Arabic Named Entity Recognition: A Corpus-Based Study

A THESIS SUBMITTED TO THE UNIVERSITY OF MANCHESTER FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN THE FACULTY OF ENGINEERING AND PHYSICAL SCIENCES

2011

Shabib AlGahtani

School of Computer Science
# Table of Contents

Publications based on this thesis.................................................................................................................. 5
List of Tables .................................................................................................................................................. 6
List of Figures ................................................................................................................................................ 8
List of Abbreviations .................................................................................................................................... 9
Abstract ......................................................................................................................................................... 11
Declaration .................................................................................................................................................. 12
Copyright Statement ...................................................................................................................................... 13
Acknowledgments .......................................................................................................................................... 14

## Chapter 1  Introduction ............................................................................................................................. 15
1.1  Named Entity Recognition (NER) Applications .................................................................................... 17
1.2  The challenge of Named Entity Recognition and Possible Solutions .................................................... 18
1.3  Research Aim and Objectives ................................................................................................................ 20
1.4  Thesis Structure ....................................................................................................................................... 23

## Chapter 2  Supervised learning in an NLP framework .............................................................................. 25
2.1  Most popular supervised ML algorithms in NLP .................................................................................... 28
2.2  Comparison of ML algorithms ................................................................................................................ 46

## Chapter 3  Arabic Language Characteristics ............................................................................................ 51
3.1  History and Current Perspective ............................................................................................................. 51
3.2  Orthography ........................................................................................................................................... 53
3.3  Transliteration ......................................................................................................................................... 54
3.4  Morphology ............................................................................................................................................ 55
3.5  Arabic language forms ............................................................................................................................ 60
3.6  Arabic Proper Nouns .............................................................................................................................. 61

## Chapter 4  Overview of Named Entity Recognition .................................................................................. 73
4.1  Definition ................................................................................................................................................ 73
4.2  NER annotation ..................................................................................................................................... 77
4.3  NER System Scoring ............................................................................................................................... 81
4.4  NER literature review .............................................................................................................................. 86
Chapter 5  Arabic POS tagger.........................................................102
  1.1 Arabic Segmentation and POS tagging................................................................. 102
  5.2 POS tagging literature review................................................................................. 104
  5.3 Transformation-Based Learning Revisited............................................................. 106
  5.4 Methodology........................................................................................................... 107
  5.5 Implementation & Experiments............................................................................... 109
  5.6 Results and discussion............................................................................................ 113
  5.7 Notes on the NNP Class.......................................................................................... 118
  5.8 Future Work............................................................................................................ 120

Chapter 6  The correlation between Arabic NER and POS.........................122
  6.1 Corpus..................................................................................................................... 123
  6.2 NER classifier.......................................................................................................... 128
  6.3 POS tagging effect.................................................................................................. 132
  6.4 Improving to Classification of Proper Nouns............................................................ 134
  6.5 Conclusion and Future work.................................................................................... 140

Chapter 7  Arabic Person Name Recognition.................................................141
  7.1 Current study........................................................................................................... 141
  7.2 Dataset and Task...................................................................................................... 142
  7.2.4 Task complexity.................................................................................................. 145
  7.3 Feature Set............................................................................................................... 145
  7.4 Experimental results............................................................................................... 154
  7.5 System comparison................................................................................................. 157

Chapter 8  Token Classification vs. Sequence Labelling for Arabic NER 159
  8.1 Hidden Markov Support Vector Machine................................................................. 159
  8.2 Data analysis............................................................................................................ 160
  8.3 Feature set............................................................................................................... 163
  8.4 Gazetteer Coverage................................................................................................ 165
  8.5 Experiments............................................................................................................. 166
  8.6 Cross Validation with HM-SVM.............................................................................. 171
  8.7 Comparison with previous work............................................................................. 172
Chapter 9  Conclusion ........................................................................................................175
  9.1  Thesis summary ........................................................................................................175
  9.2  Thesis Contribution ....................................................................................................178
  9.3  Limitation ..................................................................................................................179
Appendix A  English POSTagset .....................................................................................182
Appendix B  Arabic Collapsed Tagset ..............................................................................184
Bibliography ..................................................................................................................185

Word Count = 50,465
Publications based on this thesis


List of Tables

Table 2.1: Training data with features generated in tabular format ......................... 26
Table 2.2: Test data in tabular format just before processing the word race ................ 27
Table 2.3: Probability of each feature for the two classes .......................................... 30
Table 2.4: Iterative probability distribution over possible classes ............................... 32
Table 2.5: Feature weights ......................................................................................... 33
Table 2.6: Emission and transition probabilities of both analyses for race .................... 39
Table 3.1: Stem generating examples ........................................................................... 58
Table 3.2: Proclitics ...................................................................................................... 59
Table 3.3: Enclitics ....................................................................................................... 60
Table 3.4: Analysis of “Hmd”, diacritized and non-diacritized ..................................... 67
Table 4.1: MUC entity types ....................................................................................... 74
Table 4.2: ACE2005 Mention types and attributes ..................................................... 76
Table 4.3: NER annotation schemes example ............................................................. 79
Table 4.4: Key and system output of sentence example in MUC .................................. 83
Table 4.5: Result analysis ............................................................................................. 84
Table 4.6: CoNLL key and system output, errors underlined ....................................... 85
Table 4.7: MUC best three systems on the NE task ..................................................... 88
Table 4.8: Best participants in CoNLL 2002 ................................................................. 90
Table 4.9: Best participants in CoNLL 2003 ................................................................. 90
Table 5.1: Arabic segmentation and tagging analysis example ................................... 108
Table 5.2: Experimental results ................................................................................... 115
Table 5.3: Confusion matrix of largest error classes .................................................... 117
Table 5.4: System comparison on ATB 1.0 ................................................................. 118
Table 6.1: Corpus statistics ......................................................................................... 124
Table 6.2: Corpus split ................................................................................................. 129
Table 6.3: Baseline of most frequent classes ............................................................... 129
Table 6.4: Performance of the lexical feature ............................................................. 132
Table 6.5: Performance of the POS feature ............................................................... 134
Table 6.6: Performance of the POS baseline .......................................................... 134
Table 6.7: System performance with (LEX, POS, GAZ) ........................................ 136
Table 6.8: System performance with (LEX, POS, TRG) ....................................... 136
Table 6.9: System performance (LEX, POS, C_i) ................................................ 137
Table 6.10: System performance (LEX, POS, GLB) .............................................. 138
Table 6.11: All features with GAZ ........................................................................ 138
Table 6.12: All without GAZ ................................................................................ 138
Table 6.13: Overall accuracy with features added .................................................. 139
Table 7.1: Corpus information .............................................................................. 143
Table 7.2: Training and tests sets: details and person count per part ..................... 144
Table 7.3: Most frequent person tokens not found in the gazetteer ....................... 147
Table 7.4: Gazetteer and POS tagger accuracy ...................................................... 150
Table 7.5: Sample of the most frequent tokens in gazetteer that are not person entities... 151
Table 7.6: Most frequent non-person bigrams in the gazetteer ............................ 152
Table 7.7: Performance of baseline and different feature combinations ............... 157
Table 8.1: Corpus NE class distribution ................................................................. 161
Table 8.2: Baseline of most frequent classes .......................................................... 167
Table 8.3: Effects of combining features using different approaches .................... 168
Table 8.4: Data mapping ....................................................................................... 169
Table 8.5: Comparison of the effect of using different features (MEM vs. HM-SVM).... 170
Table 8.6: 6-fold Cross Validation ........................................................................ 171
List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Start-of-a-sentence ambiguity example</td>
<td>19</td>
</tr>
<tr>
<td>2.1</td>
<td>NB of the word race</td>
<td>29</td>
</tr>
<tr>
<td>2.2</td>
<td>One POS tagging example tree</td>
<td>35</td>
</tr>
<tr>
<td>2.3</td>
<td>Statistical version of POS tagging tree</td>
<td>36</td>
</tr>
<tr>
<td>2.4</td>
<td>HMM chain</td>
<td>39</td>
</tr>
<tr>
<td>2.5</td>
<td>ME-HMM chain</td>
<td>40</td>
</tr>
<tr>
<td>2.6</td>
<td>CRF chain</td>
<td>41</td>
</tr>
<tr>
<td>2.7</td>
<td>Separating hyperplanes</td>
<td>43</td>
</tr>
<tr>
<td>2.8</td>
<td>Separating hyperplanes</td>
<td>44</td>
</tr>
<tr>
<td>3.1</td>
<td>Buckwalter transliteration scheme</td>
<td>55</td>
</tr>
<tr>
<td>4.1</td>
<td>ACE annotation example</td>
<td>81</td>
</tr>
<tr>
<td>5.1</td>
<td>Rule templates (Brill 1995)</td>
<td>107</td>
</tr>
<tr>
<td>5.2</td>
<td>Joint tagging and segmenting algorithm</td>
<td>111</td>
</tr>
<tr>
<td>5.3</td>
<td>Tagging algorithm</td>
<td>112</td>
</tr>
<tr>
<td>5.4</td>
<td>Sample rules</td>
<td>114</td>
</tr>
<tr>
<td>5.5</td>
<td>Error distribution (top 8)</td>
<td>116</td>
</tr>
<tr>
<td>6.1</td>
<td>Internal POS of NEs, excluding OTHER, at the token level</td>
<td>125</td>
</tr>
<tr>
<td>6.2</td>
<td>POS tag distribution of NE classes</td>
<td>126</td>
</tr>
<tr>
<td>6.3</td>
<td>NE context (previous POS tag)</td>
<td>127</td>
</tr>
<tr>
<td>6.4</td>
<td>Preceding POS tag</td>
<td>128</td>
</tr>
<tr>
<td>7.1</td>
<td>POS tags of PER class tokens</td>
<td>149</td>
</tr>
<tr>
<td>7.2</td>
<td>Gazetteer and Lexicon</td>
<td>153</td>
</tr>
<tr>
<td>8.1</td>
<td>POS class distribution of corpus NE classes excluding O class</td>
<td>162</td>
</tr>
<tr>
<td>8.2</td>
<td>POS tag distribution of each NE class excluding O class</td>
<td>163</td>
</tr>
<tr>
<td>8.3</td>
<td>Comparison of the four systems without the use of unlabelled data</td>
<td>173</td>
</tr>
</tbody>
</table>
List of Abbreviations

ACE Automatic Content Extraction
AdaBoost Adaptive Boosting
ANERcorp Arabic Named Entity Recognition corpus developed by Benajiba
ATB Arabic Tree Bank
BN Broadcast News
BN Bayesian Networks
BPC Base Phrase Chunking
CLEF Cross Language Evaluation Forum
CoNLL Conference on Computational Natural Language Learning
CRF Conditional Random Fields
DARPA Defense Advanced Research Projects Agency
DT Decision Trees
EDR Entity Detection and Recognition
EDT Entity Detection and Tracking
EM Expectation-Maximization
FAC Facility
FB Forward-Backward
GIS Generalized Iterative Scaling
GPE Geo-Political Entity
HMM Hidden Markov Models
HM-SVM Hidden Markov Support Vector Machines
IE Information Extraction
IR Information Retrieval
LDC Language Data Consortium
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOC</td>
<td>Location</td>
</tr>
<tr>
<td>MD</td>
<td>Mention Detection</td>
</tr>
<tr>
<td>MEM</td>
<td>Maximum Entropy Modeling</td>
</tr>
<tr>
<td>ME-HMM</td>
<td>Maximum Entropy Hidden Markov Models</td>
</tr>
<tr>
<td>MISC</td>
<td>Miscellaneous</td>
</tr>
<tr>
<td>ML</td>
<td>Machine Learning</td>
</tr>
<tr>
<td>MSA</td>
<td>Modern Standard Arabic</td>
</tr>
<tr>
<td>MUC</td>
<td>Message Understanding Conference</td>
</tr>
<tr>
<td>NaCTeM</td>
<td>National Centre of Text Mining</td>
</tr>
<tr>
<td>NE</td>
<td>Named Entity</td>
</tr>
<tr>
<td>NER</td>
<td>Named Entity Recognition</td>
</tr>
<tr>
<td>NLP</td>
<td>Natural Language Processing</td>
</tr>
<tr>
<td>NW</td>
<td>News Wire</td>
</tr>
<tr>
<td>OOV</td>
<td>Out-of-Vocabulary</td>
</tr>
<tr>
<td>ORG</td>
<td>Organisation</td>
</tr>
<tr>
<td>PER</td>
<td>Person</td>
</tr>
<tr>
<td>POS</td>
<td>Part-Of-Speech</td>
</tr>
<tr>
<td>RRM</td>
<td>Robust Risk Minimization</td>
</tr>
<tr>
<td>SVM</td>
<td>Support Vector Machines</td>
</tr>
<tr>
<td>TBL</td>
<td>Transformation-Based Learning</td>
</tr>
<tr>
<td>TREC</td>
<td>Text REtrieval Conferences</td>
</tr>
<tr>
<td>VEH</td>
<td>Vehicle</td>
</tr>
<tr>
<td>WEA</td>
<td>Weapon</td>
</tr>
<tr>
<td>WL</td>
<td>WebLogs</td>
</tr>
<tr>
<td>WSJ</td>
<td>Wall Street Journal</td>
</tr>
</tbody>
</table>
Abstract

The task of finding and classifying proper nouns in natural language text is the core of most Named Entity Recognition (NER) systems. The NER problem has received much attention, as NER forms the basic building block of any Information Extraction system.

Although finding and classifying proper nouns in text is a very challenging task in English, the task benefits a great deal from the distinguishing orthographic feature of capitalization. When this feature is missing, as in uppercase text, or is present at the start of a sentence, ambiguity increases, and requires more knowledge sources to resolve the problem.

The lack of capitalization is, however, an intrinsic feature of Arabic, thus the NER task in Arabic becomes immediately harder than in English. The ambiguity caused by this feature is moreover increased, as most Arabic proper nouns are indistinguishable from forms that are common nouns and adjectives. Thus, a lookup approach relying on proper noun dictionaries would not be an appropriate way to tackle the problem, as ambiguous tokens that fall in this category are more likely to be used as non-proper nouns in text. In addition, Arabic is a highly morphological language, thus posing more challenges for the NER task.

We hypothesize that Arabic NER is very closely bound to Part-of-Speech (POS) tagging. However, Arabic POS taggers would normally have their worst accuracy on proper noun tagging, especially person names, given the problem just mentioned. Thus, we first built a POS tagging tool with a good coverage using the corpus-based approach. Then, we used a filtering technique to help collect unique proper nouns from large gazetteers. Combined with the POS, gazetteer, and unique names list features, we defined and used a further set of features to build a corpus-based NER classifier from labelled data. Experiments on different datasets, against a baseline and incorporating different combinations of features, resulted in demonstrating the efficiency of our final set of proposed features. The unique names list moreover assisted in reducing, in particular, the POS feature’s noise on proper nouns. Evaluation of our approach shows that it performs comparably with systems that use more, and more sophisticated, knowledge sources, and hence is easier to deploy for practical use.
Declaration

I hereby declare that no portion of the work referred to in the thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.
Copyright Statement

i. The author of this thesis (including any appendices and/or schedules to this thesis) owns any copyright in it (the “Copyright”) and he has given The University of Manchester the right to use such Copyright for any administrative, promotional, educational and/or teaching purposes.

ii. Copies of this thesis, either in full or in extracts, may be made only in accordance with the regulations of the John Rylands University Library of Manchester. Details of these regulations may be obtained from the Librarian. This page must form part of any such copies made.

iii. The ownership of any patents, designs, trade marks and any and all other intellectual property rights except for the Copyright (the “Intellectual Property Rights”) and any reproductions of copyright works, for example graphs and tables (“Reproductions”), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property Rights and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property Rights and/or Reproductions.

iv. Further information on the conditions under which disclosure, publication and exploitation of this thesis, the Copyright and any Intellectual Property Rights and/or Reproductions described in it may take place is available from the Head of School of Computer Science (or the Vice-President).
Acknowledgments

When I look at the amount of knowledge I have gained throughout my PhD journey, I realize how much time and effort spent toward the completion of this work; not only by me but also by key individuals whom I feel very indebted to.

I gratefully acknowledge the support and guidance of my supervisor John McNaught. I would not imagine completing this work without all his advices. I acknowledge the time he spent in our weekly meetings, even with his busy schedule and other commitments. I am deeply indebted to him for his patience. He assisted in all aspects of this work from discussing new ideas to writing and completing this thesis. Though, mistakes in the content of this thesis are entirely my own.

I would like to express my deep gratitude and appreciation to my first year supervisor, William Black, who was very cooperative and kind in getting me on the road when I started my PhD. Thanks are also extended to Mr. Roger Garside for his valuable comments on the writing of this thesis.

I am very grateful to my home country, Saudi Arabia, for the continuous support that I have been receiving in my whole life. I would like also to thank Technology Control Company (TCC) for hosting me in the three months internship program that gave me a very strong hand-on experience in real world problems of text analytics. Also, I would like to thank the Language Data Consortium (LDC) for providing student discount on the corpora used in this thesis.

My thanks and appreciations are also extended to all my friends in the UK and my brother Ibrahim, who were always there when I needed them.

Last but not least, my deepest thanks to my family for their emotional support and encouragement.
Chapter 1   Introduction

At the present time, we are witnessing great and increasing capabilities in storing and transferring greater volumes of information, which is an excellent factor in information technology from one perspective. From another perspective, we are overloaded with data that we cannot manage. More precisely, the rate of the growth of the capacity to store information is far larger than the rate of the development of data analysis tools. Nobody exactly knows how much information there is in our information flooded world; or how one could uniformly measure information flows from heterogeneous sources. However, Lyman & Varian (Lyman and Varian 2003) estimate that the total amount of newly created information on physical media (print, film, optical and magnetic storage) amounted to some 5 exabytes in 2002, most of it is stored in digital format. This corresponds to 9,500 billion books or 500,000 times the entire Library of Congress (which is supposed to contain approximately 10 terabytes of information). According to their measures, the surface Web contains around 167 terabytes of information, and there are indications that the deep Web, i.e., information stored in databases that is accessible to human users through query interfaces but largely inaccessible to automatic indexing, is about 400 to 500 times larger.

In terms of web pages count, WorldWideWebSize \(^1\) estimated that the indexes of the major search engines (GOOGLE, YAHOO and BING) contain 48 billion pages as of November 2011, double the estimated number of pages in the same month of last year.

The information available in the world, according to (Lyman and Varian 2003), is estimated to amount to be at least 66,800 terabytes of data. A large fraction of this information is unstructured in the form of text, images, video and audio (Moens 2006). Web content, as an important source of information, is doubling every 15 months, with 80% of the web content stored in natural language form.

\(^1\) [http://www.worldwidewebsize.com/](http://www.worldwidewebsize.com/)
Information Extraction (IE) is a subfield of Natural Language Processing (NLP) that was introduced as a solution to the information overload problem in the last decade and was defined as follows:

"Information extraction is the identification, and consequent or concurrent classification and structuring into semantic classes, of specific information found in unstructured data sources, such as natural language text, making the information more suitable for information processing tasks". (Moens 2006)

Information extraction therefore involves the creation of a structured representation (such as a database) of selected information drawn from the text. Structuring the unstructured text gives us the ability to apply techniques from data mining to find hidden patterns in the text or conduct further analysis (Grishman 1997).

Because of the importance of IE, DARPA initiated a number of efforts in The Massage Understanding Conferences (MUC) in the mid nineties. It was responsible for defining IE subtasks and running competition-based conferences to advance the field. The first and most important subtask of IE is Named Entity Recognition (NER), which is concerned with finding and classifying named entities in the text, such as persons, locations and organizations ... etc. The priority of this task comes from the fact that text always revolves around such entities. A subsequent task of IE is resolving the anaphoric references, which is concerned with finding units that refer to the same entity in the text, e.g. nouns and the pronouns referring to them. After that, there is the relation extraction subtask which aims at finding relations between extracted entities in text.

NLP in general is not a trivial task, due to language ambiguity that is an intrinsic characteristic of any natural language, occurring at all levels of representation. Humans can easily resolve this ambiguity by analyzing the context in which a particular string of words is used. However, it is very difficult to equip a machine with human-like knowledge to resolve this ambiguity. Resolving ambiguity in natural language has been of central interest to researchers and practitioners in the field from the early 1950s.
Chapter 1: Introduction

1.1 Named Entity Recognition (NER) Applications

According to Rau (Rau 1991), proper nouns are a crucial source of information in a text for extracting contents, identifying a topic in a text, or detecting relevant documents in information retrieval systems. Yet they account for a large percentage of the unknown words in a text. NER is an important task in itself; however it would sometimes not be very valuable until it is followed by the subsequent subtasks of an IE system. IE in general and NER in particular have proven successful in many NLP applications.

NER was successfully incorporated into name searching systems to identify names in queries and the underlying data source for information retrieval (Thompson and Dozier 1997). Another important application is question answering, which was driven by one of Text REtrival Conference’s tracks. NER is a core component of these systems and the effect of NER was measured in (Noguera et al. 2005). The idea was to consider only documents that have at least one named entity provided in the query. It was found that NER could reduce the data returned by the IR system by 62% without any loss of information and by 92% with acceptable loss.

Detecting NEs in a query was also successful in improving ranking, by treating named entities and context separately in query suggestions (Guo et al. 2009). In addition, NER was used to efficiently provide the user with fewer returned documents, by a factor of 2, when indexing named entities (Mihalcea and Moldovan 2001). Ontologies were used to improve IE systems and then IE systems were used to populate ontologies in a cyclic form (Nédellec and Nazarenko 2006). In the wider context, IE is used to populate the semantic web, by building a huge knowledge base (Welty and Murdock 2006).

Most NEs are not supposed to be translated by Machine Translation systems thus; detecting them would improve their performance. In document summarization, NEs were used to improve the identification of important text segments (Hassel 2003). Also, extracted entities could be exploited for better and more efficient visualization instead of raw text (Taylor 2003).

NER was also used in Story Link Detection (SLD), which is the core task of Topic Detection and Tracking (TDT) tasks. It was proven that indexing NEs is better than using a
word-based technique (C. Shah et al. 2006). In entity profiling, to find out information related to a certain entity, it is common for NEs to be written differently using various aliases. In order to link all these aliases, one needs to detect each name instance before having the chance to link them altogether.

1.2 The challenge of Named Entity Recognition and Possible Solutions

The ambiguity of NEs in text resides at two different levels; detection and classification. The first involves disambiguating NEs from non NEs in text and the second involves classifying them into classes e.g. person or location. The two processes of NER are sequential and the correctness of the identification phase is a prerequisite for the classification phase. The second level is dependent on the first one; NEs will not be classified correctly if they are not detected correctly in the first place.

The level of ambiguity that resides at these two levels differs from one language or domain to another. In English text, for instance, the identification would normally rely on orthographic case information (lower and upper) to detect NEs. If we manage to identify all NEs using that information then the fundamental problem is classifying them. For the classification, assuming that we managed to collect and group all NEs into lists for each NE class, there still would be cases where one token might fall into more than one list. For instance, the word *Brown* could be a person name, US city or US company. Thus, more information is required to classify NEs correctly; for example, the context in which the word occurs, such as the previous word. Thus, if the word *Brown* is preceded by *Mr.* then it is more likely to be a person name. Titles and designators have proved successful in NER and are called triggering words or triggers.

In languages that use upper and lower case, it is easier to detect NEs; however it is not always straightforward. In the previous brief discussion, we were considering that case information is sufficient for the identification phase. However, there are situations when the case information does not exist or is not strong enough i.e. at the start of a sentence. In such situations, more ambiguity would be added to the identification phase. For instance, *Brown* without case information, *brown*, would be a very ambiguous word as it could be a proper
noun (NNP), noun (NN), verb (VB) or adjective (JJ); these categories are called part of speech tags, see Figure 1.1. At the first level, we need to disambiguate Brown’s grammatical category. If it is classified as proper noun, then we still need to find its correct named entity class as it could be a person name (PER), location name (LOC) or organization name (ORG). Therefore, we can see that greater ambiguity requires more information to complete the disambiguation.

To be able to identify NEs successfully in such a situation, there would be a need to analyze the context, given that each word category would have a different context. For instance, an adjective typically cannot follow a verb without a determiner and also noun typically cannot be followed by an adjective in English. Thus, analyzing the surrounding lexical and POS context is crucial. If we have the means to generate the POS information correctly, then the problem would be solved to a large extent. However, it is not feasible as POS taggers rely on case information to tag proper nouns.

---

Figure 1.1: Start-of-a-sentence ambiguity example

---

Chapter 1: Introduction

The phenomenon of the lack of case information is not limited to the start of sentence in English but actually it has a wide spread in other languages and domains. The web is hosting more and more informal text that does not conform to the formal orthographic features, for example forums and social networks. Furthermore, languages not using the Latin alphabet may not distinguish NEs according to case. Our study is concerned with one language in this category, namely the Arabic Language.

Now that we have demonstrated the NER problem, one of the biggest challenges in natural language processing is how to provide a computer with the linguistic sophistication necessary for it to successfully perform language-based tasks (Brill and Mooney 1997). There have been three main approaches to tackle NLP problems. In our Brown example, we used a condition in the form of a rule of grammar to detect the token based on context; this is known as the rule-based approach. Another approach is to use statistics drawn from a large amount of labelled (annotated) data, called a corpus, where each token is labelled with its corresponding class. This rational or empirical method is mainly carried out by applying a machine learning algorithm to build a classifier from the data. Building the classifier is guided by any cue (feature) that might assist in finding the correct classification. This technique has received considerable attention in the last two decades. Corpora are increasingly recognized as an important resource for natural language processing. Statistical analysis of corpora has proved to be extremely useful in identifying the properties of texts under analysis. This approach falls within corpus-based methods and will be further discussed in chapter 2. The third approach is to combine both of the previously mentioned approaches, leading to the best results in many NLP applications.

1.3 Research Aim and Objectives

The Arabic language is one of the Semitic languages. It is the mother tongue of 317 million Arabs and the religious language of more than 1 billion Muslims. It has a number of characteristics which increase the ambiguity of its syntactic and semantic representation and hence increase the level of complexity in analyzing it. Each language has its own features that require special consideration when processing it for any NLP task, and there has not yet
been sufficient work on information extraction from Arabic. The success of previous research efforts in other languages encourages us to follow some of approaches exploited. Nowadays, with the availability of large Arabic corpora, it is possible to use corpus-based methods that have been successfully applied to other languages to resolve ambiguity. Even when there is a lack of available resources, it is now feasible to create them using tools that were produced for other well-studied languages.

The main difference in NER between English and Arabic is where the difficulty arises. In the English language the difficulty occurs in the classification phase, while in contrast it occurs in the earlier recognition phase in Arabic. This is because English rely mostly on capitalization in the detection phase as a strong indication of NEs and that feature is the main feature disambiguating NEs from other tokens.

This research aims at resolving Arabic language text ambiguity at the NER level. The complexity of this task arises from the of lack distinguishing orthographic features, which makes it a very difficult task, as discussed in previous section. In addition, Arabic NEs are mostly in the form of general purpose words; nouns or adjectives. That means that the same token could serve as an NE or a non-NE based on the context, which makes Arabic NER closely bound to POS tagging. In any language, the sequence of tags is governed by some linguistic constraints (for example a verb cannot be followed by another verb). Thus, NEs in the form of a sequence of verbs would violate that constraint. By that assumption, person names in Arabic that serve as general purpose words would be disambiguated using the POS tagging information of the context. Another feature that affects our task is the attachment of clitics such as conjunctions to Arabic words; this requires segmenting words before NER processing.

Specifically, this work will focus on the three main NE classes; person, organization and location. The main features that we plan to use are POS tagging information and NE lists. Our work requires a high accuracy part-of-speech tagger with wide coverage. The tagger would also be responsible for segmenting each token before generating the POS tag. The main hypothesis of this work is that employing part of speech tagging information and NE dictionaries could assist in substituting for the lack of capitalization in Arabic and hence compensate for the standing challenges.
To achieve our aim, we will apply a corpus-based method to a manually created corpus in addition to taking advantage of available corpora developed for the Arabic language. The available POS taggers implemented for Arabic were trained on a very limited amount of data. The amount of training data is considered to be low compared with the English taggers. Another reason for adopting our approach is that the amount of annotated text is currently four times larger than what was available at the time of implementing the other taggers. Also, the available annotated data have gone through a number of revisions and modifications to the tagset in later releases. By using a large amount of training data, the heuristic of tagging OOV as NNP would be improved. For these reasons, we decided to implement a new POS tagger, which we believe will be more suitable for our task. Also, building a lexicon from large amount of training data would be of great help in finding the unambiguous NE tokens from name lists as will be discussed later.

This research starts with the implementation of a POS tagger that considers the free order characteristics of Arabic and employs a new technique to segment (tokenize) Arabic words. We hypothesise that a technique such as Transformation-Based Learning (TBL), described in the following chapter, that considers context on both sides of the word, would perform better than other techniques that consider only the previous context, as commonly used in implementing most taggers. The TBL technique, to our knowledge, has never been investigated on standard datasets.

After generating the POS tagging information, we will use a number of machine learning techniques to build an NER classifier investigating various features (knowledge sources). These techniques rely heavily on the features that guide the learning process; correlation and quality. We try to find the optimal feature set empirically, in addition to the attempting to reduce the effect of noise caused by features like incorrect POS tagging. Our first algorithm is a token classification technique based on Maximum Entropy Modelling (MEM), described in chapter 2, which has been successfully applied to NER in other languages. It has the advantage of combining fast training and good performance. Then we approach the problem by applying a sequence labelling technique using a state of the art algorithm designed specially for NLP tasks; Hidden Markov Support Vector Machine (HM-SVM). We were encouraged by its recent success in similar problems. Our system will be tested on
different data sets to evaluate its sensitivity to training data and also for benchmarking purposes with previous work on Arabic.

1.4 Thesis Structure

After the brief introduction and the objectives of this research, we explain the structure of this thesis and briefly describe the content of each chapter.

**Chapter 2** sheds some light on statistical NLP as one effective corpus-based technique covering, and covers most widely used machine learning algorithms.

**Chapter 3** demonstrates the basic features of the Arabic language and challenges of morphology, especially those that directly affect NER. Also, we give a description of Arabic NEs and how they are formed.

**Chapter 4** gives an overview of the NER task, covering previous work and highlighting the main approaches and significant achievements. In addition, it includes a brief literature review of POS tagging in the Arabic language.

**Chapter 5** highlights the main steps conducted in this research toward implementing a POS tagger and a segmentation algorithm. It includes a comprehensive analysis of the corpus and experimental results.

**Chapter 6** starts with a corpus study to measure the relation between POS tagging and NER in the Arabic language using a manually created corpus. Then, it shows the detailed steps of building an NER token classifier using Maximum Entropy Modelling to measure the POS tagging effect on NER.

**Chapter 7** covers the extensions to the previous system and experiments performed. We describe the system built on an official data set developed by professional annotators incorporating more external resources. We also discuss a successful approach of integrating the POS tagger into the model in such a way as to reduce
noisy (inaccurate) tags generated by our tagger. This is followed with achieved improvements and discussion of the result.

**Chapter 8** provides a discussion of our NER experiment using a machine learning algorithm designed for sequence labelling, which yielded promising results in previous studies. In this experiment, we investigate the NER problem as a sequence classification problem and compare it with the token classification technique. The experiment is performed on a freely available dataset that was specifically annotated for NER. This chapter also gives a cross validation evaluation results of our approach on this dataset for the sake of comparing its performance with previous studies evaluated on same set.

**Chapter 9** describes the overall achievements and highlights the main findings of this research. Finally, we give a proposal for work planned in the near future.
The common problem for most NLP analysis levels is ambiguity i.e. having two or more analyses for a single textual unit. Thus it could be viewed as classification problem, providing motivation for the use of a space for machine learning (Daelemans et al. 1997).

In machine learning, each possible analysis is an independent class \( c \) from a set of possible classes \( C \) or analyses that an input \( x \) might fall into. In supervised learning, the task of the machine learning algorithm is to build a classification model from pre-classified training examples by inferring a mapping function from input \( X \) to output \( C \), to find the probability of input \( x \) falling into class \( c \). In practice, it actually finds a probability distribution over set of classes. The algorithm tries to generalize to new instances of input \( X \) not used in training.

Since most NLP tasks exhibit a sequential nature, some of the classification algorithms are enhanced to cope with that feature. In this case, the problem is a special classification case called sequence labelling, tagging or parsing. The input in this case is a sequence of input values, e.g. a sentence, and the purpose of the model is to find the best analysis of that sentence as a whole and not token-by-token. In contrast, classifying input one-by-one is called local classification. Features are critical cues that could be used by the algorithm to help in deciding the correct class.

In this section, we will briefly discuss some of the most widely used techniques applied to various NLP tasks, including NER, with great success. The main reason is that they will be referred to throughout the thesis as being used in previous work or in the current study. The ones that will be adopted in our experiments will be revisited in our implementation section.

To demonstrate the mechanism of these algorithms, a simple example will be used. Our example is from POS tagging for its simplicity. Thus, assume that we have the following sentence that needs to be POS tagged using the Penn tagset, (see Appendix 1):
We will not go to this race.

In this example, we have two ambiguous tokens; go, race. Let us just consider the ambiguity of the word race, which could be a verb or a noun. The task is to find the correct tag for that word using machine learning. That would require some training data to learn from. Assume that we have the following two classified sentences (POS tagged using the GENIA tagger):

The / DT race / NN will / MD start / VB tomorrow / NN but / CC Jeff / NNP will / MD not / RB race / VBP ./.

