**The Islamic University–Gaza**

**Research and Postgraduate Affairs**

**Faculty of Information Technology**

**Master of Information Technology**

الجـامعــــــــــة الإســـــلاميــة – غـزة

شئون البحث العلمي والدراسات العليا

كـليــــة تكنولوجيا المعلومات

ماجستير تكنولوجيا المعلومات

# Improving Dependency Parsing of Verbal Arabic Sentences Using Semantic Features

# تحسين الاعراب الاعتمادي للجملة العربية الفعلية باستخدام السمات الدلالية

**Eng. Heyam Khamis Elnajjar**

**Supervised by**

**Dr. Rebhi Baraka**

**A thesis submitted in partial fulfillment**

**of the requirements for the degree of**

**Master of Information Technology**

**August/2016**

إقـــرار

أنا الموقع أدناه مقدم الرسالة التي تحمل العنوان:

**Improving Dependency Parsing of Verbal Arabic Sentences Using Semantic Features**

تحسين الاعراب الاعتمادي للجملة العربية الفعلية باستخدام الدلالات السيميائية

**أقر بأن ما اشتملت عليه هذه الرسالة إنما هو نتاج جهدي الخاص، باستثناء ما تمت الإشارة إليه حيثما ورد، وأن هذه الرسالة ككل أو أي جزء منها لم يقدم من قبل الاخرين لنيل درجة أو لقب علمي أو بحثي لدى أي مؤسسة تعليمية أو بحثية أخرى.**

## Declaration

I understand the nature of plagiarism, and I am aware of the University's policy on this.

The work provided in this thesis, unless otherwise referenced, is the researcher's own work, and has not been submitted by others elsewhere for any other degree or qualification.

| | | |
|---|---|---|
| Student's name: | **هيام خميس النجار** | **اسم الطالب:** |
| Signature: | [illegible] | **التوقيع:** |
| Date: | 24/10/2016 | **التاريخ:** |

بسم الله الرحمن الرحيم

الجامعة الإسلامية-غزة
The Islamic University of Gaza

مكتب نائب الرئيس للبحث العلمي والدراسات العليا    هاتف داخلي: 1150

الرقم: ج س غ/35/ Ref:
التاريخ: 2016/10/24م Date:

## نتيجة الحكم على أطروحة ماجستير

بناءً على موافقة شئون البحث العلمي والدراسات العليا بالجامعة الإسلامية بغزة على تشكيل لجنة الحكم على أطروحة الباحثة/ **هيام خميس جودت النجار** لنيل درجة الماجستير في **كلية *تكنولوجيا المعلومات*** **برنامج تكنولوجيا المعلومات** وموضوعها:

**تحسين إعراب الجمل العربية الفعلية باستخدام السمات الدلالية**

**Improving Dependency Parsing of Verbal Arabic Sentences Using Semantic Features**

وبعد المناقشة التي تمت اليوم الإثنين 23 محرم 1438هـ، الموافق 2016/10/24م الساعة الواحدة ظهراً ، اجتمعت لجنة الحكم على الأطروحة والمكونة من:

| | | |
|---|---|---|
| د. ربحي سليمان بركة | مشرفاً و رئيساً | [illegible] |
| أ.د. علاء مصطفى الهليس | مناقشاً داخلياً | [illegible] |
| د. سناء وفا الصايغ | مناقشاً خارجياً | [illegible] |

وبعد المداولة أوصت اللجنة بمنح الباحثة درجة الماجستير في **كلية *تكنولوجيا المعلومات*/ برنامج تكنولوجيا المعلومات.**

***واللجنة إذ تمنحها هذه الدرجة فإنها توصيها بتقوى الله ولزوم طاعته وأن تسخر علمها في خدمة دينها ووطنها.***

والله ولي التوفيق ،،،

شئون البحث العلمي والدراسات العليا
Research & Graduate Affairs

نائب الرئيس لشئون البحث العلمي والدراسات العليا

[illegible]

أ.د. عبدالرؤوف علي المناعمة

# Abstract

Parsing is one of the most interesting areas in Natural Language Processing and Arabic parsing takes part of these researches. Parsing describes a word in a sentence grammatically, identifying the parts of speech, syntactic relations, etc. Dependency parsing syntactic structure consists of lexical items linked by binary asymmetric relations called dependencies. Dependency parsing community has, for the last few years shown considerable interest in parsing related to Morphologically Rich Languages with Flexible Word Order (MOR-FWO).

Linguistic semantics in Arabic language words play a big role over understanding the meaning of the sentence in a context. Because of the nature of Arabic language such as free word order, the absence of diacritics or even morphological features, and other phenomena. These lead us to the importance of using the semantics in reducing the gap in the parsing of Arabic sentences that depends on the syntax structure and improving the machine learning parsing and therefore parsing models and applications.

