of the amount of NEs which have been seen in the training phase, has obtained an F-measure of 43.36. When we have used the ME model with all the features except external resources we have enhanced the performance of our system by more than ten points (54.11). Finally, when we have included ANERgazet, the performance was slightly improved (55.23).

[7]http://bredt.uib.no/download/conlleval.txt

Table 5.6: Baseline results

Baseline	Precision	Recall	F-measure
Location	75.71%	76.97%	76.34
Misc	22.91%	34.67%	27.59
Organisation	52.80%	33.14%	40.72
Person	33.84%	14.76%	20.56
Overall	**51.39**%	**37.51**%	**43.36**

Table 5.7: ANERsys results without using external resources

ANERsys	Precision	Recall	F-measure
Location	82.41%	76.90%	79.56
Misc	61.54%	32.65%	42.67
Organisation	45.16%	31.04%	36.79
Person	52.76%	38.44%	44.47
Overall	**62.72**%	**47.58**%	**54.11**

Table 5.8: ANERsys results using external resources

ANERsys	Precision	Recall	F-measure
Location	82.17%	78.42%	80.25
Misc	61.54%	32.65%	42.67
Organisation	45.16%	31.04%	36.79
Person	54.21%	41.01%	46.69
Overall	**63.21**%	**49.04**%	**55.23**

5.2.3 Discussion

The results tables show that the ME approach and the feature-set which we have used helped significantly to capture the NEs which have not been seen in the training phase. However, the major problem, which needs to be tackled in order to enhance the performance, is the multi-word NEs identification. In order to show more clearly the great error-rate induced by this problem we have to take a look at the F-measure obtained for the different classes when a partial credit is assigned, i.e., if a single token is tagged correctly a credit is gained. Those results are shown in Table 5.9.

Table 5.9: ANERsys results using external resources in case a credit is gained when a single token of a multi-word NE is tagged correctly

ANERsys	Precision	Recall	F-measure
B-LOC	82.04%	83.79%	82.90
I-LOC	75.93%	51.90%	61.65
B-MISC	73.85%	40.85%	52.60
I-MISC	66.67%	3.53%	6.70
B-ORG	57.95%	38.80%	46.48
I-ORG	69.03%	29.21%	41.05
B-PER	75.73%	55.52%	64.07
I-PER	87.22%	43.33%	57.90
Overall	**76.34**%	**50.68**%	**60.92**

This table shows that the F-measure is considerably lower for the tokens which are part of an NE but not its first token, i.e., tokens which belong to the classes $I-X$. Figure 5.4 shows two examples of multi-word NEs which have been incorrectly tagged by ANERsys. The underlined tagged tokens represent the correct tagging. Example 1, which can be translated as "The Tunisian president *Zine El Abidine Ben Ali*", shows that ANERsys was able to capture only two words of the NE: the first one, "Zine", appeared right after the word "Tunisian", i.e., a nationality; the second one, "Ben",

very frequently appearing as part of a "Person" NE. However, since the previous word was misclassified as "O", the classifier assigned "B-PER" to the word "Ben". Example 2, illustrates the example of an organization in the sentence "pointing out the *Kurdistan Labor Party*". The last word of the NE, i.e. "Kurd", was not captured because it both was unseen in the training data and appeared in an uncommon context.

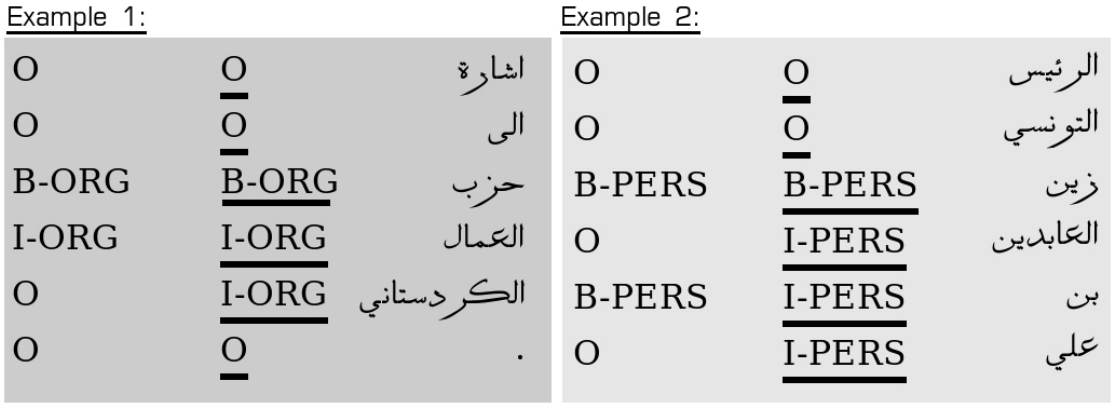


Figure 5.4: Two illustrating examples of ANERsys error in tagging multi-word NEs.
Translation of Example 1:"pointing out the *Kurdistan Labor Party*".
Translation of Example 2:"The Tunisian president *Zine El Abidine Ben Ali*"

5.3 The 2-step Approach

According to the results obtained in the first version of our system, in order to enhance the performance, we should enhance most of all its recall. This implies to investigate a method to capture all the tokens of the multi-word NEs, as we have shown earlier. For this reason, we have chosen to investigate the possibility of separating the NER task in two sub-tasks [14]. The first one would take care of detecting the NEs within the text and the second one would include this information as a feature and classify the detected NEs. In this section, we present the approach which we have

used to implement this idea. Also , we present the obtained results and a detailed error-analysis.

5.3.1 Approach and Tools

As we have illustrated in Figure 5.5, the training phase aims at training two models:

1. A model for detecting the NEs boundaries. In order to make such a model, we first change the annotations "B-PERS", "B-LOC", "B-ORG" and "B-MISC" to "B-NE" and the annotations "I-PERS", "I-LOC", "I-ORG" and "I-MISC" to "I-NE". Hence, we will obtain a model trained for only 2 classes, i.e., "B-NE" and "I-NE". The data, features and training tool are all identical to the ones employed in the first version of ANERsys (see Section 5.1).

2. A second model for classifying the NEs. This model uses the "B/I-NE" annotation as a feature. That is equivalent to supposing that we already have an ideal boundaries detection module and we train a classification model which takes its output as a feature.

The test phase, has more modules and a more complicated behaviour. As illustrated in Figure 5.5, the component models in the test phase are the following:

1. *POS-tagging*: we use a freely available Arabic POS-tagger[8] which is trained on the Arabic Treebank[9] [30]. Even though the POS-tagger has a large tags-set, we will need only the "NNP" (Proper Noun) and "NNPS" (Plural Proper Noun) tags. Those tags mark the NEs existing within an Arabic text.

2. *ME-based boundaries detection*: using the boundaries detection ME model which we have prepared in the training phase.

3. *Combination module*: once we have the output tagging of both, the POS-tag and ME based modules, we perform a union of these outupts in order to provide one single outcome where all the detected NEs are tagged.

[8]http://www1.cs.columbia.edu/∼mdiab/
[9]http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2004T02

4. *Classification model*: it is based on an ME model. It takes at the input both the raw test file and the NEs boundaries detection performed by the other modules. Thereafter, it adds the boundaries annotations to the other features which we have used in the first version of our system (see Section 5.1).

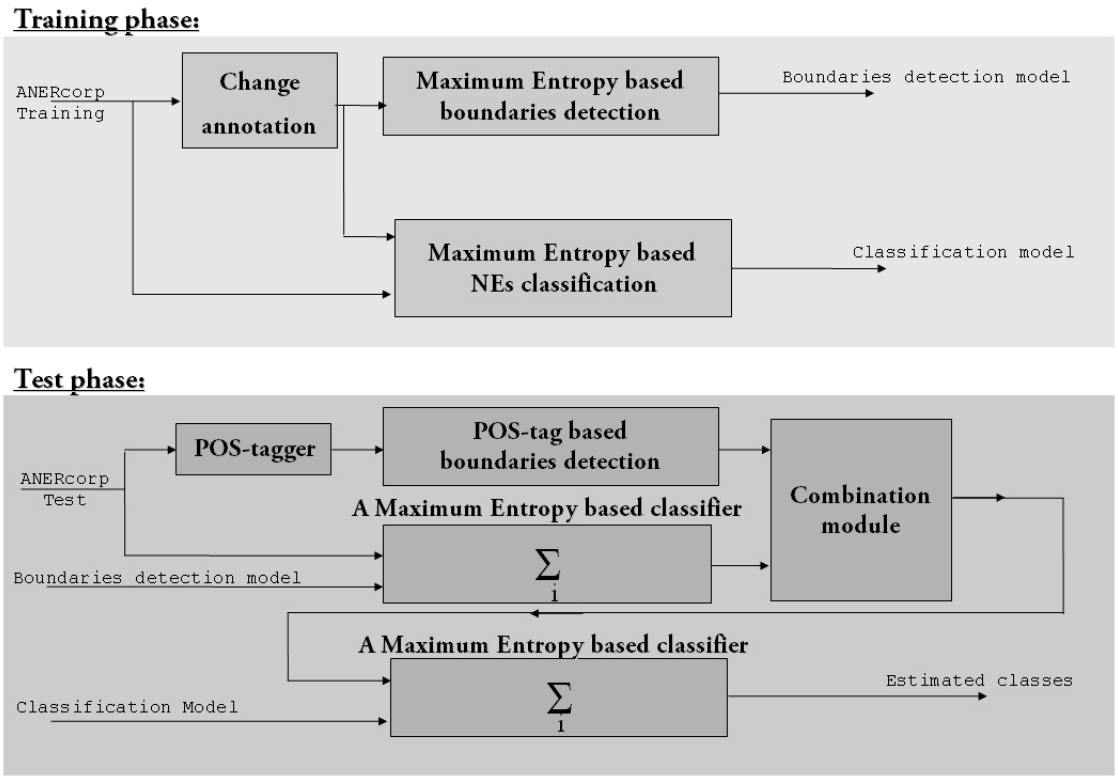


Figure 5.5: The generic architecture of the 2-step ANERsys

5.3.2 Experiments and Results

We have conducted two experiments: in the former one we have used the 2-step approach, whereas the latter was carried out in order to be able to compare the performance of our system with another system. For this purpose we have used the demo version of Siraj (Sakhr) which is available in its webpage[10]. Table 5.10 and

[10]http://siraj.sakhr.com/

Table 5.11 show the results of the 2-step ANERsys and the Siraj (Sakhr) systems, respectively.

Table 5.10: ANERsys: 2-step approach results

ANERsys	Precision	Recall	F-measure
Location	91.69%	82.23%	86.71
Misc	72.34%	55.74%	62.96
Organisation	47.95%	45.02%	46.43
Person	56.27%	48.56%	52.13
Overall	**70.24**%	**62.08**%	**65.91**

Table 5.11: Siraj (Sakhr) results

ANERsys	Precision	Recall	F-measure
Location	84.79%	67.91%	75.42
Misc	0.00%	0.00%	0.00
Organisation	0.00%	0.00%	0.00
Person	74.66%	55.84%	63.89
Overall	**78.95**%	**46.69**%	**58.58**

The results show that using a 2-step approach helped to obtained an F-score (65.91) more than 22 points above the baseline (43.36), outperforming the one-step approach (55.23) by more than 10 points and by 7 points the commercial Arabic NER system Siraj (58.58).

5.3.3 Discussion

After analyzing the results obtained with the first version of our system, we have observed that it is necessary to investigate an approach which would help to capture

the multi-word NEs. Using a 2-step approach proved to be an adequate solution to tackle this problem and thus enhance the performance of the ANERsys. In order to make a deeper analysis of the results and have a clearer vision on ANERsys 2.0, we carried out some further experiments. A first one to evaluate the performance of the first step of our new approach: i.e., the capacity of the system to delimit the NEs correctly (see Table 5.12). As for the second one, it aims at evaluating the exact error rate of the second step. For this purpose, we used a corpus where the NEs delimitations were taken directly from the manually annotated corpus (see Table 5.13), i.e. 100% correct boundaries.

Table 5.12: Evaluation of the first step of the system

ANERsys 2.0	Precision	Recall	F-measure
B-NE	82.61%	72.10%	77.00
I-NE	91.27%	42.30%	57.81
Overall	**84.27**%	**62.89**%	**72.03**

Table 5.13: Evaluation of the second step of the system

ANERsys 2.0	Precision	Recall	F-measure
Location	93.22%	88.68%	90.90
Misc	94.67%	58.20%	72.08
Organisation	76.89%	65.27%	70.61
Person	75.10%	91.37%	82.44
Overall	**83.22**%	**83.22**%	**83.22**

The results of Table 5.13 show clearly that we need to improve the first step of our system in order to enhance the performance of the global system. In case the NEs boundaries detection was ideal, we would achieve an F-score of 83.22. In order

to do so, further investigation of more statistical modeling approaches and larger feature-sets is necessary.

5.4 Concluding Remarks

The aim of this chapter is to:

1. Show our first steps towards building an NER system fully oriented to the Arabic language;

2. Give a detailed description of the employed resources, the experiments and the obtained results.

As a first experiment we have chosen to build a reliable and efficient Arabic NER system employing a very intuitive approach, i.e. Maximum Entropy. We have prepared *ourselves* manually a set of more than 150,000 tokens for training and evaluating the system. At a first stage, we have chosen a feature-set including contextual and lexical features. We have also manually built three gazetteers which can be described as lexicons of people, locations and organizations names. These gazetteers have been built in order to help measure the impact of using external resources. The obtained results and the error-analyses have showed that:

1. Using the ME approach with the feature-set which we have mentioned helps to capture the NEs which have not been seen in the training phase. This statement is supported by the results in which we have obtained 10 points more than when the ME approach was used (54.11) and than the baseline (43.36). The baseline model assigns to a word the class which it has been most frequently assigned in the training corpus, i.e., it is a good indicator of the amount of NEs which have been already seen in the training corpus.

2. Using external resources (i.e., gazetteers) has helped to increase slightly the results to 55.23. However, it is important to notice that the gazetteers which have been employed in this experiment are very small (see Subsection 5.1.2).

3. When we evaluated the obtained results considering each word apart, we found out that the major problem is to capture the multi-word NEs. Therefore, in order to enhance the performance of our NER system we had to investigate how to increase the recall of the tokens of the class $I - X$ (i.e. the tokens which are part of an NE but are not its first one, such as the token *States* in the NE *United States of America*).

In order to tackle the problem of capturing the multi-word NEs, we have investigated the possibility of dividing the NER task in two sub-tasks: the first one to detect the NEs existing within the text and the second one to classify them. In order to implement this approach we have adopted the architecture which we illustrated in Figure 5.5. The boundaries detection step is the union of the results of two modules: the first one provides the list of NEs detected by a POS-tagger (trained on the Arabic Treebank). Whereas the second one provides the list of NEs detected with an ME model (trained on the ANERcorp with the necessary changes in the annotation). We have also tagged the ANERcorp test corpus by the demo version of the commercial system Siraj[11] (Sakhr) in order to be able to compare our system with others. The obtained results may be interpreted as follows:

1. Using the 2-step approach helped to improve the overall results up to 10 points (65.91) with respect to the 1-step approach. It has also helped to obtain results which are more than 7 points above the ones of the commercial system Siraj.

2. The evaluation of the the first step of the system (i.e., the boundaries detection module) shows that this module has a F-measure of 72.03. However, the recall is only 62.89, consequently the recall of the global system has not been improved as much as we have expected. On the other hand, the major improvement has been noticed in the precision.

3. The evaluation of the second module of the system has only been performed by plugging an ideal boundaries detection module at its input. The overall results were F-measure=83.22, where both recall and precision were 83.22%. However,

[11] http://siraj.sakhr.com

the results table of this evaluation (see Table 5.13) shows that the recall of the "Miscellaneous" and "Organization" classes are not improved as of the ones of "Person" and "Location" even when the boundaries detection module is ideal (F-score=100). This is especially because the two latter ones represent the 69.4% of the NEs in ANERcorp training corpus.

The obtained results and the error-analyses which have shown that the 2-step approach should be considerably improved (almost perfect) in order to obtain a great improvement of the global system performance. Even if the first step was perfect, some classes such as "Miscellaneous" and "Organization" would still be difficult to capture. Hence, at this point of our research work we have decided to explore another direction to improve ANERsys. That direction would be using different statistical modeling approaches with a larger feature-set, increasing the size of ANERcorp and using other data-sets, if possible *standard* ones. The next chapter describes how we concretized these ideas and reports the obtained results.

Chapter 6

ME, CRFs and SVMs with Different Data-sets

"Essentially, all models are wrong, but some are useful."

- George Box -

As we have mentioned in Chapter 2 (see Subsection 2.1.1), one of the important characteristics which *the Arabic orthography lacks* in comparison with other languages, such as English, is the use of capital letters. Consequently, in the Arabic language, no special signal is used to distinguish the NEs from the other categories of words. This characteristic proved to make the NER task much harder for rule-based approaches [2] (see Subsection 3.3.4 for more details about this research work). Similarly, our first results (see Chapter 5) using an ME based approach, proved that the lack of capital letters in the Arabic language can be an obstacle to achieve high performance for ML approaches as well. Furthermore, this statement was confirmed when our experiments showed that a much better modeling of the NER problem was achieved when we performed a NEs boundaries detection step before their classification. However, it is important to point out that it is necessary to be able to detect boundaries with a very high F-measure. More precisely, the results of our study suggest that aiming at reaching an F-measure around 80 by using the 2-step approach, requires a boundaries detection system with an F-measure around 100 (an F-measure of 83.22 was obtained for the global NER system when the F-measure of boundaries detection was 100).