The task of the machine learning algorithm is to find the highest probability of the possible classes; \( P(\text{race}|VB) \) and \( P(\text{race}|NN) \) with the help of features correlated with the class. Presumably, the most intuitive features are the word itself and the previous word, so we will include them in our model in addition to the class of the previous word. Both our training and test data are shown in Table 2.1 and in Table 2.2, with proposed features generated for a better visualization.

<table>
<thead>
<tr>
<th>( f_1 ) (word)</th>
<th>( f_2 ) (previous word)</th>
<th>( f_3 ) (class of previous word)</th>
<th>Class (tag or label)</th>
</tr>
</thead>
<tbody>
<tr>
<td>The</td>
<td>-</td>
<td>-</td>
<td>DT</td>
</tr>
<tr>
<td>race</td>
<td>The</td>
<td>DT</td>
<td>NN</td>
</tr>
<tr>
<td>will</td>
<td>race</td>
<td>NN</td>
<td>MD</td>
</tr>
<tr>
<td>start</td>
<td>will</td>
<td>MD</td>
<td>VB</td>
</tr>
<tr>
<td>tomorrow</td>
<td>start</td>
<td>VB</td>
<td>NN</td>
</tr>
<tr>
<td>but</td>
<td>tomorrow</td>
<td>NN</td>
<td>CC</td>
</tr>
<tr>
<td>Jeff</td>
<td>but</td>
<td>CC</td>
<td>NNP</td>
</tr>
<tr>
<td>will</td>
<td>Jeff</td>
<td>NNP</td>
<td>MD</td>
</tr>
<tr>
<td>not</td>
<td>will</td>
<td>MD</td>
<td>RB</td>
</tr>
<tr>
<td>race</td>
<td>not</td>
<td>RB</td>
<td>VB</td>
</tr>
<tr>
<td>.</td>
<td>race</td>
<td>VB</td>
<td>.</td>
</tr>
</tbody>
</table>

Table 2.1: Training data with features generated in tabular format

3 [http://text0.mib.man.ac.uk/software/geniatagger/]
Chapter 2: Supervised learning in an NLP framework

<table>
<thead>
<tr>
<th>$f_1$ (word)</th>
<th>$f_2$ (previous word)</th>
<th>$f_3$ (class of previous word)</th>
<th>Class (tag or label)</th>
</tr>
</thead>
<tbody>
<tr>
<td>We</td>
<td>-</td>
<td>-</td>
<td>PRP</td>
</tr>
<tr>
<td>will</td>
<td>We</td>
<td>PRP</td>
<td>MD</td>
</tr>
<tr>
<td>not</td>
<td>will</td>
<td>MD</td>
<td>RB</td>
</tr>
<tr>
<td>go</td>
<td>not</td>
<td>RB</td>
<td>VB</td>
</tr>
<tr>
<td>to</td>
<td>go</td>
<td>VB</td>
<td>TO</td>
</tr>
<tr>
<td>this</td>
<td>to</td>
<td>TO</td>
<td>DT</td>
</tr>
<tr>
<td>race</td>
<td>this</td>
<td>DT</td>
<td>?</td>
</tr>
<tr>
<td>.</td>
<td>race</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>

Table 2.2: Test data in tabular format just before processing the word race

Note that all target classes have been assigned in the training table (Table 2.1) and partially in the test table. The reason is that we assume that only the word race is ambiguous and all preceding words have been already processed. Therefore, each word of our sentence now has a set of features or feature vector. Angled brackets will be used in this discussion to represent a feature vector e.g. <race>, which is equivalent to ($f_1$=race, $f_2$=this, $f_3$=DT).

To solve problems of a similar nature, there have been many machine learning techniques used in NLP which showed great success. Some are considered core concepts and some are extensions or combinations. They are either discriminative or generative in the way that they try to compute the probability of $x$ to be classified as $c$. Discriminative models use conditional probability $P(C|X)$, while generative models use joint probability $P(X, C)$. To demonstrate the difference between the two techniques, consider the task to determine the language that someone is speaking:

- Generative approach: is to learn each language and determine which language the speech belongs to.
- Discriminative approach: is to determine the linguistic differences without learning any language.
2.1 Most popular supervised ML algorithms in NLP

2.1.1 Naive Bayesian (NB)

This is one of the generative models widely used in NLP, especially in text classification. It belongs to the group of graphical models which is used to model conditional independence between random variables. The Bayesian model is based on the use of Bayes law of probability which uses the inverse of conditional probability. The term “naive” comes from its main assumption that features are conditionally independent. NB is the simplest form of Bayesian Network (H. Zhang 2004). In classification, the task is to calculate the probability of a class C given data instance X; P(C|X) where C is a set of classes and X is a feature vector \( (f_1, f_2, \ldots, f_n) \), from (Flach and Lachiche 2004):

\[
P(C|X) = \arg\max P(C | f_1, f_2, \ldots, f_n) \quad (2.1)
\]

Given the conditional probability rule of two events A and B known as Bayes rule:

\[
P(A|B) = \frac{p(B|A)p(A)}{p(B)} \quad (2.2)
\]

Probabilities in the above equation are known as:

\[
Posterior = \frac{Likelihood \times Prior}{Evidence} \quad (2.3)
\]

The probability of a class could be calculated similarly applying Bayes rule:

\[
p(C|X) = \frac{p(X|C)p(C)}{p(X)} \quad (2.4)
\]

P(X) or Evidence is always constant for all classes with no dependence on the class, so it could be ignored, leaving the nominator, which is the joint probability:

\[
p(C|X) = p(X|C)p(C) \quad (2.5)
\]

Substituting for X in the above equation with feature vector gives the following:
Chapter 2: Supervised learning in an NLP framework

\[
p(C|X) = p(f_1f_2 \ldots f_n|C) \cdot p(C) \quad (2.6)
\]

The naive assumption is used to split the first component as follows:

\[
p(f_1f_2 \ldots f_n|C) = p(f_1|C) \cdot p(f_2|C) \ldots \cdot p(f_n|C) \quad (2.7)
\]

Substituting in Eq. 1.6 yields:

\[
p(C|X) = p(C) \cdot \prod_{1}^{n} p(f_n|C) \quad (2.8)
\]

To estimate the conditional probabilities, it is common to calculate MLE from training data using relative frequency for each feature-class as:

\[
p(f_n|C) = \frac{\text{count}(f_n,C)}{\text{count}(C)} \quad (2.9)
\]

In the case of zero probability, which is the case with unseen events, a number of smoothing techniques have been developed; we will use the “add one” technique for simplicity in our example.

Even though the strong independence assumption is violated in practice, NB still gives good results, while it is the optimal classifier when total independence exists; this is proved in (Domingos and Pazzani 1996) and (Domingos and Pazzani 1997).

![Figure 2.1: NB of the word race](image)
In our POS example, we need to calculate the probabilities of the two possible tags of the word *race*; \( p(VB \mid < race >) \) and \( p(NN \mid < race >) \). An NB graph for the latter case is in Figure 2.1; note the direction of the arrow meaning that it is generating the feature.

Thus, the components required for the two cases are calculated using Eq. 1.9 and illustrated in Table 2.3. Those probabilities are substituted into Eq. 1.8 to obtain class probabilities as follows:

\[
p(NN \mid < race >) = p(NN) * p(f_1 = race|NN) * p(f_2 = this|NN) * p(f_3 = DT|NN)
\]

\[
= \frac{2}{10} * \frac{2}{3} * \frac{1}{3} * \frac{2}{3} = \frac{8}{270}
\]

\[
p(VB \mid < race >) = p(VB) * p(f_1 = race|VB) * p(f_2 = this|VB) * p(f_3 = DT|VB)
\]

\[
= \frac{2}{10} * \frac{2}{3} * \frac{1}{3} * \frac{1}{3} = \frac{4}{270}
\]

Based on the calculation above, the class NN has the higher probability of being the correct class for the word *race*.

| Class/feature | \( P(C) \) | \( p(f_1 = race|C) \) | \( p(f_2 = this|C) \) | \( p(f_3 = DT|C) \) |
|---------------|-----------|----------------|----------------|----------------|
| VB            | \( \frac{2}{10} \) | \( \frac{1+1}{2+1} = \frac{2}{3} \) | \( \frac{0+1}{2+1} = \frac{1}{3} \) | \( \frac{0+1}{2+1} = \frac{1}{3} \) |
| NN            | \( \frac{2}{10} \) | \( \frac{1+1}{2+1} = \frac{2}{3} \) | \( \frac{0+1}{2+1} = \frac{1}{3} \) | \( \frac{1+1}{2+1} = \frac{2}{3} \) |

Table 2.3: Probability of each feature for the two classes

### 2.1.2 Maximum Entropy Model (MEM)

In generative models, the need to calculate the prior probability \( P(C) \) is not favoured, as it requires more training data. The alternative is to use discriminative models which calculate \( P(C \mid X) \). The maximum entropy framework estimates probabilities based on the principle of making as few assumptions as possible, other than the constraints imposed. Such constraints
are derived from training data, expressing some relationship between features and outcome. The probability distribution that satisfies the above property is the one with the highest entropy.

For each input in the process, we try to find the distribution that allocates its probability as evenly as possible subject to the constraints. Constraints are learned from the training data induced with decisions. The main principle is to model the known and assume nothing about the unknown. Logistic regression is a special case of MEM (Ratnaparkhi 1997).

The intuition of ME is to build a distribution by continuously adding features in the form of indicator functions which select a subset of the training data. Each feature adds a constraint to the distribution. The name ME comes from the fact that the most uniform distribution has the maximum (Jurafsky and Martin 2008).

There will be a number of distributions but the equiprobable distribution has the maximum entropy:

\[
H(x) = -\sum_x P(x) \log_2 P(x)
\]  

(2.10)

In our POS tagging example, the word race is either a VB or NN. Let us assume the following constraints:

**Constraint one:** Without any information, we expect the class of the word will be one of the 48 POS tags. The most uniform probability distribution is the one that covers all classes:

\[
P(NN) + P(VB) + P(IN) + P(NNP) + \ldots = 1
\]

Thus each class will have a probability of 1/48 as in Table 2.4:

**Constraint two:** From training data, we observe that in 10 words we have 2 of them tagged as NN. That is our second constraint:

\[
P(NN) = 1/5
\]

The rest of the probabilities will be uniformly distributed over the remaining classes;
Constraint three: The last constraint is that half of the words tagged as NN are equal to *race*.

\[ P(\text{NN}, \text{race}) = \frac{1}{2} \]

| Class/Constraint | One | Two | Three | ...
|------------------|-----|-----|-------|-----
| NN               | 1/48| 1/5 | 1/2   | ..  |
| VB               | 1/48| 4/235| 1/94 | ..  |
| IN               | 1/48| 4/235| 1/94 | ..  |
| NNP              | 1/48| 4/235| 1/94 | ..  |
| ...              | 1/48| 4/235| 1/94 | ..  |

Table 2.4: Iterative probability distribution over possible classes

The process will continue until no more constraints can be extracted from the training data. After that, it chooses the highest probability to be the best class for the word *race*. If we start with the most obvious constraint in our example, we will use the determiner before *race*, which will give NN with probability equal 1. However, we used another constraint to show how probability is distributed.

The learning phase is to estimate the weight of each of these features or constraints from training data. That is done through General Iterative Scaling. In training, ME uses indicator functions which is a binary output of 0 or 1. A feature has a corresponding weight that indicates how strong a cue it is.

In decoding, the probability of a data instance \( x \) falling into a class \( c \) is calculated as follows:

\[
P(c|x) = \frac{1}{Z} \exp \sum_i w_i f_i
\]

(2.11)
Where \( f \) is a feature value and \( w \) is a feature weight, \( c \) is the class under consideration and \( c' \) is the remaining classes. Thus, given our POS example, we need to reformulate the features into the required binary output format:

\[
f_1 = \begin{cases} 
1 & \text{if } x_0 = \text{race and } c = \text{NN} \\
0 & \text{otherwise} 
\end{cases} 
\]

\[
f_2 = \begin{cases} 
1 & \text{if } c_{-1} = \text{DT and } c = \text{NN} \\
0 & \text{otherwise} 
\end{cases} 
\]

\[
f_3 = \begin{cases} 
1 & \text{if } c_{-1} = \text{TO and } c = \text{VB} \\
0 & \text{otherwise} 
\end{cases} 
\]

Assume that Table 2.5 has the weight learned by the model. Using Eq. 1.12 we could calculate the probabilities of each of the possible tags of the word *race* as follows:

\[
P(\text{NN}|\text{race}) = \frac{e^9}{e^9 e^8 + e^9} = 12.18 / 14.64 = 0.83 
\]

\[
P(\text{VB}|\text{race}) = \frac{e^9}{e^9 e^8 + e^9} = 2.46 / 14.64 = 0.16 
\]

Our calculation above gives a probability distribution. We could consider the highest probability as the best class.

\[
c = \arg\max P(c|x) 
\]

In our case, the class to be assigned to the word *race* is NN.
2.1.3 Decision Trees (DT)

DT is a discriminative model that is probably the most used technique in Data Mining, known for its great classification power. It is mostly used with numerical data but could be used on nominal or categorical values such as those used in NLP. A decision tree is a tree in which each branch node represents a choice between a number of alternatives, and each leaf node represents a classification or a decision. In classification, each node is an attribute and each branch is a value of that attribute. The first branching node is called the root which is the attribute that best splits the data based on classes. At each branch there is a question to ask to decide the possible values.

The tree is built from all the training data recursively as follows:

1. Find the best branching attribute or feature to be the root node; the one that best splits data based on the class, which is usually calculated by information gain.
2. Split that node by adding branches that represent each value of the parent node.
3. If the child node can be classified uniquely, mark it as a leaf; otherwise calculate information again on remaining attributes.
4. Iterate until either no more gain is found or all attributes have been used.

In illustrating DT, a leaf is represented by a rectangle and an internal node by a circle. The leaf is where tree stops asking further questions.

One tree of our training data is illustrated in Figure 2.2, where we assume that the highest information gain was the word feature \( f_i \). Thus, we will have a number of branches equal to the size of the vocabulary. At each branch, we will also look again for the attribute not used as ancestors. So, considering the branch \( \text{race} \), the highest gain is tag of previous word and so on. That is called tree induction from the training data. In the testing phase, the algorithm will look for the tree that matches the input, and return the appropriate decision. To find the best class for the word \( \text{race} \), it will test the class of the previous word, which is DT in our case. Thus, it will classify it as NN.
Decision trees are one of the most commonly used machine learning techniques because of their simplicity to understand and to implement (Anyanwu and Shiva 2009). DT algorithms are also capable of rule induction. There are a number of algorithms implemented for tree induction, such as ID3, C4.5 and C5.0 by Quinlan.

According to (Paliouras et al. 2000), the popularity of DT is due its applicability to a variety of learning problems, its computational efficiency and the human-readable format of the induced models, i.e., the decision trees.

Common decision trees are usually used for classification rather than sequence labelling. Another decision tree instance used for sequence labeling is the statistical trees introduced by (Magerman 1995). A statistical decision tree assigns a probability to each of the possible choices, based on the context of the decision: \( P(f|h) \), where \( f \) is an element of the future vocabulary (possible values) and \( h \) is a history (previous decisions). This probability \( P(f|h) \) is calculated by asking a question at each node about the context.

The probability of a complete parse tree (C) of a sentence (X) is the product of each decision \( d \) conditioned on all previous decisions:
Chapter 2: Supervised Learning in an NLP Framework

\[ P(C|X) = \prod P(d_i|d_{i-1}d_{i-2}...d_1) \]  
(2.13)

Each decision sequence constructs a unique parse, and the parser selects the parse whose decision sequence yields the highest cumulative probability.

As shown in Figure 2.3, the first question at node 1 would be: what is the word to be tagged? If it is the, then it is classified as a determiner with \( P(DT|the) = 1.0 \). On the other hand, if the word is race, then more questions need to be asked.

![Figure 2.3: Statistical version of POS tagging tree](image)

DT has been used in various NLP tasks, such as POS tagging in Greek by (Orphanos et al. 1999) and (Màrquez and Rodríguez 1998). Also, it was applied to co-reference resolution by (Soon et al. 2001). Binary DT was used in POS tagging by (Schmid 1994) to estimate the transition probabilities in an efficient way.
2.1.4 **Hidden Markov Model (HMM)**

This is a special case of Bayesian inference classification described above; the difference is that it performs classification in chunks instead of single instances (Jurafsky and Martin 2008).

A hidden Markov model (HMM) is a statistical construct that can be used to solve classification problems that have an inherent state sequence representation. The model can be visualized as an interlocking set of states. These states are connected by a set of transition probabilities, which indicate the probability of travelling between two given states. A process begins in some state, then at discrete time intervals, the process "moves" to a new state as dictated by the transition probabilities. In an HMM, the exact sequence of states that the process generates is unknown (i.e., hidden). As the process enters each state, one of a set of output symbols is emitted by the process. Exactly which symbol is emitted is determined by a probability distribution that is specific to each state. The output of the HMM is a sequence of output symbols. The model was first applied to speech recognition in the seventies and later was successfully applied to different NLP and non-NLP tasks.

Two important assumptions are made by the HMM model; state is independent of all predecessor states except the previous one. The observation depends on its emitting state (Sutton and McCallum 2007). Hidden comes from the assumption that the state is hidden and the observation is shown. Second, the hidden state is Markov, meaning that all information is encoded in the given state to predict the future (Ghahramani 2001).

In NLP, states of the model represent word classes that range over C and observations are the words ranging over X.

Given a sequence of words (X), the task of the HMM tagger is to find the most likely sequence of tags for a sentence. In other words, maximizing P(C|X)

\[
C = \text{argmax } P(C|X)
\]  

(2.14)

By applying Bayes rule (Eq. 1.4):
Chapter 2: Supervised learning in an NLP framework

\[
C = \arg\max P(X|C) P(C) / P(X)
\]

\[
P(C|X) = \frac{P(X|C) P(C)}{p(X)}
\]

Deleting \(P(X)\) since it is constant for this sequence of words:

\[
C = \arg\max P(X|C) P(C)
\]

Form the chain rule of probability:

\[
P(X|C) P(C) = \prod_{i=1}^{n} P(x_i|x_{i-1}c_{i-1}c_{i-1}c_i) P(c_i|x_{i-1}c_{i-1}c_{i-1})
\]

(2.15)

To calculate the above equation, the two HMM assumptions mentioned earlier are considered.

Assumption 1: the probability of a word is dependent only on its class:

\[
P(x_i|x_{i-1}c_{i-1}c_{i-1}) = P(x_i|c_i)
\]

Assumption 2: a class is dependent on the previous class (in the case of a bigram HMM):

\[
P(c_i|x_{i-1}c_{i-1}) = P(c_i|c_{i-1})
\]

Substituting into Eq. 1.15 gives the following:

\[
C = \arg\max \prod_{i=1}^{n} P(x_i|c_i) P(c_i|c_{i-1})
\]

(2.16)

The two components of the equation are estimated from training data as follows:

\[
P(c_i|c_{i-1}) \frac{\text{count}(c_{i-1}, c_i)}{\text{count}(c_{i-1})}
\]

\[
P(x_i|c_i) = \frac{\text{count}(x_i, c_i)}{\text{count}(c_i)}
\]
Chapter 2: Supervised learning in an NLP framework

It is worth noting that the latter component is not asking what is the most likely class for word $x$, but rather, if the word is $x$, how likely it is that its class is $c$; thus it is called generative.

The above calculation will find the values of each transition and emission probability of each possible class for each observation. Then, the most likely sequence of classes or paths is calculated by dynamic programming i.e., the Viterbi algorithm.

In our example POS example, we will have two Markov chains. One is displayed in Figure 2.4, the other one is similar, with one difference, which is that the last state is VB instead of NN. Assume that everything else is the same. The product of Eq. 1.15 is affected by emission and transition probabilities of the word *race*, see Table 2.6. We are assuming that all other words have only one single possible class.

![HMM chain](image)

**Figure 2.4: HMM chain**

| Class/prob | Emission = $P(x|c)$ | Transition = $P(x|c)$ |
|------------|---------------------|-----------------------|
| NN         | $P(race|NN) = \frac{1}{2}$ | $P(NN|DT) = 1/1$     |
| VB         | $P(race|VB) = \frac{1}{2}$ | $P(VB|DT) = 0$       |

Table 2.6: Emission and transition probabilities of both analyses for *race*
2.1.5 Maximum Entropy Hidden Markov Models (ME-HMM)

As the name indicates, this algorithm is considered as an extension to the HMM model previously described, augmented with the Maximum Entropy concept. The purpose is to eliminate the need to model the prior probability $P(X)$, which is part of the joint probability estimated in HMM. That elimination makes this algorithm a discriminative model, since it models the conditional probability $P(C|X)$ directly. The main advantage is the possibility of using features of wider context and not the restricted independence of states and observation used in HMM.

The modelling is the same as in HMM and the difference is in calculating the transition probability $P(c_i|c_{i-1})$.

In HMM, it is dependent on previous class, while in ME-HMM both transition and emission are combined into one equation as follows:

$$
C = \arg\max \prod_{i=1}^{n} P(x_i|x_{i-1}, c_{i-1})
$$

$$
P(C|X) = \prod_{i=1}^{n} P(c_i|x_i, c_{i-1})
$$

(2.17)

Figure 2.5 shows a modified version of the chain we used to illustrate HMM, now reflecting the combination of the two probabilities using MEM.

![ME-HMM chain](image)

Figure 2.5: ME-HMM chain
At each state, MEM Eq. 1.11 is used to calculate the conditional probability as follows:

$$P(c_i|x_i, c_{i-1}) = \frac{1}{Z(c, x')} \exp \sum_i w_i f_i(c_i, c_{i-1}, x)$$  \hspace{1cm} (2.18)

In practice, it is not only the previous word that could be used; any other feature could be added to the model. MEHMM is able to solve the joint probability overhead and model overlapping features, which is the case when states depend on previous and future observations (McCallum et al. 2000).

### 2.1.6 Conditional Random Fields (CRF)

Both previous HMM models suffer from what is called the label bias problem which is the case of zero or low transition probabilities are never visited in decoding phase as the observation is ignored. The CRF was introduced as a solution to that problem by (Lafferty et al. 2001). The fundamental theory of random fields was presented in (Hammersley and Clifford 1968). It is a log linear type of sequence tagging algorithm similar to ME-HMM in maximizing $P(C|X)$. In addition to the ME-HMM’s capability to include wider context, it has the capability of solving the label bias problem. According to (Sutton and McCallum 2007), CRF has the ability to relax the strong independence assumptions made in HMM models. For that purpose, it is constructed as an undirected graph, meaning that no adjacent dependency exists in the model; see Figure 2.6.
CRF makes the constant transition probabilities used in HMM into arbitrary functions that vary across the positions in the sequence of hidden states, depending on the input sequence. It uses an energy function instead of probabilities.

\[
P(c|x_i) = \frac{1}{Z(x)} \exp \sum_i \sum_n w_i f_i(c_i, c_{i-1}, x_n, i)
\]

From the previous equation, it can be observed that CRF is normalizing the probability globally by \(Z\), which is called the partition function. The equation combines the two states and the whole input sequence and the position in the sequence. CRF can contain any number of feature functions; the feature functions can inspect the entire input sequence \(X\) at any point. ME-HMM uses a pre-state exponential model while CRF use a single exponential model for the entire sequence. Therefore, the weights of different features at different states can be traded off against each other.

An extension to this algorithm called semi-Markov Conditional Random Fields (or semi-CRFs) was introduced in (Sarawagi and Cohen 2004). The main advantage of semi-CRFs is that they allow features which measure properties of segments, rather than individual elements, such as named entity recognition. The semi-CRF performs joint segmentation and labelling by assigning labels to each chunk.

Another semi-CRF instance was introduced in (Andrew 2006), which directly models the distribution of chunk boundaries as the previous one but able to incorporate features that indicate token is not on boundaries.

### 2.1.7 Support Vector Machines (SVM)

SVM is a discriminative technique based on the concepts of Neural Networks and Perceptron learning. According to (Burges 1998), the idea was introduced through the statistical learning theory in the late seventies by (Vapnik 1979). More recent attention was in 1992 where the term SVM was first used by Vapnik. Its first experimental use was in the
recognition of handwritten digits in 1996 by (Cortes and Vapnik 1995) when the concept of soft margins was introduced.

The technique tries to find a function that maps data points from the input space to the feature space \( x \rightarrow y \). In the case of binary classification, \( c \) will have binary values of either 0 or 1. The mapping function works to make the input in one class linearly separated from the other class. Then it tries to find the optimal separating hyperplane that separates the two classes. In Figure 2.7, we have two categories or classes where black filled and white circles represent data points of the two classes. These data points are clearly linearly separable and could have an infinite number of hyperplanes capable of separating the two classes i.e. \( H_1 \) and \( H_2 \).

\[
x \cdot w + b = 0
\]

\( b/w \) is the perpendicular distance from the hyperplane to the origin and \( w \) is norm of hyperplane.

SVM tries to find the optimum hyperplane, which is the one with the maximum distance to points of each class. Finding the optimal hyperplane is achieved by locating the points closest to the other class or that highlight the boundary facing the other class. Those points are called support vectors; they are circled in Figure 2.8. If two parallel hyperplanes pass
through the support vectors with no data points in between, the optimal hyperplane is the one that maximizes the distance between the two parallel hyperplanes, M.

![Figure 2.8: Separating hyperplanes](image)

Consider that the distance from each of the two hyperplanes to the optimal hyperplane is d, so that d = M/2. In the figure, S1 and S2 are the supporting hyperplanes and M denotes the margin or the distance between them.

Equation of S1: \( x \cdot w + b = 1 \) and points on S1 distance equals \( \frac{|1-b|}{||w||} \). D+

Equation of S2: \( x \cdot w + b = -1 \) and points on S2 distance equals \( \frac{|-1-b|}{||w||} \). D-

Given that these two vectors are parallel, they have the same norm; thus

\[
\frac{|1-b|}{||w||} = \frac{|-1-b|}{||w||} = \frac{1}{||w||}
\]

Finally, minimizing w will maximize the distance and hence the margin M (Burges, 1998).

In cases in which no linear separation is possible, SVM can work with techniques of kernel, which automatically realize a non-linear mapping to a feature space. The hyperplane found by the SVM method in the feature space corresponds to a non-linear decision boundary in the input space (Furey et al. 2000). If data points are not linearly separable, SVM uses the kernel trick to project the data to higher dimension (Yu and Kim 2009).
In the POS tagging example, the SVM training phase starts with two classes and finds the optimum hyperplane, then moves to another combination of classes. SVM uses a numerical representation of instances, so that each instance is converted into a vector of numerical features.

Let us consider the binary classification task with classes NN and VB. Also, we will only use two features for simplicity:

For NN class, there are two data points \(x= (\text{race, the}) (0,1)\) and \(x= (\text{tomorrow, start}) (1,2)\)

For VB class, there are two data points \(x= (\text{race, to}) (0,3)\) and \(x= (\text{start, will}) (2,4)\)

After the learning phase, we would have generated an equation of the separating hyperplane; \(x \cdot w + b = 0\). In the testing phase, considering the word \text{race} in our test data, it needs to be converted to vector: \((\text{race, this}) \rightarrow (0,5)\). Then, substitute the value of \(x\) in the hyperplane equation. The classification decision is based on the output value; if greater than zero it is VB, or NN otherwise.

The goal of the SVM is to optimize "generalization", the ability to correctly classify unseen data. In particular, SVMs achieve high generalization even with training data of a very high dimension.

In the field of NLP, the SVM method was applied to text categorization and syntactic dependency structure analysis, and achieved great success. Recent advances in SVM for sequence labelling augmented SVM with HMM, as presented in (Altun et al. 2003). Also, structural learning SVM was introduced by (Tsochantaridis et al. 2004) with an improved training algorithm in (Joachims et al. 2009).

### 2.1.8 Transformation-Based Learning (TBL)

TBL-Learning was firstly introduced in (Brill 1995) and was successfully used in a number of NLP tasks, achieving very interesting results in POS tagging. The main idea behind this approach is to induce the classification rules from the training data. Learning the rules is an
error-driven process guided by predefined templates. It starts by assigning each word its most frequent tag in the training data. Then, the largest error class is calculated by the confusion matrix in each iteration of the learning process. Next, the gain is calculated by applying the templates to the training data. Subsequently, the rule with the highest gain is added to the list of retagging rules. Finally, these rules are applied to the test data as in the rule-based approach. This approach will be revisited later in chapter 5 with more details when we discuss our POS tagger.

2.1.9 Joint Learning

Given that IE extraction systems involve a number of independent processes; this would result in annotation inconsistency between one and another. One example is parsing and named entity recognition, where the span of the tree might conflict with the span of the named entity. (Finkel and Manning 2009) proposed a technique to overcome this problem allowing the learning algorithm to build a joint model for both tasks. They augmented the parse trees with named entity annotation. Their experiments on six different datasets showed that the performance of both tasks were improved. To overcome the limitation caused by the lack of jointly annotated data, (Finkel and Manning 2010) proposed a technique that only requires small amount of jointly annotated training data augmented with large amount of single-task annotated data. The idea is to share some features used in the joint model with each of the single-task models. Then, the singly-annotated data can be used to influence the feature weights for the shared features in the joint model. Their experiments on the same datasets used in (Finkel and Manning 2009) showed that the hierarchal joint modelling technique was able to reduce the errors by over 20% of both tasks compared to the joint model trained only on jointly annotated data.

2.2 Comparison of ML algorithms

There are a number of parameters to be considered in evaluating and comparing ML algorithms, for example training time or the time to build the model, decoding or testing
time, accuracy, etc. In this discussion, we are concerned with the accuracy first, as it is the most important factor in our research.

Previous work on NLP has used various instances of ML. Each ML algorithm behaves differently based on many factors such as implementation, parameters, data nature, size, data sparseness, selection of feature, task complexity. There has not been total agreement on which algorithm guarantees the best performance.

Recent advances in NLP use the concept of bidirectional learning such as CRF and kernel methods such as SVM, which are claimed to be the state-of-the-art, but there is no rule of thumb for choosing between them. In addition, some of the classical methods are still doing well and sometimes outperform the recent SVM method.

The SVM method was found to be superior to NB, DT and Rules Induction in word sense disambiguation (Joshi et al. 2005). Also, SVM outperformed CRF on Vietnamese word segmentation (C. Nguyen et al. 2006). In text classification, SVM, *K-Nearest Neighbor* and NB showed very close performance in (Colas and Brazdil 2006). SVM exceeds MEHMM slightly in biomedical NER, but the performance gap was very large when using an MEM classifier not augmented with Viterbi (Kazama et al. 2002).

In more recent experiments, SVM has shown superiority over CRF and other structured learning algorithms (N. Nguyen and Guo 2007). However, this is denied by (Keerthi and Sundararajan 2007) based on their experiments that showed very close performance. They concluded that features and implementation are what makes the difference in the previous study.

In contrast, (Krishnarao et al. 2009) proved that CRF is superior on NER in Indian Languages NER compared to SVM. They inferred that the sparseness of the named entities plays a major role in deciding the final classification. Also, as a finite state machine derived from HMMs, CRFs can naturally consider state-to-state dependencies and feature-to-state dependencies. On the other hand, SVMs do not consider such dependencies.

Another comparative study concluded that CRF outperforms SVM on clinical NER (Li et al. 2008). They argued that CRF is good when combining unrelated features, while SVM is
good with overlapping features. In (Putthividhya and Hu 2011), comparison of SVM, MEM, HMM and CRF on an entity extraction problem revealed the close performance of all, with SVM and MEM performing slightly better. They also concluded that more features improve CRF performance but not SVM. Also, a very recent sequence labelling comparison study showed that CRF outperforms conventional SVM and is comparable to structural SVM (X. Zhang et al. 2011).

Another study (Peng and McCallum 2006) showed that CRF outperforms SVM in IE. It is true that there was a difference in performance, but it was very small difference. Another experiment in (PVS and Karthik 2007) showed close results between HMM and CRF.

Another sequence labeling problem, POS tagging, was studied in (Gambäck et al. 2009) using MEM, SVM and HMM. Their results showed that MEM slightly outperforms the other two. Also, MEM and SVM perform comparably on another experiment in (Snoek and Worring 2005).

One study attempted to test most algorithms of WEKA, a machine learning tool with a collection of algorithms, showed the superiority and closeness of NB and SMO, which is an SVM implementation (Andreeva et al. 2004).

A recent data mining experiment ranked DT first, followed by NB, both inferior to SVM (Douglas et al. 2011).

TBL and rule induction algorithms are good in term of the clarity of the output model, that is easy to interpret by humans and hence to debug and analyze. TBL is very slow in training but very fast in testing. It gives almost the same results as other algorithms and is sometimes better.

An official comparative study, the Pascal Challenge, was launched by Sheffield University to evaluate a number of ML algorithms on IE tasks with cross validation. It has interesting findings when comparing a number of ML algorithms on standard data and tasks. The results of the experiments found that the four best performances used different ML algorithms; Rules induction, HMM, CRF, SVM (Ireson et al. 2005).
Noisy data is the situation when we have mis-classified data or attributes (features) with low correlation with the class. It is another important factor in most real world applications especially when generating features with low quality. Logistic regression was less affected by noise in (Kalapanidas et al. 2003). It was also found that NB is the least affected by noise level although giving the worst accuracy. DT was also found to have the best improvement rate with reducing noise level. Moreover, SVM is known to be sensitive to mis-labelled data (Atla et al. 2011).