We propose a dependency parsing approach for Modern Standard Arabic MSA Arabic verbal sentences utilizing the information available in lexical Arabic VerbNet to complement the morpho-syntactic information already available in data. This complementing information is encoded as an additional semantic feature for data driven parsing. We do a series of experiments over Arabic text wirenews in Arabic dependency parsing using MaltParser. In this work, we present our experiments, with just 332 sentences training data, we are able to build a dependency parser with state-of-the-art accuracy of 71.5% Labeled Attachment Score (LAS), 77.5% Unlabeled Attachment Score (UAS),and 2% increasing in total accuracy compared to case without using semantic features

**Keywords:** dependency parsing, semantic features, Aabic VerbNet, lexical features, thematic roles, LAS, UAS.

## الملخص

الاعراب هو أحد أكثر المجالات اثارة للاهتمام في أبحاث معالجة اللغات، و الاعراب في العربية يأخذ جزءا من هذه الأبحاث. الاعراب هو تغيير أواخر الكلمات بناء على قواعد اللغة العربية (النحو) و تحديد أجزاء الكلام و العلاقات النحوية و ما الى ذلك. الاعراب الاعتمادي في اللغة يكون حيث العناصر المعجمية مربوطة بعلاقة ثنائية غير متماثلة تسمى الاعتماديات. مجتمع الاعراب الاعتمادي، على مدى السنوات القليلة الماضية أظهر اهتماما كبيرا في اعراب اللغات الغنية شكلا و ذات الترتيب المرن للكلمات.

السيميائية (معاني التعابير) أو السمات الدلالية في كلمات اللغة العربية تلعب دورا هاما لفهم المعنى العام للجملة في محتواها، بسبب طبيعة اللغة العربية مثل الترتيب الحر للكلمات، غياب علامات التشكيل، أو حتى غياب السمات الشكلية (المورفولوجية)، و غيرها من الظواهر هذا يقودنا الى أهمية استخدام معاني اللغة العربية أو السمات الدلالية لتقليص الفجوة في الاعراب للجملة العربية الذي يعتمد على بنية التركيب ولتحسين التعلم الذاتي في الاعراب و من ثم النماذج و التطبيقات للإعراب.

اننا نقترح منهجا للإعراب الاعتمادي لإعراب الجملة الفعلية في اللغة العربية الفصحى الحديثة و ذلك عن طريق الاستفادة من المعلومات المتاحة في معاجم اللغة العربية ، لإكمال المعلومات الشكلية المتاحة اصلا لدينا. هذه المعلومات التكميلية تضاف كسمات دلالية للنصوص المراد اعرابها.

لقد أجرينا التجارب المتتالية للإعراب الاعتمادي على النصوص العربية من الجرائد الالكترونية مستخدمين "المالتبارسر" بارسر . في بحثنا هذا قدمنا تجاربنا كأساس في الاعراب الاعتمادي لقد استخدمنا لذلك 332 جملة كنصوص لتعلم البارسر و استطعنا بذلك بناء بارسر اعتمادي بنتيجة 71.5% تتطابق فيه قيم المسميات للاعتماديات و المدير للمفردات مع القيم الصحيحة و بنتيجة 77.5% تتطابق فيه قيم المدير للمفردات مع القيم الصحيحة ، أي ما يعادل زيادة 2% مقارنة بعدم استخدام السمات الدلالية.

**كلمات مفتاحية:** الاعراب الاعتمادي، السمات الدلالية، قاموس الأفعال العربية، الدلالات المعجمية، الوظائف الموضوعية، دقة تطابق الاعتماديات.

## Epigraph Page

The head of wisdom is the fear of God

# Dedication

To my parents

To my husband

To my family and friends

## Acknowledgment

First of all, all thanks and appreciation to ALLAH for his unlimited blessing and for giving me Islam and the strength to complete this thesis.

I would like to thank my supervisor Dr. Rebhi Baraka without whom, this thesis would not have been possible. I would like to thank Dr. Iyad Al-Agha for his comments especially in some coding issues. I would like to thank Dr. Mohammed Attia who directed me to communicate with Dr. Mohammed ElBadrashiny who helped me in many issues in the course of the research. I would like to thank Dr. Abouenor Lahsen for the pointing me to important resources. Thanks go to Dr. Jaouad Mousser for allowing me to use the Arabic VerbNet, his dissertation, and for his clarifications along the research.

Thanks to Dan Zeman for his help in Treebank, Prof. Nizar Habash for his help in Natural Language Processing, and his remarks, Prof. Joakim Nivre and Prof. koldo Gojenola for their support and replying to my messages, and to Dr. Yuval Marton for his words encouraging NLP students “keep reading, keep experimenting… You’ll get there eventually”.