In Chapter 2 where we described the peculiarities of the Arabic language, and we have also shown that thanks to its agglutinative characteristic, one of the important peculiarities that *other languages such as English lack* in comparison with the Arabic language is a rich morphology. Such a morphology has a whole set of affixes to use and rules to apply in order to form a word correctly. Thus, studying the impact of the morphological features on Arabic NER is a key research work to be carried out in order to complete the exploration which we have started in the previous chapter: i.e., the exploration of which peculiarities of the Arabic language might be useful for the NER task. In order to fulfill this need, we have carried out experiments with a large feature space which includes almost all the possible morphological features. We have also investigated the impact of each feature individually and performed an incremental feature selection in order to find out the feature-set which helps to obtain the best performance.

Another research direction which has been triggered in the previous chapter is the

investigation of the appropriate ML approach for the Arabic NER task. As we have described in Chapters 3 and 4, the ML approaches which have proved to be successful in the NER task are ME, CRFs and SVMs. Using ME with two different techniques, was the goal of the previous chapter. In this chapter, we use all of the mentioned ML approaches and we carry out a detailed comparison among them[12][11].

Finally, in order to make the study more *robust* and to show the reliability of our results, we validate them on 9 different data-sets. The first data-set is a second version of the ANERcorp (i.e., the data-set used in the experiments which we presented in the previous chapter). This second version contains more data and has been reviewed repeatedly in order to ensure annotation coherence. The rest of data-sets are the data which have been used in the Automatic Content Extraction (ACE) 2003, 2004 and 2005 evaluations (see Subsection 3.2.3). Even if all these data-sets were (fully or partially) annotated for NER task, it is important to point out that they come from different sources (newswire, broadcast news, weblogs and Arabic Treebank) and together they complete the necessary test-platform to validate both the performance and the robustness of an NER system.

The rest of this chapter is organized as follows. We first present the feature space which we have used in Section 6.1. In Section 6.2 we describe the different data-set which we hause in our experiments. We describe our experiments and show the obtained results in Section 6.3. The obtained results are shown in Section 6.4 and draw some conlusions in Section 6.5.

6.1 Features

As we have mentioned previously, in the experiments we present in this chapter we will show the impact of different types of features with different ML approaches on the performance of an Arabic NER system. The first step to take in order to conduct such a study is to select a number of features that we want to study. In our research work, we have selected features of different types. Following we present all the necessary details about these features:

Lexical (LEX_i): These features define the lexical orthographic nature of the tokens in the text. The idea behind using these features, is to explore the usefulness of internal evidences of a token to determine whether it is an NE or not. We define them as different character n-grams of a token and they are elaborated as follows: Consider that a word is simply a sequence of characters $C_1C_2C_3...C_{n-1}Cn$ then the lexical features would be

- LEX_1=C_1
- LEX_2=C_1C_2
- LEX_3=$C_1C_2C_3$
- $LEX4$=C_n
- $LEX_5 = C_{n-1}C_n$
- $LEX_6 = C_{n-2}C_{n-1}C_n$

For instance, if we consider the word "AlErAqy" (in English "The Iraqi"), the lexical features to be extracted are:

- LEX_1= A
- LEX_2= Al
- LEX_3= AlE
- $LEX4$= y
- LEX_5= qy
- LEX_6= Aqy

Contextual (CXT): This feature may be defined similarly to the "Context unigrams and bigrams" and "Previous word's class" features which we have defined in the Subsection 5.1.2. However, a broader and more formal definition of this feature would be as follows: The contextual feature is a window of $-/+$n tokens and $-$n tags from the NE of interest. The main goal of using a $-/+$n tokens window is to help the classifier to determine the class of a token by the lexical context in which it appears, whereas the $-$n tags is more likely to help by indicating to the classifier which classes it should *not* consider in its classification (an example-based and more detailed illustration of the usefulness of the tags context is given in the Subsection 5.1.2).

Gazetteers (GAZ): The same gazetteers which we have introduced in 5.1.2 have been used for this feature. Also the way we use them has been kept unchanged.

Morphological features (M_i): This feature-set is based on exploiting the rich morphology of the Arabic language. We relied on a system for Morphological Analysis and Disambiguation for Arabic (MADA) to extract relevant morphological features [46]. MADA disambiguates words along 14 different morphological dimensions and yields an accuracy of 95%. MADA typically operates on untokenized texts (surface words as they naturally occur) and provides as an output: (i) the tokenized form of the text; (ii) the morphological anaylses for each token and their ranking according to their probability; and (iii) the morphological features of each word. Tables 6.1 and 6.2 show those features (*Feat.*), give the abbreviation used by MADA to refer to them (*Abv.*) and the possible values (*Values*). The value "NA" (for all the features) stands for "Not Applicable"; in Tables 6.1 and 6.2 we show for each feature the cases when it is not applicable (the features are presented in two tables only for a better readability). A full decription of these features is given in Chapter 2.

In our experiments we have used only 11 of these features. The features which were ommited are: (i) gender; (ii) idafa[1]; and (iii) POS-tag. The two first features were ommited because they are not descriminant for NEs, and the POS-tag because

[1]In Arabic the Idafa construction is a same syntactical sequence which can be either a possessive or a genitive contruction depending on the context

Table 6.1: Desciption of MADA morphological features

Feat.	*Abv.*	*Values*
Article	art	YES: token has the definite article attached NO: token has not the definite article attached NA: preposition and numbers
Verb Aspect	aspect	PV: Perfective IV: Imperfective NA: only applicable for verbs
Grammatical case	case	ACC: Accusative GEN: Genitive NOM: Nominative NA: only applicable for prepositions and nouns
Clitics	clitic	YES: token has a clitic attached NO: token has not a clitic attached NA: prepositions and numbers
Conjunction	conj	YES: token has a conjunction attached NO: token has not a conjunction attached
Definiteness	def	YES: token is definite NO: token is not definite NA: only applicable for adjevtives and nouns
Gender	gen	MASC: Masculine FEM: Feminine NA: Not Applicable for prepositions, numbers

we are using another POS-tagger which is more accurate and has a larger tag-set. Thus, the MADA features which we used in our experiments and the annotations which we have chosen for them are the following ones:

- M_1=article;
- M_2=aspect;
- M_3=grammatical case;

Table 6.2: Desciption of MADA morphological features

Feat.	*Abv.*	*Values*
Idafa construction	idafa	POSS: possessive NOPOSS: non possessive NA: only applicable for nouns and adjectives
Grammatical mood	mood	I: Indicative S: Subjunctive J: Jussive NA: only applicable for verbs
Number	num	SG: Singular DU: Dual PL: Plural NA: not applicable for prepositions and numbers
Particle	part	YES: token has a particle attached NO: token has not a particle attached
Person	per	1: 1st person 2: 2nd person 3: 3rd person NA: not applicable for prepositions and numbers
POS	pos	POS tag-set
Voice	voice	ACT: Active PASS: Passive NA: only applicable for verbs

- M_4=clitic;
- M_5=conjunction;
- M_6=definiteness;
- M_7=mood;
- M_8=number;

- M_9=particle;
- M_{10}=person; and
- M_{11}=voice.

Part-Of-Speech (POS) tags and Base Phrase Chunks (BPC): To derive POS-tags and BPC we employ the AMIRA-1.0 system[2] described in [31]. Like the MADA system, AMIRA-1.0 is an SVM-based set of tools. The POS tagger performs at an accuracy of 96.2% and the BPC system performs at 95.41%. It is worth noting here that the MADA system produces POS tags however it does not produce BPC, hence the need for a system such as AMIRA-1.0. The authors report that they have built the POS-tagger using the Arabic Tree Bank for training and reducing the POS tag set to 25 tags.

Nationality (NAT): This feature is both a contextual and a lexical feature. We mark nationalities in the input text. This feature is the same that we have used in our previous experiments (see Subsection 5.1.2).

Corresponding English Capitalization (CAP): MADA provides the English translation for the words it morphologically disambiguates as a side effect of running the morphological disambiguation. In the process it taps into an underlying lexicon that provides bilingual information. The insight is that if the translation begins with a capital letter, then it is most probably an NE. Using this feature, will help to investigate the error-rate induced in the Arabic NER for lacking the capital letters in its orthography.

6.2 Data

In order to evaluate the different ML approaches and the impact of the features which we have mentioned in the previous section, we have conducted our experiments

[2]http://www1.cs.columbia.edu/∼mdiab/software/AMIRA-1.0.tar.gz

using an enhanced version of our corpus (i.e., ANERcorp) and the data used in Automatic Content Extraction (ACE) evaluation 2003, 2004 and 2005. The following subsection gives more details about these data-sets.

6.2.1 ANERcorp 2.0

It is the second version of the corpus which we used in our previous set of experiments (see Section 5.1). In this second version, the same tag-set has been conserved, however it has more tokens (see Table 6.3[3]) and several rounds of reviews were performed to ensure the consistency of the data.

6.2.2 ACE data

As we have previously mentioned in Subsection 3.2.3, the ACE evaluation consists of a set of tasks. For our experiments, we are interested only in the corpora which have been used for the EDT task (see Subsection 3.2.3). In this task, the participants are asked to extract three types of entities mentions in Arabic texts, namely: Named, Nominal and Pronominal (see Subsection 3.2.3 for more details). We remind the reader that the NER task is identical to the detection and classification of "only" named mentions in the EDR task. Therefore, in order to use the ACE data in our experiments, we had to perform first a preprocessing step in order to: (i) keep only the annotation of the named mentions; (ii) change the data format from the LDC format to the IOB2 annotation scheme (see Subsection 4.3.2).
In the ACE evaluation, the data is separated per genre, i.e. type of the data source. The genres which have been used in ACE 2003, 2004 and 2005 are the following:

- ACE 2003: Broadcast News (BN) and News Wire (NW);
- ACE 2004: BN, NW and Arabic Treebank (ATB);

[3]The details about the size of the second version of ANERcorp are shown in the next subsection in order to allow the reader to easily compare it with the other data-sets which we have used in the same experiments.

Corpus	*genre*	$Size_{train}$	$Size_{test}$	$Ratio_{NE}$	N_{NE}	Avg_{span}
ANERcorp 2.0	NW	144.48k	30.28k	11%	12989	1.47
ACE 2003	BN	16.34k	2.51k	14.7%	2100	1.32
	NW	29.44k	7k	13.4%	3405	1.43
ACE 2004	BN	50.44k	13.32k	11.5%	4609	1.6
	NW	51.74k	13.4k	11.8%	4839	1.6
	ATB	21.27k	5.25k	12.6%	2072	1.6
ACE 2005	BN	22.3k	5k	19%	3553	1.46
	NW	43.85k	12.3k	15.4%	5697	1.5
	WL	18k	3.2k	6.56%	968	1.43

Table 6.3: Characteristics of ANERcorp 2.0 and ACE 2003, 2004 and 2005 data

- ACE 2005: BN, NW and WebLogs (WL).

Table 6.3 shows the average size of the training ($Size_{train}$) and test ($Size_{test}$) in number of tokens for each data-set. It also shows the ratio of NEs tokens to the the total number of tokens ($Ratio_{NE}$), the number of NEs (N_{NE}) and the average number of tokens per NE (Avg_{span}) for each corpus. We consider that ANERcorp 2.0 is NW genre because most of its text was taken from newswire webpages and it does not include any data of BN or WL.

6.3 Experiments and Results

Our experiments aimed at determining the best feature-set and best ML approach which can help achieve significant impovement in the NER task. The feature space we have explored consists of the twenty two features which we have presented in 6.1. We have also chosen three ML approaches which have proved to be efficient for the NER task, ME, CRFs and SVMs (see Chapter 4). We carried out two main sets of experiments. The first set was necessary in order to decide on two parameters which will be used systematically in the rest of the experiments and dealt with the following

issues: word tokenization (clitic segmentation) and context window. The second set of experiments consists of an incremental features selection approach which will help to: (i) determine the feature-set which leads to the best results; (ii) have enough material to make a deep analysis on the impact on each feature (such analysis would not have been possible if we had used an automatic feature selection approach). The following subsections show the necessary details about these two experiment-sets and the obtained results.

6.3.1 Parameter Setting Experiments

We needed to first establish the impact of two experimental factors on NER performance, namely tokenization and the contextual window size as a preliminary precursor to our feature engineering experiments. Clitic tokenization, in a highly agglutinative language such as Arabic, has been shown to be useful for many NLP applications [47]. Intuitively, clitic tokenization serves as a first layer of smoothing in such sparse high dimensional spaces. We needed to decide on an optimal window size, therefore we experimented with different sizes. We set the tokenization to the ATB standard tokenization scheme. In these experiments we investigate window sizes of $-1/+1$ upto $-4/+4$ tokens/words surrounding a target NE. We carry out the experiments on the ANERcorp 2.0. Table 6.4 shows the CoNLL results obtained for the untokenized corpus (UNTOK) and the tokenized corpus (TOK), respectively.

	-1/+1	-2/+2	-3/+3	-4/+4
CXT+UNTOK	**71.66**	67.45	61.73	57.49
CXT+TOK	**74.86**	72.24	67.71	64

Table 6.4: Parameter setting experiments: Comparison among different window sizes, and the impact of tokenization on the NER task

From Table 6.4 we note that clitic tokenization has a significant positive impact on NER. We see an increase of 3 absolute points in F-measure when the text is clitic tokenized. Moreover, a context size of $-1/+1$ performs the best in this task. In

fact there seems to be a degrading effect correlated with window size, the bigger the window, the worse the performance.

6.3.2 Features Engineering

We conduct different sets of experiments to explore the space of possible features. We use clitic tokenized text and we define the context (CXT) to be $-1/+1$ as established in the previous section. The rest of our experiments were organized as following:

1. **Step 1:** Using only one feature at a time, we carry out an experiment with three ML appraches and record the impact (F-measure) of each feature.

2. **Step 2:** We manually ranked each feature according to its impact. If a feature is assigned different ranks for the different genres, we give it the most frequent rank. We have performed a manual ranking because the number of impacts to rank are very reduced, thus the manual ranking is affordable.

3. **Step 3:** At this stage we evaluate the SVMs, ME and CRFs approaches combining each time the N-top elements of the ranked features list. We have carried out experiments starting from N=*1* and up to from N=*22* to find out the optimal number of top features in order to obtain the best performance.

6.3.3 Results

Baseline: We have used the CoNLL baseline (see Subsection 5.2.2).

Step 2 results: Table 6.5 shows the final ranking of the features according to their impact.

Step 3 results: In order to show the F-measure obtained as the N best features are used together, we show three figures. Figure 6.1 shows the results for the ACE 2003 data, BN genre (best results). Figure 6.2 shows the behavior of the approaches for the ACE 2004 data, NW genre. Figure 6.3 shows the results obtained with ACE

Rank	Feature	Rank	Feature
1	POS	12	NAT
2	CAP	13	LEX_1
3	M_2	14	LEX_4
4	M_9	15	M_3
5	LEX_6	16	M_8
6	LEX_3	17	M_6
7	M_4	18	LEX_2
8	BPC	19	LEX_5
9	GAZ	20	M_5
10	M_1	21	M_7
11	M_{11}	22	M_{10}

Table 6.5: Features ranked according to their impact

2005, WL genre (worst results). Finally, Table 6.6 presents the baseline and the best results obtained for each corpus together with the number of features N and the approach which was employed. In the same table we also present the results which were obtained when all the features were combined. We measure the performance using the F-measure (F).

6.4 Results Discussion and Error Analysis

Features: All the features we used in our experiments are language-independent except the morphological ones which were extracted using MADA (M_x). These features helped significantly when the approaches are used within a corpus in which the NEs might occur in very random contexts (i.e. Weblogs genre). Let us consider the following sentence:

الرءيس الروسي بلَاديمير ب # وطن في اقرب ...