A comprehensive study on class and attribute noise is in (Zhu and Wu 2004); it is noted that attribute noise caused by less correlation with the class is more widespread than class noise caused by mis-classification. However, class noise is more harmful to performance although attribute noise is still critical.

In terms of training speed and testing, NB and MEM are the fastest compared to other ML algorithms. This factor is critical when investigating different settings and features from knowledge sources. An experiment by (Koprinska et al. 2007) found that NB was the fastest while the slowest was SVM by a very large margin. Also Random Forest, an ensemble classifier with many decision trees, performed the best with an acceptable training time, one third of that of SVM. Empirical studies need efficient feasible techniques to minimize training time. Some algorithms are computationally expensive in terms of time and memory.

The previous discussion agreed with the “no free lunch” theory (Wolpert and Macready 1997); there is no single learning algorithm that universally performs best across all domains. As there is no clear superiority, it is better to use different ML methods for a given task and then compare the outcome.

Although the selection of the ML algorithm is crucial step in building NLP systems, the size of the training data proved to be more important to the performance, as demonstrated in (Banko and Brill 2001). In their experiments, they found out that the ranking of performances of four learning algorithms was reversed when the data size was increased to 1 billion words. The task that was investigated in their experiment, confusion set disambiguation, was feasible to use huge amount of data since the data is already available.
Given that not all NLP tasks have this criterion, a number of techniques have been proposed to overcome this limitation and help in annotating unlabelled data.

As these ML algorithms have different behaviours in different settings, it would not be possible to find the optimal one without evaluating them on the given task and available data.
Chapter 3    Arabic Language Characteristics

This chapter is an essential preliminary discussion that covers a general overview of the Arabic language. This brief introduction will emphasize aspects that concern the scope of this thesis, which is proper noun identification. It will cover history, orthography, morphology, structure, etc. The last part of this chapter is dedicated to issues relating to proper nouns.

Throughout this discussion, there will be a need for Arabic examples. When first introduced, the Arabic word will be accompanied by its transliteration and translation if needed in the following format; [Arabic, transliteration, translation]. The Buckwalter transliteration scheme \(^4\) will be adopted and is detailed later in Table 3.1. If the same word is used again, only the transliteration will be used and will be placed in quotes. As expected, capitalization of the first character in the translation indicates a proper noun.

3.1 History and Current Perspective

According to (Lipiński 2001), the term Semitic language was introduced in 1781 by the German historian AL Schlozezer, as it was spoken by the sons of Sem (Shem) with a long history of more than 4500 years. This family is part of the Afro-Asiatic family and its first written form was introduced in the third millennium BC. Semitic languages were among the earliest to attain a written form, with Akkadian writing beginning in the middle of the third millennium BCE. The most widely spoken Semitic language today is Arabic, followed by Amharic, Hebrew then Tigrinya (Hetzron 1997).

\(^4\) http://www.qamus.org/transliteration.htm
Chapter 3: Arabic Language Characteristics

A number of identifying features for Semitic languages family was proposed in (Versteegh 2001). According to the study, the language should have a root-pattern morphological system, the presence of emphatic/glottalised consonants, a verbal system with a prefix and suffix conjugation, as well as a large number of lexical correspondences. If a language is to be classified as Semitic, it has to exhibit the presence of all these features.

Today, Arabic is the mother tongue of more than 317 million people in the Arab states. According to UN estimates, the Arab countries will be home to some 395 million people by 2015 (UN Development Programme 2009). Moreover, Arabic is the religious liturgical language of more than 1.5 billion Muslims all over the world. It is one of the official UN languages and ranked 6th in terms of its importance. Weber, in his article about the top ten influential languages (Weber 1997) ranked Arabic 5th, based on the following measures scoring:

1. Number of primary speakers
2. Number of secondary speakers
3. Economic power of countries using the language
4. Number of major areas of human activity in which the language is important
5. Number and population of countries using the language
6. Socio-literary prestige of the language

The study is relatively old, as it covers the period of 1980-1990, considering that the parameters have direct correlation with time. However, Weber believed that his finding does not need to be updated as world population has grown relatively.

According to one official internet monitoring agency5, the world’s web users’ growth rate between 2000 and 2010 was 445%. Impressively, the Middle East, excluding African states, has scored the second highest web users’ growth rate in same period, about 1825%, just after Africa, which contains 10 Arab countries, at about 2800%. Assuming that most of these users are Arabic speakers, the Arabic web content will have to increase dramatically, which require a serious effort to address the need for Arabic NLP tools and resources. Efforts to

---

5 www.internetworldstats.com
Chapter 3: Arabic Language Characteristics

enrich the Arabic content have been initiated recently and this will result in dramatic increase of Arabic content e.g., King Abdullah's initiative organized by KACST.

3.2 Orthography

Similar to its Semitic family (Amharic, Hebrew and Tigrinya), Arabic is written from right to left. It has 28 letters, in basic forms, including three long vowels. Non-basic forms are letters generated by a combination of two letters. In addition, it includes, as diacritics, five main short vowels (not included in the alphabet) and a total of 13 short vowel combinations. Vowels and short vowels are related in a sense that a vowel is a double short vowel. These diacritics are mainly used for the accurate pronunciation of consonants, which in turn helps in clarifying the exact interpretation. They are placed above or below letters. That process is described as vocalization and text could be fully, partially or never vocalized depending on the written form (Buckwalter 2004).

Arabic letters have a cursive feature, meaning that a letter might have a different shape based on its location in a word; initial, medial, final, or isolated positions. Only three letters are not affected by that feature. Moreover, six letters in the alphabet have only two possible forms because only preceding letters could connect to them; these six letters cannot be connected to the following letters (Abdelali 2004).

One of the main features of Arabic text is the lack of capitalization, meaning it does not exhibit orthographic differences in terms of case. A further feature is that there are fewer punctuation marks, but these have been recently introduced.

The Kashida [ـ] is a special character for lengthening a letter. For instance stretching the letter [ح, H, h] in the word [محمد, mHmd, Mohammad] produces the new form [مـحمد]. It is used either to respect the constraints of calligraphy or for text justification (Elyaakoubi and Lazrek 2005).

The Arabic script is the second widely used script; it is used for: Pashto, Farsi, Kurdish, Urdu and Sindhi languages (Wagner et al. 1999).

### 3.3 Transliteration

In the era of the digital world, Arabic characters require a way to be represented in Latin characters due to the lack of support for Arabic characters in most computer software. The other reason is for non-Arabic readers to have a better understanding when demonstrating aspects of the language. One-to-one character mapping from the source language into the target script is known as transliteration, which is part of Romanization to convey spelling and not to be confused with transcription to convey pronunciation.

There has been a number of schemes used for translation in literature e.g, Buckwalter, LC, ISO and NewWay. However, the Buckwalter scheme, in Figure 3.1, was chosen in this study for its simplicity and since it does not use any unusual diacritics. Moreover, most of the literature on the Arabic language has adopted this scheme, which gives more consistency in this study. The Qamus\(^7\) website has more details of the scheme including the Unicode characters. An extension to the Buckwalter transliteration scheme was introduced in (Nizar Habash et al.) to deal with the fact that it is not intuitively easy to read.

\(^7\) [http://www.qamus.org/transliteration.htm](http://www.qamus.org/transliteration.htm)
Arabic has a complex morphological system that makes it a highly inflected language, with the presence of prefixation, suffixation, inflectional and derivational processes. Although it has a complex system, it is strongly structured (Kiraz 2002). In addition to affixation, it has the feature of clitic attachment to stems. Arabic also has a rich morphological system, where words are explicitly marked for case, gender, number, definiteness, mood, person, voice, [8](http://www.qamus.org/transliteration.htm)
tense and other morphological features (Maamouri et al. 2006). Following is an overview of Arabic word classes and how they are generated.

### 3.4.1 Word Classes

The two generic word classes are open (noun and verbs) and closed classes. According to (Khoja 2001), Arabic words can be classified into the following:

#### 3.4.1.1 Noun

A noun in Arabic is a name or a lexical unit that is used to describe a person, thing, or idea. The noun class in Arabic is further subdivided into derivatives based on the origin of the word, as follows:

- nouns derived from verbs
- nouns derived from other nouns
- nouns derived from particles
- primitives (i.e. nouns not derived from any other categories).

In addition, this class includes what would be classified as participles, pronouns, relative pronouns, demonstratives, interrogatives and numbers.

Moreover, the morphological features of 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
3.4.1.2 Verb

The Arabic inventory of verbs i.e. words describing an action, is poor compared to English verbs, which exhibit richness in tense and aspect. The deficiency of Arabic verbs is caused by the lack of precise time signification or flow of time as in English.

The morphological features for an Arabic verb and their possible values are as follows:

- Aspect : perfective (past), imperfective (present), imperative (future)
- Voice : active, passive
- Tense : past, present, future
- Mood : indicative, subjunctive, jussive

The person, number and gender features are dependent on the subject features. The values of these features are similar to those of the noun. It is worth mentioning that those features will not be present in some cases, for example, when a verb precedes the subject.

3.4.1.3 Particle

The particle class includes: prepositions, adverbs, conjunctions, interrogative particles, exceptions, interjections, negations, and subordinations.

It is worth noting that the noun and verb categories are used to classify open-class words, while the particle category classifies the closed-class words.

3.4.2 Stem Generation

Arabic derivational morphology is based on the principle of roots and patterns to generate open-class stems. A root (called radical) is a sequence of consonants, commonly trilateral. (Beesley 2001).

There is a finite set of roots and it is reported that nouns and verbs are derived from a closed set of around 10,000 roots (Al-Fedaghi and Al-Anzi 1989) and the number of possible words is estimated to be $6 \times 10^{10}$ words (Attia 2000).
Chapter 3: Arabic Language Characteristics

The pattern is a set of transformations applied to root consonants by inserting vowels between them. Vowels could be long vowels but are commonly short vowels (diacritics) (Beesley 2001).

In a given trilateral root CCC (C represents consonant) the frequent pattern is CvCvC (v represents vowel); the consonants are fixed and vowels are variable. For each v combination, a new pattern is produced from that template and hence a new stem is derived from that pattern. Thus, a single root can generate hundreds of words in the form of nouns or verbs (Ahmed, 2005). The Arabic root [كتب, ktb, notion of writing], is a trilateral root and one could generate the past tense form by substituting the v variables with vowels melody (a-a), "katab". This feature of Arabic morphology is described as non-concatenative or nonlinear morphology. Three Arabic stem generation examples are given in Table 3.1.

<table>
<thead>
<tr>
<th>Root/CCC</th>
<th>Transliteration</th>
<th>Melody/a-a</th>
<th>Translation</th>
<th>Melody/u-i</th>
<th>Translation</th>
</tr>
</thead>
<tbody>
<tr>
<td>كتاب</td>
<td>ktb</td>
<td>katab</td>
<td>wrote</td>
<td>kutib</td>
<td>to be written</td>
</tr>
<tr>
<td>درس</td>
<td>drs</td>
<td>daras</td>
<td>learned</td>
<td>Duris</td>
<td>to be learned</td>
</tr>
<tr>
<td>جمع</td>
<td>jmE</td>
<td>jamaE</td>
<td>collected</td>
<td>jumiE</td>
<td>to be collected</td>
</tr>
</tbody>
</table>

Table 3.1: Stem generating examples

Arabic roots are classified depending on the number of their consonants into biliteral, triliteral, quadriliteral and quinquiliteral. It was reported in (Elkateb et al. 2006) that 85% of Arabic words are derived from triliteral roots. There are 15 triliteral patterns, of which at least nine are in common use, and four much rarer quadriliteral patterns. All these patterns undergo some stem changes with respect to voweling (Violetta et al., 2000). Vowels are normally added to the root in the pattern CvCvC for triliterals and CvCCvC for quadriliterals.

Stems generated from the same root are semantically related and, on the other hand, stems that follow same pattern exhibit the same style. For example, certain patterns can state that the action is performed only once or many times.
3.4.3 Arabic Word Structure

The previous section covered stem generation, which form the basic building block of the Arabic word. Word-forms are complex units which encompass the following:

- **Proclitics**, morphemes occurring at the beginning of a word which include monosyllabic conjunctions and prepositions. Verbs can have only one proclitic while nouns can have up to two. The possible proclitics are listed in Table 3.2 with their functions.

<table>
<thead>
<tr>
<th>Morpheme</th>
<th>Transliteration</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>+و</td>
<td>w +</td>
<td>and</td>
</tr>
<tr>
<td>+فـ</td>
<td>f+</td>
<td>in order to</td>
</tr>
<tr>
<td>+لـ</td>
<td>l+</td>
<td>in order to</td>
</tr>
<tr>
<td>+بـ</td>
<td>b+</td>
<td>for</td>
</tr>
<tr>
<td>+و+لـ</td>
<td>w+l+</td>
<td>and+(in order to, for)</td>
</tr>
<tr>
<td>+و+بـ</td>
<td>w+b+</td>
<td>and+(in, at, by)</td>
</tr>
<tr>
<td>+فـ+لـ</td>
<td>f+l+</td>
<td>in order to+(for)</td>
</tr>
<tr>
<td>+فـ+بـ</td>
<td>f+b+</td>
<td>in order to+(in order to, for)</td>
</tr>
</tbody>
</table>

Table 3.2: Proclitics

- **Prefix**, morphemes which are commonly used with verbs to mark inflection. This category includes, for instance, the prefixes of the imperfective, e.g. *ya-*, prefixed morpheme of the 3rd person.

- **Stem**, the baseform which could be a noun or verb generated from root and pattern system, as discussed in the previous section.
- **Suffixes**, marks to indicate morphological features such as gender, number, case for nouns, mood for verbs.

- **Enclitics**, morphemes that occur at the end of a word, representing complement pronouns listed in Table 3.3. Their function is different, according to the class of the word to which they attach.

<table>
<thead>
<tr>
<th>Morpheme</th>
<th>Transliteration</th>
<th>Function (verb)</th>
<th>Function (noun)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ي +</td>
<td>+y</td>
<td>-</td>
<td>my</td>
</tr>
<tr>
<td>ك +</td>
<td>+k</td>
<td>you(singular)</td>
<td>your</td>
</tr>
<tr>
<td>كما +</td>
<td>+kma</td>
<td>you(dual)</td>
<td>your</td>
</tr>
<tr>
<td>كم +</td>
<td>+km</td>
<td>you(plural)</td>
<td>your</td>
</tr>
<tr>
<td>كن +</td>
<td>+kn</td>
<td>you(plural feminine)</td>
<td>your</td>
</tr>
<tr>
<td>ه +</td>
<td>+h</td>
<td>he</td>
<td>his</td>
</tr>
<tr>
<td>ها +</td>
<td>+hA</td>
<td>she</td>
<td>her</td>
</tr>
<tr>
<td>هما +</td>
<td>+hma</td>
<td>they(dual)</td>
<td>their</td>
</tr>
<tr>
<td>هم +</td>
<td>+hm</td>
<td>they(plural)</td>
<td>their</td>
</tr>
<tr>
<td>هن +</td>
<td>+hn</td>
<td>they(plural feminine)</td>
<td>their</td>
</tr>
</tbody>
</table>

Table 3.3: Enclitics

### 3.5 Arabic language forms

Arabic has a number of varities used in different contexts. All varieties are descended from classical Arabic, which is the Quran language. Classical Arabic is well structured and fully vowelized but is rarely used these days. The Arabic used today is the colloquial (dialect or regional) Arabic in most non-official communication activities. Dialects are less inflected than the classical language e.g, the masculine plural personal pronoun “antrm” is used for
Chapter 3: Arabic Language Characteristics

both genders in Jordan instead of “antn” in classical Arabic. Colloquial Arabic is less often written but increasingly appearing these days in on web forums and in poetry.

The official language used nowadays is a form of a “diglossia”, which is defined as two forms of the language (Farghaly and Shaalan 2009). This form of Arabic is the Modern Standard Arabic (MSA), merging classical and colloquial. Moreover, varieties of MSA forms are used today across the Arabic World, as further discussed in (Zainab, 2009).

The main feature of MSA is the total or partial absence of diacritical marks that represent vowels. MSA varieties are discussed in (Ahmed, 2004), which revealed differences in MSA among ten different countries relating to lexicon and spelling and loan words.

One critical aspect of Arabic writing today is spelling errors. Common sources of spelling error were studied in (Shaalan et al. 2003), categorizing sources of error as; hearing, writing, morphology, etc. Spelling errors are persistent in any written text of all languages. However, some errors in Arabic are not only errors but rather a common writing practice. That practice is critical when it results in changing the analysis of the generated word by producing a valid word with a totally different analysis. In English, if “Walker” was mistakenly typed as “Walked”, the sense does not change much and this might not have a strong impact on the processing. In Arabic, the letter shapes and sounds make it more susceptible to errors. Even literate writers would still commit errors frequently.

3.6 Arabic Proper Nouns

The most significant feature of Arabic proper nouns is the lack of any special orthographic feature to distinguish them from other word categories. Unlike English and other Latin character languages, Arabic has no case information. The implication of this feature is highly significant due the fact that most proper names are in the forms of verbs, adjectives or common nouns. This makes Arabic proper noun extraction more closely bound to, and highly affected by, preprocessing steps prior to the proper noun extraction task. This task is more sensitive to errors in previous analysis levels prior to proper noun extraction than in English.
The purpose of this discussion is to shed some light on the problem that makes processing proper nouns more sensitive to noise generated by previous preprocessing steps. I will start with a general overview of Arabic proper noun classes explaining how they are formed in Arabic. The discussion will cover some of the internal patterns that might help in building an Arabic proper noun recognition system. The three proper noun classes that will be considered in this discussion are; person, organization and location. At the end, challenges affecting the task of proper noun extraction will be covered.

3.6.1 Structure

3.6.1.1 Person Names

In terms of the person name structure, the smallest constituent of an Arabic person name could be classified as (1) a core component that could be a simple or compound name and (2) affixes. Affixation is used for the purpose of generating a number of name instances. In addition, name connectors are used to form a chain of patronyms spanning a number of generations. Note: the discussion found in (Auda 2003) and (Alkharashi 2009), will be used here and further enhanced by the utilizing regular expression notation. Any Arabic name could be formalized as:

\[ \text{Prefix[1,2]}? \text{Core}\text{+ Suffix}? \]  

(3.1)

Question marks indicate an optional field and the plus sign indicates a mandatory field. Thus, the above rule is interpreted as: an Arabic name is a core component preceded by optional [one or two] prefixes and followed by optional [one] suffix. Combinations of this rule are further detailed as follows:

1. **Core**: a person’s given name that could one of:
   
   - **Simple**: a single token name e.g. [محمد, mHmd, Mohammad].
Chapter 3: Arabic Language Characteristics

Most of Arabic names are common nouns or adjectives generated using the root-pattern Arabic morphological system. This category also covers names imported from other cultures and languages:

[braHy, Abraham], [موسى, mwsY, Moses], [عيسى, EysY, Jesus]

- **Compound**: multi-token name generated by annexation of two nouns. The first noun is a description of the person indicating full obedience, and the second noun is normally a definite form related to God e.g [عبد الله, Ebd Allah, servant of god]. Sometimes the second noun is related to a religious term as [صلاح الدين, SlAH Aldyn, goodness of religion]. Substituting into formula 1.1 leads to:

Prefix[1,2]? (Simple | Compound)+ Suffix?  

(3.2)

2. **Prefix**: two types of prefixes could precede an Arabic name:

- **Connected**: There is one common prefix used in family names; [الـ, Al, the] to link a person to his family e.g “AlmHmd”, formed by attaching the definite article “Al” to a person name “mHmd”.

- **Non-connected**: Two types of prefixes used for different purposes:
  
  - **Family prefix**: [آل, |l, family of] commonly written without madda (ı), “Al” and [ذوي,*wy, family of] are articles used to indicate a family name, which are similar in purpose to the connected family prefix.

  - **Kunai prefixes**: [أبو, >bw, father of] and [أم,>m, mother of] are used to form a honorific name meaning “the father or mother of someone called Kunai”. The selection is based on the gender of the person e.g., [bw mHmd, father of Mohammad], [m mHmd, mother of Mohammad]. Commonly, people are named after their eldest child, father or mother in the community. However, some Kunais might violate this criteria based on some other factors. No rule of thumb exists for the usage of Kunia; it is used sometimes as a replacement for the given name or the full name and
Chapter 3: Arabic Language Characteristics

sometimes, preceding or following the full name or the first name. Also, the prefix “>bw” has different inflection forms; changing the vowel “w” to “A” when accusative and to “y” when genitive. Substituting into formula 1.2 gives the following:

Prefix(connected | nonconnected)[1,2]? (Simple | Compound)+ Suffix?

3. **Suffix:** only one suffix can be attached to the name, which is the adjective indicator “y”, to form adjectives of relation or pertinence to a family as a last name and only applicable to simple names. Also, if the name is suffixed, it could only have the connected prefix. Substituting into formula 1.3 gives the following:

Prefix(connected)? (Simple)+ Suffix(y)?

(3.3)

Last names could be derived according to: a person's profession, name of a person's tribe of birth or family lineage, or place of residence or birth. It could also be a nickname based on personal attributes.

A name connector is used to form a patronymic or series of patronymics spanning a number of generations. This is almost similar to the “Mc” prefix used in Scottish names, with the one difference that they are used only with family names in Scottish while they could be used at different levels in Arabic. Two connectors are used in Arabic, that are based on the gender. [إبن ,>bn ,son of] is the masculine connector and [إبنة ,>bnp, daughter of] is the feminine. Name connectors are not part of the name, except in some rare cases of last names e.g [زين العابدين بن علي , Zyn AIEAbdy >bn Ely, Zine El Abidine Ben Ali] where token “>bn” here in not a connector but rather part of the last name, exactly as with the of prefix “Mc” in Scottish family names.

3.6.1.2 **Location and Organization Names**

Organization names commonly start with an organization prefix followed by one or more noun phrases connected with the conjunction [و ,w ,and] then optionally followed by
another noun phrase connected with a coordinating conjunction [ـ,ـ,ـ]. The noun phrase could be a single or compound noun (annexed) optionally followed by one or more adjectives. The main noun in the first noun phrase could be a common or proper noun (person or location). The organization prefix could also be prefixed with a determiner [ـــ,ـــ,ـــ] to produce a definite form. In that case, the prefix is followed by an adjective rather than a noun. To formalize, the form of organization names follow this rule:

(Prefix1?(Prefix2))? (noun phrase | adjective)+ (AND(noun phrase))? (FOR(noun phrase))? 

When the organization prefix is not in definite form:

(Prefix2)? (noun phrase)+ (AND(noun phrase))? (FOR(noun phrase))? 

When prefixed with “ـ”:

(Prefix1(Prefix2))? (adjective)+ (AND(adjective))? (FOR(noun phrase))? 

The main difference between person and organization names is that the latter names follow the language’s morphological and syntactic rules to a high extent and sometimes have nested structures.

In the above rule, Prefix2 represents common nouns serving as organization designators with the gender feature. As in English, these triggers are sometimes dropped if clear from the context. When an adjective is modifying a noun, they will agree on all morphological features; definiteness, number, gender. It is fairly complex to determine if the adjective is modifying the preceding noun or the organization prefix.

The connector “ـ”, proclitic “ـ” in Arabic, is used to specify the organization’s speciality, industry or business field. The following noun phrase has its own morphological agreement independent of the main noun phrase. This feature of internal clitic attachment within the phrase is special to organization names.

Location names are relatively simpler than organization names but not when they are named after people. They start with a location prefixes, e.g., city, followed by a proper noun. Most of the prefixes are in feminine form. However, unlike organizations, they do not morphologically agree with following nouns.
Chapter 3: Arabic Language Characteristics

3.6.2 Internal patterns in Proper Nouns

Most Arabic proper nouns are generated in the same manner as any other Arabic words using the root-pattern morphological system. For instance, an experiment by (Alkharashi 2009), concluded that person names are mostly generated from Arabic roots, in the same way as any other word categories. It was found that only 16 Arabic patterns contribute to the production of more than 50% of Arabic person names. The root “Hmd” alone produced 146 names. That does not apply to foreign names imported from other cultures and also transliterated names.

Proper nouns cannot have proclitics (at the end of the word) attached to them and enclitics could be attached only to the head token of the proper noun (person and locations) chunk, whereas they could slotted within organization name components. Moreover, some grammatical categories cannot be part of the proper noun such as: verbs, relative or personal pronouns.

According the previous section, person names have internal patterns such as connectors or Kunia prefixes and name connectors. Location and organization names exhibit the presence of prefixes.

3.6.3 Challenges in Proper Noun Detection and Classification

Proper noun detection in Arabic is far more difficult than in English because of the features discussed above. However, the challenges do not arise only from the nature of NEs, but actually from the previous analysis levels as well. It is a cumulative effect generated from many other sources, such as morphology and the common typing and writing practices of Arabic text.

The sources of challenges affecting the task with detailed descriptions are as follows:

3.6.3.1 Orthography

The lack of any orthographic feature marking Arabic proper nouns is considered to be the most affecting feature. This problem is expanded in MSA with the lack of diacritic marks.
which adds more ambiguity to the task. The diacritical marks would partially solve the problem, as some ambiguous tokens have exact diacritical marks regardless of whether they represent a proper noun or any other word category. However, it will help in narrowing the ambiguity space e.g. [حَمِدَ, Humida, to be thanked] in classical Arabic has only one possible POS analysis, which is VBN. In MSA, the diacritics are removed, producing the bare form حمد which have four possible analyses as shown in Table 3.4:

<table>
<thead>
<tr>
<th>Classical Arabic form</th>
<th>Trans</th>
<th>Classes</th>
<th>MSA form</th>
<th>Diacritized forms</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>حُمِدَ</td>
<td>Humida</td>
<td>VBP</td>
<td>حمد</td>
<td>حُمِدَ</td>
<td>VBP</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>حَمَدَ</td>
<td>VBD</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>حَمْدَ</td>
<td>NNP</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>حَمْد</td>
<td>NN</td>
</tr>
</tbody>
</table>

Table 3.4: Analysis of “Hmd”, diacritized and non-diacritized

3.6.3.2 Common writing practices MSA

Most common spelling errors in writing originated from Arabic orthography. The non-concatenated feature of some Arabic characters often causes spelling confusion to the writer. Two types of mistakes could happen after a non-concatenated letter: inserting a space into a word or dropping the space between two words. The insertion of a space happens when one of these characters falls into the middle of a name; for example a space delimiter could be mistakenly inserted in e.g [محمود, mHmwd, Mahmood], given that “w” is a non-concatenated letter. Using Google search, there were 176 million results returned for the correct spelled name “mHmwd” and 40k results for the misspelled variant “mHmw d”. Here, the output of tokenizing this name will be incorrect as it would produce two tokens instead of one. The reverse of this problem, dropping a space, is common in Kunia and compound names. In Kunias, the Kunia prefix “>bw” ends with “w” causing writers to
attach it directly to the succeeding core name, forming one word instead of the correct two
token name. Using Google search, the Kunia name [أبو محمود, Abu Mahmood] was searched to find out the magnitude of that error practice. Searching for “>bw mHmd”, which is the correct form, returned 19 million results and the wrong form, when space was dropped “>bwmHmd” returned 1.6 million results. In the latter case, a single word is produced which is not a name.

Also in compound names, the first noun usually ends with a non-concatenating letter. The compound name [عبد الله, Ebd Allh, Abdullah] has two components “Ebd” and “Allh”. The first component ends with the letter “d” which is a non-concatenating letter. Using Google search, the correct form “Ebd Allh” returned 55 million results while the erroneous one “EbdAllh” surprisingly returned almost double the correct form, 97 million results. The effect of these errors is crucial; the tokenization process of names that rely on spaces will not capture that phenomenon.

Shape similarities between some letters would cause the dots of some letters and hamza to be treated as diacritics and thus dropped in MSA, which causes more ambiguity. This practice would be more severe when producing a new word with a different grammatical class. The letter [ي، Y] is sometimes written with two dots below producing another letter [ي، y]. One of the most common names “Ely” is written as “ELY”, which is a preposition.

Sometimes, the reverse could happen by adding dots to a letter. Searching for the name [عيسى, EysY, Jesus] returned 43 million results using Google search engine while “Eysy” returned 1.25 million. This is less severe as the two spelling variation are always interpreted as names, as the new form will always be a name.

Another instance of writing errors is dropping dots from the feminine marker, which sometimes is more severe when it produces a morphologically ambiguous word. This happens in the case of dropping dots from the feminine indicator [ة, p] producing [ه, h] variation of spelling. This is relevant to person names as most of female names end with a feminine marker e.g. [وردة, wrdp, rose or a person name Wardah]. When the dots are dropped it generates [ورده, wrdh], which has number of segmentations, e.g., w+rd+h or
wrd+h. Each segmentation has a number of morphological analyses and none of them is a person name.

### 3.6.3.3 Morphology

Proper nouns are subject to zero or more proclitic attachment, requiring a preprocessing analysis to find the token representing the name. It is worth mentioning that clitics are normal Arabic letters, leading to difficulty in finding the lexical token of the name. This makes the Arabic NER task more complicated, as the system needs to address such a problem prior to the core task. In some cases, more ambiguity is introduced when attaching clitics to a non-name instance generates a valid Arabic verb; for example [عد, Ed, counted] when it is preceded by a conjunction “w” produces “wEd”, which a feminine person name. In this case, it is not only a semantically ambiguous word but also morphologically ambiguous as it has a number of segmentations each with a different POS tag.

In Arabic, the lack of an indefinite article makes detecting names in the form of a noun harder to disambiguate. In English, if a reader encounters the word “ROSE” in upper case text, it would generally be easily disambiguated since the noun context is different from a name context; it will be normally preceded with the article “a” if it is a noun.

Last, great the large magnitude of sparseness caused by the highly inflectional nature of Arabic requires more training data to capture enough contexts.

### 3.6.3.4 Naming system

The way names are generated makes them harder to distinguish from other non-name words. Most Arabic person names are generated in the same manner as other Arabic words, using the root-pattern morphological system investigated by (Alkharashi 2009). This could be used as a distinguishing feature. However, the nouns and adjectives account for a large portion of the open class category e.g. [صالح, SAIH, Saleh] could be a person name, adjective or 2 forms of a verb. Names imported from other cultures, such as Persian and Turkish, or Arabized foreign names, would be unique as they do not have the root-pattern feature.
Chapter 3: Arabic Language Characteristics

Multi-token and affixed, non-connected prefixes, impose a challenge in tokenizing Arabic names. The Space delimiter would not suffice in this case as it would break the unity of a single semantic concept.

The prefixes “>bw” “>m” and connectors “>bn” are common nouns that could be used in another context. Even if they are used in a names’ context, there is still some ambiguity as to whether they are part of the name or not. Nowadays, those connectors are rarely used in written Arabic and if used they commonly do not precede the final name.

The suffix “y” used in last names to produce an adjectival form is also a personal possessive pronoun. Although it is considered as a clue in last names, it could be easily confused with normal adjectives and possessive nouns.

Compound names are separated by a space delimiter, which is normally used to tokenize text. Common tokenization schemes with spaces would cause the name to be split into two tokens, losing its structure. When a name is a compound and also prefixed with the Kunia prefix, three tokens will be generated, which is even more critical.

As discussed in Section 3.9.1.1, some person names originate from the place of residence e.g [المصري, AlmSry, AlMasry], which could be a name and also a nationality. Moreover, some titles of Arabic person names could also be person names e.g. [الشيخ, Al$yk, The chief]. Also, [الله آية الله, Ayatollah] is a religious title in Iran and a person name in some parts of the Arab world.

3.6.3.5 Syntax

Given that adjectives follow nouns in Arabic, boundary detection in an organization name phrase is problematic. The reason is that organization names are usually noun phrases and could be followed by adjectives that could not be easily identified as part of the name or not and thus the ending boundary may be missed. Sometimes the following adjectives are modifying the prefix and sometimes the main noun.
Chapter 3: Arabic Language Characteristics

Although name designators are used at the start of organization and location names, which is of great assistance in recognizing them, it does not solve the problem, as these triggers are common nouns that could be used in any other context.

Given that triggers such as titles are normally nouns that may be followed by a number of adjectives, adjectives would cause the triggers to be further separated from the name, hence longer dependencies.

Annexation is used in Arabic to form compound nouns to indicate possessiveness. Given the wide spread of Arabic names in the form of nouns, full names could be a series of nouns. The lack of a possessive indicator in Arabic such as “of” and “’s” in English makes any sequence of ambiguous nouns a potential full name. For instance, the following name with no connectors and no titles is too ambiguous to detect:

[عمر فهد, Emr fhd] could be interpreted as “Fhd's age” or a person name “Emr Fhd”

Collecting proper noun action verbs would normally help in detecting the context of entities. However, this feature is affected by two syntactic feature of Arabic. First, the presence of an elliptic pronoun makes these verbs falsely indicate the presence of a proper noun while replaced with the dropped pronoun. Second, the free order language feature would detect the context but it would not be clear if the verb succeeds or precedes the proper noun.

The agreement in Arabic person names is on connectors and prefixes but not the word itself. Thus, this feature of Arabic person names would harm other processing levels as it would violate and break the morphological and syntactic structure of a sentence.