I would also like to thank the group in Language Technologies Research Centre IIIT Hyderabad for their inspirational papers. Special thanks to Riyaz Bhat for replying to my messages. Finally thanks to everyone who appears in my references.

## Table of contents

# List of Figures

## List of Tables

## List of Abbreviation

| | |
|---|---|
| **$A^3$** | Arabic Annotation Algorithm |
| **ATB** | Penn Arabic Treebank |
| **AVN** | Arabic VerbNet |
| **BPC** | Base Phrase Chunking |
| **CFG** | Context Free Grammar |
| **CoNLL** | Computational Natural Language Learning |
| **DCG** | Definite Clause Grammar |
| **DTD** | Data Type Definition |
| **HWN** | Hindi WordNet |
| **LA** | Label accuracy |
| **LAS** | Labeled Attachment Score |
| **LF** | Line Feed |
| **LFG** | Lexical Functional Grammar |
| **LIBSVN** | machine learning package for Support Vector Machines |
| **MADAMIRA** | MADA Morphological Analysis and Disambiguation of Arabic, AMIRA is a system for tokenization, part-of-speech tagging, Base Phrase Chunking (BPC) and Named Entity Recognition (NER) |
| **MOR-FWO** | Morphologically Rich Languages with Flexible Word Order |
| **MSA** | Modern Standard Arabic |
| **NER** | Named Entity Recognition |
| **NLP** | Natural Language Processing |
| **NLTK** | Natural Language Toolkit |
| **PADT** | Prague Arabic Dependency Treebank |
| **PCFG** | Probabilistic Context Free Grammar |
| **POS** | Part Of Speech |
| **SASPAS** | Syntactic Ambiguity in Single-Parse Arabic Sentences |
| **UAS** | Unlabeled Attachment Score |
| **UBG** | Unification Based Grammar |
| **UTF-8** | Unicode Transformation Format (8 Bit) |
| **WSD** | Word Sense Disambiguation |
| **XML** | EXtensible Markup Language |
| **(S,NP, VP, PP, NNP, NN, Det, V)** | Phrases categories: Statement, Noun Phrase, Verb Phrase, Prepositional Phrase, Noun Phrase, Noun, Determiner, Verb) |

# Chapter1

# Introduction

# Chapter 1
# Introduction

This chapter introduces the research. It first gives an overview of the research including a background about dependency parsing, the research problem, the research objectives, the importance of this research area, scopes and limitations, and the research methodology that is followed to achieve the research objectives. Finally, the research structure is presented.

## 1.1 Overview

The *syntactic parsing* of a sentence consists of finding the correct syntactic structure of that sentence in a given formalism. Formalisms are called grammars, and contain the structural constraints of the language. Dependency grammar and phrase structure grammar are two such formalisms. Figure 1.1 (a) and (b) shows the dependency structure and a simplified phrase structure for the sentence ‘قرأ عليّ الكتاب’ ‘Ali reads the book’.

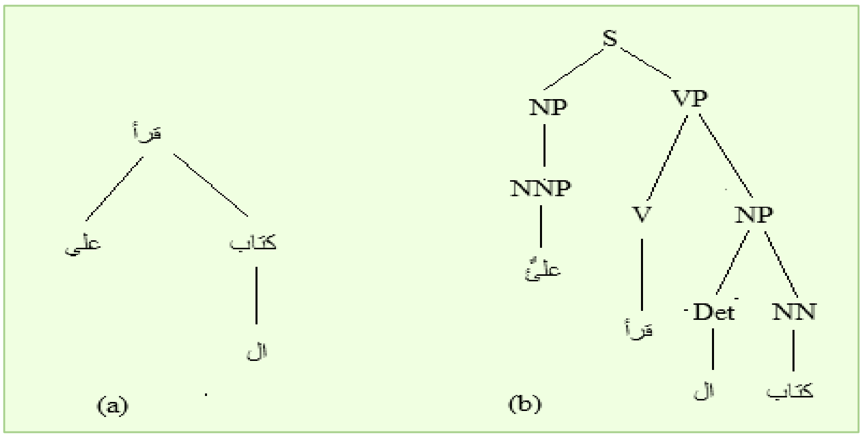


Figure (1.1): Dependency Structure and Phrase Structure for the Arabic sentence ‘Ali read the book ’’ *قرأ عليّ الكتاب*

Unlike English, Arabic is a free word order language and is also morphologically very rich. It has been suggested that free word order languages can be handled better using the dependency based framework than the constituency based one (Hudson, 1984) (Shieber, 1985) (cuk, 1988) (Bharati, Chaitanya, & Sangal, 1995). Consequently, most of the parsers for free word order languages are dependency based. The basic difference between a constituent based representation and a dependency representation is the lack of non-terminal nodes in the later. It has also been noted that use of appropriate edge labels of the dependencies gives a level of semantics. It is perhaps due to these reasons that the recent past has seen a surge in the development of dependency-based treebanks.