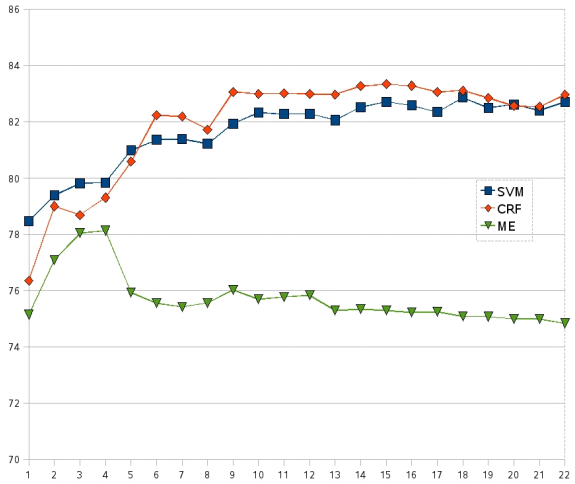


Figure 6.1: Results per approach and number of features for the ACE 2003 (Broadcast News genre) data

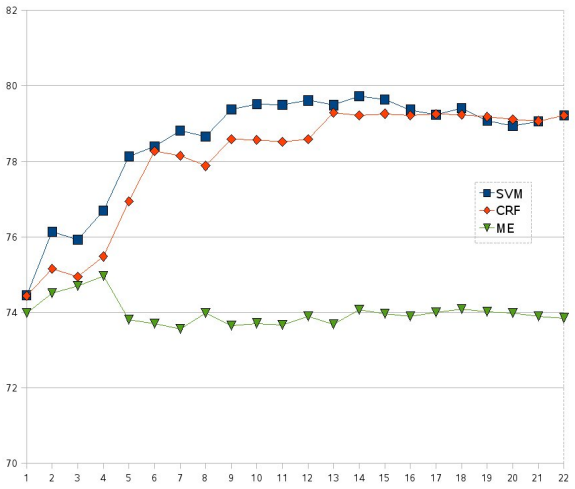


Figure 6.2: Results per approach and number of features for the ACE 2003 (Newswire genre) data

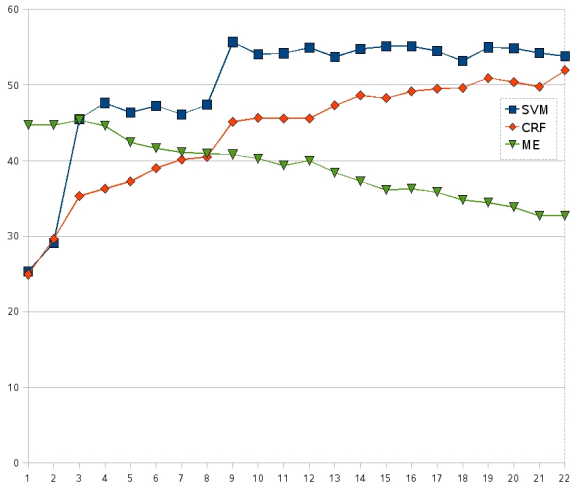


Figure 6.3: Results per approach and number of features for the ACE 2005 (Weblogs genre) data

Corpus	**genre**	**Baseline**	Best						All Features		
			SVMs		ME		CRFs		SVMs	ME	CRFs
			N	*F*	*N*	*F*	*N*	*F*	*F*	*F*	*F*
ANERcorp 2.0	NW	31.5	14	**81.04**	3	77.9	12	80.36	80.4	76.8	79.8
ACE 2003	BN	74.78	15	82.72	3	78.05	15	**83.34**	82.71	74.84	82.94
	NW	69.08	14	**79.72**	3	74.56	13	79.52	79.21	73.84	79.11
ACE 2004	BN	62.02	16	**77.61**	2	73.34	13	77.03	76.43	69.44	76.96
	NW	52.23	14	**74.13**	3	68.13	12	74.53	73.4	63.13	73.47
	ATB	64.23	15	75.43	2	69.95	13	**75.51**	75.34	64.66	75.48
ACE 2005	BN	71.06	15	**82.02**	3	77.67	14	81.87	81.47	75.71	81.1
	NW	58.63	15	76.97	3	70.31	13	**77.06**	76.19	67.41	75.67
	WL	27.66	12	**55.69**	2	44.96	14	53.91	53.81	32.66	51.81

Table 6.6: Best obtained results for each corpus. Best obtained results for each data-set are in bold characters

which can be written in Buckwalter tranliteration as:

Alr⟩ys Alrwsy blAdymyr b# wTn fy Aqrb ...

which can be translated to English as:

The Russian president Vladimir Putin in the nearest ...

The word "Putin", which in Arabic is generally spelled as بوتن (*bwtn*) was spelled differently in that specific text as بوطن(*bwTn*). On the other hand, the clitic-segmenter has mistakenly splitted the first character from the word considering it as the particle ب which generally means "in". Thus, the classifier has classified the word as *outside* because even if an NE can accept a particle as a suffix, it should be always attached to the first word of the NE (in case it is a multi-word NE as in our case). When we have added the M_{PART} feature, MADA has tagged the word وطن (*wTn*) as a word which has no particles attached. Consequently, it has been possible for the classifier to consider that the word ب # وطن (*b# w.tn*) is the second word of an NE. Other examples show when the morphological features have been helpful to the classifier when the words to classify are ambiguous: either because the surface word itself is ambiguous and a morphological disambiguation is needed or because the clitic-segmentation process has mistakenly spiltted the different components of the word and the resulting tokens are both ambiguous to the classifier and their POS-tag is completely wrong.

The performance of our NER system using the CAP feature in isolation has been ranked second (see Table 6.5) among all the others. This result confirms that the lack of capitalization in some languages such as Arabic hardens considerably the NER task. The use of lexical features (LEX_i) showed that only remarking the first and last three characters of a word (LEX_3 and LEX_6) can be useful for an NER approach. The rest of the lexical features occur randomly with all the classes and this is seen by the modeling approaches more as noise than as information. The lexical features have been mostly useful when the same NE might appear slightly different in the different parts of the corpus. For instance, in the sentence:

تقدم ايريل شَارون ...

which might be transliterated as:

tqdm Ayryl $Arwn ...

and translated to English as:

Ariel Sharon presented ...

the name "Ariel" might be tranliterated to Arabic as اريل (*Aryl*) or ايريل (*Ayryl*). In the training corpus used for the example which we have presented, this name has only appeared with the first transliteration. Hence, the classifier has classified it as *outside*. When the last trigram of the word was used as a feature (LEX_6), it has helped to indicate to the classifier that the word ايريل (*Ayryl*) shares the same last three characters with the word اريل (*Aryl*) which has been frequently seen as a person in the training data. Another similar example is the NE الوَاشنتن بوست (*AlwA$n.tn bwst*) "The Washington Post" which appears with the definite article (*Al*) only once in the corpus. Consequently, the classifier has been able to classify the word correctly only when the lexical feature LEX_6 was used. On the other hand, the lexical feature LEX_3, which concerns the first three characters of each word, has been mostly useful for the NE with different suffixes. The most remarkable example of such NEs are the nationalities which are tagged (depending on the context) as *location*, *person* or *outside*. Similarly to English the difference between the plural and the singular forms of most of the nationalities is the suffix, e.g. "Palestinian", فليسطيني (*flsTyny*) vs. "Palestinian**s**", فليسطيني **ين** (*flsTyny**yn***). In addition, the Arabic has the dual form which is very rarely used but also requires only adding a suffix to the singular form. In our data, we have seen that the LEX_i features have been very helpful to capture those cases.

Incremental Features Selection: The features incremental selection which we have used in our work helped to obtain results slightly better than using all the features together. Moreover, it is important to notice that the time to extract, train and test with only 14 or 15 features is almost half the time necessary for 22 features. Through the examples of errors which might be corrected when the best feature-set is used, we have noticed that it is simply because when we use only a selected feature-set,

we avoid providing the classifier with noisy information. One case is the NE "Holy Shrine", which is a facility, that in Arabic is said الحرم القدسي (*AlHrm Alqdsy*) and has been correctly classified. When information such as the starting bigram (which is *Al*, the definite article) has been added, the classifier has mistakenly driven the classifier to annotate both words as outside.

Approaches: The results obtained with ME were considerably lower than the ones obtained by CRFs and SVMs especially when the number of features exceeded 6. This shows that the ME approach is much more sensitive to noise and that it is more suitable to use this approach when a restricted number of accurate features is used. On the other hand, CRFs and SVMs showed very similar behaviors. Even though SVMs showed a slightly better performance when only the first 7 top features were used. Thus, according to our results it is not possible to determine an absolute superiority of the SVMs or the CRFs for the Arabic NER task. Through the data we have also observed that even if SVMs and CRFs give different 'false alarms' they tend to miss the same NEs.

Therefore, the choice of one or the other has to be based on the number of available features and their quality. In order to illustrate the difference among ME on one hand and SVMs and CRFs on the other, in Figure 6.4 we show four examples of NEs which have been captured by SVMs and CRFs and missed by ME. The underlined words are the missed NEs. For each example we give the Buckwalter transliteration and the English translation.

Classes vs. Features: In Table 6.6 we have shown only the overall F-measure obtained for each experiment. Table 6.7 shows the F-measure obtained for each class in order to give an overview on the performance of our system on the different classes. In this table, we give the results per class (FAC, LOC, PER and ORG) and we remind the overall F-measure ($Overall$) obtained when the best feature-set ($Best$) and when all the features (All) were used together with the 2003 BN data-set. The performance for the LOC and PER classes is higher, as it was expected, than the other classes because the data contains much more tokens of these classes than other classes NEs. A more interesting observation which might be made on the

1- ...يفلت صدام حسين من الحصار المفروض عليه ...

... >n yflt SdAm Hsyn mn AlHSAr AlmfrwD Elyh ...
... escape of Saddam Husein from the blockade imposed on him ...

2- ...مراسل صحيفة واشنطن بوست في القدس ...

... mrAsl Shyfp wA$nTn bwst fy Alqds ...
... the correspondent of Washington Post in Jerusalem ...

3- ...وهم القلائل موجودين في العالم من ...

... whm AlqlA}l mwjwdyn fy AlEAlm mn ...
... from the few existing in the world from ...

4- ...سفير الولايات الأمريكية بالقاهرة و عدد من كبار القادة ...

... sfyr AlwlAyAt Al>mrykyp bAlqAhrp w Edd mn kbAr AlqAdp ...
... the Embassador of United States in Cairo and a number of important leaders ...

Figure 6.4: Examples of NEs which were missed by the ME-based module and captured by SVMs and CRFs based ones

Class	*SVMs*		*ME*		*CRFs*	
	Best	*All*	*Best*	*All*	*Best*	*All*
FAC	**13.33**	**13.33**	23.64	**24**	**13.34**	0
LOC	86.66	**87.04**	**83.32**	81.29	**87.27**	87.03
ORG	**54.36**	51.31	47.56	**49.53**	**51.35**	49.12
PER	**81.55**	81.43	**76.16**	67.61	82.70	**82.83**
Overall	**82.72**	82.71	**78.05**	74.84	**83.34**	82.94

Table 6.7: Overall and per class results obtained for the 2003 BN data-set

results presented in Table 6.7 is that when we have selected the feature-set which helps to get the best performance, we did not achieve the best F-measure for each class individually. For instance, when the SVMs approach was used together with the best feature-set the F-measure obtained for the *LOC* class (86.66) is almost 0.4 points lower than what we have obtained when all the features were used (87.04). Same observation can be made on the *LOC* class when the ME approach was used (83.32 Vs. 81.29) and *PER* with the CRFs class (82.70 Vs. 82.83). In order to validate this

observation we propose to carry out another experiment which would allow to show the performance obtained for each class separately for the 2003 BN data on the CRFs approach and by selecting each time the best N features. The same ranking which we have shown on Table 6.5 would be used. Figure 6.5 shows the obtained results for this experiment. The results shown on Figure 6.5 confirm the feature-set which

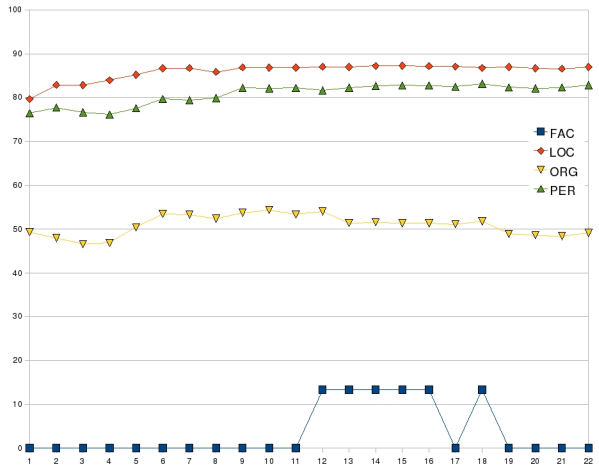


Figure 6.5: Per class results when each time the best N features are selected

helps to obtain the best overall performance (83.34 with 15 first features) is not the best feature-set when each class is considered separately. It also shows that the best feature-sets are 12, 15, 10 and 22 first features for the FAC, LOC, ORG and PER classes, respectively. Consequently, by using a feature selection approach based on the overall F-measure we have lost 3 and 0.13 points for the ORG and PER classes, respectively. The FAC class proved to be an exception in this case, obtaining either 13.34 or 0 as an F-measure is due to the rarety of the NEs of this class, thus when the system captures two (out of 15) of these NEs the F-measure is 13.34.

6.5 Concluding Remarks

In order to tackle the problem of achieving a high performance in the Arabic NER task, it is necessary to explore exhaustively the characteristics of the language and

determine the ones that are helpful, the ones that are not and the ones that are obstacles. In this chapter, we have presented a complete study of some of the most important features for the NER task. Some of these features, such as lexical and contextual features, are completely language-independent and very easy to extract. Others such as the POS-tag, BPC and morphological features need special tools to be extracted, however the study of their impact can be beneficial for all the Semitic languages NLP research community because they share almost all the morphological dimensions. Another important variable in our study is the appropriate ML approach to use. For this purpose, we have used three ML approaches which have proved to be the most efficient ones for the NER task, namely ME, SVMs and CRFs. Finally, in order to ensure the relevancy of our experiments and the significance of our results, we have used 9 data-sets of different genres (newswire, broadcast news, Arabic Treebank and weblogs) and from different sources (our corpus (i.e., ANERcorp 2.0) and the ACE 2003, 2004 and 2005 corpora). The most important conclusions which can be deduced from our study are as follows:

1. **Performance:** As we have mentioned previously, our best results (F-measure=83.34) were achieved for ACE 2003 BN data. The results for the BN genre were better than the other genres across the board. The WL data are the most noisy ones, because the texts are basically people discussions and thus an NE might appear in different surface forms and in very different contexts. SVMs and CRFs showed very similar behaviors and a slight difference in the overall performance, whereas ME performed very poorly in comparison with them. The performance using different numbers of features (see Figures 6.1, 6.2 and 6.3) show that when only few first features are used ($\sim$ 4 features), SVMs always outperforms CRFs. However, the best performace was obtained by using CRFs.

2. **Features:** The POS-tags and the capitalization features proved to have a very high impact on the performance of the NER system. The results obtained by using only those two features (79.1) with the ACE 2003 BN data yields the 94.9% of the best performance (83.34). Among the first 15 best ranked features, 6 are morphological ones. Those morphological features have been very useful

to enhance the performance of the system, especially for the WL data where the classifier decisions relies heavily on the internal evidences of the tokens. The feature which reflects the impact of using external resources (i.e., GAZ), has been ranked 9th. This means that using external resources can be very helpful because the gazetteers which we have used in our experiments are considerably small (see Subsection 5.1.2).

3. **Per Class:** Our results show that a very good performance has been obtained, together with a very good result-set to analyze the different error-types, when we have used an incremental selection approach. On the other hand, they also suggest that a selection of the adequate feature-set for each class separately can lead to a higher performance. Such a feature-selection approach seems to be necessary especially because in any data-set the frequency of appearence of the different NEs classes might be drastically different. Consequently, for a class viewpoint, we use a different training sets sizes for each class and thus different features should be used for each one.

Chapter 7

A Classifiers Combination Approach

"None of us is as strong as all of us."

- an IBM research motto -

In the experiments which we have presented in the previous chapter (see Chapter 6) we have explored a large number of features and ML approaches in order to find the key elements which could lead us to achieve a high performance in the Arabic NER task. The results showed clearly that either we use SVMs or CRFs, the performance would be approximately the same, whereas the optimization of the feature-set can be more helpful to enhance it. Using ~ 15 best features has proved to be more efficient than using the whole available feature-space (~ 22). In order to complete our error-analysis we have added a last experiment which has shown that when we state that a feature F_i has proved to be the feature with the highest impact on the NER system, it does not necessarily mean that it is the best feature for each class "separately".

This statement remains true when it is seen from a linguistic perspective. For instance, the tokens which constitute an NE of class *Person*, e.g. "Glenn Gritzner", "Barack Obama" or "Richard Branson", will always be assigned a POS-tag of type "NNP". This makes the POS-tag an important feature for the person class since it provides very strong signals to the classifier in order to capture the NEs better. Other NEs classes such as "Organization", e.g. "'United Nations Organization for Education, Science, Culture and Communications", might contain different types of tokens from a POS-tag viewpoint and thus the classifier would rather rely more on other features, e.g. use "number" and "gender" to know if different words are part of the same NE.

In this chapter, we present our experiments which attempt to prove the correctness of the reasoning which we have just stated. These experiments use a classifier per class, where each classifier uses the adequate feature-set and ML approach for the concerned class, and finally the outputs of all the classifiers are combined in one [13]. This approach differs from the one used in [59], which used the same feature-set for all the classifiers, because it optimizes not only the ML approach but also the feature-sets. Carrying out those experiments, implies the investigation of the following issues:

1. The impact of each feature individually on each class;

2. An automatic approach to rank the features according to their impact on each

class;

3. The feature-set which helps to achieve the best performance by using incremental selection; and

4. A combination strategy which resolves conflicts among classifiers.

In Section 7.1 we present the feature space which will be explored in our experiments and we emphasize the difference with the features which have been used in the experiments presented in Chapter 6. Section 7.2 describes the Fuzzy Borda Voting Scheme. This approach have been used in our experiments in order to rank each feature according to its impact on the classification of each of the NE classes. In order to optimize the needed time for our experiments and to obtain reliable results, we have made some modifications on our data-sets. In Section 7.3 we present details about our data-sets. The experiments and results are presented in Section 7.4. Finally, we discuss the obtained results in Section 7.5 and we draw our conclusions in Section 8.5.