3.6.3.6 Scarcity of Arabic Resources

The scarcity of Arabic resources might be considered the most important factor of all. Linguistic resources are crucial in developing any NLP system. Given the challenges of Arabic NER, previous preprocessing stages have a great impact on the NER level. Resources in Arabic are very limited i.e. corpora, gazetteers and preprocessing tools with good coverage such as POS taggers; given that NER task is closely bound to preprocessing
Chapter 3: Arabic Language Characteristics

stages. Furthermore, the lack of standards in foreign names Arabization introduces more sparsity to the entities themselves as they are sometimes transliterated literally and sometimes phonetically.
Chapter 4   Overview of Named Entity Recognition

4.1 Definition

Named Entity Recognition (NER) is considered to be the most fundamental task of any IE system. It is described in (Leaman and Gonzalez 2008) as the most basic problem in automatic text extraction. Therefore, it has received extensive attention from the research community and has been the focus of many research efforts since the mid nineties. Events were organised as competitions to boost the NER field. Each of these events targeted a different domain, requiring a different definition of the task. They were driven by the standing challenges and needs. They started with the general domain, then turned to military reports and later to blogs, moving from a single to a multilingual framework. In each evaluation, tasks were defined and data provided to the participant, then evaluation of participating systems was carried out.

Here, we give a brief description of each evaluation and the NE definition task for each of them:

4.1.1 Message Understanding Conference (MUC)

The Message Understanding Conferences (MUC) was a series of conferences organized by the (U.S.) National Institute of Standards and Technology and the U.S. Department of Defense Advanced Research Projects Agency. These events were held between 1987 and 1998 as part of the TIPSTER Text program. MUC held seven evaluation conferences for Information Extraction systems. Each conference brought Information Extraction one step forward, setting more difficult tasks, and providing more resources for testing and evaluation.
Chapter 4: Overview of Named Entity Recognition

The task was to extract information about relevant events from newswire texts and use it to fill the slots in a scenario template. The domain covered was news text on terrorism then joint ventures (Chinchor 1998). The NER task was first introduced in MUC-6 in 1995 and then was again included in the last MUC-7. All MUC conferences focused on the English language except the Multilingual Entity Task (MET) that was run in parallel with MUC-7, targeting NE in the Chinese, Japanese and Spanish languages. The NE task defined three subtasks; ENMAX for proper nouns, TIMEX for temporal time and date and NUMEX for measurements including money. Details of each type and its corresponding subtypes with description are given in Table 4.1.

The Information Retrieval and Extraction Exercise (IREX) 1999 was oriented toward the Japanese language and included retrieval and entity extraction. Basically, it is similar to the MUC-NE and MET tasks. There were minor differences; "artifact" was added, which includes product names, names of services, etc.

<table>
<thead>
<tr>
<th>TYPE</th>
<th>Description</th>
<th>SUB-TYPE</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENMAX</td>
<td>proper names, acronyms, and perhaps miscellaneous other unique identifiers</td>
<td>PERSON</td>
<td>named person or family</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LOCATION</td>
<td>name of politically or geographically defined location</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ORGANIZATION</td>
<td>named corporate, governmental, or other organizational entity</td>
</tr>
<tr>
<td>TIMEX</td>
<td>&quot;absolute&quot; temporal expressions only</td>
<td>DATE</td>
<td>complete or partial date expression</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TIME</td>
<td>complete or partial expression of time of day</td>
</tr>
<tr>
<td>NUMEX</td>
<td>numeric expressions, monetary expressions and percentages; expressed in either numeric or alphabetic form.</td>
<td>MONEY</td>
<td>monetary expression</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PERCENT</td>
<td>Percentage</td>
</tr>
</tbody>
</table>

Table 4.1: MUC entity types

4.1.2 Conference on Computational Natural Language Learning (CoNLL)

CoNLL 2002 and 2003 were successors of the MUC evaluations, organized to advance work on language-independent frameworks. In contrast to MUC that covered IE in general, CoNLL was solely focused on the named entity recognition task. The task defined four classes, which are the ones in ENMAX of MUC, in addition to Miscellaneous names similar to artifact defined in IREX. The languages considered by CoNLL were Spanish and Dutch in 2002 and German and English in 2003.

4.1.3 Automatic Content Extraction (ACE)

The ACE program was initiated by National Institute of Standards and Technology (NIST) and attempts to take the NER task “off the page”. That is in the sense that the research objectives are defined in terms of the target objects (i.e., the entities, the relations, and the events) rather than in terms of the words in the text. NER as defined in MUC, is to identify those words (on the page) that are names of entities. In ACE, on the other hand, the corresponding task is to identify the entity so named. Reference resolution thus becomes an integral and critical part of solving the problem i.e. when referring to a person with his nationality; it is labeled as a PERSON.

The Entity Detection and Tracking (EDT) task includes the Mention Detection (MD) subtask that defined the same classes defined by CoNLL in addition to FACILITY such as “Empire State Building”. Also, Global Political Entity was added as a hybrid entity to cope with different interpretations of locations; sometimes a location plays the role of an organization.

In ACE 2004, VEHICLE and WEAPON classes were added to those of ACE 2003. Another important change was the introduction of entity subtypes, such as COUNTRY and DISTRICT subtypes of the LOCATION class. The PERSON class did not have any subtypes until ACE 2005. All entity details of the EDT task are shown in Table 4.2.
### Chapter 4: Overview of Named Entity Recognition

<table>
<thead>
<tr>
<th>TYPE</th>
<th>SUBTYPE</th>
<th>Class: Description</th>
<th>Mention Type: Descr.</th>
</tr>
</thead>
<tbody>
<tr>
<td>FAC (Facility)</td>
<td>Airport, Building-Grounds, Path, Plant, Subarea-Facility</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPE (Geo-Political)</td>
<td>Continent, County-or-District, GPE-Cluster, Nation, Population-Center, Special, State-or-Prov</td>
<td>SPC: A particular, specific and unique real world entity</td>
<td></td>
</tr>
<tr>
<td>LOC (Location)</td>
<td>Address, Boundary, Celestial, Land-Region-Natural, Region-General, Region-International, Water-Body</td>
<td>GEN: A kind or type of entity rather than a specific entity</td>
<td>NAM (Name): A proper name reference to the entity</td>
</tr>
<tr>
<td>ORG (Organization)</td>
<td>Commercial, Educational, Entertainment, Governmt, Media, Medical-Science, Non-Gov, Religious, Sports</td>
<td>NEG: A negatively quantified (usually generic) entity</td>
<td>NOM (Nominal): A common noun reference to the entity</td>
</tr>
<tr>
<td>PER (Person)</td>
<td>Group, Indeterminate, Individual</td>
<td>USP: An underspecified entity (e.g., modal/uncertain/…)</td>
<td>PRO (Pronominal): A pronominal reference to the entity</td>
</tr>
<tr>
<td>VEH (Vehicle)</td>
<td>Air, Land, Subarea-Vehicle, Underspecified, Water</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WEA (Weapon)</td>
<td>Biological, Blunt, Chemical, Exploding, Nuclear, Projectile, Sharp</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4.2: ACE2005 Mention types and attributes

---

10 Added in ACE 2004
4.2 NER annotation

In order to conduct any experimental analysis or build efficient systems, textual data would always be critical. The data is required to have some meta-data included to denote the phenomena to be studied. A set of textual data is called a corpus and adding annotation produces an annotated corpus. In (Leech 2005), the following were among the significant factors of corpus creation:

- Manual examination of corpus
- Automatic analysis of corpus
- Reusability of annotations
- Multi-functionality

Corpus annotation is very laborious work that normally require human linguist to label each word with its corresponding class. There have been various schemes for the annotation format of corpora for different NLP tasks. The selection of the scheme is driven by the definition of the task. The following are the main two schemes used in NER annotation:

4.2.1 Inline annotation

This kind of annotation is done by inserting entity tags directly into text and has two main formats:

4.2.1.1 SGML format

This kind of annotation was adopted by MUC which was done by inserting SGML tags into the text to mark up named entities. The markup will have the following form:

<ELEMENT-NAME ATTR-NAME="ATTR-VALUE" ...>text-string</ELEMENT-NAME>

Example: Yesterday, John William Adams met Frank in London UK.
Chapter 4: Overview of Named Entity Recognition

We will use abbreviations for NE classes for simplification;


The markup is defined in SGML Document Type Descriptions (DTDs), written for MUC-6 use by personnel at MITRE and maintained by personnel at NRaD. The DTDs enable annotators and system developers to use SGML validation tools to check the correctness of the SGML-tagged texts produced by the annotator or the system. Annotators used a software tool provided for MUC-6 by the SRA corporation to assist in generating the answer keys to be used for system training and testing.

4.2.1.2 Column-based format

This is a simpler form of annotation which places each word on a single line with its corresponding class delimited by tab or space. It is sufficient when there is no nested annotation required by the task definition, such as POS tagging. According to (Kudo and Matsumoto 2001), two schemes were used in text chunking, “inside/out” and “start/end”.

I. inside/out:

This was introduced in (Ramshaw and Marcus 1995) and later extended in (Sang and Veenstra 1999). It is based on annotating a token with its position (P); inside a named entity or outside, and attaching that position to its entity class (CCC). The full label would have the form (P-CCC). One generic outside class was used for all (O). To mark entity boundaries, (P) would take (B) at the beginning boundary of an entity in the IOB scheme and (E) marks the ending boundary in the IOE scheme. The two schemes have the following variants:

a. IOB

- IOB1: assigns B only if followed immediately by another token of same entity type.
- IOB2: assigns B whenever starting a new entity; head entity
Chapter 4: Overview of Named Entity Recognition

b. IOE

- IOE1: assigns E for end token if immediate preceding another token of same type.
- IOE2: assigns E whenever ending an entity.

II. Start/end

In this scheme, S was added in addition to all tags used in inside/out schemes to represent single token entities. B and E tags were assigned regardless of the preceding token class.

An illustrating example showing all discussed schemes is given in table 4.3:

Example: Yesterday, John William Adams met Frank in London UK.

<table>
<thead>
<tr>
<th></th>
<th>IOB1</th>
<th>IOB2</th>
<th>IOE1</th>
<th>IOE2</th>
<th>Start/End</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yesterday</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>,</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>John</td>
<td>I-PER</td>
<td>B-PER</td>
<td>I-PER</td>
<td>I-PER</td>
<td>B-PER</td>
</tr>
<tr>
<td>William</td>
<td>I-PER</td>
<td>I-PER</td>
<td>I-PER</td>
<td>I-PER</td>
<td>I-PER</td>
</tr>
<tr>
<td>Adams</td>
<td>I-PER</td>
<td>I-PER</td>
<td>I-PER</td>
<td>E-PER</td>
<td>E-PER</td>
</tr>
<tr>
<td>Met</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>Frank</td>
<td>I-PER</td>
<td>B-PER</td>
<td>I-PER</td>
<td>I-PER</td>
<td>S-PER</td>
</tr>
<tr>
<td>In</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>London</td>
<td>I-LOC</td>
<td>B-LOC</td>
<td>E-LOC</td>
<td>E-LOC</td>
<td>S-LOC</td>
</tr>
<tr>
<td>UK</td>
<td>B-LOC</td>
<td>B-LOC</td>
<td>I-LOC</td>
<td>E-LOC</td>
<td>S</td>
</tr>
</tbody>
</table>

Table 4.3: NER annotation schemes example
Chapter 4: Overview of Named Entity Recognition

Different schemes were used without a decision as to which one is best (Collobert et al. 2011). The most widely used is IOB2 but classifiers built on different schemes have given better results as in (Kudo and Matsumoto 2001).

IOB2 was adopted for CoNLL evaluations and data consists of two columns separated by a single space. Each word has been put on a separate line and there is an empty line after each sentence. The first item on each line is a word and the second is a named entity class. The tags used in CoNLL were: person names (PER), organizations (ORG), locations (LOC) and miscellaneous names (MISC).

4.2.2 Standoff annotation

When multiple layers of annotation are required, such as co-reference, it would be impractical to have all layers within the text. Standoff annotation is used to separate text from annotation by having the text in one file and annotation indexes in another file. It is a more efficient way to collaborate and exchange and analyse. It was adopted in ACE since it required a complex representation of IE tasks.

Example text: Yesterday, John William Adams met Frank in London UK.
4.3 NER System Scoring

Evaluating NER systems is performed by comparing the output of the system to the human annotated test data. Evaluation schemes used in NER share the same measures imported from Information Retrieval. However, they use different methods in calculating these measures. It is very important to have a close look at each evaluation method since some of
the participating systems in the three events; (MUC, CoNLL and ACE) discussed later. It would give a better judgment on the performance of each.

**Precision** is the percentage of correct positive predictions returned by the system. It is computed as the ratio between the number of NEs correctly identified by the system True Positives (TP) and the total number of NEs returned by the system. The precision is calculated by dividing TP by the sum of TP and false positives (FP)

$$Precision = \frac{TP}{TP + FP} \quad (4.1)$$

**Recall** indicates the percentage of positive cases recognized by the system. It is computed as the ratio between the number of NEs correctly identified by the system (TP) and the number of NEs that the system was expected to recognize. Thus, Recall is the number of (TP) divided by the sum of (TP) and false negatives (FN)

$$Recall = \frac{TP}{TP + FN} \quad (4.2)$$

**F-measure** is the common weighted harmonic mean between Precision and Recall defined as:

$$F = \frac{(\beta + 1)Precision \times Recall}{\beta \ (Precision + Recall)}$$

where $\beta$ is the weighting factor.

When the Precision and Recall have the same weight $\beta=1$, it is called F1:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (4.3)$$
4.3.1 MUC

The MUC scoring scheme gives credit if the system was able to detect a named entity (TEXT) regardless of its type, and even partial detection is credited. Also, it gives credit if the system is successful in assigning the correct class, regardless of span (SPAN). That way it is testing all kinds of errors that the system might produce. Each correct SPAN gets 1 point and also TEXT gets 1 point.

For example consider the sentence that we have previously used for the annotation schemes example.

Yesterday, John William Adams met Frank in London UK.

<table>
<thead>
<tr>
<th>Key</th>
<th>System</th>
</tr>
</thead>
</table>

Table 4.4: Key and system output of sentence example in MUC
Chapter 4: Overview of Named Entity Recognition

<table>
<thead>
<tr>
<th>Key</th>
<th>System output</th>
<th>Correct criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;ENAMEX TYPE=&quot;PER&quot;&gt;John William Adams&lt;/ENAMEX&gt;</code></td>
<td><code>&lt;ENAMEX TYPE=&quot;PER&quot;&gt;John William&lt;/ENAMEX&gt;</code> Adams</td>
<td>None</td>
</tr>
<tr>
<td><code>&lt;ENAMEX TYPE=&quot;PER&quot;&gt;Frank&lt;/ENAMEX&gt;</code></td>
<td><code>&lt;ENAMEX TYPE=&quot;PER&quot;&gt;Frank&lt;/ENAMEX&gt;</code></td>
<td>Type &amp; Text</td>
</tr>
<tr>
<td>in</td>
<td><code>&lt;ENAMEX TYPE=&quot;GPE&quot;&gt;in&lt;/ENAMEX&gt;</code></td>
<td>None</td>
</tr>
<tr>
<td><code>&lt;ENAMEX TYPE=&quot;LOC&quot;&gt;London&lt;/ENAMEX&gt;</code></td>
<td><code>&lt;ENAMEX TYPE=&quot;PER&quot;&gt;London&lt;/ENAMEX&gt;</code></td>
<td>Text</td>
</tr>
<tr>
<td><code>&lt;ENAMEX TYPE=&quot;LOC&quot;&gt;UK&lt;/ENAMEX&gt;</code></td>
<td><code>&lt;ENAMEX TYPE=&quot;LOC&quot;&gt;UK&lt;/ENAMEX&gt;</code></td>
<td>Type &amp; Text</td>
</tr>
</tbody>
</table>

Table 4.5: Result analysis

We have 3 TEXT and 2 TYPES correct and, using equations 4.1, 4.2 and 4.3:

\[
\text{Precision} = \frac{5}{5 + 5} = \frac{1}{2} = 50\%
\]

\[
\text{Recall} = \frac{5}{5 + 3} = \frac{5}{8} = 63\%
\]

\[
F1 = \frac{2 \times 0.5 \times 0.63}{0.5 + 0.63} = 56\%
\]
4.3.2 CoNLL and IREX

In these evaluations, there was no partial credit but rather, an exact match of type and text is required. Table 5 shows the output of the same sentence:

<table>
<thead>
<tr>
<th>KEY</th>
<th>System Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yesterday</td>
<td>O O</td>
</tr>
<tr>
<td>,</td>
<td>O O</td>
</tr>
<tr>
<td>John</td>
<td>B-PER B-PER</td>
</tr>
<tr>
<td>William</td>
<td>I-PER I-PER</td>
</tr>
<tr>
<td>Adams</td>
<td>I-PER O</td>
</tr>
<tr>
<td>Met</td>
<td>O O</td>
</tr>
<tr>
<td>Frank</td>
<td>B-PER B-PER</td>
</tr>
<tr>
<td>In</td>
<td>O B-GPE</td>
</tr>
<tr>
<td>London</td>
<td>B-LOC B-PER</td>
</tr>
<tr>
<td>UK</td>
<td>B-LOC B-LOC</td>
</tr>
</tbody>
</table>

Table 4.6: CoNLL key and system output, errors underlined

Using the same equations 4.1, 4.2 and 4.3:

\[
Precision = \frac{2}{2 + 3} = \frac{2}{5} = 0.4
\]

\[
Recall = \frac{2}{2 + 2} = \frac{2}{4} = 0.5
\]

\[
F1 = \frac{2 \times 0.4 \times 0.5}{0.4 + 0.5} = \]

85
4.3.3 ACE

This evaluation is based on a complex algorithm where different named entities have different weights in calculating the $EDR_{value}$. The EDR_value score for a system is defined to be the sum of the values of all of the system's output entity tokens, normalized according to the sum of the values of all reference entity tokens. The maximum possible EDR value score is 100 percent. The value of each system token is based on its attributes and on how well it matches its corresponding reference token. The value of a system token is defined as the product of two factors that represent both the inherent value of the token and how accurately the token’s attributes are recognized and the token’s mentions are detected.

4.4 NER literature review

Previous work on NER started much earlier for English than for Arabic. The work on Arabic was very limited until the past decade. In this chapter, an overview of NER research will be discussed, starting with English language, as it preceded other languages’ efforts to tackle the NER problem. Then, a comprehensive overview of previous efforts in the Arabic language will be discussed. Given that similar challenges to Arabic NER might be present in other languages and domains, a brief overview of relevant efforts will be discussed, specifically targeting the lack of capitalization and ambiguous tokens that could serve as NEs or nouns and adjectives. The motivation for this is the fact that it would assist in choosing the best features and approach to tackling Arabic NER.

There have been a number of classification schemes presented in the literature for NER approaches. Borthwick subdivides NER approaches based on the mechanism of providing the system with the required knowledge to: rule-based, automated and hybrid-based (Borthwick 1999). The automated approach is where machine learning algorithms are used to build the classification model from labelled data. Further classification was introduced in (Nadeau and Sekine 2007) which states that NER studies differ based on a number of factors; language, genre, learning method and feature space. (Wattarujeekrit and Collier 2005) classified NER approaches as lookup, rules and machine learning.
In this discussion, NER approaches will be classified similarly to Borthwick, except that the automated approach will be referred to as corpus-based.

4.4.1 English NER

4.4.1.1 Rule-based

This approach relies on handcrafted rules that require strong linguistic skills. One of the earliest works to approach proper nouns in text was explained in (Kuhns 1988). The focus of the system was on parsing news articles then classifying and extracting relevant details. Another example, a rule-based NER system, described in (McDonald 1996), was developed using internal and external evidence. Internal evidence is found within the name, e.g., company designators such as Co., and external evidence is found in the context, such as personal titles. The rules of this system focused on organizations first as they might contain person and location names. These two systems tackled the problem of proper noun identification prior to the definition of the NER task in MUC.

Rule-base systems have performed well in the MUC-6 and MUC-7 competitions. The best participating system in MUC-6 was NameTag, designed by SRA. The same system (then managed by IsoQuest) ranked second in MUC-7. The second best system in MUC-6 was also built using this approach; FASTUS by SRI. More details of the three participating systems are given in Table 4.7.

We have provided tables of the best three systems in each NER evaluation conference, as it would not be accurate to list them in one table, given the different parameters of each evaluation event.
Table 4.7: MUC best three systems on the NE task

### 4.4.1.2 Corpus-based

As the name indicates, corpus-based methods rely on large corpora to address an NLP task. With the advent of large linguistic corpora annotated with named entity classes, it was feasible to use machine learning techniques to tackle the NER challenges. The approach is based on statistics drawn from large corpora and the task of the machine learning algorithms, such as the ones discussed in chapter 2, is to find the most probable NE outcomes (classes) of each word in a text. Those systems are specifically known as supervised learning as they are trained on pre-annotated data. Supervised learning systems are dependent on the availability of data. Recent studies on NER mostly follow this approach.

One example is IdentFinder by BBN, covered in (Bikel et al. 1997) and (Bikel et al. 1999), which participated in MUC-6 and MUC-7, ranking third in both. It was part of two different IE systems used in the two MUC events; PULM and SIFT. Identfinder is based on Hidden Markov Models (HMM) where the states of the HMM were organized into regions, one for each entity class. One region is added, which is NOT-A-NAME. For each region, a statistical bigram model is used to compute the likelihood of the state sequence, using the most indicative internal features. Using a similar modeling technique, another system was
Chapter 4: Overview of Named Entity Recognition

enhanced with global features and triggers, yielding a performance of F1 96.6% and F1 94.1% on MUC-6 and MUC-7 data, respectively (Zhou and Su 2002).

The technique of Maximum Entropy Models was also applied to NER described, in (Chieu and Ng 2002a). This latter model was improved by a set of global features extracted from other occurrences of the word. The improvement gained by these features was F1 2.5% on MUC-6 data.

All systems that participated in CoNLL 2002 and 2003 were corpus-based, using various machine learning algorithms. Interestingly, the top systems were very close to the human annotator tagging accuracy. Some of the participating systems used a single learning algorithm, while others combined more than one. The learning algorithm itself is not the only factor that governs the performance but also feature sets and how features are employed, as has been discussed in chapter 2.

In CoNLL 2002, (Carreras et al. 2002) combined a number of binary classifiers built with the same learning algorithm (Decision Trees). Combining a number of weak classifiers is called Boosting. Three classifiers were used to classify words as beginning, inside or outside of NE. Two other classifiers focus on the boundaries of the NE phrase. The third one is a global classifier that combines the latter two classifiers. The system ranked first in both languages of CoNLL 2002. Details of the best three participating systems are given in Table 4.8.

In CoNLL 2003, (Florian et al. 2003) combined four learning algorithms with the same very rich resources such as large gazetteer and the output of two other NER systems. Also, they incorporated POS tagging information and word’s leading and trailing character. Combining different classifiers produced by different algorithms efficiently led to taking advantage of the strengths of each one. The system ranked first in both languages covered in ConLL 2003. Details of the best three participating systems are given in Table 4.9.

Although supervised learning is proving successful, the scarcity of training data in some languages and domains triggered the emergence of two recent methods; semi-supervised learning and unsupervised learning. Semi-supervised learning uses bootstrapping techniques
to capture context incrementally starting with a few seed examples whereas, unsupervised learning uses clustering of similar context (Nadeau and Sekine 2007).

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>F1 (S,D)</th>
<th>Rank (S,D)</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boosting (DT)</td>
<td>81.39 (S)</td>
<td>1 (S)</td>
<td>(Carreras et al. 2002)</td>
</tr>
<tr>
<td></td>
<td>77.05 (D)</td>
<td>1 (D)</td>
<td></td>
</tr>
<tr>
<td>TBL</td>
<td>79.05 (S)</td>
<td>2 (S)</td>
<td>(Florian 2002)</td>
</tr>
<tr>
<td></td>
<td>75.36 (D)</td>
<td>3 (D)</td>
<td></td>
</tr>
<tr>
<td>Bootstrapping</td>
<td>77.15 (S)</td>
<td>3 (S)</td>
<td>(Cucerzan and Yarowsky 2002)</td>
</tr>
<tr>
<td>Boosting (MEM)</td>
<td>75.36 (D)</td>
<td>2 (D)</td>
<td>(Wu et al. 2002)</td>
</tr>
</tbody>
</table>

Table 4.8: Best participants in CoNLL 2002

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>F (E/G)</th>
<th>Rank (E/G)</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBL,HMM,RRM,MEM</td>
<td>88.76 (E)</td>
<td>1 (E)</td>
<td>(Florian et al. 2003)</td>
</tr>
<tr>
<td></td>
<td>72.41 (G)</td>
<td>1 (G)</td>
<td></td>
</tr>
<tr>
<td>MEM</td>
<td>88.31 (E)</td>
<td>2 (E)</td>
<td>(Chieu and Ng 2003)</td>
</tr>
<tr>
<td>HMM, MEM</td>
<td>86.07 (E)</td>
<td>3 (E)</td>
<td>(Klein et al. 2003)</td>
</tr>
<tr>
<td></td>
<td>71.90 (G)</td>
<td>2 (G)</td>
<td></td>
</tr>
<tr>
<td>WINNOW</td>
<td>71.27 (G)</td>
<td>3 (G)</td>
<td>(T. Zhang and Johnson 2003)</td>
</tr>
</tbody>
</table>

Table 4.9: Best participants in CoNLL 2003

### 4.4.1.3 Hybrid

The term *hybrid* in this discussion is concerned with the combination of rule-based and corpus-based methods; this should not be confused with the hybrid technique used when two or more machine learning algorithms are combined to build the classification model.

---

11 S denotes Spanish language and D for Dutch language
12 E denotes English language and G for German language, “x” indicates absence
MEM was used to combine a number of rule sets based on their probability of making correct predictions on the training data (Mikheev et al. 1998). Processing was performed in stages starting with the highest probability rules, sure-fire rules. The performance of the system developed by the Language Technology Group (LTG) of the University of Edinburgh was the best ranked in the MUC-7 evaluation with F1 93.4%. Another hybrid system that also used MEM was the MENE system, which participated in MUC-7 and was ranked fourth. It was combined with the three weakest systems in MUC-7 and achieved results of F1 97.1%, outperforming the single best system in MUC-7 (by IsoQuest), with comparable performance to human annotator performance (Borthwick et al. 1998a).

A comprehensive survey was presented in (Nadeau and Sekine 2007) covering the main aspects of NER over a period of 15 years, from 1990 to 2006.

4.4.2 Non-Capitalized English NER

NER ambiguity exists in almost all languages and domains with difference magnitudes. The problem of NER without capitalization is not specific to the Arabic language. It has a wide spread in many languages and even the standard form of the English language may have that feature, for example English text all in upper case. With no capitalization, it is purely a lexical problem with highly critical data size. One other domain that represents a great challenge in English NER is social media content and weblogs, where most content is written as informal text, often dropping capitalization from proper nouns.

Another of these domains is the output of Automatic Speech Recognition systems, which lack both case information and punctuation. That format is called the SNOR formats, which also has numbers spelled as words.

If case information is not available, the NE task becomes significantly harder, even for humans.

An experiment performed by (Kubala et al. 1998) with Identfinder of BNN Technologies on upper case text reported a degradation of less than F1 2% points on two different corpora.
The result of Identfinder on MUC-6 data was F1 94.9% on mixed case text and F1 93.6% on upper-case text. The closest to Arabic is upper case text with punctuation where the difference was 2 % in (Miller et al. 1999).

The effect of the lack of case information was reported on OCR output by (Miller et al. 2000), where Identfinder was evaluated and the performance is still above 90%, with only 2.3% degradation in performance due to missing case, but including the presence of punctuation.

In (Chieu and Ng 2002b), the problem of missing case was approached by first training the MEM algorithm on two versions of human annotated corpus, mixed and upper case. Then, the resulting models were used to tag unlabelled data. The output of the two models is compared and only the human tagged ones are considered if they differ. After that, the algorithm was trained on original and machine labelled corpora, giving more weight to the human tagged examples. The result was significant on their testing data, improving the accuracy by 3%.

(Srihari et al. 2003) proposed an approach of restoring orthographic features of text in an attempt to convert to mixed case from the upper case text that exists in degraded documents such as the output of speech recognition systems or emails. The idea was to split the process into case restoration and then to apply normal NER. That approach is in contrast to the common approach of retraining an NE tagger on the degraded documents themselves. Three orthographic tags for each word are defined in this model: (i) initial uppercase followed by lowercase, (ii) all lowercase, and (iii) all uppercase. A maximum entropy based Hidden Markov Model (MEHMM) was used to build the model. N-gram context and long distance co-occurrence evidence were used as features of the system. Accuracy degradation in the NER process was 2% from upper to mixed case and claimed to be the best in the literature.

This approach could be applicable to Arabic, considering that the NNP tag is serves as the capitalization in POS tagged text. So the task would start with NNP tagging (correction) then normal NER would follow.
Chapter 4: Overview of Named Entity Recognition

The magnitude of ambiguity in English single case text is not exactly identical to that of Arabic, as ambiguous entities (proper nouns in the form of a common noun or adjective) are rare in English. Moreover, no system has been designed specifically for text without case information.

Another domain that represents a great challenge for the research community is biomedical text NER with a vast and rapidly increasing amount of data available. More details of the difficulties in biomedical domain were discussed in (Wattarujeekrit and Collier 2005). The domain NEs are highly generative and hard to tokenize.

(Leaman and Gonzalez 2008) used CRF to tackle the problem by exploiting internal features such as lexical, leading/trailing character ngrams, POS tagging, shallow parsing and lemmatization (converting to base form). There was no measure of ambiguity included in their publication and reported accuracy was F1 81.96% on the BioCreative Corpus, outperforming other systems. Interestingly, their system outperforms other systems without the use of gazetteers and deep syntactic parsing. They noticed that the IO annotation scheme gives better results than IOB scheme, which might also be the case with Arabic.

A more recent work was detailed in (Chowdhury and Lavelli 2010) where a feature-specific approach was used for detection of disease mentions. The features used were orthographic, POS tagging information generated by the GENIA tagger and patterns. The system was able to outperform (Leaman and Gonzalez 2008) possibly due to being specifically tailored for disease mentions.

In another study, the CRF technique was improved to capture long distance dependency by linking similar words and linking words having typed dependencies (J. Liu et al. 2010).

The problem in the English Biomedical domain is more an identification problem. This is not the case with disambiguation of Arabic general text because a biomedical entity could rarely serve as a verb or adjective.
4.4.3 Non-capitalized Languages NER

Most Asian languages such Chinese, Urdu, Farsi, Hebrew, and African's Ethiopian lack proper noun capitalization. However, research into NER in these languages is far behind that of European languages. South Asian languages also suffer from ambiguous proper nouns as in Arabic. Some of these languages also share the free order syntactic feature with the Arabic language. Moreover, these languages suffer from a lack of NER resources such as corpora and gazetteers.

(Goyal 2008) used CRF on the Hindi language with internal and external features; word N-gram, POS tagging, morphology, chunking, affixes lists that indicate NEs, trigger words and stemming. The processing was split into recognition and then classification. The reported accuracy was F1 64.3 % on the NLPAI 2007 NER contest Corpus. When the system was trained and tested on the CONLL 2003 English data with only language-independent features, the performance was relatively higher at F1 75%, even if random person entities were replaced with common nouns. The reason, Goyal concluded, was that the data set was better quality in terms of POS tagging and chunking, which substitutes for missing capitalization. However, when POS tagging information was generated on upper case text, the performance was comparable to that of the Hindi experiment.

(Saha et al. 2008) used MEM on the Hindi language and was able to improve the performance from 75 to 81% using context patterns. These patterns were induced semi-automatically by using seed entities of each NE class on the corpus and collecting the most frequent context. The patterns with highest coverage were added as a feature in the system.

A combination of three machine learning algorithms (MEM, SVM, CRF) was used in (Ekbal and Bandyopadhyay 2010) to tackle NER in the Bengali language. Very rich features both language dependent and independent were employed. Unlabelled data was also used to provide the system with more context patterns. Significant improvement was achieved, shifting the accuracy from a baseline of F1 76% using only language-independent features to F1 92.55% with all proposed features.
A recent survey of Indian languages which share almost the same challenges as Arabic is presented in (Sharma et al. 2011). The survey discussed methods, and the results of approaches to the main five south Asian languages were discussed.

### 4.4.4 Arabic NER

Previous work on Arabic NER was almost entirely focusing on news text, including this thesis, and the main efforts are as follows:

#### 4.4.4.1 Rule-based

TAGARAB was developed by (Maloney and Niv 1998) using the NetOwl pattern matching engine. Their system contains two modules, a morphological analyzer to generate word type and morphological features, and a name finder, which uses a word list with morphological features and pattern-actions rules. Their system was tested on 14 articles of the AlHayat newspaper with 3214 tokens. The accuracy scored was F1 90% on the training set and F1 85% on the test set. Their system was highly affected by the removal of the morphological features, yielding F1 75%.