Parsing sits between POS Tagging and semantic level as shown in the classical diagram of NLP. Figure 1.2 Parsing needs, Part-Of-Speech Tagging, which must finished earlier. Parsing is related to the structure of the sentence.

| |
|---|
| Semantic Extraction |
| Parsing Syntactic Processing<br>(structure detection of the sentence ) |
| POS Tagging |
| Morphological Processing |

Figure (1.2): Classical diagram of NLP (Bhattacharya, 2010)

When we read a verbal sentence like ‘درب الفريق مدرب الألعاب الرياضية’ ‘the sports game coach trained the team’. It is correct syntactically and grammatically, ’درب’ ’train’ is the verb, ‘الفريق’ ‘team’ is the object "theme", and the ‘مدرب’ ‘the sports game coach’ is the subject "agent" which is not located directly after the verb and is also allowed in classical and Modern Standard Arabic. But when we are able to understand the sentence ‘نظم الفريق مدرب الألعاب الرياضية’ ‘the sports game coach trained the team’ we are invoking actually a higher level knowledge. We are invoking the semantic in spite of the order of this sentences. We can make out the ‘مدرب’ ‘the sports game coach’ as the agent because of the ‘مدرب’ ‘the sports game coach’ is more associated to the action and has the ability to organize the team. So when the order of the sentence is distorted we have to resort to deeper level of meaning to understand the sentence. So the cognitive loads become higher. We sacrifice the order (syntax) but these come at the cost of more challenging and difficult processing, namely the semantic processing.

Bhattacharya, (2010) Notice this fact, that if a language processing entity or information processing entity does not have this back up mechanism, does not have this robustness or does not have the layers of intelligence inbuilt then it will fail when something goes wrong at lower level. Figure 1.2 depicts this fact. Whenever anything goes wrong at a particular level. When something goes wrong at $L_i$ would lead to robust recovery at $L_{i+1}$. So something going wrong at the level of the syntax semantics will be invoked.

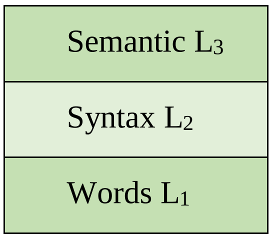


Figure (1.3): NLP Layers (Bhattacharya, 2010)

Dependency parsing community has for the last few years, shown considerable interest in parsing morphologically rich languages with flexible word order such as Arabic.

Modern Standard Arabic (MSA) is the literary standard Arabic currently used across the Middle East and North Africa, and one of the official six languages of the United Nations. All printed books, newspapers, magazines, official and educational documents are written in MSA. Furthermore, MSA is used in news, documentary and scientific programs across different media and online portals.

In this research, we intend to improve the Arabic dependency parsing of the verbal sentence. To this goal, we add extended semantic features to morpho-syntactic features that already exists in the data. The semantic features extracted from Arabic VerbNet. Data driven dependency parser, MaltParser is used. Malteval tool is used to record the scores.

We have used freely available MaltParser (version 1.9.0) (Hall, Nilsson, & Nivre, 2014) (Nivre et al., 2007) parser. The research focuses on choosing the best option for a certain parameter/feature keeping the other parameter/feature fixed (Nivre, Nilsson, & Hall, 2005) (Hall, 2006). MaltParser is a data driven deterministic dependency transition-based parser that obtains a dependency tree in linear-time in a single pass over the input using a stack of partially analyzed items and the remaining input sequence, by means of history-based feature models (Hall, 2006) (Nivre, et al., 2007b). The features are added to inspect the semantic feature at the top of the stack and the next input token. The semantic information is employed due to linguistic semantics of the verb and its qualifiers (such as subject or object) from lexical Arabic VerbNet.

Arabic VerbNet is one of the first verb lexica, which classify the most used verbs of Modern Standard Arabic (MSA) on the basis of Levin's (Levin, 1993) verb classes using the development procedure of Schuler (Schuler, 2005). The verb lexicon in its current state has 334 classes, which contain 7672 verbs (Mousser, 2010). We extract semantic features such as class of the verbs and thematic roles for these verbs from lexical Arabic VerbNet and incorporate them into our parsing model.