7.1 Features

The space of features which we have used for the experiments we present in this chapter are the same as the ones we have introduced in Chapter 6, except some of the morphological features which we have ommited for their unreliability. The idea of removing these features came after a discussion with the authors of [46] and the developpers of the Morphological Analysis and Disambiguation of Arabic (MADA) tool which we use to extract the morphological features. The authors informed us that even if the accuracy of extraction of the features is very high, some of them have a very low F-measure and a better performance of the NER system is expected if they are removed from the feature-set. Those features are:

1. Mood;

2. Grammatical case; and

3. Voice.

We have also decided to remove other morphological features because they present unnecessary information. Following, we present these features and we explain why they are not necessary:

1. Article: We have another features which indicates if a word is definite or not, i.e. Definiteness.

2. Clitic: The text we are using is already segmented, thus no word is supposed to have attached clitics.

3. Conjunction: The conjunction tokens are also segmented from the stem, and similarly to the "Clitic" feature, no word is supposed to have any attached conjunctions.

Hence, the remaining morphological features which we have used in our experiments are: (i) aspect; (ii) person; (iii)definiteness; (iv) gender; and (v) number.

Moreover, we have modified the value-set of some of these features in order to render it more NER task oriented. The value-sets of these feature are as follows:

1. **Aspect (M_{ASP})**: In Arabic, a verb may be imperfective, perfective or imperative. However, since none of the NEs is verbal, we decided to turn this feature into a binary feature, namely indicating whether a token is marked for Aspect (APP, for applicable) or not (NA, for not applicable).

2. **Person (M_{PER})**: In Arabic, verbs, nouns, and pronouns typically indicate person information. The possible values are *first, second* or *third* person. Again, similar to *aspect*, the applicability of this feature to the NEs is more relevant than the actual value of *first* versus *second*, etc. Hence, we converted the values to APP and NA, where APP applies if the person feature is rendered as *first, second* or *third*.

3. **Definiteness (M_{DEF})**: All the NEs by definition are definite. The possible values are DEF, INDEF or NA.

4. **Gender (M_{GEN})**: All nominals in Arabic bear *gender* information. According to MADA, the possible values for this feature are masculine (MASC), feminine (FEM), and neuter (or not applicable NA), which is the case where gender is not applicable for instance in some of the closed class tokens such as prepositions, or in the case of verbs. We use the three possible values MASC, FEM and NA, for this feature. The intuition is that since we are using a sequence model, we are likely to see agreement in *gender* information in participants in the same NE.

5. **Number (M_{NUM})**: For almost all the tokens categories (verbs, nouns, adjectives, etc.) MADA is able to provide the grammatical *number* with a high F-measure. In Arabic, the possible values are singular (SG), dual (DU) and plural (PL). The correlation of the SG value with most of the NEs classes is very high. Heeding the underlying agreement of words in Arabic when they are part of the same NE, the values for this feature are SG, DU, PL and NA (for cases where *number* is not applicable such as closed class function words).

7.2 Classic and Fuzzy Borda Voting Scheme

In this section, we describe the approach which we have chosen to rank the different features acording to their impact. In order to do so, we needed an approach which is able to deduce a final ranking from several rankings and takes into consideration the weight assigned to each feature as well, e.g. Fuzzy Borda Voting Scheme (FBVS). This approach satisfies the conditions which we have mentioned and has been successfuly used in other NLP tasks such as Geographical Information Retrieval [83] and Word Sense Disambiguation [20].

7.2.1 Classic Borda Voting Scheme

In some elections, the voters are asked not only to provide the name of the candidate they think deserves to win but to give a ranking of all the candidates. Thereafter, in order to decide on the final winner, a method is needed to provide a final

ranking of the candidates from all the rankings receveived from the voters, i.e., The Classic Borda Voting Scheme (CBVS) (also known as "Borda Count"). The best way to explain how CBVS works is by providing a clear example such as the following one:

Let us suppose that we had five experts $e1$ .. e_5 who provided a ranking of three candidates c_1, c_2 and c_3. These rankings are shown in Table 7.1.

e_1	e_2	e_3	e_4	e_5
c_1	c_1	c_2	c_1	c_2
c_2	c_3	c_3	c_3	c_3
c_3	c_2	c_1	c_2	c_1

Table 7.1: Experts rankings

The first step is to convert, for each expert, the ranking into a "preference" matrix. This matrix contains simply 0's and 1's in order to express if the concerned expert prefers one candidate to another. Thus, the rankings shown in Table 7.1 would be transformed in the following matrices:

$$M_{e_1} = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}, M_{e_2} = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, M_{e_3} = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}, M_{e_4} = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$
$$, M_{e_5} = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

For instance, the matrix M_{e_1} can be read as follows:

- It is not possible to express a preference of a candidate to itself $\Rightarrow$ the element $1,1$ of the matrix should contain value 0[1].

[1]Through this same example it is possible to show that assiging a value of 1 or 0 to the matrix diagonal elements does not change the final ranking of the candidates.

- The ranking of the expert e_1 shows a preference of c_1 to c_2 $\Rightarrow$ the element $1,2$ (i.e. row 1, column 2) of the matrix should contain value 1.
- The ranking of the expert e_1 shows a preference of c_1 to c_3 $\Rightarrow$ the element $1,3$ (i.e. row 1, column 3) of the matrix should contain value 1.
- The ranking of the expert e_1 shows that c_2 is not prefered to c_1 $\Rightarrow$ the element $2,1$ (i.e. row 2, column 1) of the matrix should contain value 0.
- ...

Once all the matrices are derived from the experts ranking, a sum of the element of each row (i.e., each candidate) of each matrix is calculated. In the case of our example we obtain the following vectors:

$$V_{e_1} = \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix}, V_{e_2} = \begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix}, V_{e_3} = \begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix}, V_{e_4} = \begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix}, V_{e_5} = \begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix}$$

Therafter, we sum up for each row the elements of all the vectors in order to obtain one final vector. The following vector is obtained in our case:

$$V_{final} = \begin{pmatrix} 6 \\ 5 \\ 4 \end{pmatrix}$$

Finally, the candidates are ranked according to the their correspondant element in V_{final}. Table 7.2 reminds the rankings provided by the experts at the beginning and shows the final ranking obtained by CBVS.

e_1	e_2	e_3	e_4	e_5	*CBVS*
c_1	c_1	c_2	c_1	c_2	c_1
c_2	c_3	c_3	c_3	c_3	c_2
c_3	c_2	c_1	c_2	c_1	c_3

Table 7.2: Experts rankings and CBVS result ranking

7.2.2 Fuzzy Borda Voting Scheme

Subsection 7.2.1, showed how CBVS is able to deduce from many rankings of different voters (or experts) a final ranking. It also showed that CBVS is based on binary values and does not take into consideration the degree of preference of an candidate c_i to a candidate c_j for the different experts. For instance, let suppose that instead of the rankings shown in Table 7.1, the experts have provided both a *weight* for each candidate (see Table 7.3). In such a case, we need a more sophisticated version of CBVS which takes into account the weights provided by the experts, i.e. FBVS.

e_1	e_2	e_3	e_4	e_5
$c_1: 9$	$c_1: 9.5$	$c_2: 9$	$c_1: 8.5$	$c_2: 9.5$
$c_2: 8.5$	$c_3: 9$	$c_3: 4$	$c_3: 8$	$c_3: 3$
$c_3: 8$	$c_2: 8.5$	$c_1: 2$	$c_2: 7.5$	$c_1: 1$

Table 7.3: Experts rankings and weights

FBVS performs in a very similar way as CBVS. However, in order to include the weights in the decision making of the final ranking, some changes were necessary when the experts preference matrices and vectors (see Subsection 7.2.1) are calculated. In order to illustrate those changes, we use the example shown in Table 7.3.

According to FBVS, $r^i_{j,k}$ (the j,k element of the matrix M_{e_i}) may be computed using the fomula 7.1.

$$r^i_{j,k} = \frac{w^i_j}{w^i_j + w^i_k} \tag{7.1}$$

$$M_{e_1} = \begin{pmatrix} 0.5 & 0.51 & 0.53 \\ 0.49 & 0.5 & 0.51 \\ 0.47 & 0.49 & 0.5 \end{pmatrix}, M_{e_2} = \begin{pmatrix} 0.5 & 0.53 & 0.51 \\ 0.47 & 0.5 & 0.48 \\ 0.49 & 0.52 & 0.5 \end{pmatrix}, M_{e_3} = \begin{pmatrix} 0.5 & 0.18 & 0.33 \\ 0.82 & 0.5 & 0.69 \\ 0.66 & 0.31 & 0.5 \end{pmatrix}$$

$$, M_{e_4} = \begin{pmatrix} 0.5 & 0.53 & 0.51 \\ 0.47 & 0.5 & 0.48 \\ 0.49 & 0.52 & 0.5 \end{pmatrix}, M_{e_5} = \begin{pmatrix} 0.5 & 0.1 & 0.25 \\ 0.9 & 0.5 & 0.76 \\ 0.75 & 0.24 & 0.5 \end{pmatrix}$$

Consequently, an important property which can be observed on the produced matrices is: $r^i_{j,k} = 1 - r^i_{k,j}$. Thus, in order to compute the corresponding vectors with a method which is coherent with the one used in CBVS (see Subsection 7.2.1), we may sum up only the elements which are > 0.5 (corresponding to the elements $= 1$ in CBVS) for each row of each matrix[2]. The resulting vectors are as follows:

$$V_{e_1} = \begin{pmatrix} 1.04 \\ 0.51 \\ 0 \end{pmatrix}, V_{e_2} = \begin{pmatrix} 1.04 \\ 0 \\ 0.52 \end{pmatrix}, V_{e_3} = \begin{pmatrix} 0 \\ 1.51 \\ 0.66 \end{pmatrix}, V_{e_4} = \begin{pmatrix} 1.04 \\ 0 \\ 0.52 \end{pmatrix}, V_{e_5} = \begin{pmatrix} 0 \\ 1.66 \\ 0.75 \end{pmatrix}$$

Thereafter, we sum up the weights for each row. The final weights are the following:

$$V_{final} = \begin{pmatrix} 3.12 \\ 3.68 \\ 2.45 \end{pmatrix}$$

Finally, we rank the candidates accrodingly to their correspondant weight in V_{final}. Hence, the final ranking using FBVS is shown in Table 7.4 together with the rankings and weights provided by the experts at the beginning.

In the CBVS raking result (see Table 7.2), c_1 was ranked first simply because it

[2]Similarly to CBVS, the final ranking is the same whether we include the diagonal elements in the sum or not.

e_1	e_2	e_3	e_4	e_5	*FBVS*
c_1 : 9	c_1 : 9.5	c_2 : 9	c_1 : 8.5	c_2 : 9.5	c_2 : 3.68
c_2 : 8.5	c_3 : 9	c_3 : 4	c_3 : 8	c_3 : 3	c_1 : 3.12
c_3 : 8	c_2 : 8.5	c_1 : 2	c_2 : 7.5	c_1 : 1	c_3 : 2.45

Table 7.4: Experts rankings and CBVS result ranking

was the candidate which most times was ranked first by the experts (three times, whereas c_2 was ranked two times as first and c_3 zero times). The confidence of the voters is not taken into consideration at all. Using FBVS, we were able to employ the voters confidences (i.e. weights). As a result, c_2 was ranked first because even though it was ranked as first candidate only two times, the experts (i.e. voters) assigned it a very high confidence in comparison with the other candidates. On the other hand, in all the cases where c_1 was ranked first, its confidence was realtively close to the confidences of the other candidates. Hence, the FBVS is an approach which combines both *frequency* and *confidence* in order to come up with a final ranking of the candidates.

7.3 Data

As we have previously mentioned, the research study which we present in this chapter has the goal of finding an optimized feature-set for each NE class. In order to do so, we will carry out an incremental features selection approach based on the performance obtained for each feature individually (see Section 7.4). Therefore, if the measure used to decide on the degree of usefulness of a feature to a certain class is the F-measure obtained, it is necessary to split the data into three parts:

1. *train*: the training set;
2. *dev*: the developement set to measure the impact of the different features;
3. *test*: the test set which could be used for final test.

Corpus	*genre*	$Size_{train}$	$Size_{dev}$	$Size_{test}$
ACE 2003	BN	12.41k	4.12k	5.63k
	NW	23.85k	9.5k	9.1k
ACE 2004	BN	45.68k	14.44k	14.81k
	NW	45.66k	15.2k	16.9k
	ATB	19.04k	6.16k	6.08k
ACE 2005	BN	18.54k	5k	8.4k
	NW	40.26k	12.5k	13.83k
	WL	13.7k	6.2k	6.4

Table 7.5: Statistics of ACE 2003, 2004 and 2005 data

In order to do so, we have used for each genre 3 folds for *train*, 1 fold for *dev* and 1 fold for *test*. Consequently, the training set is smaller than the one used in the experiments which we have presented in Chapter 6 and achieving a high performance becomes very challenging. Table 7.5 shows the average size of the corpora which will be used in our experiments. We have not used the ANERcorp 2.0 in our experiments because its size is much bigger than the ACE data and the obtained performance is very similar to the ACE 2003 BN data (see Table 6.6).

7.4 Experiments and Results

As we have previously mentioned, our objective is to find the optimum set of features per NE class and then combine the outcome in a global NER system for Arabic. According to the experiments which we have carried out in Section 6.3.1, using the tokens context of size $-1/+1$ and tags context window of -1 empirically yields the best performance. In all the experiments which we present in this section, we will keep these context windows unchanged. It is also important to mention that we will only report results for the CRFs and SVMs approaches because ME showed a very poor performance in comparison to the mentioned approaches (see Section 6.3).

7.4.1 Training per Individual NE Class

In order to observe the exact impact of each feature on each class we have trained at each time using only one feature and turning off the other annotations for the other classes in the training set. We have observed that there are generally two possible ways in which we can change the data for this purpose. Those two possible settings are the following:

1. **3-way classification**: Setting all the other NE classes (i.e. others than the concerned class) to O, similar to non-NE words, thereby yielding a 3-way classification, namely, B-NE for the class of interest, I-NE for the class of interest, and O for the rest including the rest of the NEs and other words and punctuation;

2. **4-way classification**: This second setting discriminated between the other NE classes that are not of interest and the rest of the words. The intuition in this case is that NE class words will naturally behave differently than the rest of the words in the data. Thereby, this setting yields a 4-way classification: B-NE for class of interest, I-NE for class of interest, NE for the other NE classes, and O for the other words and punctuation in the data.

Let consider the following sentence:

"John Hennessy the President of Stanford University lives in California"

Three NEs appear in this sentence, namely "John Hennessy" as a person NE, "Stanford University" as an organization NE and "California" as a location NE. In the case we want to build a classifier which only focuses on the person NEs, we have two possible ways in which we can change the annotation of the NEs "Stanford University" and "California", i.e., using the 3-way classes annotation or the 4-way one. Table 7.6 shows the tokens of our example, the initial annotation (*Init. annot.*), the 3-way classes annotation (*3-way*) and the 4-way one (*4-way*).

In order to contrast the 3-way vs the 4-way classification, we run experiments using SVMs that we evaluate using the ACE 2003 data set with no features (apart from 'CXT' and 'current word') using SVMs. Table 7.7 illustrates the yielded results:

Tokens	*Init. annot.*	*3-way*	*4-way*
John	B-PER	B-PER	B-PER
Hennessy	I-PER	I-PER	I-PER
the	O	O	O
President	O	O	O
of	O	O	O
Stanford	B-ORG	O	B-NE
University	I-ORG	O	I-NE
lives	O	O	O
in	O	O	O
California	B-LOC	O	B-NE

Table 7.6: Illustrating example of the difference between the 3-way and 4-way classes annotations

Class	*Num(classes)*	*BN genre*	*NW genre*
GPE	3	76.72	79.88
	4	**76.88**	**80.99**
PER	3	64.34	42.93
	4	**67.56**	**44.43**
ORG	3	41.73	25.24
	4	**46.02**	**37.97**
FAC	3	**23.33**	15.3
	4	**23.33**	**18.12**

Table 7.7: F-measure Results using 3-way vs. 4-way class annotations using SVMs

For all the NE classes we note that the 4-way classification yields the best results. Moreover, we counted the number of 'conflicts' obtained for each NE classification. A 'conflict' arises when the same token is classified as a different NE class by more than one classification system (an ML technique together with an NE class); for example, a classification system may tag a token as a B-GPE while another would tag it as B-ORG. Our findings are summarized as follows:

- **3 classes**: 16 conflicts (8 conflicts in BN and 8 in NW). 10 of these conflicts happened between GPE and PER classes, and 6 between GPE and ORG classes.
- **4 classes**: 10 conflicts (3 conflicts in BN and 7 in NW). 9 of these conflicts happened between GPE and ORG classes, and only one between GPE and FAC classes.