(Abuleil and Evens 2004) worked on NER by splitting the task into three phases; possible NE phrase finder, relationship graph of the phrase and NE identification rules. Finding a possible phrase is guided by a set of keywords and verbs that surround person names with a fixed window distance. In the relationships graph technique, they represented words as nodes and relationship as edges with weights assigned based on the number of times these words co-occur in name phrases. The system was built for four classes; person, organization, location, event (e.g. conference) and was tested on 500 articles from AlRaya newspaper with 533 entities. Their system was able to extract 78.4% of named entities correctly. Abuleil further developed a hybrid system by supporting the system with statistics from the training corpus. The system was tested on 3347 tokens test set and the accuracy was 97%, we could not confirm if it was on a blind set (Abuleil 2006).
Chapter 4: Overview of Named Entity Recognition

(Mesfar 2007) used the NooJ platform to build an NER system targeting all MUC entities. The system exploits a morphological analyzer, gazetteer, triggers and rules to identify NEs. It was evaluated on part of the Arabic version of “Le Monde Diplomatique” giving an average accuracy of F1 87%.

Shaalan and Raza have developed a rule-based system for person name recognition, extended later for all named entities defined by MUC (Shaalan and Raza 2008). The system relied on dictionaries and grammars in the form of regular expressions. They reported accuracy on their corpus used to write the rules was above 90% on average for all MUC named entities. The system’s dictionaries have deeper semantic information, e.g., jobs, positions, geographic, political features, considering how named entities are formed.

In (Traboulsi 2009), local grammars were used to address Arabic NER, finding consistent structures of person names that occur frequently in news text. Corpus-based techniques were used to induce grammars for proximity of Reporting Verbs (declared, said, etc.) which are considered to be sufficiently frozen as they contain slots that can only be filled in with specific types of linguistic units. Frequency analysis, Collocation Analysis and Concordance Analysis were conducted on ArabiCorpus 13 to collect the data required for building grammars based on reporting verbs and the function words that collocate with them.

(Al-Shalabi et al. 2009) approached Arabic NER in two stages, finding NEs in context then extracting NEs from that context. They used special keywords (titles and designators) and special verbs for each class to write the system’s rules. Their algorithm employed function words and word patterns. It was tested on 20 articles from the AlRaya newspaper and the reported accuracy was 86.1% overall and 81% for the person class.

Another rule-based person name extractor for Arabic was developed using lists of verbs, triggers and stop words (Elsebai et al. 2009). Grammars were used to select person name candidates, which are passed to a morphological analyzer. If one of the analyses indicates a proper noun, it is tagged as a person entity, provided it does not exist in a dictionary of organizations and locations. The accuracy was F1 89% on the associated test corpus, which was manually built from an Arabic news website.

13 http://arabicorpus.byu.edu/
Chapter 4: Overview of Named Entity Recognition

Another rule-based system was implemented by (Zaghouani et al. 2010) exploiting language-independent rules in the Europe Media Monitor, which is a multi-lingual news analysis service. All languages are processed with the same generic rules but referencing language-specific word lists. If a rule is to be applied only to a specific language it is added in the language-specific parameter file. The word lists were built using a bootstrapping technique by capturing the most frequent context of named entities. The system was evaluated on 35 news articles (34k words) achieving an average accuracy of 75%.

4.4.4.2 Corpus-based

(Zitouni et al. 2005) built an ME-HMM classifier trained on the ACE 2003 data and part of the ACE 2004 data, for the entity detection and recognition (EDR) task. Features including lexical, syntactic, gazetteer and the output of another NER system were employed in the system. The overall accuracy was 69.2% on part of the ACE 2004 corpus\(^\text{14}\).

Most of Arabic NER research using corpus-based methods is accredited to Benajiba from Valencia University, where he explored a wide range of algorithms. (Al-Onaizan et al. 2007) used MEM on a manually annotated University of Valencia (UPV) corpus of 150k words to build an NER model targeting NE classes of person, organization, location and other\(^\text{15}\). They used simple lexical contextual features, gazetteers and a stop word list, achieving an accuracy of F1 55.2% on 15% of corpus and 46.7% on the person class.

Later, POS tagging feature was exploited by (Benajiba and Rosso 2007) improving the performance on the same corpus to 65.9% and to 52.1% for the person class. The NER task was split into two modules; NE phrase recognition then NE classification. The performance of the classification module was very good while the identification phase was poor.

(Benajiba and Rosso 2008) used CRF on the same corpus, achieving an overall accuracy of 79.2%, and for the person class 73.4%. The improvement was not only dependent on the use of CRF but also on the extended feature set.

\(^{14}\) [http://projects.ldc.upenn.edu/ace/data/](http://projects.ldc.upenn.edu/ace/data/)

\(^{15}\) [http://www1.ccls.columbia.edu/~ybenajiba/downloads.html](http://www1.ccls.columbia.edu/~ybenajiba/downloads.html)
A combination of CRF and SVM was then adopted by (Benajiba et al. 2008a) employing a comprehensive feature set including morphological features. They used voting scheme to select the best output of the two algorithms. Also, an incremental feature selection method was used to select an optimized feature set. An independent classifier with a different feature set for each named entity was trained on the corpus. This system’s best result was F1 83.5% on the ACE 2003 data and the worst result was on the AEC 2005 weblogs genre with F1 57.3%.

(Benajiba et al. 2009a) further enriched the feature set with language independent features. The model was built using SVM on a combined corpus of UPV and ACE. The best result was 82.17% on ACE 2003 Broadcast News genre.

A bootstrapping technique was explored to equip the system discussed in (Benajiba et al. 2009a) with a richer context of NEs. A parallel corpus was used with a state of the art English NER system to generate notations then project the annotation back to the Arabic version. A new feature was added, which is the head-word provided by Collin parser with noisy accuracy. The approach was able to improve the accuracy by F1 1.5% compared with what was achieved in their previous experiment on ACE 2003 BN data (Benajiba et al. 2010).

An Arabic NER experiment was conducted using the LingPipe 16 NER and HMM-based NER chunker. The NER algorithm depends on word N-grams of size 8. The system was evaluated on ANERCorp (UPV) data17 with 5-fold cross-validation settings. The reported average accuracy was F1 67% on all classes and F 65% on the person class.

Another N-gram-based technique has been investigated in (AbdulHamid and Darwish 2010). The work relied on word boundary character N-grams (leading and trailing) in addition to word N-gram. Their system was evaluated on part of the ACE 2005 corpus, achieving F1 81% and on the ANERCorp (UPV) achieving F1 82%. They concluded that character N-grams capture most Arabic morphological features. Interestingly, the system did not use any external resources such as a gazetteer.

16 http://alias-i.com/lingpipe/demos/tutorial/ne/read-me.html
17 http://www1.ccls.columbia.edu/~ybenajiba/downloads.html
An integration approach was investigated in (AbdelRahman et al. 2010) by combining bootstrapping semi-supervised pattern recognition and Conditional Random Fields. The corpus used was UPV and a 6-fold cross validation experiment showed that their system yielded 67%, 88%, 65% for the person, location, and organization classes respectively.

It is hard to compare the Arabic systems that have been discussed as they were not tested on identical settings in terms of the data used and splitting. Standard data sets and settings are a requirement that needs to be addressed immediately by the Arabic NLP research community to promote research in the field.

To help in creating an annotated data and making it freely available, it would be very efficient to organize a corpus annotation project by a specialized committee, such as the ACL Special Interest Group on Computational Approaches to Semitic Languages (ACL SIGSEM)\textsuperscript{18}. We recommend that the first step is to contact interested researchers in the field to form a group responsible of carrying out the plan toward building an Arabic NER corpus. The group would first decide on selecting the corpus that should range over the different regions of the Middle East, given that it has been found that there is a verity of MSA. There exist some corpora available which need to be assessed for their suitability for the task. The size of the corpus should be large enough for conducting corpus-based studies since supervised learning approaches benefits largely from the size of the data. Next, a definition of the task needs to be highlighted by the group. Previous definitions of the NER task could be adopted or modified. Then, survey the available annotation tools that will be used to annotate the corpus. It could be possible to pre annotate the corpus using one of the NER systems that have been already developed to speed up the annotation process. Moreover, it is also possible to use active learning tool such as ECELA, explained in (Tsuruoka et al. 2008), which is developed by the National Centre of Text Mining (NaCTeM)\textsuperscript{19}. We recommend using the standoff annotation scheme to help in collaborating and sharing the corpus annotation. This will also enable researchers to add further annotations of other tasks to the corpus. The group should allocate enough annotators to annotate the corpus according to the task definition. The annotators do not need to be experts in NLP as basic

\textsuperscript{18} http://www.semitic.tk/
\textsuperscript{19} http://www.nactem.ac.uk/acela/
understanding of Arabic language would suffice. The corpus will then be split according the number of participants (annotators) and the same fold needs to be annotated by at least two annotators and then adjudicated by third annotator who should be knowledgeable of the task.

### 4.4.5 Arabic NER Resources

Two main resources are of high importance in NER system design; corpora and gazetteers. In highly inflectional languages such as Arabic, they are even more critical. One of the successful factors that benefit NER systems in English is the availability of large annotated corpora. With the lack of case information, the NER challenge is a purely lexical problem. The development of these resources for Arabic is still limited compared to other languages, and some are commercial with no details published. Some of the main resources are the following:

#### 4.4.5.1 Corpora

- ANERCorp is a corpus of more than 150,000 words annotated for the NER task developed by Benajiba\(^{21}\) following the CoNLL 2003 task definition.
- A parallel Arabic-Spanish NER corpus\(^{22}\) was developed by Doaa Samy and is discussed in (Samy et al. 2004).
- A multilingual (English, French, Arabic) 1 million word corpus developed by LREC (Mostefa et al. 2009). (not available yet)

#### 4.4.5.2 Gazetteers

- New Mexico State University bilingual Arabic-English lexicon containing NEs\(^{23}\)
- CJK developed Database of Arab Name Variants (DAN) which has 3 million English entries of transliterated Arabic names\(^{24}\)

---

\(^{20}\) [http://projects.ldc.upenn.edu/ace/data/](http://projects.ldc.upenn.edu/ace/data/)


\(^{22}\) [http://www.illl.uam.es/ESP/Arabe_espanol.html](http://www.illl.uam.es/ESP/Arabe_espanol.html)

- Three NE gazetteers of (Persons, Locations and Organisations) collected by Benajiba and freely available.

- Automatically extracted lexicon of 45,000 names using Arabic Wordnet with the Arabic Wiki was developed with an accuracy of 95% (Attia et al. 2010).

### 4.4.5.3 Efforts to overcome scarcity of annotated corpora

Given the scarcity of Arabic NER corpora, there have been a number of attempts to address that problem. Samy used a parallel Spanish-Arabic corpus to generate Arabic NE annotations. It was done by annotating the Spanish version with a Spanish NER tool then mapping them into the Arabic version. The corpus was a set of UN web documents and accuracy on 300 sentences was 90% (Samy et al. 2005). Furthermore, (Zitouni and Florian 2009) investigated propagating from a rich-resource language to a poor-resource language (Arabic). The process was done by running a mention detection system on English data then translating the data to Arabic followed by text alignment.

(Al Khalifa and Rodríguez 2009) discusses the development of a semi-automatic approach to extend the Arabic NE coverage of the WordNet using Wikipedia.

According to (R. Shah et al. 2010), this machine translation approach is not limited by the size of parallel data but is affected by extra noise from the translation, added to the error produced by the NER system. However, they followed that approach targeting Arabic and Swahili. They reported an accuracy of F1 83.5% on 25k of the Arabic UPV corpus.

The parallel corpus approach was again investigated using a different source language. It was investigated on English-Arabic corpus by (Benajiba et al. 2010). The size of the corpus was 900k words and performance was improved by F 1.5%.

---

24 [http://www.kanji.org/cjk/arabic/araborth.htm](http://www.kanji.org/cjk/arabic/araborth.htm)
Chapter 5  Arabic POS tagger

Part-of-speech (POS) tagging is the process of assigning a morphosyntactic role to each word in a text and hence is considered to be a crucial step that highly affects other subsequent NLP tasks. With respect to Modern Standard Arabic (MSA), the official written language, the importance of POS tagging is even larger due to its characteristics that impose a number of processing challenges. For example, POS tagging is vital for Arabic named entity recognition, due to the absence of capitalization in proper nouns. In Semitic languages including Arabic, the phenomenon of clitic attachment is another challenge added to POS tagging complexity. The process of finding the boundaries between the stem and the clitics attached to it is called word tokenization or segmentation. The ambiguity of a word has two parts; finding the correct segmentation and finding the correct tag for each segment.

In this chapter, we cover the implementation of our POS tagger that is intended to be used in NER. For that purpose, we focus on proper noun tagging since named entities are mostly proper nouns.

1.1 Arabic Segmentation and POS tagging

MSA processing is highly affected by the “missing diacritic” problem, adding more complexity to both syntactic and semantic analysis. This is due to the fact that diacritics reduce the number of possible classes of the word. This feature is not present in English, but can be imagined by dropping vowels from words. For example, dropping the vowel from *is* would result in three possible interpretations: *us*, *is* and *as*. Still, vowels would have to be restored by the context to decide on the correct word.
An Arabic word is composed of stem plus affixation to indicate tense, gender and number. In addition to affixes, clitics are attached to the beginning, the end or to both. Clitics are segments that represent an independent syntactic role: mainly conjunctions, prepositions and pronouns. Prepositions and conjunctions are attached to the beginning of the word and pronouns at the end (Diab et al., 2004). Clitics are composed of general Arabic characters that could be part of the stem, and hence pose problems for tokenization. To appreciate the problem of clitic attachment in English, consider passing English text through a noisy channel with the possibility of dropping the space delimiter between words, resulting in word concatenation. Assume the following (noisy) sentence is received:

*Those cars useless fuel.*

The wordform *useless* has two interpretations as it is a candidate that might have been formed by concatenation due to noise; one interpretation is correct and the other is the result of concatenating the words *use* and *less*. If we use the POS tagging information of the previous word *cars*, it would be more sensible to choose the interpretation *use less*, since verbs are more likely to follow nouns than adjectives.

Bar-Haim et al. (2005) refer to each unit of the word that represents an independent tag as a segment. In Arabic, the word [*ولدك*, wldk, your child] has three valid segmentations: wld+k, w+l+d+k and w+I+d+k. Each of these corresponds to a number of POS tagging annotations; for example, the segmentation w+l+d+k, might have the POS tagging sequence of CC+NN+PRP$. Combining both the segmentation with the tagging information constitutes a full analysis; w/CC+Id/VBD+k/PRP. These two tasks are bound together in such a way that the correct tagging analysis always encodes the correct segmentation.
5.2  POS tagging literature review

5.2.1  English language POS

POS tagging started early in the NLP field, in the late 1950s. However, the first large-scale tagger was TAGGIT (Greene and Rubin 1971), a rule-based system that relies on the word pattern and previous tag to disambiguate the current word. Later, CLAWS was developed by the University of Lancaster in the early 1980s as a probabilistic version of TAGGIT (Garside 1987).

Since then, a wide range of corpus-based methods has been applied to NLP tasks including POS tagging leveraging large annotated corpora. Ratnaparkhi used Maximum Entropy Models to build a POS classifier which successfully combined a wider context of tagging history and morphological features to yield an accuracy of 96.6% on the Penn Tree Bank (Ratnaparkhi 1996). Popular taggers were developed using Hidden Markov Model (HMM) techniques adopted from speech recognition and applied to tagging such as (Kupiec 1992) which achieved 96.3% accuracy. An error driven approach, called Transformation-Based Learning (TBL), was introduced by Brill in 1994 and achieved an accuracy of 97.2% on the same (Penn Tree Bank) corpus, outperforming HMM tagging (Brill 1995).

Recent advances in POS tagging have introduced the concept of bidirectional learning, which has resulted in the now state-of-the-art accuracy of above 97% for English. Bidirectional learning uses previous and successive context explicitly to find the tag of the current word. One instance of bidirectional learning is the bidirectional dependency network proposed and discussed in (Toutanova et al. 2003), which yielded 97.20% on the WSJ corpus. Moreover, the same concept was also adopted to develop a biomedical text tagger, discussed in (Tsuruoka et al. 2005). Their results showed the robustness of the tagger when tested on different domains. Another instance of bidirectional learning is the perceptron-like guided learning explained in (Shen et al. 2007), which also yielded comparable results.

Most corpus-based methods produce models that are not easily analyzed and improved, compared to a set of clear and concise transformation rules as are produced by TBL. TBL shares with such methods the idea of automatic extraction of language regularities from
corpora in the training phase but its tagging phase uses fully rule-based techniques (Brill and Mooney 1997)

5.2.2 Arabic Language POS

In Arabic POS tagging, Khoja used a hybrid technique of statistical and rule-based analysis with a morphosyntactic tagset (2001). Later, Support Vector Machines were used to separately implement a character based word-tokenizer and a POS tagger with a collapsed tagset of the Arabic Tree Bank, achieving scores of 99.7% and 95.5% on word-tokenization and tagging respectively (Diab et al., 2004). An enhancement of this system is discussed in (Diab, 2009). With the help of the rich morphological features of Arabic, Habash and Rambow were able to tackle both tokenization and tagging in one step, achieving an accuracy of 97.5% (Habash and Rambow 2005). Later, their system was extended in (Habash et al. 2009). An HMM Hebrew tagger was ported to Arabic, yielding an accuracy of 96.1% (Mansour et al. 2007).

A recent morphological analyzer and POS tagger was implemented and discussed in (Sawalha and Atwell 2010). With 22 morphological features, they worked on defining a new very rich tagset aimed at improving the POS tagging accuracy. The new tagset could also be used in other NLP applications. The tool produces all possible analysis of Arabic words including lemma, root, pattern and vowelization (adding diacritical marks).

In (Sabtan and Ramsay 2009), a combination of rule-based and corpus-based approaches was used with the help of leading and trailing characters. TBL was used to capture errors made by the rule-based module. The generated model is used to tag undiacritised text.

The performances of the systems discussed in this brief review are given if the tool has been tested on a standard dataset.
5.3 Transformation-Based Learning Revisited

Transformation-based learning was briefly covered in chapter 2 of this thesis as one of the corpus-based techniques used in NLP. Here, we will provide a deeper insight into its algorithm and concept. It is an error-driven approach to inducing retagging rules (yielding improved accuracy) from a training corpus. The learning algorithm starts by building a lexicon containing each word with all possible tags and the frequency of occurrence of each tag. Then, it maintains two versions of the corpus: a gold standard corpus that contains word/tag pairs and a training corpus that contains only words. Next, it assigns the most frequent tag to each word in the training corpus, a step referred to as initial state tagging. After that, it compares the resulting initially annotated corpus with the gold-standard corpus and determines the class (tag) with the largest error. Focusing on that error class, it applies a set of predefined templates to correct the errors and chooses the rule with highest gain, where gain means the number of corrections in the training corpus. This rule is stored in a list after being applied to the training corpus. Then, the algorithm calculates the largest error class from the updated training corpus once again, and so on until no further correction can be made. In the tagging phase, the tagger will use the lexicon to tag each word with its most frequent tag, then the list of learned rules is applied.

Usually, each rule template has a tag transformation and a triggering condition. The tag transformation will be fired only if the triggering condition is met. The predefined templates are divided into two categories: non-lexicalized and lexicalized rules. A non-lexicalized rule depends only on surrounding tagging information to change the tag of the current tag, while a lexicalized rule can make reference to words. Some of the 24 rule templates used for English (Brill 1995) are listed below in Figure 5.1. The first rule is a non-lexicalized one that is interpreted as: change tag A to tag B if tag C occurs at position -1 (previous tag). The underlined rule is a lexicalized one interpreted as change tag A to tag B if word C occurs at position -1. In general, A is the original tag, B the transformed tag and C the triggering tag with @ the index of the triggering tag. If the trigger is a word then C is the lexical unit.
5.4 Methodology

Given the clitic attachment feature in Arabic, the POS tag of a word could be compound in nature, leading to tagset extension which, in turn, adds more complexity to this task. Also, this adds the problem of data sparseness (fewer forms with specific compound tags). Thus, the decision was taken to consider a segment-level tagger instead of a word-level one. The first stage implied by this approach is the maintaining of a segment-level annotated corpus, which will be used to produce retagging rules. This involves segmenting the available word-
level corpus as a pre-processing step. The rule induction algorithm described above is run on the pre-processed corpus. The induced rules are applicable to segment-level text and not raw text. In the tagging phase, the algorithm exploits the close relation between the tagging and segmentation processes in Arabic to perform tagging and segmentation of words at the same time. This process relies on a morphological analyzer to produce all possible analyses and uses bigrams to choose the correct analysis. As clitics are limited in number, words that seem to have clitics, i.e., that are ambiguous, are processed, while words that do not have clitics are tagged with their most frequent tag, taken directly from the lexicon. By focusing on specific words, the tagging accuracy of the initial state is leveraged and hence adds more confidence to the N-gram module. In the N-gram module, the task is to choose only the correct segmentation. This is a task which does not require consideration of long previous context and hence does not add the burden of data sparseness. As an example, consider the following example shown in table 5.1:

[قرأ ولدك الكتاب, qr> wldk AlktAb, Your child read the book]

<table>
<thead>
<tr>
<th>Word</th>
<th>Transliterated</th>
<th>Translated</th>
<th>Full Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>قرأ</td>
<td>qr&gt;</td>
<td>read</td>
<td>qr&gt;/VBD</td>
</tr>
<tr>
<td>ولدك</td>
<td>wldk</td>
<td>your boy and diverted you and to demolish</td>
<td>wld/NN+k/PRP$ w/CC+ld/VBD+k/PRP w/CC+l/IN+dk/NN</td>
</tr>
<tr>
<td>الكتاب</td>
<td>AlktAb</td>
<td>the book</td>
<td>AlktAb /NN</td>
</tr>
</tbody>
</table>

Table 5.1: Arabic segmentation and tagging analysis example

Here, the only word that is ambiguous in terms of its segmentation is “wldk” since it starts and ends with clitic-like segments: “w” could be a conjunction and “k” could be a pronoun (see Table 5.1 for transliteration correspondence). In the initial state tagging of this sentence, words will be tagged with their most frequent tag, assuming that they exist in the lexicon, while “wldk” will be tagged as unknown. A morphological analyzer is used only to process
"wldk" and produce the three analyses listed in Table 5.1. Now, the bigram of the previous tag and the segmentation of "wldk" will be used to select the correct segmentation based on the frequencies drawn from the corpus. The analysis associated with the highest of the following probabilities will be selected:

\[ P(wld+k \mid \text{VBD}), P(w+ld+k \mid \text{VBD}) \text{ and } P(w+l+dk \mid \text{VBD}) \]

If all these bigrams never occur in the corpus, resulting in zero probability, then we select the tagging analysis with the highest probability, given the previous tag:

\[ P(\text{NN PRP$\mid$VBD}), P(\text{CC VBD PRP} \mid \text{VBD}) \text{ and } P(\text{CC IN NN} \mid \text{VBD}) \]

In the previous example, it would be more sensible to select the first segmentation of the word, wld+k, as it is most likely for NN to follow VBD.

5.5 Implementation & Experiments

5.5.1 Corpus

The corpus used in this experiment is the Arabic Tree Bank (ATB), which was produced in four parts by LDC and contains news text from four official newspapers of different regions in the Middle East (Maamouri et al. 2004). The total number of words is some 770k. The annotations include morphological analysis and syntactic trees of sentences. For our task, only the morphological analysis is needed.

The morphological analysis annotation was first mapped to the Arabic collapsed tagset distributed with ATB, which comprises 24 tags, see Appendix B. Then, each word with a compound tag was split so that annotation was at the segment-level. The resulting corpus has segment/tag units. The total number of segments in the corpus after segmentation was some 920k.

In our lexicon, it was found that 10% of unique tokens were ambiguous. However, because of the frequency of these tokens, 35% of the data was ambiguous: segments with more than one tag. This high percentage was due mainly to ambiguity in pronouns that are used
differently as personal and possessive. That class would normally have large occurrences in any language. The other ambiguous class was conjunction; [، w, and] was tagged with highest frequency as a conjunction but in 60 cases as noun when used as an abbreviation of one news agency. Thus, it has two possible tags in our lexicon and hence all occurrences of that token were considered ambiguous.

### 5.5.2 Algorithm

The morphological analyzer used to produce word analysis was the Buckwalter Morphological Analyzer (BMA) (Buckwalter 2002). The same mapping scheme was used to map the output of BMA (in BMA’s own tagset) to be consistent with the collapsed tagset used in mapping the ATB. The TBL training phase was then performed on segmented text. As output, a lexicon was built and a set of retagging rules induced.

The tagging phase has two different algorithms which are used, depending on the format of the input text. Figure 5.3 shows the algorithm used to segment and tag an un-segmented text, the general case. In the initial state annotator, not only words that do not exist in the lexicon but also words that seem to have clitic attachments are passed to the BMA. The main concern in this stage is finding only the correct segmentation and not the correct full analysis. If the tagging of the selected analysis is not correct, it will be corrected afterwards by the retagging rules induced in the training phase.

If the tagger is to be run on a segmented text, only segments that do not exist in the lexicon will be tagged as unknown. These are sometimes referred to as out-of-vocabulary items. The N-gram module’s task is to select the highest tag probability of the current segment produced by the BMA. The tag for an unknown segment is conditioned by the previous tag. After construction of the initial state with unknown word guessing, retagging rules are applied straightforwardly to the initially annotated text.

The joint segmentation and tagging algorithm is further illustrated in the block diagram in Figure 5.4. There are two shaded processes in the diagram that differentiate segmented from unsegmented text. The first is the condition that checks if a token exists in the lexicon; that
condition is enhanced in the case of unsegmented text to check also if the word has a clitic-like segment. The second shaded process is at the bottom, which is only included if the text is unsegmented.

1. Assign the most frequent tag to all words and OOV to unknown words in the input list of words. Any word that starts or ends with a clitic-like sequence of characters will also be tagged as OOV.
2. Pass the list of words with their initial tagging to the BMA.
3. The BMA will only process words tagged as OOV and find their solutions. Each solution of OOV words outputted by the BMA is mapped to the collapsed tagset; each solution has a number of possible segmentations and taggings.
4. Each word in the input list will have one of the following:
   - Single tag if it exists in the lexicon
   - One or more analyses if found by the BMA
   - Tagged as NNP if not found by the BMA.
5. Use the frequency of occurrence of the bigram constructed from the previous tag and the segmentation of the word in focus to select the correct solution produced by the BMA. As a back-off scheme, use the bigram constructed from the previous tag and the tag of the current word’s segments.
6. Split the word according to the selection done in the previous step such that the input list contains segments only.
7. Apply retagging rules learned from the pre-tokenized corpus.

Figure 5.2: Joint tagging and segmenting algorithm
Figure 5.3: Tagging algorithm
5.5.3 Experiments

In order to evaluate the performance of our TBL tagger, two experiments were conducted on two different corpora. The first experiment was carried out only on ATB 1.0 for the sake of comparing tagger performance with previous work, as most recent available taggers were evaluated against that part, due to its availability when they were being developed. The second experiment was carried out on the four parts of the ATB to see how our tagger would perform on diverse genres, assuming discrepancy. In both experiments, the corpus was split into 90% training set and 10% test set. The rules are induced first from the pre-segmented corpus. Then, the first evaluation was conducted on the training pre-segmented corpus. This does not sound reasonable, but it was an attempt to examine the quality of the induced rules. Then, the next run took place on the pre-segmented test set. Finally, the tokenization module was evaluated on the non-segmented (word-level) training set.

5.6 Results and discussion

Using the same rule templates that have been used for English, a set of non-lexicalized and lexicalized rules was produced from the pre-segmented corpus in both experiments. The total number of rules was 255 in experiment 1 while that number increased to 1500 in experiment 2. The first portion of the rules was concerned with changing PRP to PRP$, which captures the Arabic feature of pronouns serving as either personal pronoun or possessive pronoun, depending on whether the previous tag is noun or verb. A sample of the induced rules is listed in Figure 5.5. The first four rules are non-lexicalized and the rest are lexicalized.

Table 5.2 shows the results obtained at each stage of the two experiments. The quality of the induced rules was superior on the training set, achieving an accuracy of 98.6% and 97.9%, in experiments 1 and 2, respectively. When evaluated on the test set, the tagger achieved an accuracy of 96.9% in experiment 1 and 96.15% in experiment 2. The main cause for the accuracy drop was the fall in accuracy of the initial state annotator, caused by the tagging inconstancy and extension in different parts of the ATB, e.g., months were tagged as NNP.
in ATB 1.0 while they were tagged as NN in the other parts of the ATB. Empty slots in the table indicate that the test was not completed.

<table>
<thead>
<tr>
<th>Rule</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRP$ → PRP tag IN @ [-1] .</td>
<td></td>
</tr>
<tr>
<td>PRP → PRP$ tag NN @ [-1] .</td>
<td></td>
</tr>
<tr>
<td>PRP$ → PRP tag VBP @ [-1] .</td>
<td></td>
</tr>
<tr>
<td>PRP$ → PRP tag VBD @ [-1] &amp; tag WP @ [-2] .</td>
<td></td>
</tr>
<tr>
<td>JJ → NN word AlEAm @ [0] &amp; tag CD @ [1] .</td>
<td></td>
</tr>
<tr>
<td>IN → NN word bEd @ [0] &amp; tag IN @ [-1] .</td>
<td></td>
</tr>
<tr>
<td>VBD → NN word b @ [-1] .</td>
<td></td>
</tr>
</tbody>
</table>

Figure 5.4: Sample rules

The N-gram module used for OOV guessing achieved an accuracy of 85% in experiment 1 and 80% in experiment 2, due to the fact that the BMA 1.0 was developed from ATB 1.0. However, the accuracy was not highly affected, because larger training data enriches the lexicon and hence reduces the possibility of unknown words except proper nouns, that would be tagged as NNP if not found by the BMA.

The segmentation module achieved an accuracy of 99.6% in experiment 1 and 99.2% in experiment 2. The coverage of the BMA has a larger effect on the segmentation module, as the usage of the lexicon is reduced, since all words that seem to have clitics attached are passed to the BMA. That superior accuracy was achieved due to the low number of words having multiple segmentations in the corpus. We plan to conduct further experiments in order to precisely evaluate the segmentation module as it was tested on a small amount of data. Furthermore, the F measure would give a more meaningful measure of the performance of this module and provide the ability to compare it with other techniques. This aspect also will be included in further study.
## Chapter 5: Arabic POS tagger

<table>
<thead>
<tr>
<th></th>
<th>Exp1</th>
<th>Exp2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATB part</td>
<td>1</td>
<td>1,2,3,4</td>
</tr>
<tr>
<td>Corpus size</td>
<td>165k</td>
<td>920k</td>
</tr>
<tr>
<td>Train set size</td>
<td>150k</td>
<td>828k</td>
</tr>
<tr>
<td>Lexicon size</td>
<td>15k</td>
<td>43k</td>
</tr>
<tr>
<td>Number of rules</td>
<td>255</td>
<td>1500</td>
</tr>
<tr>
<td>Initial state accuracy on train set</td>
<td>95.65%</td>
<td>--</td>
</tr>
<tr>
<td>Accuracy after rule application</td>
<td>98.6%</td>
<td>97.9%</td>
</tr>
<tr>
<td>Test set size</td>
<td>15k</td>
<td>92k</td>
</tr>
<tr>
<td>Unknown words in test set</td>
<td>5.3%</td>
<td>2.8%</td>
</tr>
<tr>
<td>Accuracy of initial state when OOV as NN</td>
<td>91.1%</td>
<td>--</td>
</tr>
<tr>
<td>Accuracy on test set OOV as NN + rules</td>
<td>93.67%</td>
<td>--</td>
</tr>
<tr>
<td>Accuracy of initial state when OOV as NNP</td>
<td>92.37%</td>
<td>--</td>
</tr>
<tr>
<td>Accuracy on test set OOV as NNP + rules</td>
<td>94.91%</td>
<td>--</td>
</tr>
<tr>
<td>Accuracy of initial state when using N-gram</td>
<td>94.52%</td>
<td>93.4%</td>
</tr>
<tr>
<td>Words not found by BMA</td>
<td>18%</td>
<td>25%</td>
</tr>
<tr>
<td>Accuracy of unknown word guessing</td>
<td>85%</td>
<td>80%</td>
</tr>
<tr>
<td>Accuracy with all modules</td>
<td><strong>96.90%</strong></td>
<td><strong>96.14%</strong></td>
</tr>
<tr>
<td>Accuracy improvement</td>
<td>2.38%</td>
<td>2.7%</td>
</tr>
<tr>
<td>Transformation Accuracy (correct/all)</td>
<td>94%</td>
<td>92%</td>
</tr>
<tr>
<td>Largest error class</td>
<td>NN as JJ</td>
<td>NN as JJ</td>
</tr>
<tr>
<td>Accuracy of segmentation module</td>
<td>99.6%</td>
<td>99.2%</td>
</tr>
</tbody>
</table>

Table 5.2: Experimental results
The pie chart in Figure 5.6 shows the largest classes of the errors committed by the tagger. The largest one was the NN, constituting one third of the errors. That error rate is affected by the frequency of occurrence of that class in the corpus. Also, nouns share most of the adjective and some verb forms. To check the ambiguous classes, we have generated the confusion matrix of our tagger errors, shown in Table 5.3. The largest error class was tagging NN as JJ. Adjectives normally follow adjectives; however, nouns also follow other nouns. Thus, if the adjective has the noun form then it would be hard to capture. The second largest error class was tagging NNP as NN which was due to the general case of Arabic proper nouns that are in the form of general nouns. The NNP tagging accuracy will be further discussed in the following section.