Experiments are conducted in our work have been performed on a subset of the HamleDT 3.0. Which is a collection of linguistic textual data in multiple languages. It is based on pre-existing data sets (original treebanks). Arabic dataset is based on Prague Arabic Dependency Treebank (PADT) (Prague, 2004). The annotated labels on the arcs of a dependency tree are syntactico-semantic. The subset contains around 370 sentences with an average length of 29 words and has over 10.900 unique tokens.

CoNLL[1] format is the standard format being used in the dataset. CoNLL format represent data in strict construction in ten columns. We add our semantic features to complement the existing morphological features in FEATS column. Semantic predicate, for instance is one of the features being added. Semantic predicate means that some verbs like **(وَدَّعَ)** have many situations, in the sentences

"ودّع فلانٌ فلاناً ، ودّع الشيءُ تركه و خلاه ، ودّع فرسَه"

(1) It is a Column-based format, Token info separated by new line, Sentences separated by blank line, 10 fields for each token.

‘A person consign someone, let the thing and left it, and let his horse’ here we have three predicates for the object (non-animate and animate “human and animal”) or three options for the semantic feature (Semantic predicate).

The contribution of our research depends on higher-level knowledge needed for disambiguation of the verbal Arabic sentence parsing. The semantic features is added to our data. The features categories under these knowledges of the verbs are: semantic predicate, thematic roles, and semantic classes’ identification.

To this end we build our dependency parsing architecture in three stages, data preparation, parsing model, and results evaluation to proof the benefit and usefulness of using the semantic features behind Arabic lexicals such as Arabic VerbNet to candidate the verbs' qualifiers such as subject and object and to achieve parsing improvement. We evaluate our experiments based on Unlabeled Attachment Score (UAS)[2], Labeled Attachment Score (LAS)[3] and Label Accuracy (LA). These scores will show and give us an open discussion about the results for using semantic features, and how specific real world aspect of lexical items could improve verbal sentences parsing with MSA which have been improved using MaltParser tool in our research.

## 1.2 Statement of the Problem

The sentence in Modern Standard Arabic language, such as those verbal sentences appearing in MSA such as wirenews, has a free order construction, which leads to unsuitability of traditional morphological dependency in the syntax of the sentences. Therefore, there is a need for dependency parsing approaches that take such sentences and their free structure into accounts.

(2) % of tokens with correct HEAD, where HEAD: 7th columns and is head of token (ID or 0).

(3) % of tokens for which a system has predicted the correct HEAD and DEPREL, where DEPREL 8th column and is dependency relation to head.

The problem of this research is how to use semantic features, such as those extracted from lexical Arabic VerbNet, to extend the existing morpho-syntactic features in the data in order to improve Arabic dependency parsing of verbal sentences.

## 1.3 Research Objectives

### Main Objective

To build a parser for a Modern Standard Arabic Language verbal sentences utilizing specific real world aspects using lexical-based semantic features that achieves parsing improvement over existing parser.

### Specific Objectives

- To prepare data required for the dependency parsing by adding LEMMA[4] and extracting needed semantic features from the lexical VerbNet.
- To perform classification and learning on the prepared (training) data to predict the next state in the dependency parsing derivation.
- To perform dependency parsing on the verbal sentences of the test data based on the trained parsing model.
- To obtain the parsing model results by comparing the final resulting parsed data with the test data (gold data).
- Evaluate parsing performance by computing the official scoring metric Unlabeled Attachment Score (UAS), Labeled Attachment Score (LAS) and Label Accuracy (LA).

## 1.4 Importance of the Research

Parsing is one of the major tasks, which helps in understanding the natural language. It is useful in several natural language applications. Machine translation, anaphora resolution, word sense disambiguation, question answering, summarization are few

(4) Lemma: the canonical form or dictionary form of a set of words for example fly, flies, flew and flying all has the lemma fly.

of them (Ambati, 2011). Due to the availability of annotated corpora in recent years, data driven parsing has achieved considerable success.

The parsing of Arabic sentence is a necessary prerequisite for many natural language processing applications such as machine translation and information retrieval (Buchholz & Marsi, 2006).

Also, language modeling used in the parsing process has essential effect on different languages. In this modeling a sentence might be guided by either frequency, language knowledge or word knowledge as shown in Table (1.1) (Bhattacharya, 2010). In our research we use a model that is guided by word knowledge. This model has characteristics such as: probabilistic dependency grammar and semantic rules with probabilities. This model proved its effectiveness and importance when used with Arabic dependency parsing with morphological features (Marton, Habash, & Rambow, 2010).

Table (1.1): Language Modeling

| Guided by frequency | Guided by language knowledge | | Guided by word knowledge | |
|---|---|---|---|---|
| **N-gram** | CFG | Probabilistic CFG | Dependency Grammar "MaltParser" | **Probabilistic Dependency Grammar** |
| | e.g. S-> VP N | | Sematic rules are always b/w head | **Semantic rules with probability** |

Finally, this research demonstrates the importance of adding semantic features to the structure of the Arabic verbal sentence and its role in enhancing the parsing process of such verbal sentences.