An example of a conflict observed using the 3-way classification that disappeared when we apply the 4-way classification is the sentence shown in Figure 7.1 (transliterated as *n$rt SHyfp WA$nTn tAyms tqryrA*), which is translated as 'The Washington Times newspaper published a report'.

نشرت صحيفة واشنطن تايمس تقريرا

Figure 7.1: Illustrating example for classifiers conflict

When trained using a 3-way classifier, 'Washington' is assigned the tag GPE for the GPE classifier system and as an ORG for the ORG classifier system. However, when trained using the 4-way classification approach, this conflict is resolved as an ORG in the ORG classifier system and an NE in the GPE classifier system. Thereby, confirming our intuition that a 4-way classification is better suited for the individual NE classification systems. Accordingly, for the rest of the experiments in this chapter reporting on individual classification systems, we use the 4-way classification approach.

7.4.2 Measuring the impact of Individual features per class with FBVS Ranking

An experiment is run for each fold of the data. We train on data annotated for one NE class, one ML method (i.e., SVMs or CRFs), and one feature. For each experiment we use the tuning set for evaluation (i.e., obtaining the F-measure performance value).

After obtaining the F-measures for all the individual features on all the data genres and using the two ML techniques, we rank the features (in a decreasing order) according to their impact (F-measure obtained) using FBVS (see Section 7.2.2). This results in a ranked list of features for each ML approach and data genre per class. Tables 7.8 and 7.9 show the obtained rankings for SVMs and CRFs, respectively.

7.4.3 Feature set/class Generalization

Once the features are ranked, we incrementally experiment with the features in the order of the ranking. Thus, we train with the first feature and measure the performance on the tuning data, then we train with the second one together with the first feature, i.e. the first two features and measure performance, then the first three features and so on. Thereafter, we select the first n features that yield to the best performance (after which additional features do not impact performance or cause it to deteriorate). We use the top n features to tag the test data and compare the results against the system when it is trained on the whole feature set. Since the total number of features is 16, each ML classifier is trained and evaluated on the tuning data 16 times for each genre. The best number of features per class per genre per ML technique is determined based on the highest obtained F-measure. Finally, the last step is combining the outputs of the different classifiers for all the classes. In case of conflict, where the same token is tagged as two different NE classes, we use a simple heuristic based on the classifier precision for that specific tag, favoring the tag with the highest precision. Table 7.10 illustrates the obtained results. For each data set and each genre it shows the F-measure obtained using the best feature set

Feats	PER	GPE	ORG	FAC	VEH/WEA
LEX_1	16	12	12	15	4
LEX_2	3	15	7	12	5
LEX_3	10	6	15	10	6
LEX_4	7	16	4	8	7
LEX_5	15	14	16	16	8
LEX_6	12	4	10	9	9
GAZ	14	7	9	11	3
BPC	4	13	13	6	1
POS	1	5	1	4	16
NAT	8	3	2	3	15
M_{ASP}	13	2	5	2	10
M_{PER}	11	11	3	5	14
M_{DEF}	9	9	6	7	11
M_{GEN}	5	8	11	13	12
M_{NUM}	6	10	14	14	13
CAP	2	1	8	1	2

Table 7.8: Ranked features according to FBVS using SVMs for each NE class

Feats	PER	GPE	ORG	FAC	VEH/WEA
LEX_1	6	12	14	1	4
LEX_2	2	10	1	16	5
LEX_3	5	3	10	5	6
LEX_4	7	7	3	15	7
LEX_5	3	5	2	6	8
LEX_6	10	4	4	7	9
GAZ	9	6	6	11	3
BPC	8	9	8	4	1
POS	1	1	5	14	16
NAT	13	8	7	13	15
M_{ASP}	16	15	12	8	10
M_{PER}	11	16	9	12	14
M_{DEF}	12	14	11	9	11
M_{GEN}	15	13	16	10	12
M_{NUM}	14	11	13	3	13
CAP	4	2	15	2	2

Table 7.9: Ranked features according to FBVS using CRFs for each NE class

and ML approach. We show results for both the dev and test data using the optimal number of features *Best Feat-Set/ML* contrasted against the system when using all 16 features per class *All Feats/ML*. The table also illustrates three baseline results on the test data only. *FreqBaseline*: For this baseline, we assign a test token the most frequent tag observed for it in the training data, if a test token is not observed in the training data, it is assigned the most frequent tag which is the O tag. *MLBaseline*: In this baseline setting, we train an NER system with the full 16 features for all the NE classes at once. We use the two different ML approaches yielding two baselines: $\textbf{MLBaseline}_{SVMs}$ and $MLBaseline_{CRFs}$.

It is important to note the difference between the *All Feats/ML* setting and the *ML-Baseline* setting. In the former: all 16 features are used per class in a 4-way classifier system and then the classifications are combined and the conflicts are resolved using our simple heuristic whereas in the latter case of *MLBaseline* the classes are trained together with all 16 features for all classes in one system. Since different feature-sets and different ML approaches are used and combined for each experiment, it is not possible to present the number of features used in each experiment in Table 7.10. However, Table 7.11 shows he number of features and the ML approach used for each genre and NE class.

7.5 Results Discussion and Error Analysis

Performance: As illustrated in Table 7.10, our *Best Feat-set/ML* setting outperforms the baselines and the *All Feats* $\{SVM—CRF\}$ settings for all the data genres and sets for both the test data. Moreover, the *Best Feat-set/ML* setting outperforms both *All Feats* $\{SVM—CRF\}$ settings for the *dev* data for all genres.

The results yielded from the ML baselines are comparable across all the data genres and the two ML approaches.

Comparing the global ML baseline systems against the All Feature Setting, we see different performances across the different genres and different data-sets where the ML baseline outperforms the All Feature Setting and vice versa.

Comparing the performance per genre across the different data-sets, we note bet-

		ACE 2003		*ACE 2004*			*ACE 2005*		
		BN	*NW*	*BN*	*NW*	*ATB*	*BN*	*NW*	*WL*
	FreqBaseline	73.74	67.61	62.17	51.67	62.94	70.18	57.17	27.66
	$MLBaseline_{SVMs}$	80.58	76.37	74.21	71.11	73.14	79.3	73.9	54.68
	$MLBaseline_{CRFs}$	81.02	76.18	74.67	71.8	73.04	80.13	74.75	55.32
dev	*Best Feat-set/ML*	**83.41**	**79.11**	**76.9**	**72.9**	**74.82**	**81.42**	**76.07**	**54.49**
	All Feats. SVMs	81.79	77.99	75.49	71.8	73.71	80.87	75.69	53.73
	All Feats. CRFs	81.76	76.6	76.26	71.85	74.19	79.66	74.83	36.11
test	*Best Feat-set/ML*	**83.5**	**78.9**	**76.7**	**72.4**	**73.5**	**81.31**	**75.3**	**57.3**
	All Feats. SVMs	81.76	77.27	69.96	71.16	59.23	81.1	72.41	55.58
	All Feats. CRFs	81.37	75.89	75.73	72.36	74.21	80.16	74.43	27.36

Table 7.10: Final Results Obtained with selected features contrasted against all features combined

	BN		*NW*		*ATB*		*WL*	
	N	*ML*	*N*	*ML*	*N*	*ML*	*N*	*ML*
Person	12	SVM	14	SVM	9	SVM	11	SVM
Location	10	SVM	7	SVM	16	CRF	14	SVM
Organization	9	CRF	6	CRF	10	CRF	12	CRF
Facility	10	CRF	14	CRF	14	SVM	16	CRF
Vehicle	3	SVM	3	SVM	3	SVM	3	SVM
Weapon	3	SVM	3	SVM	3	SVM	3	SVM

Table 7.11: Number of features and ML approach used to obtain the best results

ter performance across the board for BN data over NW per year. The worst results are obtained for ACE 2004 data for both BN and NW genres. There is no definitive conclusion that a specific ML approach is better suited for a specific data genre. We observe a slightly better performance for the CRFs ML approach in the $MLBaseline_{CRFs}$ condition for both BN and NW.

The worst performance is yielded for the WL data. This may be attributed to the small amount of training data available for this genre.

Comparing *dev* and *test* performance, we note that the overall results on the dev data are better than those obtained on the test data. This is somehow expected given that the weights for the FBVS ranking are derived on the basis of the dev data used as a tuning set. The only counter example for this trend is with the WL data genre, where the test data yields a significantly higher performance for all the conditions except for *All Feats CRFs*.

Inconsistencies in the data lead to many of the observed errors. The problem is that the ACE data is annotated primarily for a mention detection task which leads to the same exact words not being annotated consistently. For instance, the word 'Palestinians' would sometimes be annotated as a GPE class whereas in similar other contexts it is not annotated as a named entity at all. Since we did not manually correct these cases, the classifiers are left with mixed signals.

Features: The quality of the performance of the different feature extraction tools such as AMIRA (for POS tagging and BPC) and MADA (for the morphological features) are optimized for NW data genres. Thereby, yielding suboptimal performance on the WL genre, leading to more noise than signal for training.

However, comparing relative performance on this genre, we see a significant jump from the most frequent baseline **FreqBaseline** ($F_{\beta=1}$=27.66) to the best baseline $\textbf{MLBaseline}_{CRFs}$ ($F_{\beta=1}$=55.32). We see a further significant improvement when the **Best Feat-set/ML** setting is applied yielding an $F_{\beta=1}$=57.3. Interestingly, however the $\textbf{MLBaseline}_{CRFs}$ yields a much better performance ($F_{\beta=1}$=55.32) than **All**

Feats CRF with an $F_{\beta=1}$=27.36. This may indicate that a global system that trains all classes at once using CRFs for sparse data is better than training separate classifiers and then combining the outputs. It is worth noting the difference between **MLBaseline**$_{SVMs}$ and **All Feats SVM**, F-measure=54.68 and F-measure=55.58, respectively. This result suggests that SVMs are more robust to less training data as illustrated in the case of the individual classifiers in the latter setting.

Features vs. Classes: As observed in Tables 7.9 and 7.8, the ranking of the individual features could be very different for two NE classes. For instance, the BPC is ranked 4th for the PER class, is ranked 13th for GPE and ORG classes. The disparity in ranking for the same individual features strongly suggests that using the same features for all the classes cannot lead to a global optimal classifier.

With regards to morphological features, we note in Table 7.8, that Definiteness, M_{DEF}, is helpful for all the NE classification systems, by virtue of being included for all optimal systems for all NE classification systems. Aspect, M_{ASP}, is useful for all classes except PER. Moreover, M_{GEN} and M_{NUM}, corresponding to Gender and Number, respectively, contributed significantly to the increase in recall for PER and GPE classes. Finally, the Person feature, M_{PER} contributed mostly to improve the classification of ORG and FAC classes. Accordingly, observing these results, contrary to previous ones by [35], our results strongly suggest that significant impact morphological features have on Arabic NER, if applied at the right level of granularity.

The VEH and WEA classes both exhibit a uniform ranking for all the features and yield a very low performance. This is mainly attributed to the fact that they appear very rarely in the training data. For instance, in the ACE 2003, BN genre, there are 1,707 instances of the class PER, 1,777 of GPE, 103 of ORG, 106 of FAC and only 4 for WEA and 24 for VEH.

SVMs vs. CRFs Comparing SVMs and CRFs, we note that they both show a high performance in the Arabic NER task.

However, as illustrated in Table 7.11, SVMs outperformed CRFs on most of the classes. Interestingly, CRFs tend to model the ORG and FAC entities better than

SVMs. Hence, it is not possible to give a final word on the superiority of SVMs or CRFs in the Arabic NER task. In fact, it is necessary to conduct a per class study, as the one we present in this chapter, in order to determine the right ML approach and features to use for each class. Our best global NER system combined the results obtained from both ML approaches.

7.6 Concluding Remarks

In the previous chapter, we have adopted an incremental features selection approach in order to determine the feature-set which helps obtain the best F-measure for Arabic NER. However, our error analysis showed that when each NE class is considered individually the best feature-set might be different.

In this chapter, we have presented the experiment-set which we have used in order to confirm the previous chapter conclusions. In order to do so, we have:

1. Manually filtered the morphological features used in the previous chapter in order to keep only the ones which the extraction tool developers report that they perform at high F-measure;

2. Conducted experiments using at each step one feature and modifying the data by keeping only one NE class. These experiments helped to show the impact of each feature for each NE class;

3. Useed the Fuzzy Borda Voting Scheme approach in order to rank the features for each NE class. The oucome of this ranking shows which features are important for which class;

4. Performed experiments by selecting at each iteration for each class the n best features. Thus, the best feature-set for an NE class would be simply the n best features which helped to obtain the best F-measure.

These experiments have been conducted for both the SVMs and CRFs ML approaches. We also have splitted our data into training, developement and test sets in

order to perform the features optimization steps on the development set and obtain the final results using the test set.

Our best results yielded an F-measure of 83.5 on the ACE 2003 BN data. This result outperforms the best result obtained in the experiments which we have presented in the previous chapter (83.34) even though the training data which has been used is 0.75 as much of the ones which have been used in the previous chapter experiments.

Moreover, the most significant conclusions suggested by our experiments are as follows:

1. When it is needed to build a classifier for only one NE class (e.g. B-PER and I-PER), a better performance is obtained if the rest of the classes (e.g. B-ORG, I-ORG, B-FAC ... I-LOC) are turned into a virtual class (e.g. B-NE and I-NE) than if they are turned into the outside class (i.e., O). In addition, when a virtual NE class is used it has been observed that there are less conflicts (tokens tagged as NE by more than one classifier) between the different classifiers. The intuition behind using this 4-way classification is to avoid that a classifier would consider some feature signals (e.g. NNP of the POS-tag feature) to be indicators for the token to be a non-NE.

2. Although SVMs and CRFs did not show a significant difference in the final results of the previous chapter, the results suggest that those approaches have shown to be different in many ways. First of all, the features which each of these approaches have considered to be important for a certain NE class might be very different. For instance, using the first unigram of a token as a feature (i.e., LEX_1) has been considered to be the least important feature(ranked 16th) for the PER class by SVMs, whereas CRFs has ranked it as the 6th most important feature for that class (see Tables 7.8 and 7.9). However, it is not possible to explain why exactly one or another feature has been ranked differently for CRFs and SVMs. It was expected that each of these ML approaches would behave differently with the different features because as the literature states (see Chapter 3), CRFs assign a weight to each feature and uses those weights afterwards

for classification. On the other hand, SVMs use mainly the so-called Support Vectors (SVs), i.e., closest data points to the classes separation hyperplan, in order to determine the most adequate hyperplane. This characteristic of SVMs has classified this approach as very robust to noise because it previliges some data points (i.e., SVs) to others and thus not all the data points are treated equally as it is the case in ME and CRFs. What our experiments suggest then is that for some classes, where we have few data points such as ORG and FAC, all the features and all the data points should be used (i.e., CRFs). Whereas for others, where we have a considerable amount of data points (i.e., PER and LOC), it is more adequate to use the most robust ML approach as there are more chances to have noise (i.e., SVMs).

Apart from the morphological feature which we have used in our experiments, the rest of the feature-space elements which we have explored are all language-independent and affordable. Therefore, our experiments show how a significantly accurate Arabic NER system can be built by using freely available tools and resources. Differently from other comparative studies on the adequate Ml approach which can be used for the NER task, we have not only reported the best F-measure obtained but we have also shown the performance of each of the ML approaches by class and which feature-set is more suitable to be used with each one of them. Finally, our research study strongly suggests that an accurate Arabic NER system is ought to use a classifier/class approach where each classifier uses the appropriate feature-set and ML approach to the concerned class.

Chapter 8

Using Large External Resources

"He who says 'I know!' is more ignorant than the ignorant; one should always seek to learn from the others."

- Ostad Elahi -

8.1 Introduction

The hypotheses which have been stated, the experiments which have been carried out and the results which have been obtained in the previous chapters have shown that using a multi-classifier approach helps to achieve a very good performance for Arabic NER. Our research work has lead to obtain a framework that employs the best ML approach and feature-set for each NE class. This framework has been used to carry out a study to show the impact of each feature on Arabic NER. Our results claim that (see Chapter 7 for more details):

1. POS-tags and capitalization are the features which have the greatest impact in terms of F-measure;

2. Lexical features are very helpful to capture NEs which tend to appear in different surface forms (e.g. for non-Arabic NEs which have different possible transliterations), and NEs which appear with different affixes;

3. The rich morphology of the Arabic language could be used to improve the NER system. In order to do so, it is necessary to use a tool which helps to extract the morphological features of the Arabic words and the features values are ought to be changed to fit with the needs of the global systems.

4. The use of gazetteer-based features has proved to be very benificial for all the NE classes. As shown in Table 7.8, these features has been ranked among the first eleven ones for all the NE classes even though our gazetteers are very small.

Whereas the three first claims do not suggest any clear directions to expand our research work, the last one clearly points that it is necessary to use larger amount of external resources and study their impact on the global NER system.