We observed that an unexpectedly large number of cardinal numbers (CD) was tagged as NNP. It was found that our regular expression used in the initial state to capture number digits fails when the number has a time format. Hence, they were tagged as OOV and later as NNP when not found by BMA.

We also observed that although we have a large number of pronouns, the tagger had a good accuracy on tagging them, even though the same form is used to serve as both possessive and personal. The reason is that they have a rule of thumb; if following a noun then they are possessive and personal otherwise.

![Error distribution (top 8)](image-url)

Figure 5.5: Error distribution (top 8)
Table 5.4 shows a performance comparison of our tagger on segmented text with three other taggers described in (Diab et al, 2004; Habash and Rambow, 2005; Mansour et al, 2007). The exact test data used to evaluate those taggers was not available but a similar selection scheme was used instead. 10% of ATB 1.0 was randomly selected for testing our TBL
tagger, comparing its performance with the reported accuracy of these others. Under this condition, our TBL tagger outperformed Diab et al’s and Mansour et al’s, which do not rely on any morphological features like ours, and is slightly inferior to Habash and Rambow’s, that exploits morphological features. In addition, our tagger yielded concise and easily interpreted rules.

<table>
<thead>
<tr>
<th>System</th>
<th>Technique</th>
<th>Accuracy %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diab et al 2004</td>
<td>SVM</td>
<td>95.49</td>
</tr>
<tr>
<td>Habash &amp; Rambow 2005</td>
<td>SVM</td>
<td>97.5</td>
</tr>
<tr>
<td>Mansour et al 2007</td>
<td>HMM</td>
<td>96.12</td>
</tr>
<tr>
<td>Our tagger</td>
<td>TBL</td>
<td>96.9</td>
</tr>
</tbody>
</table>

Table 5.4: System comparison on ATB 1.0

5.7 Notes on the NNP Class

Since this study is a pre-processing step for the NER system, we have reviewed some aspects of the POS tagger that would influence our NER system considerations. The first is the Arabic syntactic feature of free order of nouns and verbs. This feature also affects proper nouns as they have a similar role to nouns. We measured the percentage of NNP following a VBD and VBD following NNP. It was found that NNP was followed by VBD in 40% of the cases; almost half of the occurrences of this combination. This observation is critical in the case of using action verbs as a feature in Arabic NER system, since lists of action verbs used as a feature would not be able judge whether the previous or next word is a proper noun. However, it still could be used as an indicator that a proper noun is in the vicinity.

For the clitic attachment feature, we found out that 9% of NNP have clitics attached to them; mainly proclitics, that is conjunctions. That suggests that the importance of segmenting the Arabic text is also critical in NER. It would be very rare to have enclitics
such as PRP$ following proper nouns, although it happened in the corpus in a few cases to indicate belonging to a country such as, [لبناننا، IbnAnnA, our Lebanon].

The last feature to be checked is the ambiguous proper noun (serving as non-NNP). We found that 13% of the NNP in the corpus lexicon are ambiguous i.e., they have other possible classes. Also, we observed from the counts in our lexicon that tokens of this type are more likely to occur in text as non-NNP, given the normal low percentage of NNP in text compared to other classes. That would affect the initial state accuracy, as it relies on the most frequent tag of the word, in which case the NNP would always be defeated by other potential tags.

With respect to the NNP tagger accuracy and based on the error pie chart in Figure 5.6, the largest erroneous classes were NN, JJ and NNP in descending order. However, this measure does not indicate how accurate our NNP tagging is. Thus, we have calculated the accuracy of each class using the standard IE metrics previously discussed in chapter 4, precision, recall, F1.

The count of NN tokens was 29085, of which there were 24961 true positives and 1133 false positives. The false negative count was 1087. Using equation 4.1, 4.2, 4.3:

\[ \text{Precision} = \frac{24961}{24961 + 1133} = 95.7 \]

\[ \text{Recall} = \frac{24961}{24961 + 1087} = 95.8 \]

\[ F1 = \frac{2 \times 95.8 \times 95.7}{95.8 + 95.7} = 95.8 \]

With respect to JJ class, the total count was 8599, 7875 true positives, 673 false positives and 722 false negatives. The three measures are:

\[ \text{Precision} = \frac{7875}{7875 + 673} = 92.1 \]

\[ \text{Recall} = \frac{7875}{7875 + 722} = 91.6 \]
For the NNP class, the count was 6439 tokens; 5938 true positives, 494 false positives and 501 false negatives. The results are:

\[
F1 = \frac{2 \times 92.1 \times 91.6}{92.1 + 91.6} = 91.8
\]

\[
\text{Precision} = \frac{5938}{5938 + 494} = 92.3
\]

\[
\text{Recall} = \frac{5938}{5938 + 501} = 92.2
\]

\[
F1 = \frac{2 \times 92.3 \times 92.2}{92.3 + 92.2} = 92.2
\]

Based on these parameters, NNP tagging accuracy was very low compared to NN even if the NN class contributed to a greater number of errors made by the tagger. The NNP and JJ performances are very close. However, there is another factor with larger effect on the NNP class than the JJ class. That factor is the likelihood of being an OOV token.

In the test set, there were 2662 OOV tokens; of which there were 994 NNP and 360 JJ. That is, 37% of OOV are NNP whereas 13% are JJ. The large size of our corpus assisted in limiting OOV tokens to the NNP class.

5.8 Future Work

This study has showed that TBL outperforms other techniques used for Arabic POS tagging, without word features, as well as being simple and less complex and with the same templates used as with other languages. With respect to segmentation, it is quite telling that the short previous tagging context using most frequent tag information will still perform well for this task.

Encouraged by the performance of the tagger, we plan to train the tagger on un-segmented text. In this way, compound tags will have both tagging and segmentation information, thus eliminating the need for a morphological analyzer. This new experiment will involve
modifying the learning algorithm to consider validating that the new compound tag is one of the possible tags for the word, based on the presence of clitic-like segments, exploiting the phenomenon that tagging analysis always maps to only one segmentation. Also, the conditions of the contextual templates have to be applied based on single tag context rather than on compound tag context. Furthermore, Brill has introduced other TBL templates to deal with unknown words using only trailing and ending characters: we assume these would be appropriate for Arabic words if the templates were modified. The proposed templates would use clitics attached to the word as a feature along with the context of tags to guess unknown words, rather than using morphological features.
Chapter 6  The correlation between Arabic NER and POS

Our first attempt to address Arabic NER problem was by measuring how the Arabic POS tagging and NER are related. We carry a quantitative analysis of that relation using a small manually annotated corpus. This work concerns the main three NE classes; PERSON (PER), LOCATION (LOC) and ORGANIZATION (ORG). Our analysis is carried out both collectively and per class. The POS tagging information included in the corpus was done by expert annotators to guarantee the accuracy of our analysis, as machine tagged text will not be perfect. This study would examine our main hypothesis that NER is highly related to POS tagging in Arabic. The main assumption is that NEs follow specific POS tag classes and fall into specific classes.

Next, we measure the POS tagging information effect on the performance of an NER system. We adopt a corpus-based method that constructs a classification model from labelled data and apply it to test data that was not used in the training. The proposed classifier is built using Maximum Entropy Modelling, which has proven successful in many NLP tasks, especially in NER.

This was followed by adding more features based on the performance of the POS feature to help where POS tagging features fail. At each step, the classifier performance is evaluated on a test set of the corpus that was not used in the training.
6.1 Corpus

6.1.1 Corpus overview

Due to the lack of Arabic NER corpora, we have built our own corpus with manual annotation. We had the choice of taking a number of news article and generating the POS feature using our POS tagger discussed in the previous chapter. However, the tagger would not normally give a perfect tagging, leading to inaccurate measures. Thus, we decided to use a corpus annotated with POS by an expert.

The corpus used in this experiment is part of the ATB, that was used in developing our POS tagger. We have selected 200 articles of the Arabic TreeBank (ATB) 2.0\textsuperscript{25} comprising news articles from the AlHayat newspaper. The total number of words was 60,000. The ATB annotation includes morphological analysis for each word, and syntactic trees of sentences. The morphological tagset includes 131 tags. We have used the collapsing scheme distributed with the ATB to map each morphological tag to its corresponding collapsed tag, yielding a tagset of 24 basic tags. This is because we are not including any morphological features in the proposed NER system. Given that proper nouns are subject to clitic attachments producing compound tags, we have split words with compound tags. The total number of tokens (segments) was 70,000. The corpus was then divided to 60,000 segments for training and 10,000 segments for testing purposes. Our reason for using the gold standard segmented corpus is that we try to accurately measure the effect of POS tagging by eliminating any other factors such as segmentation errors.

6.1.2 Corpus Annotation

We have used Callisto\textsuperscript{26}, a stand-off annotation tool, to manually label each word with its NE class: PER, LOC, ORG and a fourth class of Other (O) for tokens not belonging to any of the three classes. The annotation guidelines were the same as those given by the MUC.

\begin{footnotesize}
\begin{itemize}
\item \textsuperscript{25} [http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2004T02]
\item \textsuperscript{26} [http://callisto.mitre.org/]
\end{itemize}
\end{footnotesize}
Chapter 6: The correlation between Arabic NER and POS

website\textsuperscript{27} and the job was carried out by one annotator; the Phd candidate who is an Arabic native speaker. A Java XML parser was implemented to extract each word with its NE class from the resulting XML files. Then, the result is combined with the gold-standard POS tagging produced in the previous step. Each line in the corpus contains: word, POS tag and NE class, following the IOB2 scheme described in chapter 3. Thus we have two labels for each class (B and I) and one O label to denote “OTHER” class.

6.1.3 Corpus Analysis

The total number of named entities is 5379 tokens: 24.5\% PERSON, 33.7\% LOCATION and 41.8\% ORGANIZATION. Some organization tokens were not words but rather punctuation marks because news articles tend to have entities between quotations following the designator, e.g., "منظمة حقوق الإنسان". According to MUC guidelines, designators are part of the named entity, so the quotation marks were also tagged with the rest of the name. Token and NE distributions are shown in Table 5.1. The overall average of tokens per entity is 1.8\%, which is probably the case for news text and not for other genre such as forums and weblogs.

<table>
<thead>
<tr>
<th>Class</th>
<th>Token count</th>
<th>NE count</th>
<th>Tokens/Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person</td>
<td>1317</td>
<td>675</td>
<td>1.9</td>
</tr>
<tr>
<td>Organization</td>
<td>2251</td>
<td>905</td>
<td>2.4</td>
</tr>
<tr>
<td>Location</td>
<td>1811</td>
<td>1302</td>
<td>1.3</td>
</tr>
<tr>
<td>Total</td>
<td>5379</td>
<td>2882</td>
<td>1.8</td>
</tr>
</tbody>
</table>

Table 6.1: Corpus statistics

\textsuperscript{27} http://www-nlpir.nist.gov/related_projects/muc/proceedings/ne_task.html
Chapter 6: The correlation between Arabic NER and POS

With respect to POS tagging, more than half of the NEs were tagged as proper noun (NNP). Figure 6.1 shows the POS distribution over NE classes excluding the “OTHER” class. To have precise statistics, Figure 6.2 shows the POS tags of each NE class including OTHER. Based on the statistics, we would expect that the person class would have the highest gain when the POS feature is employed. Unexpectedly, not all person names where tagged as NNP, hence we carried out more analysis on these occurrences. It was found that most of the errors were POS tagging errors from the gold standard tagging and a few were annotation errors from when the corpus was annotated with NE classes. We could not confirm if the tagging errors were made by the expert linguist since the corpus went through a number of pre-processing steps to its final format. Even with these errors, the majority of NEs were tagged as NNP. Location names were more ambiguous with 20% non NNP tags. The reason is that designators, i.e. city, were annotated by definition as part of location entities, which are tagged as NN. The most ambiguous class was the organization class, which was expected due to forming them from noun and adjectives. This class would benefit least from the POS tagging feature. In general, detection of NEs would definitely improve with POS tagging information, consistency with our intuition that NEs fall mostly within specific POS tags.

![Image of a pie chart showing POS distribution over NE classes excluding the "OTHER" class.](image)

Figure 6.1: Internal POS of NEs, excluding OTHER, at the token level\(^{28}\)

---

\(^{28}\) O-POS denotes the remaining pos tagging categories

125
Chapter 6: The correlation between Arabic NER and POS

Figure 6.2: POS tag distribution of NE classes
Regarding the POS tag context of NEs, Figure 6.3 demonstrates the previous tag of NE classes. Figure 6.4 shows the previous POS context of each class. The most regular class was LOC as it was preceded in more than half the occurrences by prepositions. Thus, using the POS tagging of the current and previous words would have a different impact in the two classes (PER, LOC). The PER class is more regular with respect to the current word tag and LOC is more regular with respect to the previous word. However, the tagging accuracy of our tagger was very good with preposition as it is a closed class token, in contrast to NNP as the most erroneous class in our evaluation. The organization class was very much affected by the quotation marks; it was preceded in more than a third of the occurrences by punctuation.

Figure 6.3: NE context (previous POS tag)
6.2 NER classifier

To measure the effect of POS tagging information when used as a feature we split our corpus into 85% training set and 15% test set. The detailed statistics of the two sets are shown in Table 6.2. The percentage of NE found in the test set suggests that our split was appropriate.
### Table 6.2: Corpus split

<table>
<thead>
<tr>
<th>Class</th>
<th>Train NE tokens</th>
<th>Train NEs</th>
<th>Test NE tokens</th>
<th>Test NEs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person</td>
<td>1317</td>
<td>568</td>
<td>210</td>
<td>107</td>
</tr>
<tr>
<td>Organization</td>
<td>2251</td>
<td>773</td>
<td>313</td>
<td>132</td>
</tr>
<tr>
<td>Location</td>
<td>1811</td>
<td>1102</td>
<td>291</td>
<td>200</td>
</tr>
<tr>
<td>Total</td>
<td>5379</td>
<td>2443</td>
<td>814</td>
<td>439</td>
</tr>
</tbody>
</table>

#### 6.2.1 Task Complexity

In all experiments, we have used the CoNLL shared task evaluation methodology discussed in chapter 3; the tool used for the evaluation was provided by CoNLL-2002\(^{29}\). It is a standard method used in most NER literature and simplifies the process of comparing with previous work. Our first experiment was to measure the complexity of the task through two different measures. The first is by constructing a baseline of assigning the most frequent NE class in the training set to the test data. We have used the baseline utility also provided by CoNLL-2002\(^{30}\). For a named entity to be assigned the class, it needs to occur in full in the training data. The baseline results are shown in Table 6.3, using the standard IE measures described in chapter 3:

<table>
<thead>
<tr>
<th>Class</th>
<th>Accuracy%</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>PER</td>
<td>63.49</td>
<td>34.78</td>
<td>44.94</td>
<td></td>
</tr>
<tr>
<td>LOC</td>
<td>87.75</td>
<td>81.00</td>
<td>84.24</td>
<td></td>
</tr>
<tr>
<td>ORG</td>
<td>46.22</td>
<td>41.04</td>
<td>43.48</td>
<td></td>
</tr>
<tr>
<td>All</td>
<td>70.98</td>
<td>29.21</td>
<td>41.39</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.3: Baseline of most frequent classes

---

\(^{29}\) [http://www.cnts.ua.ac.be/conll2002/ner/bin/conlleval.txt](http://www.cnts.ua.ac.be/conll2002/ner/bin/conlleval.txt)

\(^{30}\) [http://www.cnts.ua.ac.be/conll2002/ner/bin/baseline.txt](http://www.cnts.ua.ac.be/conll2002/ner/bin/baseline.txt)
Chapter 6: The correlation between Arabic NER and POS

The other measure was on a token level to find the percentage of tokens that are ambiguous in terms of the number of NE classes they could take. We have built a lexicon from the whole corpus listing each token with possible NE tags. The lexicon size was 10052 tokens with 6% having more than one class. Then, we calculated the percentage of tokens that has more than one class in the lexicon. It was found that 28.5% of the corpus was ambiguous. This high percentage is due to the fact that the most frequent ambiguous tokens were preposition and conjunction that were tagged when within organization names. That also includes quotation marks since they are encountered within named entities. It was inferred by finding the number of tokens in the corpus that have more than one entry in the NE lexicon.

The lexicon tokens that have more than two classes have their ambiguity within NE classes themselves, given that we have 7 classes. There were 141 tokens with that level of ambiguity. One example is [ناصر, nASr, Nasser] with the following classes and frequencies:

“nASr” -> I-LOC 3 ; I-ORG 1 ; I-PER 1 ; B-PER 1 ;

The percentage of ambiguous words is not accurate as I found one instance in the corpus of a preposition tagged as person and it has a large number of occurrences in the corpus.

6.2.2 Maximum Entropy Modeling Revisited

Maximum Entropy Modeling has been widely used in various NLP tasks including NER. It is known for its ability to combine features from diverse knowledge sources successfully. According to (Borthwick et al. 1998b) and (Ratnaparkhi 1996), the main components of the ME model are futures, histories and features. The purpose is to calculate the probability of history $h$ to have an outcome $f$; $p(f \mid h)$. 
Chapter 6: The correlation between Arabic NER and POS

History is all the information that helps in assigning weights to features. In the NER task, \( f \) is one of the named entity classes (PER, ORG, LOC, etc.). Calculating \( p ( f \mid h ) \) can be reformulated as finding the probability of \( f \) associated with token \( t \) in the corpus:

\[
p ( f \mid h ) = \{ f \mid \text{information relative to token } t \ (\text{history})\}
\]

History is the possible context that helps in predicting the class of token \( t \):

\[
h_i = \{ t_i, t_{i+1}, t_{i-1}, f_{i-1}, \text{POS}_i, \text{POS}_{i-1} \}
\]

where \( f \) is the named entity class.

Features in Maximum Entropy are binary features, meaning they could have only two outcomes. One possible feature is:

\[
g_j (h_i, f_i) = \begin{cases} 
1 & \text{if } \text{(POS}_{i-1}\text{) is NN and } C_i = \text{PERSON} \\
0 & \text{otherwise}
\end{cases}
\]

Features are relationships between history and outcome and not only attributes of the word or context. For the above feature \( g \), if the previous POS tag of the history \( h \) is a noun then \( t_i \) is most likely to be classified as a PER. Given a feature space, the job of the model is to associate a weight to each feature in the feature space using a method called General Iterative Scaling.

In the decoding (testing) phase, the conditional probability \( p ( f \mid h ) \) is calculated for the token being processed for all features in the feature space. Then, we ignore the ones that are not active, i.e., that yield zero output. Finally, \( p ( f \mid h ) \) is the product of the active features normalized over the product of all features. The active features are the ones that have a value of 1.

Fortunately, the implementation of this algorithm does not require a binary feature format as it is enough to choose the features to include in the model. We have used a Java implementation available in the openNLP project\(^{31}\).

\(^{31}\) http://maxent.sourceforge.net/
6.3 POS tagging effect

In this stage of our study, we are trying to measure only the effect of the POS tagging feature. The system will be enhanced later with more features. The two features used in this experiment are lexical and POS tagging features (POS).

The lexical feature (LEX) is defined as four sub-features: the word itself, the previous word, the next word and the word after next. This feature is built from the training data. The intuition is that NEs follow and precede certain tokens. The POS feature was defined for each of current word, previous word and next word.

We have built a classifier from the training corpus and evaluated the model on the test set. The lexical features improved the accuracy by almost 20% over the baseline. That improvement was due to the fact that our baseline module was very strict and considers the full entity name to be tagged while the algorithm deals with tokens. Thus, if one token occurs in the training set as part of an NE then there is a chance to correctly detect it in the test set if it was a single token. The details of performance are in Table 6.4.

<table>
<thead>
<tr>
<th></th>
<th>Accuracy %</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>PER</td>
<td>67.02</td>
<td>54.31</td>
<td>60.00</td>
<td></td>
</tr>
<tr>
<td>LOC</td>
<td>82.18</td>
<td>75.11</td>
<td>78.49</td>
<td></td>
</tr>
<tr>
<td>ORG</td>
<td>28.86</td>
<td>32.09</td>
<td>30.39</td>
<td></td>
</tr>
<tr>
<td>All</td>
<td>61.12</td>
<td>57.75</td>
<td>59.39</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.4: Performance of the lexical feature

After that, we have added the POS features to the model, increasing the accuracy by 7%. Detailed in Table 6.5, the most improved class of that feature was the person class, as person names are always tagged as NNP in contrast to organization names, that could have nouns and adjectives. The location class was not improved over the baseline since they are
repeated in the corpus, suggesting that the POS feature was overwhelmed by the lexical feature. In addition, location NE are more likely to occur as single tokens, which have less effect on performance if missed than missing part of multiple token NE instances, which are often the case for person and organization NE. Nevertheless, it is not appropriate to generalize that result to any other work as this experiment used the gold standard POS tagging. Thus, we have replaced the POS tagging information in the corpus with the most frequent POS tag from the lexicon built in the construction of our POS tagger, which is the bottom line of the POS tagger. This is will not be the worst performance of our tagger given the following:

- The segmentation factor is not included here as a corpus is pre segmented with gold standard segmentation.
- The text quality of the corpus is very good as it has been carefully prepared by experts with limited spelling mistakes.
- The ATB corpus was built from news text that would share common lexical items.

For the above reasons, we would expect the tagging accuracy on free text to be lower and this will be investigated more in later chapters when we use a different corpus. Thus, the purpose here is to measure the effect of different POS tagging qualities on the NER task.

When the classification model was built from the machine POS tagged corpus, the performance was degraded by a 4% drop in the quality of the POS feature. That confirms that ambiguous tokens are more likely to occur in text as non-NE. Tokens of the person class for instance were tagged as NNP in 95% of occurrences with gold standard tagging while it dropped to 80% when using the POS tagging baseline. Also considering the effect on the accuracy of each class in Table 6.6, the person class tokens are more likely to be used as non NE tokens. That means also that collecting NEs in dictionaries will be neither helpful nor detrimental to the performance; this is to be confirmed later in the following section.

However the performance is still better than when only lexical features were employed. That would suggest that using the POS features with similar quality would always have a positive impact on the accuracy of the model.
Table 6.5: Performance of the POS feature

<table>
<thead>
<tr>
<th></th>
<th>Accuracy %</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>PER</td>
<td>76.92</td>
<td>68.97</td>
<td>72.73</td>
<td></td>
</tr>
<tr>
<td>LOC</td>
<td>82.21</td>
<td>77.38</td>
<td>79.72</td>
<td></td>
</tr>
<tr>
<td>ORG</td>
<td>39.57</td>
<td>41.04</td>
<td>40.29</td>
<td></td>
</tr>
<tr>
<td>All</td>
<td>67.85</td>
<td>64.97</td>
<td>66.38</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.6: Performance of the POS baseline

<table>
<thead>
<tr>
<th></th>
<th>Accuracy %</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>PER</td>
<td>67.00</td>
<td>57.76</td>
<td>62.04</td>
<td></td>
</tr>
<tr>
<td>LOC</td>
<td>84.90</td>
<td>73.76</td>
<td>78.93</td>
<td></td>
</tr>
<tr>
<td>ORG</td>
<td>37.69</td>
<td>36.57</td>
<td>37.12</td>
<td></td>
</tr>
<tr>
<td>All</td>
<td>66.11</td>
<td>59.24</td>
<td>62.49</td>
<td></td>
</tr>
</tbody>
</table>

6.4 Improving to Classification of Proper Nouns

The tag NNP is a generic tag assigned to any of the three NE classes. Using the POS tagging information, the performance of our NER system was improved by 7% without the use of any other features helping to classify proper nouns tagged as NNP except the lexical feature. The accuracy of detecting NE was 74%, indicating the number of NEs that were detected but wrongly classified. Thus, we use additional features to help in detecting non-NNP tokens and classifying them. We propose three simple features and measure the system performance when each one is added to our baseline of lexical feature and POS information.

6.4.1 NE Gazetteers (GAZ)

A gazetteer of each class was used in a window of three tokens, i.e., current, previous, next. The person gazetteer was provided by a government agency and contains 100k person names. The location gazetteer consisted mainly of countries and cities names collected from the web and the size was 2k. The organization gazetteer contains only the names extracted from the training set. After the addition of this gazetteer feature, there was no significant effect on the overall accuracy, as shown in Table 6.7. To justify that performance even with the large coverage of our person gazetteer, we have calculated the accuracy of the gazetteer
without building the model. We adopted the IE standard metrics of precision, recall and F1. The true positives, false positive and false negative were calculated as follows:

\( \text{TP} = \) the number of tokens that exist in gazetteer and are labelled as PER

\( \text{FP} = \) the number of tokens that exist in gazetteer and are not labelled as PER

\( \text{FN} = \) the number of tokens that do not exist in gazetteer and labeled as PER

Using the three measures equations:

\[
\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}
\]

\[
\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}
\]

\[
F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}
\]

The results were as follows:

\[
\text{Precision} = \frac{803}{803 + 2452} = 24.66 \%
\]

\[
\text{Recall} = \frac{803}{803 + 514} = 60.97\%
\]

\[
F1 = \frac{3007.04}{85.63} = 35.12 \%
\]

The low precision of the gazetteer, caused by false positives, could explain the insignificance in the system performance even if our gazetteer has good recall. The overall accuracy of the gazetteer is affected by the fact that most person names are nouns and adjectives. These categories are more frequent in text than a proper noun. For that reason, the gazetteer feature was removed from the system.
Chapter 6: The correlation between Arabic NER and POS

<table>
<thead>
<tr>
<th></th>
<th>Accuracy %</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>PER</td>
<td>79.41</td>
<td>69.83</td>
<td>74.74</td>
<td></td>
</tr>
<tr>
<td>LOC</td>
<td>83.09</td>
<td>77.83</td>
<td>80.37</td>
<td></td>
</tr>
<tr>
<td>ORG</td>
<td>39.06</td>
<td>37.31</td>
<td>38.17</td>
<td></td>
</tr>
<tr>
<td>All</td>
<td>69.34</td>
<td>64.33</td>
<td>66.74</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.7: System performance with (LEX, POS, GAZ)

### 6.4.2 Trigger feature (TRG)

Three tables of NE triggers were manually handcrafted. Person triggers include titles, prefixes and roles. Organization triggers include designators such as company. Location triggers include prefixes such as city. The three tokens before and after the token in focus are matched against the trigger tables, since the named entity might not be preceded directly by the trigger, due to the fact that Arabic adjectives follow nouns. Also, the trigger could come after the named entity. If one of the tokens within that window is found in one of the trigger tables, the value of the associated gazetteer feature is set to the name of the corresponding class. We have built another system that includes this feature in addition to the lexical and POS tagging feature. The performance was improved by 1.3% and the best single class gain was the ORG class since it has the most irregular POS information, Table 6.8.

<table>
<thead>
<tr>
<th></th>
<th>Accuracy %</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>PER</td>
<td>82.11</td>
<td>67.24</td>
<td>73.93</td>
<td></td>
</tr>
<tr>
<td>LOC</td>
<td>82.16</td>
<td>79.19</td>
<td>80.65</td>
<td></td>
</tr>
<tr>
<td>ORG</td>
<td>41.48</td>
<td>41.79</td>
<td>41.64</td>
<td></td>
</tr>
<tr>
<td>All</td>
<td>69.75</td>
<td>65.61</td>
<td>67.61</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.8: System performance with (LEX, POS, TRG)
6.4.3 Class of previous word feature (C₁)

Given that named entities come in chunks of tokens, the class of the previous word is a good indicator of the current word class. Also, the trigger feature previously discussed might not be in the window of the current word but in the window of the previous word. We have added the previous word class in an attempt to model the correlation between classes of adjacent words. That feature had a good effect on our system with a 5% increase, as shown in Table 6.9.

<table>
<thead>
<tr>
<th></th>
<th>Accuracy %</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>PER</td>
<td>84.95</td>
<td>68.10</td>
<td>75.60</td>
<td></td>
</tr>
<tr>
<td>LOC</td>
<td>85.57</td>
<td>77.83</td>
<td>81.52</td>
<td></td>
</tr>
<tr>
<td>ORG</td>
<td>60.82</td>
<td>44.03</td>
<td>51.08</td>
<td></td>
</tr>
<tr>
<td>All</td>
<td>79.28</td>
<td>65.82</td>
<td>71.93</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.9: System performance (LEX, POS, C₁)

6.4.4 Global feature (GLB)

This single feature is defined as the class of the other occurrences of the token within the same article, that have already been processed. If the same word is found, the value of global feature is assigned the class of the previously processed token. The class is used as a feature regardless of whether the token is at the start or in the middle of an NE. This feature tokens takes into consideration the name alias phenomenon, which is especially prevalent in news articles. The performance is detailed in Table 6.10.
Chapter 6: The correlation between Arabic NER and POS

<table>
<thead>
<tr>
<th>Accuracy %</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>PER</td>
<td>79.44</td>
<td>73.28</td>
<td>76.23</td>
</tr>
<tr>
<td>LOC</td>
<td>84.80</td>
<td>78.28</td>
<td>81.41</td>
</tr>
<tr>
<td>ORG</td>
<td>40.85</td>
<td>43.28</td>
<td>42.03</td>
</tr>
<tr>
<td>All</td>
<td>79.44</td>
<td>67.28</td>
<td>68.23</td>
</tr>
</tbody>
</table>

Table 6.10: System performance (LEX, POS, GLB)

6.4.5 All features effect

When we include all the proposed features, we can observe a very good improvement over the baseline of lexical and POS tagging features i.e., 7%. The details of per class accuracy are in Table 6.12. We tried our gazetteer feature one more time when all features are included and the system was significantly degraded by 3%, shown in Table 6.11. The overall results of our experiments in this chapter are listed with the accuracy of each feature in Table 6.13.

<table>
<thead>
<tr>
<th>Accuracy %</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>PER</td>
<td>86.02</td>
<td>68.97</td>
<td>76.56</td>
</tr>
<tr>
<td>LOC</td>
<td>86.70</td>
<td>79.64</td>
<td>83.02</td>
</tr>
<tr>
<td>ORG</td>
<td>66.29</td>
<td>44.03</td>
<td>52.91</td>
</tr>
<tr>
<td>All</td>
<td>81.82</td>
<td>66.88</td>
<td>73.60</td>
</tr>
</tbody>
</table>

Table 6.11: All features with GAZ

<table>
<thead>
<tr>
<th>Accuracy %</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>PER</td>
<td>91.40</td>
<td>73.28</td>
<td>81.34</td>
</tr>
<tr>
<td>LOC</td>
<td>85.92</td>
<td>80.09</td>
<td>82.90</td>
</tr>
<tr>
<td>ORG</td>
<td>72.34</td>
<td>50.75</td>
<td>59.65</td>
</tr>
<tr>
<td>All</td>
<td>83.40</td>
<td>70.06</td>
<td>76.39</td>
</tr>
</tbody>
</table>

Table 6.12: All without GAZ
Comparing these results with previous work is not applicable as different settings are used in each study. However, the results are still quite informative, regarding the effect of the added features. Moreover, the PERSON class was the most affected by the quality of POS tagging as it is tagged as NNP and the nature of person names makes it more likely to be used as non NE. Corpus size has a major impact on performance as shown in previous studies: most systems exhibit performance increase proportional to data size.
6.5 Conclusion and Future work

Early results obtained in this work have proved the strong correlation between NER and POS tagging in the Arabic language. External resources such as gazetteers with good coverage do not guarantee the improvement Arabic NER systems. Also, external cues can be used successfully in Arabic NER corpus-based approaches. However, these features are currently just included blindly in the model without any effort to engineer them properly. Utilizing a grammar that governs NE creation would definitely add to the performance, for example an entity currently recognized as Organization could be better recognized as a Geographical entity following a company designator. The accuracy of POS tagging has a major impact on performance, especially on the person classification. However, there is no system that guarantees the accuracy of the ATB tagging. Changing the sequence of processing such as carrying out NER before POS tagging, could also improve both analysis levels. To overcome the problem of annotated corpus size and its quality, we could obtain the AEC corpus, which will also help in comparing results with those of others.
Chapter 7  Arabic Person Name Recognition

In our previous chapter, the gazetteer did not have any effect on the performance for two different reasons; the quality of POS tagging, as we were using gold standard tagging and the quality of the gazetteer itself. The performance of the system when the gazetteer feature was included is affected by the fact that most Arabic tokens that could be used as proper nouns could also be used as general nouns or adjectives.

In this chapter, we aim to investigate the effect of our POS tagger on a standard dataset. Also, we use a standard set with a relatively larger size than what has been used in the previous chapter. We will see how we could improve the gazetteer performance with a subset of unique person names, using a filtering technique that involves our POS tagger lexicon.

We also investigate the effect of exploiting novel contextual features for the task of Arabic person name recognition. These features are generated from a corpus with the help of a gazetteer that has a large coverage and through leveraging negative context.

7.1 Current study

The present study focuses on the identification phase of person names as they have the most complexity compared to other entity classes and they might occur in compound non-person entities, e.g. an organization named after a person. Thus, the correct classification of a person name would require these compound entities to be classified. Another motivation was that each class has its own context and hence different distinguishing features. This work has a strong emphasis on finding novel features (for Arabic NER), based on human intuition to tackle this task. Our experiments are based on the concept of Maximum
Entropy. The work starts with building a named entity corpus for the PERSON class from an existing corpus. After that, Maximum Entropy Modeling is used to build a classifier from the data testing one feature at a time. Some features have been employed previously in NER and some are novel in the Arabic language. For each feature employed, the classifier performance is evaluated on the test corpus (that was not used in training) to confirm its effect before the final system test design. Finally, we build a model with all features proven to be effective. For the data standardization issue, we publish deep analysis and documentation of our corpus.