## 1.5 Scope and Limitations

**Modern Arabic and Domain:** MSA Modern Arabic is considered in our research. The domain consists of 52 distinct newswire stories from the Lebanese publication. It could appear in its nature as a free order structure because of its social nature as compared to classical Arabic.

**Parsing Type and Model:** Using dependency deterministic data driven parser, MaltParser is used in our research due to its performance and increasing accuracy score. We use a deterministic parsing algorithm to map a dependency graph in MaltParser (Hall, 2006) (Hall, Nilsson, & Nivre, 2014) and show the improving of dependency parsing of Arabic Verbal sentence, by adding the semantic features to our model

**Lexical Features:** We use Arabic VerbNet since it meets the expectations by increasing score, having relation-based concept Arabic structure, and overcomes the absence of many verbs in AVN documents.

**Kind of a Sentence:** We are only dealing with syntactic processing and structural ambiguity in verbal sentence, and features are added where they belong to verbs and their qualifiers, this is because of using the features in AVN.

**The Cases Intended in the Research:** In our research we take into account several cases such as:

- *Free order* verbal Arabic sentences: *VSO*, *VOS*, and *OVS* like :

(اشترى أخوك كتاباً = اشترى كتابا أخوك = كتابا اشترى أخوك)

'Your brother bought a book' do not have short vowel: diacritics.

- *Absent of morphological feature* like: تأبى العرب الضيم ، حضر الرجال.

'Arabs refuse injustice', 'the men attended'.

- *Active and Passive* verb have the same form **(يَكتب ، يُكتب)**

'Write, wrote'

- The same form for the verb has different meaning like :

دَرَسَ

درس الكتاب و نحوه : كرر قراءته ليحفظه

The student *studied* the book and towards: repeated reading to keep it Familiarize

*درس* الدارس سنابل القمح : داسها بمدراة أو بدراسة

He threshed the wheat *spikes* with the harvester

*درس* الثوب : صيره باليا

He *consumed* the dress

- *Subject* (in active verb) is likely to be a humanitarian and non-humanitarian like :

فتح أحمدُ الباب

فتح المفتاحُ الباب

Ahmed open the door

The key open the door

- *Scope*, the agent should be the (الغزال ، الأسد) for the verb أكل regard to free order in the statement

ركض الأسد و أكل الغزال

The lion ran and eat the deer.

- *Inversion* :

"هو رائع" قالت سلمى

"He is wonderful" said Salma.

- *Long distance dependencies* like :

هذا الكتاب ، يسهل قراءته

This book, easy to read

- *Overhead* :

'الرجل القناصة قتل الرجل بالبندقية عندما وقف بجانب المبنى'

'The camera man shot the man with the gun when he was near building'

قتل تعود على من (الرجل/القناصة)، بالبندقية تعود على من (الرجل/ القناصة) وقف تعود على (الرجل/الرجل القناصة).

- Also multiple attachment points of preposition phrases and *clause attachment points*.

رأيت الشجرة بالتلسكوب

I saw the tree with a telescope

رأيت الشجرة بأوراقها (MacKinlay, Dridan, McCarthy, & Baldwin, 2012)

I saw the tree their leaves

- Absence of many grammatical rules in social media and may news website and others (الأفغاني، 2003).
- And others.

But the size and type of the trained data not enough to cover these cases.

The contribution depends on higher level knowledge needed for disambiguation the Arabic Verbal Parsing. Also How much grammatical linguistic remain in MSA will help for our topic, see Appendix F

## 1.6 Research Methodology

We follow these methodology steps to achieve the main and the specific objectives of the research and hence solve the problem:

**First Step:** State of the Art and Literature Review

We provide descriptions about concepts and principles presented in our research such as dependency parsing and its approaches, Arabic VerbNet and semantic features, and theories and topics are needed in our research. We describe in some details the literature and previous research studies, which includes the works and efforts apply to improve  dependency parsing in different languages, by adding semantic features such as annotated, lexical-based, or ontology-based semantic features,

**Second Step:** Build our Architecture

Our architecture include three stages as follow

*Data preparation*

We prepare the data by adding the value of LEMMA, and due to this value extract the semantic features from AVN document. MADAMIRA tool is used to do this task and determine the semantic features that we need from the AVN documents and adding them to the data as additional features.

*Parsing model*

We create our model using MaltParser to perform classification and learning (training) task over the data before prepared and after. Then parse data using the parsing model and test data, the test data does not contain the extended features.