The idea of using external resources attempts, mainly, to help the NER system to better capture the NEs which do not appear in the training data (i.e. Out-Of-Vocabulary NEs). What we have used so far as external resources in our experiments are manually extracted dictionaries with very reduced size. In this chapter we use a unsupervised approach which helps to acquire much larger dictionaries from parallel

aligned corpora. This approach (which we describe with more detailes in Section 8.2) is based on transferring knowledge about NEs from another language. In order to do so, we have automatically annotated the English part of a large English-Arabic aligned parallel corpus with an accurate English NER model. Thereafter, we have propagated the annotation from the English to the Arabic part and then extracted different features based on different linguistic motivations. Hence, in addition of being a continuation of the previous chapters, the study which we present in this chapter shows similtaneously:

1. How to use an NER model of a source language to enhance a target language NER model;
2. The impact of three different features extracted from parallel corpora;
3. A detailed discussion of the obtained results and error analysis of the system outputs.

In order to carry out our experiments we have used the same multi-classifier framework which we have described in Chapter 7. Apart from building an optimized classifier for each NE class, this framework has also the advantage of integrating new features without repeating the experiments for the features which have been already ranked. Thus, for the new features which we have extracted from the parallel corpus, we just have to:

1. Measure their impact separately;
2. Find out their ranking among the other features; and
3. Carry out the experiments with incremental selection to find out an optimized feature-set for each class.

The remainder of this chapter is organized as follows. In Section 8.2 we describe in details our approach of extracting features from the English-Arabic parallel corpus. Thereafter, we show the characteristics of the parallel corpora which we have used and give the obtained results in Section 8.3. We discuss the results and give an error analysis in Section 8.4 and we finally draw our conclusions in Section 8.5

8.2 Extracting Feature from Aligned Corpus

As we have previously stated, our purpose is to not only use gazetteers as external resources but rather extract many features of different types from a parallel corpus and study closely their impact on the global NER system. The idea behind using an aligned parallel corpus is basically to be able to use an NER model built on another language (English in our case). Such a model would be much more accurate since the English language has a simpler morphology than Arabic, has access to much more accurate POS-taggers and has much more available enriched semantic resources. The English NER model which we use in our experiments is originally designed to be used for the Mention Detection (MD) task (see Chapter 3, Subsection 3.2.3). In our experiments we keep only the named mentions to reduce the system from MD to NER. This model is MEMM-based[113][1] and uses a large variety of features which might be categorized as follows:

- **Contextual**: which consist of using the current word and a context window of $-/+2$;
- **Syntactical**: POS-tags issued from three different POS-taggers and BPCs;
- **Classifiers**: outcomes of many classifiers which have been trained on other corpora;
- **Semantic**: word sense tags using the English WordNet to give the synonyms and hyperonyms of each word;

More details about the English MD model can be found in [113].
As illustrated in Figure 8.1, the first step consists of extracting necessary features and running the English NER model on the English side of the parallel corpus, resulting in a tagged text. Thereafter, we use the word alignment file (parallel data) to project NEs from English text to Arabic: the alignment file has a one-to-many structure and describes which words of the English side correspond to which words of the Arabic

[1] We use the IBM's NER model.

one. The result of this step is an annotated Arabic text with NEs obtained by propagation from English.

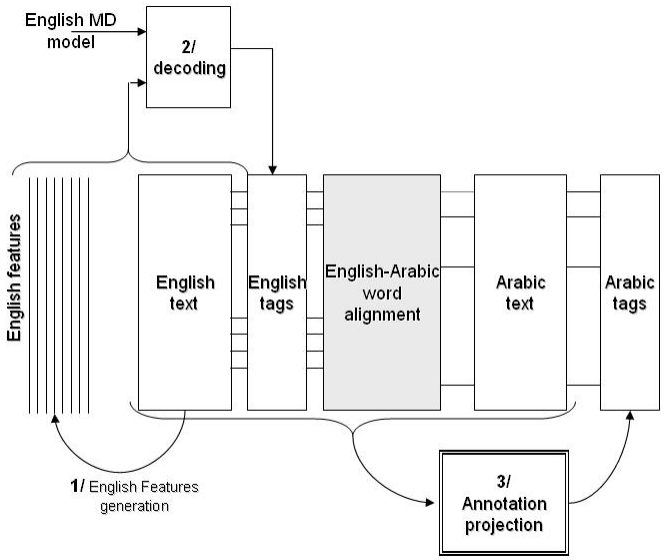


Figure 8.1: Annotation and projection steps

It is also important to mention that this approach has two major noise sources, which might have a negative impact on the final results, namely: (i) the*English model errors*; and (ii) *Alignment errors*: those type of errors are relatively rare when the data is manually aligned.

8.2.1 Features extraction

Once the corpus in the target language,i.e. Arabic, is tagged with NEs obtained by propagation, we extract different kinds of features that we use to enhance the Arabic NER model. Those features are as follows:

1. Gazetteers: we group NEs by class in different dictionaries. During decoding, when we encounter a token or a sequence of tokens that is part of a dictionary, we fire its corresponding class; the feature is fired only when we find a complete match between sequence of tokens in the text and in the dictionary.

2. n-gram context features: it consists of using the annotated corpus in the target language to collect n-gram tokens surrounding an NE. We organize those contexts by NE class and we use them to tag tokens which appear in the same context during decoding. These tags will be used as additional feature in the NER model. For instance, if we consider that the person NE صدَام حسين (SdAm Hsyn - Sadam Husein) appears in the following sentence:

صرح أمس أن صدَام حسين يترأَّس نظَامَا فَاشلَا

which might be transliterated as:

SrH Ams An SdAm Hsyn ytrAs nZAmA fA$lA

and translated to English as:

declared yesterday that Sadam Husein governs a failed system

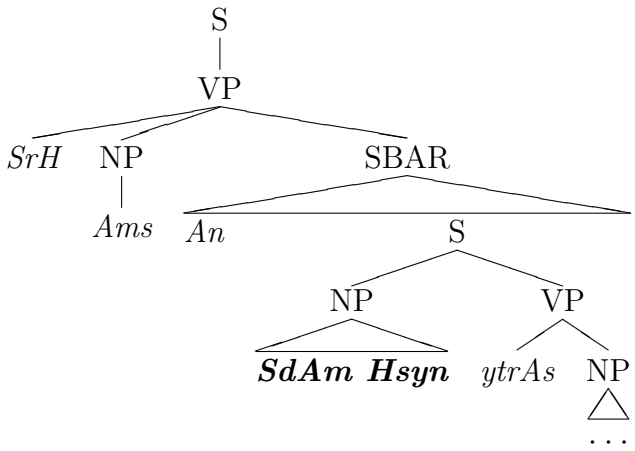
the context n-grams that would be extracted are:

. **Left n-grams:** L_1=أن (An - that), L_2=أمس أن (Ams An - yesterday that), etc.

. **Right n-grams:** R_1=يترأَّس (ystrAs - governs), R_2=يترأَّس نظَامَا (ytrAs nZAmA - governs a system), etc.

During decoding we fire as a potential person every sequence of tokens (e.g. Ben Ali) that appears in the same n-gram context as a person in the annotated Arabic data: preceded by L_i or followed by R_i. We proceed similarly for other NE classes.

3. Head-word based features: it considers that the lexical context in which the NE appeared is the sequence of the parent sub-trees head words in a parse-tree. For instance, if we consider the sentence which we have used in the previous example, the corresponding parse tree would be the following:



The first parent sub-tree head word is the verb 'ytrAs' (governs), the second one is 'An' (that) and the third one is the verb 'SrH' (declared). During decoding, we fire an information saying that a sequence of token (e.g. John) can be a person if it appears with the same n first parent sub-tree head words as a person in the annotated Arabic language data. This is used as additional information in the NER model. The parse-tree parent head words represent a more global context than the one provided by the n-grams.

4. Parser-based features: it attempts to use the *syntactic environment* in which an NE might appear. In order to do so, for each NE in the target language corpus we consider only labels of the parent non-terminals. For instance, according to the parse tree of the example which we have used earlier, for the person 'SdAm Hsyn', the first parent non-terminal label is 'S', the second one is 'SBAR' and the third one is 'VP'. During decoding, we fire as potential person every occurrence of a sequence of tokens (e.g. Mohamed VI) that appear with the same n first parent non-terminals labels as a real person in the annotated Arabic data. We proceed similarly for other NE classes: organization, facility, location, etc.

Gazetteer based features are the most natural and expected kind of features that one would extract from Arabic text annotated by propagation from English. On the other hand, n-gram context, head-word based and parse-based features are motivated by the works of [23, 34] for language modeling. These authors show that building a language model based on n-grams together with head-word based and parse-based features led to a considerable reduction on perplexity and an interesting improvement

on speech recognition system performance.

8.3 Experiments and Results

8.3.1 Parallel Data

Half of the parallel corpus which we have used in our experimens is hand-aligned by professional annotators at IBM T. J. Watson Center, the other half is publicly available data at the LDC. The corpus has texts of five different genres, namely: newswire, news group, broadcast news, broadcast conversation and weblogs. The Arabic part contains 941,282 tokens, after propagating the annotation from the English part we obtained 57,290 NEs. Table 8.1 shows the number of NEs for each class.

Table 8.1: Number of NEs per class in the Arabic part of the parallel corpus annotated by propagation from English

Class	Number of NEs
FAC	998
LOC	27,651
ORG	10,572
PER	17,964
VEH	85
WEA	20

8.3.2 Feature Individual Impact

After propagating the annotation from the English to the Arabic side of the parallel corpus and extracted the new features which we have described in Subsection 8.2.1, we have conducted some experiments, using both SVMs and CRFs, with each of the extracted features separately in order to measure the impact of each one of them in

terms of F-measure improvement points. In Table 8.2 we show the obtained results when no features are used (*No Feat.*) and when each of the features extracted from the parallel corpus was used individually. Those features are the following:

1. *autoGazet.*: automatically extracted gazetteers;
2. L_1, L_2 and L_3: left unigram, bigram and trigram , respectively;
3. R_1, R_2 and R_3: right unigram, bigram and trigram , respectively;
4. HW_1, HW_2 and HW_3: the first, first and second, first, second and third parse-tree head words, respectively;
5. PB_1, PB_2 and PB_3: the first, first and second, first, second and third parse-tree parent non-terminals, respectively;

The experiments were carried out using the training and developement sets of the data-sets which have been used in Chapter 7 (i.e., ACE 2003, 2004 and 2005 data-sets).

Similarly to the experiments which we have described in Chapter 7, we have performed an incremental selection of the features for each NE class separately. Thereafter, we have built a classifier for each NE class which uses the feature-set yielding the best results together with the most adequate ML approach. Table 8.3 shows the final results. Similarly to the table of final results in Chapter 7, Table 8.3 shows for each data set and each genre the F-measure obtained using the best feature set and ML approach. It shows results for both the dev and test data; using the optimal number of features *Best Feat-Set/ML* contrasted against the system when using all 29 features per class *All Feats/ML*. The table also illustrates three baseline results on the test data only. *FreqBaseline*: For this baseline, we assign rach token in the test data the most frequent tag observed for it in the training data, if a this token is not observed in the training data, it is assigned the most frequent tag which is the O tag. *MLBaseline*: In this baseline setting, we train our NER system with the full 16 features for all the NE classes at once. We use the two different ML approaches yielding two baselines: $MLBaseline_{SVMs}$ and $MLBaseline_{CRFs}$. As we have mentioned in

Table 8.2: Individual impact of the features extracted from the hand-aligned parallel data using SVMs.

	ACE 2003		*ACE 2004*			*ACE 2005*		
	BN	*NW*	*BN*	*NW*	*ATB*	*BN*	*NW*	*WL*
No Feat.	*70.78*	*65.34*	*68.87*	*60.47*	*61.61*	*72.18*	*62.06*	*38.69*
autoGazet.	72.95	65.4	69.9	59.96	65.43	75.54	63.12	40.38
L_1	73.45	65.52	70.76	60.02	66.34	76.67	**63.86**	45.06
L_2	71.81	65.3	69.91	59.56	65.7	75.14	62.56	43.77
L_3	71.77	64.82	69.31	59.54	64.22	74.94	61.97	43.5
R_1	71.97	64.92	69.53	59.99	65.49	75.03	61.98	44.12
R_2	72.06	65.1	70.01	60.0	66.12	75.12	61.96	43.53
R_3	72.01	64.93	69.93	59.77	65.98	75.09	62.01	42.19
HW_1	72.45	66.11	69.65	59.96	64.55	75.33	62.14	42.94
HW_2	71.94	65.73	69.23	59.54	64.65	74.78	62.34	43.78
HW_3	71.99	65.82	68.91	59.51	63.76	73.93	61.91	43.60
PB_1	**76.93**	**66.39**	**73.35**	**60.44**	**66.9**	**77.11**	62.54	**47.69**
PB_2	75.39	66.13	71.48	60.33	66.01	76.63	63.13	45.69
PB_3	74.14	65.65	71.32	59.57	64.7	75.55	62.69	43.68

Table 8.3: Final results Obtained with selected features contrasted against all features combined

		ACE 2003		*ACE 2004*			*ACE 2005*		
		BN	*NW*	*BN*	*NW*	*ATB*	*BN*	*NW*	*WL*
	FreqBaseline	73.74	67.61	62.17	51.67	62.94	70.18	57.17	27.66
	$MLBaseline_{SVMs}$	81.52	76.57	76.33	70.03	73.24	79.64	74.3	55.52
	$MLBaseline_{CRFs}$	81.74	76.79	77.01	71.02	72.93	79.92	74.93	56.70
dev	*Best Feat-set/ML*	**83.93**	**79.72**	**78.54**	**72.8**	**74.97**	**81.82**	**75.92**	**55.65**
	All Feats. SVMs	82.32	78.65	77.33	71.75	74.39	81.11	75.73	55.32
	All Feats. CRFs	82.55	78.13	77.89	71.92	74.76	79.65	75.01	40.03
test	*Best Feat-set/ML*	**84.32**	**79.40**	**78.12**	**72.13**	74.54	**81.73**	**75.67**	**58.11**
	All Feats. SVMs	82.45	79.25	76.02	71.45	73.61	81.67	72.66	56.31
	All Feats. CRFs	82.38	78.87	77.27	71.76	**74.78**	80.79	75.06	34.51

the previous chapter, the difference between the *All Feats/ML* setting and the *ML-Baseline* setting consists in that, the former: all 29 features are used per class in a 4-way classifier system and then the classifications are combined and the conflicts are resolved using our simple heuristic (the token gets the class asigned by the classifier with the highest precision), whereas in the latter case of *MLBaseline* the classes are trained together with all 29 features for all classes in one system.

Since different feature-sets and different ML approaches are used and combined for each experiment, it is not possible to present the number of features used in each experiment in Table 8.3. However, Table 8.4 shows the number of features and the ML approach used for each genre and NE class.

8.4 Results Discussion and Error Analysis

The results presented in Table 8.2 show that the features which we have extracted from the parallel corpus have been helpful for most of the data-sets. The highest

Table 8.4: Number of features and ML approach used to obtain the best results

	BN		*NW*		*ATB*		*WL*	
	N	*ML*	*N*	*ML*	*N*	*ML*	*N*	*ML*
Person	20	SVM	21	SVM	17	SVM	24	SVM
Location	19	SVM	15	SVM	26	CRF	24	SVM
Organization	18	CRF	16	CRF	20	CRF	22	CRF
Facility	15	CRF	21	CRF	22	SVM	25	CRF
Vehicle	3	SVM	3	SVM	3	SVM	3	SVM
Weapon	3	SVM	3	SVM	3	SVM	3	SVM

improvements have been obtained for the 2003 BN and 2005 WL data-sets, whereas no improvement has been obtained for the 2004 NW corpus, for instance. The improvement varies significantly from one data-set to another because it highly depends on the number of NEs which the model has not been able to capture using the contextual, lexical, syntactical and morphological features and were correctly tagged in the parallel corpus.

Impact of the features extracted from the parallel corpus per class: Similarly to the other features, the new features which we have extracted from the parallel corpus had different levels of impact for the NE classes. Here we present the three most important features for each class:

1. **FAC**: L_1, PB_2 and *autoGaz.*;
2. **LOC**: PB_2, L_1 and HW_1;
3. **PER**: L_1, R_3 and HW_2;
4. **ORG**: *autoGaz.*, PB_2 and L_2; and
5. **VEH and WEA**: as we have stated in Chapter 7, there are only few VEH and WEA NEs in the corpus which does not allow to measure properly the impact of the different features on these classes.

Weblogs vs. context: In the previous chapter, our results have shown that the random contexts in which the NEs tend to appear in the WL documents stand against obtaining a significant improvement. The results which we have obtained in this chapter confirm this statement because the features which use a more global context (PB_i and HW_i) have helped to obtain better results than the ones which we have obtained using local context (L_i and R_i).

Manually and automatically built gazetteers: The former one benefits from a 100% noise-free items whereas the latter has been extracted automatically and thus even if the size is significantly larger it contains some noisy elements which resulted from a wrong annotation of the English model. However, our results have shown that a model which uses the automatically extracted gazetteers outperforms the manually built one across the board.