As this is a corpus-based study, discussion of features will be preceded by a comprehensive discussion of the corpus used and an analysis of the POS tagger. Insights gained from the corpus will govern the system design.

Note: The Buckwalter transliteration scheme\(^{32}\) will be used throughout this chapter when Arabic examples are introduced.

We have used a Java implementation available in the openNLP project\(^{33}\).

#### 7.2 Dataset and Task

##### 7.2.1 Corpus

Due to the scarcity of Arabic corpora annotated for the NER task, we investigated another alternative that could be used in our experiments. The decision was to use the ACE corpus distributed by LDC\(^{34}\). This corpus is a multilingual annotated corpus for the task of Mention Detection (MD), a slightly different task from the MUC task. One of its subtasks consists of the recognition and classification of all the "pronominal", "nominal" and "named" mentions of entities in the text. Data in the corpus is separated by genre, i.e. type of the data source. The genres which have been used in ACE 2003, 2004 and 2005 are the following:

---

\(^{32}\) [http://www.qamus.org/transliteration.htm](http://www.qamus.org/transliteration.htm)


\(^{34}\) [http://projects.ldc.upenn.edu/ace/data/](http://projects.ldc.upenn.edu/ace/data/)
Broadcast News (BN) and News Wire (NW); Arabic Treebank (ATB); WebLogs (WL). Details of each part are shown in Table 7.1.

<table>
<thead>
<tr>
<th>Part</th>
<th>Source</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACE 2003</td>
<td>BN, NW</td>
<td>40171</td>
</tr>
<tr>
<td>ACE 2004</td>
<td>BN, NW, ATB</td>
<td>155951</td>
</tr>
<tr>
<td>ACE 2005</td>
<td>BN, NW, WL</td>
<td>101244</td>
</tr>
<tr>
<td>All</td>
<td>BN, NW, ATB, WL</td>
<td>297366</td>
</tr>
</tbody>
</table>

Table 7.1: Corpus information

7.2.2 Corpus preprocessing

The ACE corpus is in the form of raw text files of news stories with stand-off annotation in XML files. We have implemented an XML parser to extract text and annotation from the standoff annotated corpus. As this corpus requires preparation for the NER task, the only annotations kept are the person named mentions which are closely consistent with the MUC guidelines. The new corpus format is formatted according to the IOB2 scheme (B for Begin, I for Inside, O for Outside), e.g.:

John       B-PER
met         O
James       B-PER
William     I-PER
.
.
.
O

We have split the corpus into 90% for training and 10% for testing. Table 1 shows the distribution and number of person entities per genre.
Chapter 7: Arabic Person Name Recognition

<table>
<thead>
<tr>
<th>Part</th>
<th>Source</th>
<th>Training set size</th>
<th>Training PER count</th>
<th>Test set size</th>
<th>Test PER count</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACE 2003</td>
<td>BN</td>
<td>13517</td>
<td>361</td>
<td>2062</td>
<td>46</td>
</tr>
<tr>
<td></td>
<td>NW</td>
<td>22521</td>
<td>403</td>
<td>2071</td>
<td>32</td>
</tr>
<tr>
<td>ACE 2004</td>
<td>BN</td>
<td>52875</td>
<td>1090</td>
<td>10554</td>
<td>271</td>
</tr>
<tr>
<td></td>
<td>NW</td>
<td>56640</td>
<td>1395</td>
<td>8469</td>
<td>162</td>
</tr>
<tr>
<td></td>
<td>ATB</td>
<td>23297</td>
<td>483</td>
<td>4116</td>
<td>70</td>
</tr>
<tr>
<td>ACE 2005</td>
<td>BN</td>
<td>19622</td>
<td>536</td>
<td>2301</td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>NW</td>
<td>52643</td>
<td>1091</td>
<td>5122</td>
<td>135</td>
</tr>
<tr>
<td></td>
<td>WL</td>
<td>18356</td>
<td>373</td>
<td>3200</td>
<td>40</td>
</tr>
<tr>
<td>Total</td>
<td>All</td>
<td>259370</td>
<td>5732</td>
<td>37886</td>
<td>807</td>
</tr>
</tbody>
</table>

Table 7.2: Training and tests sets: details and person count per part

7.2.3 Corpus Quality

The corpus contains some noise represented by the insertion of spaces into tokens. Sometimes, the generated words are valid Arabic words. It was not possible to precisely measure how frequent that happens in the corpus. That kind of noise is commonly present in media channels and it is believed that the corpus was not cleaned to maintain this feature.

The example below is found in the file named “NTV20001213.1530.1474” in the ACE 2005 corpus:

[زوجته هيل ري, Zwjth Hyl ry, His wife Hilla ry ]

In that example, the two tokens generated from “Hillary” are valid non-person Arabic words; both words are common nouns that would never be person names. Below is the syntactic information of that phrase:

Zwjth/NN+PRP$ Hyl/NN  ry/NN
Moreover, space insertion into a non-entity token might generate an entity-like token.

According to ACE annotation guidelines, the entity type “PER” refers to any mention that represents people. The subtype class “NAM” covers either names or nicknames\(^35\) of individuals, groups and nations. In the ACE 2005 when “NAM” subtype was further classified to indicate these derivatives. For instance, [العرب, AlErb, Arabs] was tagged as a person of subtype “NAM” in ACE 2003 and 2004. Also, people referred to by their nationalities were annotated as “NAM”. The effect is clear on our dictionary coverage and POS tagging accuracy, as will be discussed later.

### 7.2.4 Task complexity

This task is far more complex in Arabic than in Latin alphabet languages given the lack of capitalization in Arabic proper nouns. However, a more descriptive measure of this complexity is required. To achieve this, we have calculated the percentage of ambiguous tokens in the corpus. Ambiguous tokens are tokens that might fall into the two classes PERSON and OTHER. We have calculated the tag-per-word rate as:

\[
\text{tag - per - word rate} = \frac{\text{ambiguous tokens} \times 2 + \text{unambiguous tokens}}{\text{token count}}
\]

\[= 1.24\%\]

### 7.3 Feature Set

#### 7.3.1 Lexical (LEX)

Three lexical features were used: current, previous and next word. The intuition is that named entities would follow or precede certain keywords or collocations. Moreover, the feature would help to capture entities occurring in both the training and test set.

---

\(^{35}\)http://projects.ldc.upenn.edu/ace/docs/ArabicEDTV4-2-3.pdf
7.3.2 Gazetteer (GAZ)

The use by others of a gazetteer has proven to be effective in this task. So, we have used a list of some 100k Arabic and Arabized names, provided by a government agency. The list contains a large amount of noise, apparently because of data entry issues. Having a large gazetteer means larger coverage but does not guarantee performance improvement, as a larger number of ambiguous tokens would fall into it.

Our first step was to evaluate the gazetteer, measuring its coverage. The number of person tokens in the corpus that also exist in the gazetteer was 5168 out of 11534, about 50% of all person entities in the corpus. However, that figure is not sufficient to predict the effectiveness of the gazetteer.

Further analysis was needed to precisely measure the coverage and quality of our gazetteer before starting our experiments. For that purpose, we turned to the standard IE metrics, Precision, Recall and F-measure. These metrics were calculated on the corpus without building the statistical model:

For the gazetteer coverage, we measure the correctness of the gazetteer based on the class of the token in the corpus as follows:

\[
TP = \text{the number of tokens that exist in the gazetteer and are labelled as PER}
\]

\[
FP = \text{the number of tokens that exist in the gazetteer and are not labelled as PER}
\]

\[
FN = \text{the number of tokens that do not exist in the gazetteer and are labelled as PER}
\]

\[
Precision = \frac{7742}{7742 + 84775} = 8.37\%
\]

\[
Recall = \frac{7742}{7742 + 5465} = 58.62\%
\]

\[
F1 = \frac{981.30}{66.99} = 14.65\%
\]
The low precision was due to the large number of false negatives, that is words existing in the gazetteer and not labelled as person entities in the corpus. This proves the claim that person names are mostly common nouns and adjectives.

The low recall is affected by the large amount of foreign names in the corpus. Also, the definition of the task would be another reason, since names of nations and groups would not be in the gazetteer. Table 7.3 gives a sample of the most frequent person entity tokens that do not exist in the gazetteer. Only one of them is an Arabic token, which is the second one in the list. This token is annotated in the corpus as person class since it refers to Arab as a nation. The remaining tokens in the table are mainly senior international non Arabic politicians.

<table>
<thead>
<tr>
<th>Arabic word</th>
<th>Transliteration</th>
<th>Translation</th>
<th>Number of occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>كلنتون</td>
<td>klyntwn</td>
<td>Clinton</td>
<td>175</td>
</tr>
<tr>
<td>العربية</td>
<td>AlErbyp</td>
<td>Arabic</td>
<td>141</td>
</tr>
<tr>
<td>ايهود</td>
<td>Ayhwd</td>
<td>Ehud</td>
<td>112</td>
</tr>
<tr>
<td>آل</td>
<td>Al</td>
<td>81</td>
<td></td>
</tr>
<tr>
<td>شارون</td>
<td>$Arwn</td>
<td>Sharon</td>
<td>73</td>
</tr>
<tr>
<td>غور</td>
<td>gwr</td>
<td>Gore</td>
<td>52</td>
</tr>
<tr>
<td>بوتين</td>
<td>bwtyn</td>
<td>Butin</td>
<td>48</td>
</tr>
<tr>
<td>اولبرايت</td>
<td>AwlbrAyt</td>
<td>Albright</td>
<td>47</td>
</tr>
<tr>
<td>بوب</td>
<td>bwb</td>
<td>Bob</td>
<td>45</td>
</tr>
<tr>
<td>استرادا</td>
<td>AstrAdA</td>
<td>Astrada</td>
<td>45</td>
</tr>
<tr>
<td>عنان</td>
<td>AnAn</td>
<td>Anan</td>
<td>34</td>
</tr>
<tr>
<td>فلاديمير</td>
<td>flAdymyr</td>
<td>Vladimir</td>
<td>34</td>
</tr>
</tbody>
</table>

Table 7.3: Most frequent person tokens not found in the gazetteer
7.3.3 POS tagging (POS)

POS tagging information did not improve the accuracy of NER systems from English news text as it was subsumed by the capitalization. The lack of capitalization in Arabic along with general-purpose entities suggests we should employ POS tagging. An assigned NNP tag is a strong indication of a proper noun and also, names cannot follow certain tags.

We ran our POS tagger on the corpus, to yield a new corpus format compared to the previous example:

<table>
<thead>
<tr>
<th>Name</th>
<th>POS</th>
<th>Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>NNP</td>
<td>B-PER</td>
</tr>
<tr>
<td>met</td>
<td>VBD</td>
<td>O</td>
</tr>
<tr>
<td>James</td>
<td>NNP</td>
<td>B-PER</td>
</tr>
<tr>
<td>William</td>
<td>NNP</td>
<td>I-PER</td>
</tr>
<tr>
<td></td>
<td>PUNC</td>
<td>O</td>
</tr>
</tbody>
</table>

It was found that 68% of person tokens were tagged as NNP. To better evaluate the tagging accuracy, we used the same IE metrics as above.

Note that most locations and some organizations tokens are also tagged as NNP, which results in an inaccurate measure of performance, which is why precision is highly affected. However, recall is relatively accurate and informative. Moreover, we are calculating how much the NNP tag indicates a person name.

Similar to our calculation of the coverage of the gazetteer, we noted the following for the POS tagging:

\[
TP = \text{the number of tokens tagged as NNP and labelled as person}
\]

\[
FP = \text{the number of tokens tagged as NNP and not labelled as person}
\]

\[
FN = \text{the number of tokens tagged as non-NNP and labelled as person}
\]

Using the same metrics of IE evaluation equations:
The definition of the task would affect the NNP tagging accuracy as some of the person tokens were tagged as adjectives e.g. nationalities and some tagged as nouns e.g. groups. Also, there were 300 English tokens in the corpus, mainly from the WL corpus which represent English user names. There were 72 of them labelled as person names.

Words that were not found in the tagger lexicon were tagged as OOV. The OOV count was 4047 tokens, of which 1638 are person entities, 40.49%. This figure suggests that the tagger accuracy is more affected by the foreign names and not the task definition.

\[
Precision = \frac{8836}{8836 + 12528} = 41.36\%
\]

\[
Recall = \frac{8836}{8836 + 4371} = 66.90 \%
\]

\[
F1 = \frac{5533.97}{108.26} = 51.11 \%
\]

Figure 7.1: POS tags of PER class tokens\(^{36}\)

\(^{36}\) O-POS denotes the remaining POS tagging categories
Table 7.4 shows the summary of performance of the main two features of the system described above:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Gazetteer</th>
<th>POS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision</td>
<td>8.37</td>
<td>41.36</td>
</tr>
<tr>
<td>Recall</td>
<td>58.62</td>
<td>66.90</td>
</tr>
<tr>
<td>F-measure</td>
<td>14.65</td>
<td>51.11</td>
</tr>
</tbody>
</table>

Table 7.4: Gazetteer and POS tagger accuracy

The POS tagging is also used to segment the corpus since we would have compound tags. We are always concerned with the stem of the current word and not clitics. However, we do not want to remove them totally as they have important information. Thus, lexical features are governed by the POS features as follows:

- If the current token is has a proclitic then the previous word is set to the proclitic and the current word is set to the stem.
- If the current word has an eclitic then the next word is set the enclitic and the current word to the stem.
- If the previous word has an enclitic then the stem of the previous word is discarded and only the enclitic is kept as the previous word.
- If next word had a proclitic then the next word is discarded and only the proclitic is kept as the next word.

The POS tagging features are also flattened accordingly. Also, normalization is performed on each token for the letters in the Arabic alphabet that are most likely to cause spelling mistakes discussed in chapter 3 i.e. “أ” and “ي”.
7.3.4 Most frequent non-person tokens (AMB)

In an attempt to improve precision when employing a gazetteer, we use a list of the most frequent non-person tokens. This list is a subset of the gazetteer and was built by finding the most frequently occurring words in the gazetteer that are not annotated as a person entity in the training corpus. Table 5 shows a sample of the highest counts. These tokens are mainly uncommon female person names. However, these are also prepositions known to have a high frequency of occurrence in any text. “An” and “>n” are variants of the same transliterated (Arabized) foreign name “Ann”, which serves as a common particle in Arabic.

<table>
<thead>
<tr>
<th>Token</th>
<th>Transliterated</th>
<th>POS</th>
<th>Number of occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>في</td>
<td>Fy</td>
<td>IN</td>
<td>9195</td>
</tr>
<tr>
<td>من</td>
<td>Mn</td>
<td>IN</td>
<td>5286</td>
</tr>
<tr>
<td>على</td>
<td>Eli</td>
<td>IN</td>
<td>3374</td>
</tr>
<tr>
<td>ان</td>
<td>An</td>
<td>IN</td>
<td>2540</td>
</tr>
<tr>
<td>التي</td>
<td>Alty</td>
<td>WH</td>
<td>1544</td>
</tr>
<tr>
<td>أن</td>
<td>&gt;n</td>
<td>IN</td>
<td>1515</td>
</tr>
<tr>
<td>الرئيس</td>
<td>Alr\ys</td>
<td>NN</td>
<td>748</td>
</tr>
<tr>
<td>بين</td>
<td>Byn</td>
<td>ADV</td>
<td>728</td>
</tr>
<tr>
<td>بعد</td>
<td>bEd</td>
<td>ADV</td>
<td>707</td>
</tr>
</tbody>
</table>

Table 7.5: Sample of the most frequent tokens in gazetteer that are not person entities

7.3.5 Effective Preceding Unigrams (UNIG)

N-grams have been used as an effective feature to capture context surrounding entities. That is, this feature exploits the collocation phenomenon. We collected the most frequent words
preceding person entities. The effect of this feature depends on the technique used to select feature instances. The simplest technique is to use frequency of occurrence, considering the highest frequency to be strongly correlated with the person class. As person entities are not frequent in text, we also generated another list of “tokens not preceding person entities”.

7.3.6 List of most frequent non-person bigrams (BIAMB)

Since we plan to use lexical features that span current, previous and next words, and due to the high false positive ratio caused by our large gazetteer, we used a list of bigrams that we constructed from current and previous words that occur in the gazetteer but that are not person entities. Another list was also generated and employed, of the current and next words. The size of the list was 10k.

<table>
<thead>
<tr>
<th>Arabic</th>
<th>Transliterated</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>شرم الشيخ</td>
<td>$rm Al$yx</td>
<td>125</td>
</tr>
<tr>
<td>حزب الله</td>
<td>Hzb Allh</td>
<td>99</td>
</tr>
<tr>
<td>في مدينة</td>
<td>Fy mdynp</td>
<td>61</td>
</tr>
<tr>
<td>الرئيس الفلسطيني</td>
<td>Alr}ys AlflsTyny</td>
<td>60</td>
</tr>
</tbody>
</table>

Table 7.6: Most frequent non-person bigrams in the gazetteer

7.3.7 List of unambiguous names (UNQ)

Some person entities do not require any processing by the system, as they are unique person names and they will always be so, regardless of the context. It seems to be impossible to collect tokens that could only be person names and not anything else. To generate this feature, we built a subset of the gazetteer without using the corpus. Instead we used the lexicon of our POS tagger, which contains words and their possible POS tags extracted from the Arabic TreeBank. Each word of the gazetteer was checked for its existence in the POS tagger lexicon. If it was always tagged as NNP, it was added to the unambiguous name list.
Chapter 7: Arabic Person Name Recognition

If it did not exist in the lexicon, it was added to the list. The intuition here is that person entities are not frequent in text but, if they are ambiguous, they would have to appear in our ATB corpus. Locations and organizations are also tagged as NNP meaning that, if they exist in the gazetteer, they will be part of this list too. Figure 1 gives an overview of the process with Area (B – C) as our target list. The eventual size of the list was 92k.

![Diagram](image)

Figure 7.2: Gazetteer and Lexicon

### 7.3.8 Trigger (TRG)

Triggers are tokens that occur around person names. They include titles, positions, etc. They have been used with English and proven effective. Here, we have collected a list of triggers containing 200 tokens. The trigger feature is a Boolean feature, testing whether the word is preceded by a token found in the trigger list.

### 7.3.9 Previous Class (Cₙ)

Based on the results of the previous chapter, we used this feature to model the dependency between adjacent words.
7.3.10 Tag of other occurrences (GLB)

This is a global feature used to check if the current token has been previously tagged as a person in the same article. It is common to use distinguishing context such as triggers when an entity is first introduced. Later in the same article, it may be referred to by the first or last name, so with less indicative context.

7.4 Experimental results

7.4.1 Baseline

To demonstrate the effect of our feature set on the model, we first constructed a baseline which involved tagging NEs with their most frequent class in the training corpus. For that purpose we have used CoNLL baseline utility. The performance of the baseline is shown in Table 7.7. It would normally be affected by the tag per word percentage, the repetition of entities in the corpus and degree of human annotation consistency.

7.4.2 Effect of Proposed Feature Set

The results of proposed feature combinations are given in Table 7.7. In general, all features had a positive effect on the model. Some of them give a very low improvement when they are not combined with some other features, given the correlation between them.

After adding the POS tagging feature to the system, the performance was increased significantly, proving the strong correlation between POS tagging and NER in Arabic language. This effect is mainly because of NNP tags. The POS feature improved the performance by about 7%.

In the case of Arabic, we sometimes have combined POS tag. We did not split them in the corpus but rather, that was done during processing. The reason is that we avoid any error caused by the segmentation module.

The GLB feature of a person entity in the same article did not benefit the system much when employed on its own, in addition to lexical features. However, the system with all
features active was negatively affected when the global feature was removed. For this feature to work, the "other occurrences" should have been correctly labelled.

The GAZ feature did not improve the performance much when first employed. Thus, we added some constraints to govern the process of generating the gazetteer and unambiguous features, based on the POS tag. If the POS of the current word is OOV, the gazetteer and unambiguous features are set to true, as 40% of OOV words are person names. The same is done for the previous and next words, altering only the gazetteer feature. The result after this modification is the one given in Table 7.7.

With respect to AMB feature, we ran a series of experiments to find the best list size empirically. A list of the 500 most frequent ambiguous tokens showed the best accuracy.

Collecting the preceding unigrams of the person class, UNI, also did not add any gain to the performance. Probably this information is already captured by the lexical feature used. Thus, we used a simple equation to calculate the correlation, relying on relative frequency of occurrence of the word preceding the person class tokens, divided by the number of all occurrences:

\[
\text{Correlation} = \frac{\text{Count}(w_{-1}, \ w \ \text{tagged as} \ C)}{\text{Count}(w_{-1})}
\]

; where \text{Count} denotes the number of occurrences.

For each class C, unigrams were ranked based on their correlation and the highest ranked were included in the list of preceding unigrams.

The effect of the correlation equation is illustrated by considering the conjunction [و , \ w, and], which was ranked second of the most frequent words preceding person entities, but it was also among the most frequent preceding non-person entities. Using the correlation measure, this token was not included.

When we first employed the UNIQ list in our system, there was no significant improvement. The reason was that we had not checked the content of this list against the corpus. We then checked it and found out that some entities in the list were not person
entities. We removed the most frequent non-person entities from the list, leading to some improvement. Then, we noticed that, for a token to be in this list, it should have been found in the gazetteer first. That is, OOV words will not be in both lists. So, we used POS tag information to change this feature: any word that was tagged as OOV was now labelled as unambiguous. In this part of our experiment, we used POS tagging information as a selection criterion, not as a feature in the system.

In similar systems, the previous word was checked against the trigger list and, if found, the trigger feature \((\text{TRG})\) was set to true. However, in Arabic, this is not sufficient, as adjectives follow nouns. For example, without an adjective we have:

الرئيس جورج بوش

President George Bush

However, if the trigger is followed by an adjective, the trigger will be one token away from the entity, in Arabic. The trigger would still be adjacent to the entity in English. The example below shows the difference; the trigger is underlined and the adjective is in italic.

الرئيس السابق جورج بوش

Former President George Bush

This problem expands when we have more than one adjective. Testing on our corpus led to the system being improved when the trigger window was increased to three tokens.

The class of the previous word \(C_{-1}\) improved the system with all features by 4.8% and that is because we are using a token classification algorithm. We would like to note that all features would have a higher effect if this feature was included in all experiments.

Our last experiment was to leverage the previous context to correct the POS feature of the previous word. When processing a given word, if the previous word was labelled as PER
then the POS feature of previous word is set to NNP. This caused the performance to increase to 75.59%.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>37.99</td>
<td>50.21</td>
<td>43.26</td>
</tr>
<tr>
<td>LEX</td>
<td>61.04</td>
<td>58.45</td>
<td>59.72</td>
</tr>
<tr>
<td>LEX+GAZ</td>
<td>68.22</td>
<td>59.30</td>
<td>63.45</td>
</tr>
<tr>
<td>LEX+UNIQ</td>
<td>68.71</td>
<td>59.51</td>
<td>63.78</td>
</tr>
<tr>
<td>LEX+GAZ+UNIQ</td>
<td>69.50</td>
<td>60.04</td>
<td>64.42</td>
</tr>
<tr>
<td>LEX+POS</td>
<td>68.22</td>
<td>63.66</td>
<td>65.86</td>
</tr>
<tr>
<td>LEX+POS+Gaz+UNIQ</td>
<td>71.11</td>
<td>63.55</td>
<td>67.12</td>
</tr>
<tr>
<td>Lex+POS+GAZ+UNIQ+AMB+TRG+GLB</td>
<td>72.40</td>
<td>68.01</td>
<td>70.14</td>
</tr>
<tr>
<td>LEX+POS+GAZ+UNIQ+AMB+TRG+GLB+C₁</td>
<td>79.76</td>
<td>70.35</td>
<td>74.76</td>
</tr>
<tr>
<td>All with POS₁ based on C₁</td>
<td>81.81</td>
<td>70.24</td>
<td>75.59</td>
</tr>
</tbody>
</table>

Table 7.7: Performance of baseline and different feature combinations

7.5 System comparison

There have been a number of research experiments that involved the ACE dataset. (Benajiba et al. 2008a) reported evaluation on the overall classes of entities (6 classes) without giving per-class accuracy. Moreover, this system was trained and tested per genre. Their best result was on BN 2003 which is the smallest part of the corpus, achieving 83.5% on F1. Their worst accuracy was on WL 2005, with 57%, which is justified by the noise exhibited in web forums. The second worst performance was NW 2004 (72.4%), and this part is the largest among the ACE datasets. Thus, it is not possible to compare what was
achieved only on the person class with their reported results on 6 classes and on individual genre experiment.

(AbdulHamid and Darwish 2010) selected part of the ACE corpus for their experiments. They used ACE 2005, excluding WL dataset, which is the very noisy part of the corpus. Also, they excluded entities with subtypes “GROUP” and “NATION”, keeping only “INDIVIDUAL” person entities, which are the most appropriate for the NER task, but this feature was not available in previous ACE datasets. Their reported accuracy on the person class was 81% F1. It is worth noting that their system relied only on character N-grams and word roots.

Another experiment by (Benajiba et al. 2009b) where they used voting to combine SVM, CRF and MEM classifiers adopted the same experimental settings as above without per class accuracy.

A further recent experiment in (Benajiba et al. 2009a) was conducted using a per genre evaluation. However, their results on the person class varied from 56.1% on the 2005 WL genre to 81.4% on the ACE 2003 BN genre. Their system with only the context feature achieved more 70% in most genres which could be the reason for their very high accuracy. In addition, they have used more features than what is used our experiment.
Chapter 8  Token Classification vs. Sequence Labelling for Arabic NER

In this chapter we evaluate our approach in previous chapter on another dataset as it was built for the Arabic NER task. Moreover and based on our previous chapter results, we try to find a way to integrate adjacent context efficiently into our design.

Thus, we move to NER as a sequence labelling problem. This technique has been widely used in various NLP problems, POS, parsing ... etc including NER. The most widely used algorithms have been discussed in chapter 2. In this chapter, we compare NER as a token classification problem to a sequence problem using state-of-the-art technique. In addition, we compare our design with previous work. Also, this experiment will be enhanced to include location and organization in addition to person class.

8.1 Hidden Markov Support Vector Machine

The Hidden Markov Support Vector Machine (HM-SVM) was proposed and explained in (Altun et al. 2003), for labelling sequence data. HM-SVM is a discriminative learning technique based on a combination of the two most successful machine learning algorithms: Support Vector Machine (SVM) and Hidden Markov model (HMM). HM-SVM addresses all of the shortcomings of HMM, while retaining some of the key advantages of HMM, namely the Markov chain dependency structure between labels and an efficient dynamic programming formulation. Both HM-SVM and CRF adopt a discriminative approach to modeling and can account for overlapping features (labels can depend directly on features of past or future observations). In addition, HM-SVM includes two additional crucial properties inherited from SVM: the maximum margin principle and a kernel-centric approach to learning non-linear discriminant functions.
Chapter 8: Token Classification vs. Sequence Labelling for Arabic NER

The one-sack reformulation of the training problems is solved by using the cutting-plane algorithm, which makes the complexity of the training step for HM-SVM linear in the number of training samples (Joachims et al. 2009).

HM-SVM has been used for many applications recently, showing greater success than other classification and sequence labelling techniques. When it was first introduced in (Altun et al. 2003), it was compared with a number of state-of-the-art techniques (CRF and HMM) on NER and POS tagging experiments, where it produced lower error rates than either. Since then, it has been receiving attention from the field.

HM-SVM outperformed conventional SVM and MEM by 3% on functional label assignment in the Chinese language (YUAN and REN 2009). In the biomedical domain, it was compared with Neural Networks and CRF in the task of finding protein interactions. The HM-SVM outperformed the other two approaches in a cross-validation experiment (B. Liu et al. 2009).

A recent experiment on sign language showed a big gap in performance between HMM to HM-SVM by 18%. (Michael et al. 2011).

In information extraction from literature, HM-SVM was used to parse references to generate author, date, event, etc. HM-SVM was also compared to SVM and CRF, showing closeness of the three, but HM-SVM had a better accuracy (X. Zhang et al. 2011).

SVM\textsuperscript{hmm} toolkit\textsuperscript{37} version 3.10 is used as our implementation of HM-SVM. We adopt the first-order Markov HM-SVM with linear kernel. We have used the default parameters setting except parameter c, which was set to 1000 as it gave the best result.

8.2 Data analysis

8.2.1 Corpus

The corpus used in this experiment was the ANERCorp, developed by Benajiba and freely available on his website\textsuperscript{38}. There were a number of reasons for this selection:

\textsuperscript{37} http://www.cs.cornell.edu/people/tj/svm_light/svm_hmm.html

\textsuperscript{38} http://www.cs.cornell.edu/people/benajiba/ANERCorp.html
1. There has been number of research studies involving this dataset with very detailed result analysis.

2. The annotation specifically follows the NER task. Thus it is more appropriate to validate our approach discussed in the previous chapter, where our results might be affected by the definition of the task it was built for.

3. It has a simple IOB2 format and is contained in one file so that it is easier and more obvious to split it to conduct cross-validation.

4. Although it was annotated only by a single annotator, we believe he is an expert in the field.

5. Our tagger was built from the ATB corpus, which was also used as part of the ACE dataset used in the previous chapter, and would thus give a better tagging accuracy.

<table>
<thead>
<tr>
<th>Class</th>
<th>NE token count</th>
<th>NE count</th>
<th>Token/Entity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person</td>
<td>6438</td>
<td>3238</td>
<td>1.9</td>
</tr>
<tr>
<td>Organization</td>
<td>3402</td>
<td>1912</td>
<td>1.7</td>
</tr>
<tr>
<td>Location</td>
<td>5003</td>
<td>3825</td>
<td>1.3</td>
</tr>
<tr>
<td>All</td>
<td>14843</td>
<td>8975</td>
<td>1.6</td>
</tr>
</tbody>
</table>

Table 8.1: Corpus NE class distribution

Table 8.1 shows the NE classes of the corpus; both tokens and phrases. Also, it shows the token rate per phrase. The rate confirms the fact that location names tend to be single tokens, in contrast to person and organisation classes, especially in news articles.

### 8.2.2 Corpus POS tagging

We ran our tagger discussed in chapter 5 on the corpus to make it ready for the learning algorithm. Figure 8.1 shows the overall POS tagging of the NE classes.

---

The number of OOV of the NEs is larger than in the ACE dataset and this might be due to using the ATB as part of ACE, which also was used in building the POS tagger and hence provided better tagging decisions.

The OOV word count was 2829, 1.8% of the whole corpus. That is a very good coverage of our tagger. However, 72% of OOV were NE tokens. Person class POS tagging is illustrated in Figure 8.2 where it was the most affected class by the OOV tokens. We observed that the corpus has a high percentage of foreign names due to being from international news coverage. The OOV percentage was higher than that of the ACE datasets by 6% in the person class, suggesting that the tagger coverage was affected by being built from part of the data used. Also, we found some spelling mistakes that would also be a source of OOV tokens, but we did not attempt to correct them. With respect to location class, 73% of location names were tagged as NNP compared to 79% in our gold standard set used in chapter 6, this indicates that locations are generally limited in news articles and also that our tagger has a good coverage of location names. Person and location classes are more regular in terms of their POS tags than organizations. Also, location names are limited given that OOV is ranked 4th, while OOV ranked second in the person class. It is worth noting that the location class was not very different from the gold standard tagging experiment in chapter 6.
where it was 97%, while here it was 73%. The organization class is still the most ambiguous class in terms of its POS tagging.

![Figure 8.2: POS tag distribution of each NE class excluding O class](image)

### 8.3 Feature set

We have used a different feature set for each class depending on the availability of the feature as follow:

#### 8.3.1 Person Class

The person class feature set is the same as the one used in chapter 7. However, since the task is different, we have recompiled the N-gram lists that were created from the ACE corpus. Our unique names list was not altered as the ACE dataset was not involved in creating it, only the gazetteer and the POS tagger lexicon.
8.3.2 LOCATION

For this experiment, we used only the following features, without any lists built from the corpus:

- **Lexical (LEX)**: defined for previous, current and next tokens.

- **Gazetteer (GAZ)**: defined for previous, current and next tokens.

The gazetteer that we have used was built in an automated way. The countries list was not hard to collect from the web. For cities, we have downloaded the list of world cities from GeoWorldMap which has only an English version of 37k cities. The format of the data consisted of the name of each city with various longitude and latitude and time zone. To create an Arabic version, we have used GOOGLE translate service to translate the list. The first translation was not good enough so we have added the term “City of” before each city, which improved the translation as it was affected by the context. Then we removed any occurrences of words that were not translated and kept in English, which means that no translation was provided. We used same steps as before to build our unique person names list. Each token is searched in our lexicon and if it was found to be tagged other than with NNP, it is removed, otherwise it was kept in the list. The final list contains 24k tokens.

- **POS tagging (POS)**: defined for previous, current and next tokens.

- **Trigger (TRG)**: defined for the preceding three tokens. The trigger list has 30 location prefixes.

8.3.3 ORGANISATION

- **Lexical (LEX)**: defined for previous, current and next tokens.