*Results Evaluation*

We evaluate parsing performance by computing the officinal scoring metrics Unlabeled Attachment Score (UAS), Labeled Attachment Score (LAS) and Label Accuracy (LA). To this end we use MaltEval tool.

**Third step:** Results and conclusion

The outputs of this research are the scores in step two which intend on dependencies that include labeled and dependent of the tokens, we present a discussion for these outputs and finally, the conclusion and recommendation of the study were drawn up.

## 1.7 Thesis Structure

**Chapter 2: State of the Art** presents dependency parsing and its different approaches, then gives a general overview of treebank, Arabic VerbNet and description of the MaltParser used in the research. Additionally it describes the related works for incorporating features in dependency parsing. In **chapter 3: Dependency Parsing of Verbal Arabic Sentences**, we present basic concepts and steps that are used to perform dependency parsing and present our parsing model and build its architecture including the parsing model. **Chapter 4: Implementation** describes the implementation process of the dependency parsing approach including the tools used to build the parsing model as specified in the architecture presented in the previous chapter. **Chapter 5: Results and Discussion** evaluates the performance of the parsing model using MaltEval tool and presents a discussion of the results.

Finally, **chapter 6: Conclusion and Future Work** concludes the research by stressing that the results meet the research objectives.

# Chapter 2

# State of the Art

# Chapter 2
# State of the Art

In this chapter, we present dependency parsing and the data-driven dependency parser used in our research. We start with a brief description of dependency parsing. Then we mention different approaches commonly followed for dependency parsing. We describe a state-of-the-art data-driven dependency parser used in our research. We give a general overview of the Arabic VerbNet and mention the main features that are used in improving dependency parsing of Arabic verbal sentences. We describe in details the related work for incorporating features starting with morphological features in Arabic dependency parsing, semantic annotation features and finally lexical based semantic features in dependency parsing.

## 2.1 Dependency Parsing

Dependency graphs represent words and their relationship to syntactic modifiers using directed edges. Figure 2.1 shows a dependency graph for the sentence, "John hit the ball with the bat" (Ambati, 2011). This example belongs to the special class of dependency graphs that only contain projective (also known as nested or non-crossing) edges. Assuming a unique root as the left most word in the sentence, a projective graph is one that can be written with all words in a predefined linear order and all edges drawn on the plane above the sentence, with no edge crossing another. Figure 2.1 shows this construction for the example sentence. Equivalently, we can say a dependency graph is projective if and only if an edge from word *w* to word *u* implies that there exists a directed path in the graph from *w* to every word between *w* and *u* in the sentence.

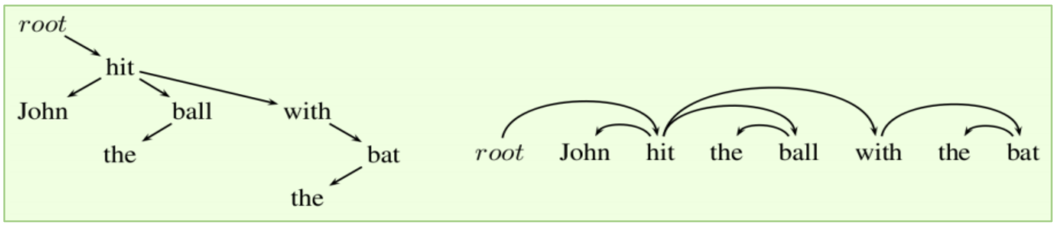


Figure (2.1): An example dependency graph (Ambati, 2011)

However, there are certain examples in which a non-projective graph is preferable. Consider the sentence, "John saw a dog yesterday which was a Yorkshire Terrier".

Here the relative clause “which was a Yorkshire Terrier” and the noun it modifies (the dog) are separated by a temporal modifier of the main verb. There is no way to draw the dependency graph for this sentence in the plane with no crossing edges, as illustrated in Figure 2.2. In languages with flexible word order, such as Arabic, Czech, Dutch, German, and Hindi, non-projective dependencies are more frequent.

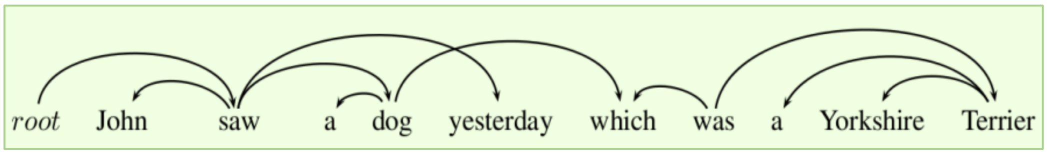


Figure (2.2): A non-projective dependency graph (Ambati, 2011)

In the most common case, every valid dependency graph has the following properties,

1. Each word has exactly one incoming edge in the graph (except the root, which has no incoming edge).
2. There are no cycles (tree).
3. If there are *n* words in the sentence (including root), then the graph has exactly *n - 1* edges.