8.5 Concluding Remarks

In this last chapter, we have presented a research study which attempts at propagating knowledge about NEs from another language (i.e., English) in order to enhance the Arabic NER model. The purpose behind such a research work is to explore the impact of using large amount of automatically extracted external resources. In order to do so, we have used a manually aligned English-Arabic parallel corpus of almost one million words. We have used an accurate Engish MD model to annotate the English part of the parallel corpus. Thereafter, we have kept only the named mentions, that is, we have removed the tags of the nominal and pronominal mentions, and propagated the obtained annotation to the Arabic part of the parallel corpus using the alignment information. Finally, we have extracted a number of features for each of the NEs and have carried out experiments in order to measure the impact of each one of them.
The obtained results have shown that:

1. A significant improvement has been obtained for almost all the data-sets. For those where only a slight improvement has been observed (e.g. 2004 NW and

2005 NW), we have noticed that we have not achieved a significant gain either because the feature did not fire for the NEs which were not captured when only contextual, lexical, syntactical and morphological features were used or because it has mistakenly fired for tokens which are not NEs (i.e., created false alarms);

2. The model which employs automatically extracted gazetteers has yielded better results than the one using manually built gazetteers; and

3. The performance obtained when a local context feature has been added is lower than the one obtained when a more global one (based on parse-tree information) was used for almost all the data-set. The results for the WL data confirm that one of the main obstacles to obtain a high performance for WL data is that NEs tend to appear in very random contexts.

Last but not least, our framework which selects and uses the best feature-set and ML approach for each NE class has proved to be flexible to add additional features without requiring any adaptation.

Chapter 9

Conclusions

In this document we have presented our achievements in the Arabic Named Entity Recognition task. We remind the reader that this task aims at the identification and classification of the NEs within an open-domain text. An accurate NER system might be used to enhance the performance in other NLP tasks such as Information Retrieval [102], Machine Translation [8], Text Clustering [105] and especially Question Answering [36][44][81]. For each of these tasks, an NER system is used with the appropriate class-set in order to preprocess the data and thus provide more information to the global system (see Chapter 3). From a ML viewpoint, the NER task is a sequence classification problem where each word has to be classified as a non-NE (oftenly called "Outside" words), as the beginning of an NE of class c (tagged as $B-c$) or as inside an NE of class c (tagged as $I-c$). In the literature, researchers have shown that the supervised approaches are the most adequate ones to tackle the NER problem. The most successful supervised approaches have been ME, SVMs and CRFs.

The English NER has been subject of research during many years and a considerable amount of published works show the obtained results using different ML approaches and feature-sets. Moreover, many evaluation campaigns such as MUC-6, CoNLL 2002 and 2003 have provided test-beds to the research community in order to encourage the investigation of novel approaches for the English NER task and allow the participants to compare their results. Consequently, the English NER has reached a high

performance which has not been achieved in almost any of the other languages. In this document, we have presented a research study which concerns mainly Arabic NER. The Arabic language has its own scripture and inherits the agglutinative characteristic from the Semitic languages family (see Chapter 2). From the NER viewpoint, the fact that the Arabic scripture lacks capitalization (i.e., classifiers cannot rely on a special signal which indicates the existence of an NE) makes the detection of NEs within the text more challenging than in the languages which support capitalization. Its agglutinative peculiarity causes sparseness in the data and this decreases the quality of training significantly. In brief, these two charactersitics of the Arabic language, among others, have brought new challenges which need to be addressed directly. Therefore, language-independent approaches cannot be used to build a high performance Arabic NER system. Not many research works on the Arabic NER task have been published. Thus, it was necessary to conduct a full study which:

1. Shows what characteristics of the Arabic language are obstacles to the NER task, suggests solutions, proves their efficiency by emperical results and attempts to use the rich morphology of the Arabic language in order to enhance the performance.

2. Conducts a comparative study among the different ML approaches, namely ME, SVMs and CRFs, and presents them in a beneficial way to the research community: i.e., reports the obtained performance for each class separately and shows the difference of behavior of each ML approach.

3. Uses different data-sets of different genres in order to confirm the efficiency and robustness of the approaches which have helped obtain a high performance.

The main subject of this thesis was to satisfy this need by conducting different experiment-sets which we have presented in Chapters 5, 6, 7 and 8.

9.1 Findings and Research Directions

As we have previously mentioned, our experiment-sets have been conducted with the aim of exploring the different feature-sets and approaches which would help obtain a high performance NER system. We have carried out our experiments on 9 different data-sets which cover four genres, namely Newswire, Broadcast News, Weblogs and Arabic Treebank. In this section, we summarize our major findings wheareas the details of each experiment can be found in Chapters 5, 6, 7 and 8.

1. *Arabic language vs. NER task*: In order to tackle the data sparseness problem caused by the complex morphology of the Arabic language, we have performed a *tokenization* preprocessing step which consists in separating the stem word from the clitics attached to it. Our experiments show that a gain of 3.1 points might be obtained after performing tokenization. Concerning the lack of capitalization in the Arabic language, we have imported capitalization from the English language by using a lexicon-based translation and using it as a feature. Imported capitalization has been ranked as the second most important feature in a set of 22 features (POS-tag is the first one). Almost 3.5 points of F-measure have been obtained as a gain when the capitalization feature was used. We have also used the Morphological Analysis and Disambiguation for Arabic (MADA) tool in order to extract 14 different morphological features for each Arabic word. This experiment was an attempt to use the Arabic rich morphology in order to enhance the performance of Arabic NER. Our experiments show that more gain might be obtained if each of these features is used for the NE class where it has shown to have the greatest impact: i.e. none of these features has shown to be striclty efficient or non-efficient for all the classes. Thus, the NER task might benefit from the rich morphology of the Arabic language if each of the morphological features is used with the appropriate NE class.

2. *Features vs. Performance*: In our experiments, we have used lexical, contextual, syntactical and morphological features. Concerning the contextual ones, we have first performed a pre-setting experiment where we have explored different

possible contextual window sizes. Our experiments have shown that a window of $-1/+1$ yields to the best performance. The syntactical and morphological features have shown to be the best ones for the NER task in terms of overall F-measure impact. However, when we have explored the impact of each feature for each class separately, our observation was that the impact of a feature on one class might be different from its impact on another one. For instance, The third most important feature to capture the person NEs was the POS-tag. The same feature has ranked the fifth most important feature to capture location NEs. For this reason, we have adopted a multi-classifier approach in which each classifier deals with only one NE class. For each classifier we have selected the feature-set which most suits the concerned class. Finally, we have combined the outputs of the different classifiers in one outcome and the results outperformed those obtained when the feature-set was optimized using the overall F-measure even when we have used only 75% as much of the training data which we have used in the one-classifier approach.

3. *ML approaches vs. NE classes*: In our experiments, we have used ME, SVMs and CRFs. The ME approach has proved to behave very poorly in comparison with the rest of the ML approaches. Whereas SVMs and CRFs have shown that their behaviors are very similar when the comparison is done on the overall F-measure level. When the results are compared per class, it has been observed that CRFs and SVMs might obtain very different results. For instance, the obtained results when CRFs have been used for NE classes such as organization and facility have been higher than when the SVMs were used. On the other hand, SVMs have obtained better results on the rest of the NE classes. Similarly to the feature-sets, our results have proved that it might not be possible to state that one ML approach is better than another for the NER task. Thus, the best results were obtained when we have used a multi-classifier approach where each classifier uses the best ML technique for the concerned class and combines the outputs of the different classifiers into a single one.

4. *External knowledge:* One of the most important experiments which we have carried out concerned the study of importing knowledge about NEs from another language. In order to do so we have used a one-million words English-Arabic parallel corpus. Our study has shown that the improvement obtained from such an approach highly depends on the evaluation corpus: that is, the improvement is much higher when the NEs which were uncorrectly tagged by the baseline system exist within the Arabic part of the parallel corpus.

9.2 Thesis Contributions

We consider that the major contributions of this thesis can be classified in two different categories:

I. Contributions to NER in general:

1. We compared the obtained results from different ML approaches, namely, Maximum Entopy, Support Vector Machines and Conditional Random Fields. The comparison has been shown on an overall F-measure level and per class. The results which we have presented could be very useful as a solid background for anyone who needs to build an efficient NER system.

2. We have used an incremental approach to optimize the feature-set for each NE class for a multi-classifier approach. For this purpose, we have explored the use of the Fuzzy Borda Voting Scheme to rank the features according to their impact for each NE class. Our results and error analyses show that a very efficient multi-classifier approach is to select the adequate ML technique and feature-set for each NE class separately and combine their outcomes at the end.

3. We have conducted one of the first attempts to transfer NER knowledge from a resource rich language, such as English, to another language, such as Arabic. In our experiments we extract contextual and parse-tree based features and we show the obtained impact both when they are used individually and when they are combined with the other features.

II. Contributions to Arabic NER:

1. Provide a deep study of the Arabic NER task and show the approaches and features which help obtain a high performance.

2. Show how the Arabic rich morphology might be employed in order to build a robust NER system.

3. Build an NER corpus, the ANERcorp, of more than 150k tokens freely available for the research community[1]. ANERcorp has been annotated by following the CoNLL 2003 and 2003 guidelines and has been reviewed several times to ensure annotation coherence.

4. Build NER models based on SVMs and CRFs ready to be used and freely available for the research community[2]. These models can be used in order to have an Arabic NER system which can be tuned by using the study which we provide in this document.

9.3 Further Challenges

According to our understanding of the Arabic NER task, the research directions which might be taken in order to achieve even higher performance are as follows:

1. According to the research work which we have described in Chapter 8, the obtained improvement when we import knowledge from another language highly depends on the number of uncorrectly tagged tokens which can be found in the parallel corpus. One way to increase this number is by using a larger parallel corpus. However, manually aligned parallel corpora are very costly to build. A possible research direction would be to use an automatically aligned parallel corpus. The hardest part would be dealing with all the noise induced by the uncorrectly aligned segments.

[1] http://www.dsic.upv.es/grupos/nle/downloads.html
[2] http://www.dsic.upv.es/grupos/nle/downloads.html

2. *Enrichment of the Arabic WordNet (AWN)*: The existing AWN offers a complete platform which allows to be enriched for most of the NLP tasks. If the AWN is enriched for the NER task, it would allow to explore the impact of using semantic features for Arabic NER. Such an approach is promising because the use of synonymy might allow to have more information about the words in the test set which have not been seen in the training set (OOVs).

Bibliography

[1] S. Abney, M. Collins, and S. Amit. Answer Extraction. In *Proc. of the ANLP 2000*, 2000.

[2] S. Abuleil and M. Evens. Extracting names from arabic text for question-answering systems. *Computers and the Humanities*, 2002.

[3] ACE-2003. The ACE 2003 (ACE03) Evaluation Plan. In *Proc. of the Automatic Content Extraction 2003*, 2003.

[4] ACE-2004. The ACE 2004 (ACE04) Evaluation Plan. In *Proc. of the Automatic Content Extraction 2004*, 2004.

[5] ACE-2005. The ACE 2005 (ACE05) Evaluation Plan. In *Proc. of the Automatic Content Extraction 2005*, 2005.

[6] C. Amaral, A. Cassan, H. Figueira, A. Martins, A. Mendes, P. Mendes, C. Pinto, and D. Vidal. Priberam's Question Answering System in QA@CLEF 2007. In *Working Notes of CLEF 2007*, Budapest, Hungary, 2007.

[7] D. Applet, J. Hobbs, J. Bear, D. Israel, M. Kameyama, A. Kehler, D. Martin, K. Myers, and M. Tyson. SRI International FASTUS System: MUC-6 Test Results and Analysis. In *Proc. of the 6th Conference on Message Understanding*, pages 237–248, 1995.

[8] B. Babych and A. Hartley. Improving Machine Translation Quality with Automatic Named Entity Recognition. In *Proc. of EACL-EAMT*, 2003.

[9] R. Baeza-Yates and B. Ribeiro-Neto. *Modern information retrieval*. New York: ACM Press; Addison-Wesley, 1999.

[10] BBN. BBN: Description of the PLUM System as Used for MUC-6. In *Proc. of the 6th Conference on Message Understanding*, pages 55–70, 1995.

[11] Y. Benajiba, M. Diab, and P. Rosso. Arabic Named Entity Recognition: A Feature-driven Study. *IEEE Transactions on Audio, Speech and Language Processing. Special Issue on Processing Morphologically Rich Languages (in press)*, July, 2009.

[12] Y. Benajiba, M. Diab, and P. Rosso. Using Language Independent and Language Specific Features to Enhance Arabic Named Entity Recognition. *To be published in the International Arabic Journal of Information Technology.*

[13] Y. Benajiba, M. Diab, and P. Rosso. Arabic Named Entity Recognition using Optimized Feature Sets. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 284–293, 2008.

[14] Y. Benajiba and P. Rosso. ANERsys 2.0: Conquering the NER Task for the Arabic Language by Combining the Maximum Entropy with POS-tag Information. In *Proc. of the 3rd Indian International Conference on Artificial Intelligence*, 2007.

[15] Y. Benajiba and P. Rosso. Towards a Measure for Arabic Corpora Quality. In *Proc. of CITALA-2007*, 2007.

[16] Y. Benajiba, P. Rosso, and J.M. Gómez. Adapting the JIRS Passage Retrieval System to the Arabic Language. In *CICLing 2007 Conference*, volume 4394 of *Lecture Notes in Computer Science*, pages 530–541. Springer-Verlag, 2007.

[17] Y. Benajiba, P. Rosso, and A. Lyhyaoui. Implementation of the ArabiQA Question Answering System's Components. In *Proc. ofWorkshop on Arabic Natural Language Processing, 2nd Information Communication Technologies Int. Symposium, ICTIS'07*, pages 3–5, 2007.

[18] Y. Benajiba, P. Rosso, and J.M. Benedí Ruiz. ANERsys: An Arabic Named Entity Recognition System Based on Maximum Entropy. In *CICLing 2007 Conference*, volume 4394 of *Lecture Notes in Computer Science*, pages 143–153. Springer-Verlag, 2007.

[19] D. Buscaldi, J.M. Gómez, P. Rosso, and E. Sanchis. The UPV at QA@CLEF 2006. In *Working Notes for the CLEF 2006 Workshop*, 2006.

[20] D. Buscaldi and P. Rosso. UPV-WSD: Combining Different WSD Methods by Means of Fuzzy Borda Voting. In *Fourth International Workshop on Semantic Evaluations (Semeval-2007)*, pages 434–437, 2007.

[21] J. Carbonell, D. Harman, E. Hovy, S. Maiorano, J. Prange, and K. Sprack-Jones. Vision Statement to Guide Research in Question Answering (QA) and Text Summarization. Technical report, NIST, 2000.

[22] X. Carreras, L. Marquez, and L. Padro. Named Entity Extraction Using AdaBoost. In *Proc. of CoNLL 2002*, 2002.

[23] C. Chelba and P. Xu. Richer syntactic dependencies for structured language modeling. In *In Proceedings of the Automatic Speech Recognition and Understanding Workshop*, 2001.

[24] A. Chen and F.C. Gey. Building an Arabic Stemmer for Information Retrieval. In *Proc. of the TREC 2002*, page 631, 2002.

[25] R.J. Cooper and S.M. Ruger. A Simple Question Answering System. In *Proc. of the TREC 2000*, 2000.

[26] C. Cortes and V. Vapnik. Support Vector Networks. *Machine Learning*, 1995.

[27] S. Cucerzan and D. Yarowsky. Language Independent NER using a Unified Model of Internal and Contextual Evidence. In *Proc. of CoNLL 2002*, 2002.

[28] H. T. Dang, D. Kelly, and J. Lin. Overview of the TREC 2007 Question Answering Track. In *TREC 2007*, 2007.

[29] E. Daya, D. Roth, and S. Wintner. *Learning to Identify Semitic Roots*, chapter 8. Springer, 2007.

[30] M. Diab, K. Hacioglu, and D. Jurafsky. Automatic Tagging of Arabic Text: From raw text to Base Phrase Chunks. In *Proc. of HLT-NAACL-2004*, , 2004.

[31] M. Diab, K. Hacioglu, and D. Jurafsky. *Arabic Computational Morphology: Knowledge-based and Empirical Methods*, chapter 9. Springer, 2007.

[32] O. Douglas and F. Gey. The TREC 2002 Arabic/English CLIR Track. In *Proc. of the TREC 2002*, 2002.

[33] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, 1973.

[34] A. Emami, P. Xu, and F. Jelinek. Using a connectionist model in asyntactical based language model. In *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 372–375. Hong Kong, 2003.

[35] B. Farber, D. Freitag, N. Habash, and O. Rambow. Improving ner in arabic using a morphological tagger. In European Language Resources Association (ELRA), editor, *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, 2008.

[36] S. Ferrández, O. Ferrández, A. Ferrández, and R. Mu noz. The Importance of Named Entities in Cross-Lingual Question Answering. In *Proc. of Recent Advances in Natural Language Processing, RANLP-2007*, Borovets, Bulgaria, 2007.

[37] D. Ferrés and H. Rodríguez. TALP at GeoCLEF-2006: Experiments Using JIRS and Lucene with the ADL Feature Type Thesaurus. In *Working Notes for the CLEF 2006 Workshop*, 2006.

[38] R. Florian. Named Entity Recognition as a House of Cards: Classifier Stacking. In *Proc. of CoNLL 2002*, 2002.

[39] R. Florian, A. Ittycheriah, H. Jing, and T. Zhang. Named Entity Recognition through Classifier Combination. In *Proc. of CoNLL 2003*, 2003.