- **POS tagging (POS)**: defined for previous, current and next tokens. Also, we have used POS bigrams of current word with previous in addition to the current word with the next

---

word POS tag. The intuition is that the POS context of organization names is more likely to be a sequence of nouns or nouns and adjectives.

- **Trigger (TRG):** defined on a window of three tokens preceding the current word.

### 8.4 Gazetteer Coverage

Prior to carrying out the proposed experiments, we have calculated the coverage of the gazetteers since they are a very critical resource in our system design. We have used the IE metrics of precision, recall, and F1, which were previously discussed in chapter 4 and have been used throughout this thesis.

For the person class gazetteer:

\[
TP = \text{the number of tokens that exist in the gazetteer and are labelled as PER}
\]

\[
FP = \text{the number of tokens that exist in the gazetteer and are not labelled PER}
\]

\[
FN = \text{the number of tokens that do not exist in the gazetteer and are labelled as PER}
\]

\[
\text{Precision} = \frac{5742}{5742 + 46668} = 10.96 \%
\]

\[
\text{Recall} = \frac{5742}{5742 + 689} = 89.29 \%
\]

\[
F1 = \frac{2 \times 10.96 \times 89.28}{10.96 + 89.28} = 19.52 \%
\]

Looking at these figures, we inferred that a large coverage gazetteer would not guarantee a better accuracy. This is caused by person names that are in form of nouns and adjectives. Compared to the accuracy of our gazetteer in the previous chapter when using a different dataset, we have a 5% increase in the F1 measure, which is caused by the difference in the task definitions used in creating the two datasets.
For the location class gazetteer:

TP = the number of tokens that exist in the gazetteer and are labelled as LOC

FP = the number of tokens that exist in the gazetteer and are not labelled as LOC

FN = the number of tokens that do not exist in the gazetteer and are labelled as LOC

\[
\text{Precision} = \frac{3614}{3614 + 27310} = 11.67 \%
\]

\[
\text{Recall} = \frac{3614}{3614 + 1419} = 71.81 \%
\]

\[
F1 = \frac{2 \times 11.67 \times 71.81}{11.67 + 71.81} = 20.01 \%
\]

Even though our gazetteers had a different acquisition method, we had almost the same accuracy in both. It is worth noting that the location class has a better precision even though it was created automatically, since location names are less used as non-NE tokens.

8.5 Experiments

We split the corpus into 6 folds and used the top fold as a test set and the remaining 5 as a training set. First, we built a baseline that involved tagging each NE with its most frequent NE class if present in the training set. The baseline was built using CoNLL’s baseline utility\(^{40}\). The results of the baseline are given in Table 8.2. Location names seemed to be limited in international news articles and hence performed better than the other two classes. Also, it benefit from being mostly single token entities.

\(^{40}\) http://www.cnts.ua.ac.be/conll2002/ner/bin/baseline.txt
Chapter 8: Token Classification vs. Sequence Labelling for Arabic NER

<table>
<thead>
<tr>
<th>Accuracy %</th>
<th>PRE</th>
<th>REC</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>PER</td>
<td>52.19</td>
<td>34.05</td>
<td>41.21</td>
</tr>
<tr>
<td>LOC</td>
<td>77.34</td>
<td>67.91</td>
<td>72.32</td>
</tr>
<tr>
<td>ORG</td>
<td>44.74</td>
<td>47.22</td>
<td>45.95</td>
</tr>
<tr>
<td>All</td>
<td>41.54</td>
<td>26.67</td>
<td>32.48</td>
</tr>
</tbody>
</table>

Table 8.2: Baseline of most frequent classes

In the following two experiments, we will build a classifier from training set using the two compared algorithms (MEM and HM-SVM) and evaluate it on the test set, considering only the person class. Based on the results, we will carry out our cross-validation on the three classes.

8.5.1 Maximum Entropy Modeling Classifier

We have used the same implementation of MEM provided by OpenNLP that was used in chapter 7. The second column in Table 8.3 shows the results of the performance (F1) for the main features used in our model. Results in the third column shown the result using the same sets of features, but using the previous class as a feature and changing the POS tag of the previous word based on that class, as we described in chapter 7.

Our results in Table 8.3 prove the efficiency of the proposed feature set. The best feature when combined with lexical feature was the POS feature in both approaches. A 9% increase in performance was observed when using only the POS feature combined with LEX feature, rising to 16% when changing the previous tag based on the previous class. Although our unique list has a positive impact on the performance, especially when combined with the gazetteer, the improvement in the performance was not as large as in the experiments described in the previous chapter since we had a larger number of OOV person names that would not fall in both dictionaries. The most significant improvement was achieved by integrating the POS feature with the previous tag.
The “All” measure indicates the performance of the system with all proposed features including bigrams and the list of ambiguous tokens. We did not include details of them in the table as they did not cause a very significant improvement of the system.

We observe that the lexical feature alone scored 59% in the previous chapter while it scored 50% on this dataset. This might be due to the size of the dataset, since ANERCorp is half the size of the ACE dataset, which means more contexts of NEs.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Without Cₜ</th>
<th>With Cₜ</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEX</td>
<td>50.37</td>
<td>50.37</td>
</tr>
<tr>
<td>LEX+GAZ</td>
<td>54.12</td>
<td>55.07</td>
</tr>
<tr>
<td>LEX+UNIQ</td>
<td>55.10</td>
<td>56.59</td>
</tr>
<tr>
<td>LEX+GAZ+UNIQ</td>
<td>56.23</td>
<td>57.12</td>
</tr>
<tr>
<td>LEX+POS</td>
<td>60.00</td>
<td>66.54</td>
</tr>
<tr>
<td>All</td>
<td>61.11</td>
<td>69.18</td>
</tr>
</tbody>
</table>

Table 8.3: Effects of combining features using different approaches

8.5.2 HM-SVM classifier

8.5.2.1 SVM Format

SVM does not accept categorical features, which are the usual format of most NLP features. It accepts only binary numerical features. Thus, it was required to convert our data to SVM data format. The data format is explained on the SVM_hmm tool developer’s website.⁴¹

Each line in the corpus file holds one class and the feature vector of the associated token, and an optional comment, in the following format:

`TAG qid:EXNUM FEATNUM:FEATVAL FEATNUM:FEATVAL .. #comment`

---

TAG is a natural number that identifies the NE class \( C \) that is assigned to the example. The EXNUM gives the example number. The first line with a given EXNUM value is interpreted as the first element of the sequence, the second line as the second element, etc. All lines with the same EXNUM should be in the same sentence and have to be in consecutive order.

We have used a simple method that involved a hash-table. For each “feature/value” in our feature vector we generate one feature number and set it to 1 and store it in the hash-table. If the same “feature/value” is encountered again we retrieve the same feature number from the table. Each sentence is a sequence and would have a different unique \( qid \). Then, the resulting feature numbers are sorted in increasing order. Consider the following example:

John met Mike. Mike was in UK.

If we have these three feature available: \( f_1 \) : the word itself, \( f_2 \) : the POS tag, \( f_3 \) : the POS tag of the previous word, and \( C \) as the NE class or label. Table 8.4 shows the mapping from symbolic to numeric feature format required by SVM\(^{HMM} \).

<table>
<thead>
<tr>
<th>Categorical features</th>
<th>SVM mapping</th>
<th>SVM feature vector</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f_1 )</td>
<td>( f_2 )</td>
<td>( f_3 )</td>
</tr>
<tr>
<td>John</td>
<td>NNP</td>
<td>PUNC</td>
</tr>
<tr>
<td>met</td>
<td>VBD</td>
<td>NNP</td>
</tr>
<tr>
<td>Mike</td>
<td>NNP</td>
<td>VBD</td>
</tr>
<tr>
<td>.</td>
<td>PUNC</td>
<td>NNP</td>
</tr>
<tr>
<td>Mike</td>
<td>NNP</td>
<td>PUNC</td>
</tr>
<tr>
<td>was</td>
<td>VBD</td>
<td>NNP</td>
</tr>
<tr>
<td>in</td>
<td>IN</td>
<td>VBD</td>
</tr>
<tr>
<td>UK</td>
<td>NNP</td>
<td>IN</td>
</tr>
</tbody>
</table>

Table 8.4: Data mapping
8.5.2.2 HMM-SVM Results

Our approach used in the MEM experiment, discussed in chapter 7, that involves modifying the tag of the previous word based on the previous class, was not applicable when using SVM\textsuperscript{hmm}. That is because it would require modifying the algorithm source itself. When applying HM-SVM, this feature will be disabled since we do not have access to the features in the testing phase. Also, the global feature, that stores the class of other occurrences of the word is also disabled, as it is built online, which also requires the code to be modified.

Table 8.5 shows the effect of combining features when using HM-SVM combined with our best results obtained when using MEM. HM-SVM was able to outperform MEM, which suggests that it had a better way of integrating adjacent word features.

<table>
<thead>
<tr>
<th>Feature</th>
<th>MEM</th>
<th>HM-SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEX</td>
<td>50.37</td>
<td>61.40</td>
</tr>
<tr>
<td>LEX+GAZ</td>
<td>55.07</td>
<td>64.63</td>
</tr>
<tr>
<td>LEX+UNIQ</td>
<td>56.59</td>
<td>65.04</td>
</tr>
<tr>
<td>LEX+GAZ+UNIQ</td>
<td>57.12</td>
<td>65.63</td>
</tr>
<tr>
<td>LEX+POS</td>
<td>66.54</td>
<td>69.75</td>
</tr>
<tr>
<td>All</td>
<td>69.18</td>
<td>74.49</td>
</tr>
</tbody>
</table>

Table 8.5: Comparison of the effect of using different features (MEM vs. HM-SVM)

It is not correct to compare the two algorithms as they do not use the same internal approach in finding the class of a token. The conventional SVM could be compared with MEM but not SVM-HMM. Thus, our aim here is to find the best way to integrate our features into the NER model.
8.6 Cross Validation with HM-SVM

Encouraged by the performance of the HM-SVM algorithm on the first fold of the corpus, we have conducted a cross-validation experiment on the whole corpus. We have adopted a 6 fold split of the corpus in order to compare with others.

On each fold, we have built an independent classifier for each class. Results of these experiments are shown in Table 8.6.

Our best average performance was on the location class, as it was easier in terms of the context of POS tags and also due to the repetition of location entities in the corpus.

Our worst result was on the organization class, since there was no gazetteer employed and also due to the syntactic structure of their naming systems that make them look like any other Arabic phrase.

We have analyzed the differences between the results of the best and worst fold, fold 0 and fold 2. It was found that the number of person tokens tagged as NNP in fold 0 was double that in fold 2, which indicates the POS accuracy effect of the NER.

<table>
<thead>
<tr>
<th>Class</th>
<th>PERSON</th>
<th>LOCATION</th>
<th>ORGANIZATION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PRE</td>
<td>REC</td>
<td>F1</td>
</tr>
<tr>
<td>Fold</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>80.23</td>
<td>69.52</td>
<td>74.49</td>
</tr>
<tr>
<td>1</td>
<td>74.64</td>
<td>60.34</td>
<td>66.73</td>
</tr>
<tr>
<td>2</td>
<td>73.11</td>
<td>55.46</td>
<td>63.07</td>
</tr>
<tr>
<td>3</td>
<td>75.84</td>
<td>67.50</td>
<td>71.43</td>
</tr>
<tr>
<td>4</td>
<td>78.44</td>
<td>69.43</td>
<td>73.66</td>
</tr>
<tr>
<td>5</td>
<td>80.66</td>
<td>55.58</td>
<td>65.81</td>
</tr>
<tr>
<td>Avg.</td>
<td>77.15</td>
<td>62.47</td>
<td>69.2</td>
</tr>
</tbody>
</table>

Table 8.6: 6-fold Cross Validation
8.7 Comparison with previous work

(Benajiba and Rosso 2008) reported an accuracy of 89%, 61% and 73% for location, organization and person classes, respectively. A separate classifier was built for each class using CRF. Their system employed POS tagging, a gazetteer, base phrase chunking and nationalities list. We could not compare our results with this system nor with the previous one without knowing the exact dataset split, given the large variation in the performance of each fold.

Another cross validation experiment in (Benajiba et al. 2008b) reported an overall accuracy of 80.3%. Later, (Benajiba et al. 2009b) reported an accuracy of 81% on an updated version of this dataset, which is a very high accuracy. However, we are not sure how the dataset was enhanced and how much the annotation was affected. Moreover, no per class accuracy was given to compare with.

LingPipe 42 is a collection of NLP tools that contains a state-of-the-art HMM-based NER that was applied to a number of languages including Arabic. A cross validation experiment on ANERCorp was published on its website.

(AbdelRahman et al. 2010) used a bootstrapping technique in order to enrich their system with more NE patterns. Hence, unlabelled data was used, mainly crawled from the web. Their processing was a sequence of three modules. The system is targets ten NE classes including job, device, car, etc. The first module was a CRF classifier built using POS tagging, base phrase chunk, morphological, character N-grams and semantic group features.

In (Benajiba et al. 2009a), the authors reported per class cross-validation of 78.5% on the person class, 89.6% on the location class and 64.3% on the organization class. Interestingly, the use of the context feature alone achieved 74% accuracy on the three classes, which is higher than the performance achieved by the other compared systems even with the use of unlabelled data used by (AbdelRahman et al. 2010). Their gazetteer did not improve the system but rather degraded performance.

42 http://alias-i.com/lingpipe/demos/tutorial/ne/read-me.html
We shared the same criteria used by (AbdelRahman et al. 2010) to split the corpus into 6 folds of 25,000 words each. LingPipe shares the same number of folds but the split was based on number of sentences, with each fold having 815 sentences.

We have selected the three systems that used exact or close experimental settings to ours; (Benajiba et al. 2009a), (AbdelRahman et al. 2010) and Lingpipe. In Figure 8.3, a comparison of the four systems is illustrated, showing the superiority of (Benajiba et al. 2009a). Our system outperforms (AbdelRahman et al. 2010) and Lingpipe on person and location classes. The performance shown for the system of (AbdelRahman et al. 2010) is the one without the use of any unlabelled data to bootstrap the system. In the organization class, the performance of our system is between the other two, although all three systems perform with similar accuracy. It is worth mentioning that LingPipe did not use any features other than character N-grams in their system. Even when the unlabelled data was used in (AbdelRahman et al. 2010), our system still performs better on the person class. However, their bootstrapping technique was very powerful on the other two classes, specially on the organization class.

Figure 8.3: Comparison of the four systems without the use of unlabelled data
Another experiment on the same dataset was conducted in (AbdulHamid and Darwish 2010) where they reported accuracies of 88%, 73% and 82% for Location, Organization and Person classes, respectively. They used CRF to build a classifier for each class. Their test set was selected randomly, meaning that we cannot compare our results with theirs. Interestingly, they relied mainly on character N-grams in their design, without the use of any other morphological, POS tagging and gazetteer features. They suggested that leading and trailing N-grams are sufficient to capture most of the Arabic morphological features and hence POS tagging.
Chapter 9 Conclusion

9.1 Thesis summary

Arabic NER is a very challenging task for various reasons. Challenges arise at different levels of analysis. These do not affect the NER task directly, but rather affect pre-processing steps required to approach NER successfully.

The lack of capitalization in Arabic is the main challenge for Arabic NER. This lack is more significant due to the fact that names are indistinguishable from adjectives and common nouns. Our main hypothesis of this research is that Arabic NER is very closely bound to POS tagging.

We shed light on those Arabic language features that affect Arabic NER both directly and indirectly. Subsequently, we conducted a comprehensive survey of the task and techniques of NER in Arabic and in other languages, and in different domains.

To prove our hypothesis, we have built an Arabic POS tagger, using TBL, from the Arabic Tree Bank, which yielded 96.6% accuracy on an unseen dataset. This result was higher than (Diab et al, 2004; Mansour et al, 2007), who, similar to our approach, did not employ any morphological features in their designs. We found that the NNP class achieved the lowest accuracy, proving our subsidiary hypothesis that ambiguous tokens are more likely used as non-NNP tokens in text. Details of our tagger design are given in Chapter 5 and results are given in Table 5.2.

Then, to provide suitable training and test data, we annotated 70,000 words of the Arabic Tree Bank with their named entity classes. A first experiment was conducted to measure the ambiguity caused by proper nouns in the form of common nouns and adjectives. It was
found that 28% of the tokens in the corpus were ambiguous: they have two possible NE classes.

We analysed our corpus to find the correlation between Arabic NER and POS tagging. Our analysis revealed that proper nouns have a large degree of regularity in terms of their POS tagging context. It was found that location entities follow a preposition in 56% of their occurrences in the corpus.

Next, we built an Arabic NER classifier using Maximum Entropy Modelling employing lexical and POS features. The resulting accuracy was 59% F1. Given that the POS tagging was gold standard, we used the lexicon of our tagger to tag each token in the corpus with its most frequent tag in the POS tagging corpus, which caused the accuracy to drop by 4%, proving the sensitivity of NER to POS tagging errors.

Our attempt to improve the performance was by the use of gazetteer features. When they were employed, there was no significant improvement to the accuracy. We investigated the reason for these results; we found that although the gazetteer has a very good recall, it also has a very low precision, which has been discussed. We then investigated using the class of other occurrences of the word being processed, together with a list of triggers. Both had a positive impact on the accuracy. The last feature that was investigated is the class of the previous word, as named entities mostly consist of a sequence of tokens. Our system when employing the most effective features achieved 76.4% F1.

As our first experiment was carried out on small dataset, we decided to investigate the features on standard set. Using ACE Arabic part, we have selected annotation that indicates entity name and excluded others used for mention tracking task. We have selected the person class to investigate our approach. Our POS tagger discussed in chapter 5 was run on the corpus first then we conducted some analysis of POS tagger and gazetteer accuracy. The accuracy of the gazetteer, which includes 100k person names, was 14.7%. This is due to very low precision, because of the task definition, on which names of nations and groups are labelled as person entities. Testing our tagger on the corpus showed that it was able to tag 68% of person entity tokens as NNP. It was also found that 40.5% of the person tokens are OOV. To deal with the low precision of our gazetteer, we carried out the following steps:
- We collected the most frequent words in the training set which occur in our gazetteer and are not labelled as person entity.

- We also collected the most frequent bigrams that occur in the gazetteer, as the system will be highly affected by adjacent words.

- We have used our POS tagger lexicon built in chapter 5 to collect gazetteer items that were never tagged as a non-NNP. Words that are in the gazetteer and not in lexicon were considered unique person names. The list of unique names list was used to generate a new feature.

- Encouraged by the increase in performance caused by the previous word class in chapter 6, we have also used the previous class to modify the POS tag of the previous word. If the previous word is labelled as PERSON by the classifier, its POS tag is set to “NNP”.

We then built a classifier using MEM with the above features, along with all features used in chapter 6. The results in Table 7.7 proved the effectiveness of our approach improving the accuracy from the baseline of 43.3% to 75.6% on F1. Using the unique names list was able to improve the accuracy more than the gazetteer used to build it. When both were employed, the performance was better than when they were used individually. Our results were compared with previously reported results on the same dataset although the splitting of data into training and test sets was not the same in each case. Researchers who worked on this dataset ran their experiments per genre, which would definitely give better results. Thus, it was not feasible to compare our work with theirs. However, we noted their results in section 7.5.

Our features had a different impact on the system when the previous word class was employed. This indicates the correlation between classes and adjacent word features, which was the focus of chapter 8. Sequence labelling techniques were proposed to deal with such a case; HM-SVM is one such state-of-the-art technique. We used a freely available dataset that was used by other researchers in order to compare their work with ours. Our experiment discussed in chapter 8 was on person, location and organization classes. To have a good coverage location gazetteer, we used an MT translation tool (GOOGLE translate) to
translate the English version of GeoWorldMap\textsuperscript{43}, then we have used our POS tagging lexicon to clean errors. If the Arabic word generated by the translation tool has a tag other than NNP in the lexicon, it is not stored in the unique location names list, as is the case with our person names unique list. The location gazetteer had almost the same quality as the person gazetteer when we calculated the recall and precision of lookup against our corpus.

Our experiment of evaluating classifiers built with MEM and HM-SVM for the person class showed the superiority of the latter, as shown in Table 8.5. The superiority was due to the fact that the classifier was able to model the dependencies between adjacent classes when augmented with HMM and also that SVM uses state of the art kernel method techniques for classification.

We carried a cross-validation experiment on 6 folds of the corpus for each NE class. The results in Table 8.6 showed our best result was on the location class, followed by the person class. We compared our results with four systems using the same settings, and our system ranked second for person and location classes, and ranked third for the organization class since we did not employ any specific features for the organization class, see Figure 8.3. We were able to achieve good results with simple features, compared to the other systems that used many knowledge sources in their designs.

\section{Thesis Contribution}

The main contributions of this thesis are summarized as follows:

1. We surveyed Arabic language features and how these features affect the NER task.
2. We surveyed NER in general and the most successful approaches.
3. We surveyed the most relevant work on resolving ambiguity of Arabic NER.
4. We measured and proved the strong correlation between Arabic NER and POS tagging.
5. We integrated Arabic POS tagging with a large gazetteer efficiently.

\footnote{\text{http://geobytes.com/FreeServices.htm}}
6. We measured the effectiveness of a unique name list.

7. We demonstrated that careful integration of features makes a difference.

8. We showed that focusing on non NEs in bigrams is a promising feature due to the normally low rate of occurrences of NEs in text.

9. We built a location gazetteer automatically using the GOOGLE translate tool and then used a POS tagger lexicon to filter incorrect output. This gazetteer had 71% recall on our test set.

10. We investigated and proved that SV-HMM is a very powerful sequence labelling technique compared to MEM.

11. Our approach of employing a unique names list along with the large gazetteer and POS tagging features is recommended for other languages and domains that lack proper noun distinguishing features.

9.3 Limitation

This work did not attempt to tackle the problem of noisy text caused by spelling errors. It does not using any character N-grams, which make a word with a spelling error equal to any OOV word. Given the high percentage of NEs that are OOV, words with spelling mistakes would be more likely to be classified as an NE.

Our approach was evaluated on news text. Using features that have a sequential nature would not be appropriate for other genres in which NEs often occur as single tokens and titles are dropped, such as emails and weblogs.

The system that we built is presumably very sensitive to the locale of the named entities. We expect that its performance will be degraded if the text has a high percentage of foreign proper names.
9.4 Future Directions

1. To deal with OOV tagging that highly affects our system, we plan to use GOOGLE translate to disambiguate OOV words. Any word not found by the tagger will be passed with its context to the translation tool. If the word was transliterated then it is treated as a proper noun.

2. NER is a POS tagging problem: two problems that are very closely bound and traditionally tackled in a sequential order. Moreover, POS tagging is related to morphological analysis. We aim to approach the problem as a single step rather than the traditional way of sequential processing. Given that ATB is annotated with morphological analyses that are mapped to the POS tag, we will use it as our corpus. To add NE annotation to it, we will run our NER tool on the corpus then manually correct and add missing labels. The NNP tag would also be beneficial in the manual work as it is a strong indication of NEs. We will use more features that proven to be successful such as character n-grams and stemming.

3. We plan to investigate the use of joint hierarchal learning technique discussed in chapter 2, to build a multi task model from single-task annotated data since we already have one annotated corpus for each task.

4. The performance of the Semi-CRF algorithm is very encouraging to be examined on the task of Arabic NER. Also, we would like to apply it to the Arabic word segmentation.

5. We plan to extend our feature set to include bag of words features, given the complex syntactic characteristics of Arabic, such as the free order feature of verb and nouns. Also, we plan to enrich our trigger list automatically from unlabelled data. To overcome OOV word caused by spelling errors, we plan to use character N-grams to, as currently this is left to the surrounding context. Thus, a word with a spelling mistake is seen as a foreign name. In other words, both fall into the same category.

6. We shall review our segmentation module, since the design published in (Benajiba et al. 2009a) obtains 74% accuracy on the same dataset without any feature employed
except the contextual feature, which we assume along with the lexical features in our design.

7. We plan to build one multi-class classifier instead of one per class, as NNP could be in any of the three classes (person, location and organization). If the word is not found in the dictionaries then it will be competing with other classes.

8. We still need to work more on names that are in the person gazetteer but are not person names. It is probable if we use character N-grams built from the gazetteers, we could get better results by taking advantage character patterns found in person names.

9. The annotation scheme has an impact on the performance, as has been proved in other research: IOB2 does not differentiate between tokens in the middle of a NE and at the end. Arabic proper nouns, especially person names, have some pattern in the last names. Also, such considerations would be appropriate for organization names.

10. We plan to work on modifying the HM-SVM algorithm so that it could modify the previous word's POS tag, based on the previous class as done with the MaxEnt classifier in chapter 7.
# Appendix A  English POS Tagset

<table>
<thead>
<tr>
<th>POS Tag</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC</td>
<td>coordinating conjunction</td>
<td>and</td>
</tr>
<tr>
<td>CD</td>
<td>cardinal number</td>
<td>1, third</td>
</tr>
<tr>
<td>DT</td>
<td>determiner</td>
<td>the</td>
</tr>
<tr>
<td>EX</td>
<td>existential there</td>
<td>there is</td>
</tr>
<tr>
<td>FW</td>
<td>foreign word</td>
<td>d'houvre</td>
</tr>
<tr>
<td>IN</td>
<td>preposition/subordinating conjunction</td>
<td>in, of, like</td>
</tr>
<tr>
<td>JJ</td>
<td>adjective</td>
<td>green</td>
</tr>
<tr>
<td>JJR</td>
<td>adjective, comparative</td>
<td>greener</td>
</tr>
<tr>
<td>JJS</td>
<td>adjective, superlative</td>
<td>greenest</td>
</tr>
<tr>
<td>LS</td>
<td>list marker</td>
<td>1)</td>
</tr>
<tr>
<td>MD</td>
<td>modal</td>
<td>could, will</td>
</tr>
<tr>
<td>NN</td>
<td>noun, singular or mass</td>
<td>table</td>
</tr>
<tr>
<td>NNS</td>
<td>noun plural</td>
<td>tables</td>
</tr>
<tr>
<td>NNP</td>
<td>proper noun, singular</td>
<td>John</td>
</tr>
<tr>
<td>NNPS</td>
<td>proper noun, plural</td>
<td>Vikings</td>
</tr>
<tr>
<td>PDT</td>
<td>predeterminer</td>
<td>both the boys</td>
</tr>
<tr>
<td>POS</td>
<td>possessive ending</td>
<td>friend's</td>
</tr>
<tr>
<td>PRP</td>
<td>personal pronoun</td>
<td>I, he, it</td>
</tr>
<tr>
<td>PRP$</td>
<td>possessive pronoun</td>
<td>my, his</td>
</tr>
<tr>
<td>RB</td>
<td>adverb</td>
<td>however, usually, naturally, here, well</td>
</tr>
<tr>
<td>RBR</td>
<td>adverb, comparative</td>
<td>better</td>
</tr>
<tr>
<td>RBS</td>
<td>adverb, superlative</td>
<td>best</td>
</tr>
<tr>
<td>RP</td>
<td>particle</td>
<td>give up</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>SYM</td>
<td>Symbol</td>
<td></td>
</tr>
<tr>
<td>TO</td>
<td>to</td>
<td>to go, to him</td>
</tr>
<tr>
<td>UH</td>
<td>interjection</td>
<td>uhhuhhhuhh</td>
</tr>
<tr>
<td>VB</td>
<td>verb, base form</td>
<td>take</td>
</tr>
<tr>
<td>VBD</td>
<td>verb, past tense</td>
<td>took</td>
</tr>
<tr>
<td>VBG</td>
<td>verb, gerund/present participle</td>
<td>taking</td>
</tr>
<tr>
<td>VBN</td>
<td>verb, past participle</td>
<td>taken</td>
</tr>
<tr>
<td>VBP</td>
<td>verb, sing present, non-3d</td>
<td>take</td>
</tr>
<tr>
<td>VBZ</td>
<td>verb, 3rd person sing. present</td>
<td>takes</td>
</tr>
<tr>
<td>WDT</td>
<td>wh-determiner</td>
<td>which</td>
</tr>
<tr>
<td>WP</td>
<td>wh-pronoun</td>
<td>who, what</td>
</tr>
<tr>
<td>WP$</td>
<td>possessive wh-pronoun</td>
<td>whose</td>
</tr>
<tr>
<td>WRB</td>
<td>wh-abverb</td>
<td>where, when</td>
</tr>
<tr>
<td>#</td>
<td>Pound sign</td>
<td></td>
</tr>
<tr>
<td>$</td>
<td>Dollar sign</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td>Sentence-final punctuation</td>
<td></td>
</tr>
<tr>
<td>,</td>
<td>Comma</td>
<td></td>
</tr>
<tr>
<td>:</td>
<td>Colon, semi-colon</td>
<td></td>
</tr>
<tr>
<td>(</td>
<td>Left bracket character</td>
<td></td>
</tr>
<tr>
<td>)</td>
<td>Right bracket character</td>
<td></td>
</tr>
<tr>
<td>&quot;</td>
<td>Straight double quote</td>
<td></td>
</tr>
<tr>
<td>'</td>
<td>Left open single quote</td>
<td></td>
</tr>
<tr>
<td>&quot;</td>
<td>Left open double quote</td>
<td></td>
</tr>
<tr>
<td>'</td>
<td>Right close single quote</td>
<td></td>
</tr>
<tr>
<td>&quot;</td>
<td>Right close double quote</td>
<td></td>
</tr>
</tbody>
</table>
## Appendix B  Arabic Collapsed Tagset

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JJ</td>
<td>Adjective</td>
</tr>
<tr>
<td>RB</td>
<td>Adverb</td>
</tr>
<tr>
<td>CC</td>
<td>coordinating conjunction</td>
</tr>
<tr>
<td>DT</td>
<td>determiner/demonstrative pronoun</td>
</tr>
<tr>
<td>FW</td>
<td>foreign word</td>
</tr>
<tr>
<td>NN</td>
<td>common noun, singular</td>
</tr>
<tr>
<td>NNS</td>
<td>common noun, plural or dual</td>
</tr>
<tr>
<td>NNP</td>
<td>proper noun, singular</td>
</tr>
<tr>
<td>NNPS</td>
<td>proper noun, plural or dual</td>
</tr>
<tr>
<td>RP</td>
<td>particle</td>
</tr>
<tr>
<td>VBP</td>
<td>imperfect verb (present tense)</td>
</tr>
<tr>
<td>VBN</td>
<td>passive verb</td>
</tr>
<tr>
<td>VBD</td>
<td>perfect verb</td>
</tr>
<tr>
<td>UH</td>
<td>interjection</td>
</tr>
<tr>
<td>PRP</td>
<td>personal pronoun</td>
</tr>
<tr>
<td>PRP$</td>
<td>possessive personal pronoun</td>
</tr>
<tr>
<td>CD</td>
<td>cardinal number</td>
</tr>
<tr>
<td>IN</td>
<td>subordinating conjunction (FUNC_WORD) or preposition (PREP)</td>
</tr>
<tr>
<td>WP</td>
<td>relative pronoun</td>
</tr>
<tr>
<td>WRB</td>
<td>wh-adverb</td>
</tr>
<tr>
<td>,</td>
<td>punctuation, token is , (PUNC)</td>
</tr>
<tr>
<td>.</td>
<td>punctuation, token is . (PUNC)</td>
</tr>
<tr>
<td>:</td>
<td>punctuation, token is : or other (PUNC)</td>
</tr>
</tbody>
</table>
Bibliography


Attia, M. 2000. ‘A Large-Scale Computational Processor of the Arabic Morphology, and Applications’. Master’s Thesis, Cairo, Egypt: Faculty of Engineering, Cairo University.


194
Koprinska, I., J. Poon, J. Clark, and J. Chan. 2007. ‘Learning to Classify E-mail’. 
*Information Sciences* 177 (10): 2167–2187. 
<http://dx.doi.org/10.1016/j.ins.2006.12.005>.

Krishnarao, A., H. Gahlot, A. Srinet, and D. Kushwaha. 2009. ‘A Comparison of 
Performance of Sequential Learning Algorithms on the Task of Named Entity 
Recognition for Indian Languages’. In *Proceedings of the 9th International Conference on 

Krupka, G. 1995. ‘SRA: Description of the SRA System as Used for MUC-6’. In *Proceedings 
<http://dx.doi.org/10.3115/1072399.1072419>.

Krupka, G., and K. Hausman. 1998. ‘IsoQuest Inc.: Description of the NetOwl (TM) 
Extractor System as Used for MUC-7’. In *Proceedings of the Seventh Message 
Understanding Conference (MUC-7)*. Vol. 7. 


Kudo, T., and Y. Matsumoto. 2001. ‘Chunking with Support Vector Machines’. In 
*Proceedings of the Second Meeting of the North American Chapter of the Association for 
<http://dx.doi.org/10.3115/1073336.1073361>.

<http://dx.doi.org/10.3115/991635.991707>.


Models for Segmenting and Labeling Sequence Data’. In *Proceedings of the Eighteenth
<http://dl.acm.org/citation.cfm?id=645530.655813>.


<http://eprints.lancs.ac.uk/1303/>.


<http://dx.doi.org/10.1186/1471-2105-10-381>.

<http://dl.acm.org/citation.cfm?id=1869961.1869963>.


Translation SIG of the British Computer Society.


Nédellec, C., and A. Nazarenko. 2006. ‘Ontologies and Information Extraction’.


202


204