[40] D. Freitag. Trained Named Entity Recognition Using Distributional Clusters. In *Proc. of Empirical Methods in Natural Language Processing*, 2004.

[41] Y. Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2):256–285, 1995.

[42] J.M. Gómez, D. Buscaldi, E. Bisbal-Asensi, P. Rosso, and E. Sanchis. QUASAR, The Question Answering System of the Universidad Politecnica de Valencia. In *CLEF 2005, Accessing Multilingual Information Repositories*, volume 4022 of *Lecture Notes in Computer Science*, pages 439–448, Vienna, Austria, 2005. Springer-Verlag.

[43] J.M. Gómez, M. Montes y Gómez, E. Sanchis, and P. Rosso. A Passage Retrieval System for Multilingual Question Answering. In *8th International Conference of Text, Speech and Dialogue 2005 (TSD'05)*, volume 3658 of *Lecture Notes in Artificial Intelligence*, pages 443–450, Karlovy Vary, Czech Republic, 2005. Springer-Verlag.

[44] M. Greenwood and R. Gaizauskas. Using a Named Entity Tagger to Generalise Surface Matching Text Patterns for Question Answering. In *Proceedings of the Workshop on Natural Language Processing for Question Answering (EACL03)*, pages 29–34, 2007.

[45] R. Grishman and B. Sundheim. Design of the MUC-6 Evaluation. In *Proc. of the 6th Conference on Message Understanding*, pages 1–11, 1995.

[46] N. Habash and O. Rambow. Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop. In *Proc. of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 573–580, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

[47] N. Habash and F. Sadat. Arabic Preprocessing Schemes for Statistical Machine Translation. In *Proc. of HLT-NAACL-2006*, Brooklyn, New York, 2006.

[48] L.C. Hai and T.N. Hwee. Named Entity Recognition with a Maximum Entropy Approach. In *Proc. of CoNLL-2003*, 2003.

[49] B. Hammou, H. Abu-salem, S. Lytinen, and M. Evens. QARAB: A question answering system to support the Arabic language. In *Proc. of the workshop on computational approaches to Semitic languages, ACL*, 2002.

[50] S. Harabagiu, D. Moldovan, C. Clark, M. Bowden, A. Hickl, and P. Wang. Employing Two Question Answering Systems in TREC-2005. In *Proc. of the TREC 2005*, 2005.

[51] K. S. Hasan, M. A. ur Rahman, and Vincent Ng. Learning-Based Named Entity Recognition for Morphologically-Rich, Resource-Scarce Languages. In *12th Conference of the European Chapter of the ACL*, pages 354–362. Association of Computational Linguistics, 2009.

[52] A. Hickl, K. Roberts, B. Rink, J. Bensley, T. Jungen, Y. Shi, and J. Williams. Question Answering with LCCs CHAUCER-2 at TREC 2007. In *TREC 2007*, 2007.

[53] H. Isozaki and H. Kazawa. Efficient support vector classifiers for named entity recognition. In *Proceedings of the 19th international conference on Computational linguistics*, pages 1–7, 2002.

[54] E. T. Jaynes. Information Theory and Statistical Mechanics. *The Physical Review*, 1957.

[55] F. Jensen. *An Introduction to Bayesian Networks*. Springer, 1996.

[56] Z. Junsheng, H. Liang, D. Xinyu, and C. Jiajun. Chinese named entity recognition with a multi-phase model. In *Proc. of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 213–216, Sydney, Australia, 2006. Association for Computational Linguistics.

[57] D. Klein, J. Smarr, H. Nguyen, and C. Manning. Named Entity Recognition with Character-Level Models. In *Proc. of CoNLL-2003*, 2003.

[58] R. Koeling. Chunking with maximum entropy models. In *Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning*, pages 139–141, 2000.

[59] Z. Kozareva. *Resolving Named Entity Problems: from Recognition and Discrimination to Semantic Class Learning*. Dissertation, Universidad de Alicante, Spain, Nov 2008.

[60] M. K. Kozlov, S. P. Tarasov, and L. G. Khachian. Polynomial Solvability of Convex Quadratic Programming. *Dokl. Akad. Nauk SSSR*, 1979.

[61] T. Kristjansson, A. Culotta, P. Viola, and A. McCallum. Interactive Information Extraction with Constrained Conditional Random Fields. In *Proc. of the Nineteenth National Conference on Artificial Intelligence (AAAI-2004)*, 2004.

[62] G.R. Krupka. SRA: Description of the SRA System as Used for MUC-6. In *Proc. of the 6th Conference on Message Understanding*, pages 221–236, 1995.

[63] T. Kudo and Y. Matsumoto. Chunking with support vector machines. In *NAACL '01: Second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies 2001*, pages 1–8. Association for Computational Linguistics, 2001.

[64] S. Kullback and R.A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[65] J. Lafferty, A. McCallum, and F. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *In Proc. of the 18th International Conf. on Machine Learning*, Williamstown, MA, USA, 2001.

[66] L. S. Larkey, L. Ballesteros, and E. Connell. *Light Stemming for Arabic Information Retrieval*, chapter 12. Springer, 2007.

[67] S.L. Larkey, J. Allan, E.C. Margaret, A. Bolivar, and C. Wade. UMass at TREC 2002: Cross Language and Novelty Tracks. In *Proc. of the TREC 2002*, 2002.

[68] D. Laurent, P. Séguéla, and S. Négre. Cross Lingual Question Answering using QRISTAL for CLEF 2007. In *Working Notes of CLEF 2007*, 2007.

[69] S.L. Lauritzen and D.J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society, Series B*, 50:157–224, 1988.

[70] W. Li and A. McCallum. Rapid development of hindi named entity recognition using conditional random fields and feature induction. *Special Issue of ACM Transactions on Asian Language Information Processing: Rapid Development of Language Capabilities: The Surprise Languages*, 2003.

[71] JDC. MacKay. *Bayesian Interpolation*. MIT Press, 1992.

[72] P. Makagonov and M. Alexandrov. Some Statistical Characteristics for Formal Evaluation of the Quality of Text books and Manuals. In *Computing Research: Selected papers.*, pages 99–103, 1999.

[73] J. Maloney and M. Niv. TAGARAB, A Fast, Accurate Arabic Name Recognizer Using High-Precision Morphological Analysis. In *Proc. of the Workshop on Computational Approaches to Semitic Languages*, 1998.

[74] R. Malouf. Markov Models for Language-Independent Named Entity Recognition. In *Proc. of CoNLL-2003*, 2003.

[75] G.S. Mann. A Statistical Method for Short Answer Extraction. In *Proc. of the ACL-2001 Workshop on Open-domain Question Answering*, 2001.

[76] D.C. Manning and H. Schütze. *Foundations of statistical natural language processing*. MIT Press, 1999.

[77] A. McCallum and W. Li. Early Results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-enhanced Lexicons. In *In Proc. of the Seventh Conference on Natural Language Learning (CoNLL-2003)*, Edmonton, Canada, 2003.

[78] F.A. Mohammed, K. Nasser, and H.M. Harb. A knowledge based arabic question answering system (aqas). *ACM SIGART Bulletin*, pages 21–33, 1993.

[79] D. Moldovan, C. Clark, and M. Bowden. Lymba's PowerAnswer 4 in TREC 2007. In *TREC 2007*, 2007.

[80] D. Mollá and B. Hutchinson. Dependency-based Semantic Interpretation for Answer Extraction. In *Proc. of the Australian NLP Workshop 2002*, 2002.

[81] D. Mollá, M. van Zaanen, and D. Smith. Named Entity Recognition for Question Answering. In *Proc. of the Australasian Language Technology Workshop Sancta Sophia College*, pages 51–58, 2006.

[82] D. Nadeau and S. Sekine. A Survey of Named Entity Recognition and Classification. *Linguisticae Investigationes*, 30(7).

[83] J.M. Parea-Ortega, L.A. Ure na López, D. Buscaldi, and P. Rosso. TextMESS at GeoCLEF 2008: Result Merging with Fuzzy Borda Ranking . In *Working Notes of CLEF 2008*, Denmark, 2008.

[84] J. Pearl. Fusion, propagation, and structuring in belief networks. *Artificial Intelligence*, 29:241–288, 1986.

[85] M. Pérez-Coutio, M. Montes y Gómez, A. López-López, L. Villaseor-Pineda, and A. Pancardo-Rodríguez. A Shallow Approach for Answer Selection based on Dependency Trees and Term Density. In *Working Notes for the CLEF 2006 Workshop*, 2006.

[86] D. Pinto, A. McCallum, X. Wei, and W. B. Croft. Table extraction using conditional random fields. In *SIGIR '03: Proceedings of the 26th annual international*

ACM SIGIR conference on Research and development in informaion retrieval, pages 235–242, New York, NY, USA, 2003. ACM.

[87] X. Qiu, B. Li, C. Shen, L. Wu, X. Huang, and Y. Zhou. FDUQA on TREC2007 QA Track. In *TREC 2007*, 2007.

[88] L.A. Ramshaw and M.P. Marcus. Text Chunking using Transformation-based learning. In *Proc. of the Third ACL workshop on Very Large Corpora*, pages 82–94, 1995.

[89] A. Ratnaparkhi. A simple introduction to maximum entropy models for natural language processing. Technical report, Institute for Research in Cognitive Science, University of Pennsylvania.

[90] J.C. Reynar and A. Ratnaparkhi. A Maximum Entropy Approach to Identifying Sentence Boundaries. In *Proc. of the Fifth Conference on Applied Natural Language Processing*, pages 803–806, 1997.

[91] M.D. Riley. Some Applications of Tree-based Modelling to Speech and Language. In *Proc. of DARPA Speech and Language Workshop*, pages 339–352, 1989.

[92] I. Rish. An Empirical Study of the Naive Bayes classifier. In *Proc. of Workshop on Empirical Methods in Artificial Intelligence in IJCAI-01*, 2001.

[93] R. Rosenfeld. *Adaptative Statistical Language Modeling: A Maximum Entropy Approach*. Dissertation, Carnegie Mellon Universit, PA 15213, Sep 1994.

[94] P. Rosso, A. Lyhyaoui, J. Pe narrubia, M. Montes y Gómez, Y. Benajiba, and N. Raissouni. Arabic-English Question Answering. In *Proc. of ICTIS-2005*, 2005.

[95] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing Management*, 24(5):513–523, 1988.

[96] K. Sato and H. Saito. Extracting word sequence correspondences with support vector machines. In *COLING-2002*, 2002.

[97] B. Settles. Biomedical Named Entity Recognition Using Conditional Random Fields and Rich Feature Sets. In *Proc. of NLPBA/COLING*, 2004.

[98] F. Sha and F. Pereira. Shallow Parsing with Conditional Random Fields. In *Proc. of Human Language Technology*, 2003.

[99] R.D. Shachter. Probabilistic inference and influence diagrams. *Operations Research*, 36:589–604, 1988.

[100] C.E. Shannon. A mathematical theory of communication. *Bell Syst. Techn. J.*, 27:379–423, 1948.

[101] J.R. Shewchuk. An introduction to conjugate gradient method without the agonizing pain. 1994.

[102] P. Thompson and C. Dozier. Name Searching and Information Retrieval. In *In Proc. of Second Conference on Empirical Methods in Natural Language Processing*, Providence, Rhode Island, 1997.

[103] E.F. Tjong. Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition. In *Proc. of CoNLL 2002*, 2002.

[104] E.F. Tjong and Fien DeMeuler. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proc. of CoNLL 2003*, 2003.

[105] H. Toda and R. Kataoka. A Search Result Clustering Method using Informatively Named Entities. In *In Proc. of the 7th annual ACM international workshop on Web information and data management*, pages 81–86, 2005.

[106] S. Tomlinson. Experiments in Named Page Finding and Arabic Retrieval with Hummingbird SearchServerTM at TREC 2002. In *Proc. of the TREC 2002*, page 248, 2002.

[107] T.Q. Tran, T.X.T. Pham, H.Q. Ngo, D. Dinh, and N. Collier. Named entity recognition in vietnamese documents. *Progress in Informatics*, 4:5–13, 2007.

[108] H. Turtle and W. B. Croft. Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems*, pages 187–222, 1991.

[109] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

[110] M. Montes y Gómez, L. Villase nor Pineda, M. Pérez-Couti no, J.M. Gómez, E. Sanchis, and P. Rosso. A Full Data-Driven System for Multiple Language Question Answering. In *CLEF 2005, Accessing Multilingual Information Repositories*, volume 4022 of *Lecture Notes in Computer Science*, Vienna, Austria, 2005. Springer-Verlag.

[111] T. Zhang, F. Damerau, and D. Johnson. Text chunking based on a generalization of winnow. *Journal of Machine Learning Research*, 2:615–637, 2002.

[112] G.K. Zipf. *Human behaviour and the principle of least effort*. Addison-Wesley, 1949.

[113] I. Zitouni and R. Florian. Mention detection crossing the language barrier. In *Proceedings of EMNLP'08*, Honolulu, Hawaii, October 2008.

[114] I. Zitouni, J.S. Sorensen, X. Luo, and R. Florian. The Impact of Morphological Stemming on Arabic Mention Detection and Coreference Resolution. In *Proc. of the ACL Workshop on Computational Approaches to Semitic Languages*, pages 63–70, 2005.

Appendix A

Bayesian Networks

Bayesian Networks are graphical models which encode the probabilistic relationships between a set of variables [55]. A BN is a directed acyclic graph, Figure A.1 shows an illustrating example of a BN. If there is an arrow from a node N_1 to another

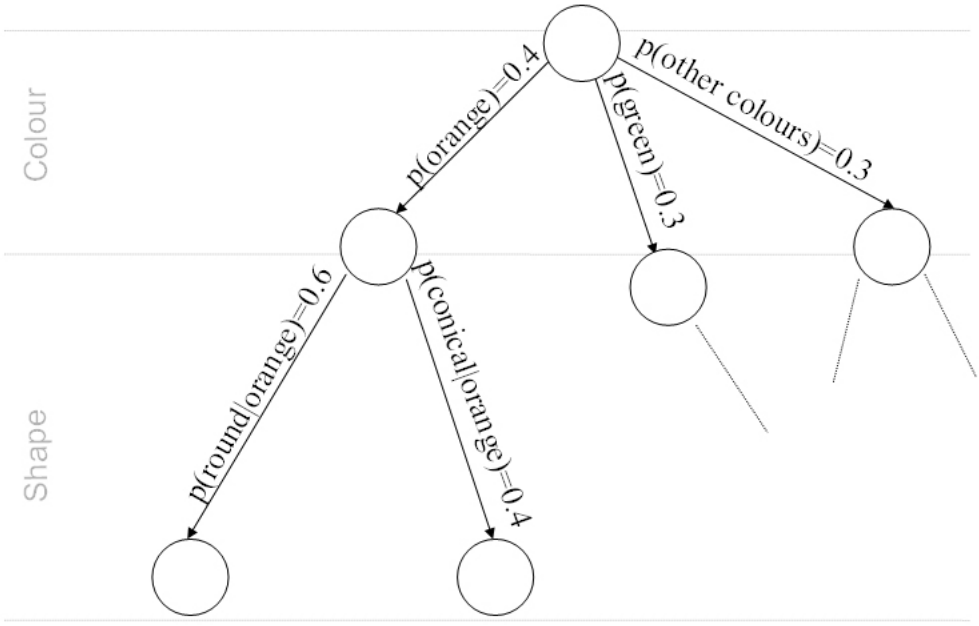


Figure A.1: Illustrating example of a Bayesian Network: the vegetables case study

node N_2, then N_1 is a parent of N_2 and N_2 is a child of N_1. For each node (also

referred to as *variable*) there is conditional probability table. This table defines a distribution over the values of the associated variable for each joint instantiation of the parents. Let us employ the same example of the exergue of this chapter. If we consider that when a vegetable has a round shape and an orange colour is surely a pumpkin then it is easy to calculate the probability of that a vegetable is a pumpkin using the BN as follows:

$$\begin{aligned} p(pumpkin) &= p(orange, round) \\ &= p(round|orange).p(orange) \\ &= 0.4\ .\ 0.6 \\ &= 0.24 \end{aligned}$$

Hence, if we consider any simple BN $a \rightarrow b \rightarrow c \rightarrow d$, we may compute $P(a|d)$ with the following formula:

$$\begin{aligned} p(a|d) &= \frac{p(a,d)}{p(d)} \\ &= \frac{\sum_{b,c} p(a,b,c,d)}{\sum_{a,b,c} p(a,b,c,d)} \end{aligned}$$

In order to make an efficient computation of this probability, it is possible to exploit one of the main characteristics of BNs, i.e., the conditional independencies and thus equation A.1 can be written as:

$$P(a|d) = \frac{p(a).\sum_b p(b|a).\sum_c p(c|b)p(d|c)}{\sum_a p(a).\sum_b p(a)p(b|a).\sum_c p(c|b)p(d|c)} \tag{A.1}$$

However, the main issue of classification is to compute the probability of an object x belonging to a class c. In order to compute this probability using BNs, there are methods of *inference* supposing that some variables are absent or unknown. In the case of classification, the unknown variables are the classes of the objects. In the literature, there are many inference methods based on different approaches [99], [84], [69].