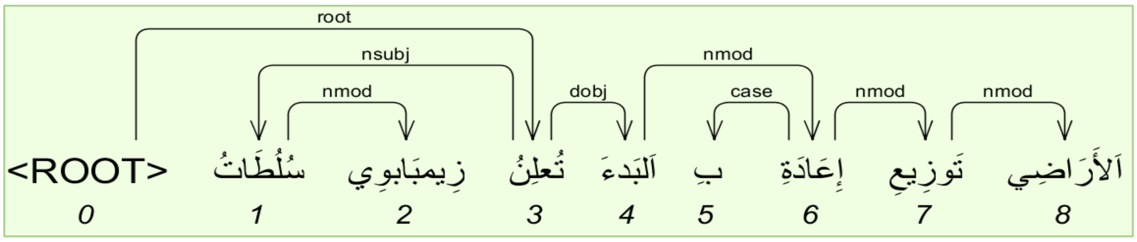


Figure (2.3): An example of a labeled dependency graph

Figure 2.3 shows an example of a labeled dependency graph for the sentence:

**‘تعلن سلطات زيمبابوي البدء بإعادة توزيع الاراضي ’**

‘Zimbabwe authorities announce the start of redistributing land’

A dependency graph that satisfies these constraints must be a tree, and call such graphs *dependency trees*. Dependency syntax postulates that syntactic structure consists of lexical items linked by binary asymmetric relations (‘arrows’) called dependencies. The arrows are commonly typed with the name of grammatical relations (labels) such as (subject, object, noun-modification, etc.). The arrow connects a *head* (governor, superior, regent), the verb ‘**تعلن**’ ‘announce’in Figure 2.3

with a dependent (modifier, inferior, subordinate), such as 'سلطات، اعادة،... الخ' 'authorities, re, etc.' in the same figure. Usually, dependencies from a tree (connected acyclic, single-head). The *ROOT* is the *head* of the sentence.

We always refer to words in a dependency relationship as the head and modifier. The dependency structures can be principled showing only one kind of modification such as grammatical, or syntactic and even semantic properties of the head-modifier relationships.

In general, dependency parsing can be broadly divided into two approaches grammar-driven and data-driven (Chang & Lin, 2001). Most of the modern grammar-driven dependency parsers parse by eliminating the parses, which do not satisfy some set of grammatical constraints.

Data-driven dependency parsers are different from the grammar driven parsers in that they use a corpus to induce a probabilistic model for disambiguation. Nevertheless many data-driven parsers also combine dependency formalism with the probabilistic model (Nivre et al., 2007b).

There are several state-of-the-art Arabic statistical and rule-based parsers such as Bikel parser (Bikel, 2002) (Gabbard & Kulick, 2008) (Kulick, Gabbard, & Marcus, 2006) , Malt parser (Habash & Roth) (Nivre, Nilsson, & Hall, 2005) (Hall, 2006), Stanford parser (Green, Sathi, & Manning, 2009) (Mousser, 2010), and Attia's rule-based parser for Modern Standard Arabic (Attia, 2006) (Tounsi, Attia, & Genabith, 2009). These parsers require the availability of a treebank (Habash N, 2010).

## 2.2 The Prague Arabic Dependency Treebank

The Prague Arabic Dependency Treebank (PADT) contains a multi-level description comprising functional morphology, analytical dependency syntax, and tectogrammatical representation of linguistic meaning. These linguistic annotations are based on the Functional Generative Description theory (Sgall, Hajiˇcová, & Panevová, 1986) and the Prague Dependency Treebank project (Hajiˇ c, Hladká, & Pajas, 2001). The corpus of PADT 1.0 consists of morphologically and analytically annotated newswire texts of Modern Standard Arabic (Pennsylvania, 2016).

HamleDT 3.0 is a collection of linguistic textual data in multiple languages. Arabic language is our interest. HamleDT 3.0 data portion is used in the research contains different stories from wirenews and comprises over 10,900 tokens of data annotated analytically and provided with the disambiguated morphological information. In addition, the release includes complete annotations of MorphoTrees resulting. HamleDT 3.0 Is based on pre-existing data sets ("original treebanks"). Arabic dataset is based on Prague Arabic Dependency Treebank (PADT) (Prague, 2004). Data is presented in CoNLL format, CoNLL format represent data in strict construction in ten columns, which are ID, FORM, LEMMA, CPOSTAG, POSTAG, FEATS, HEAD, DEPREL, PHEAD, and PDEPRAL. Figure 2.4 display a portion of the data.

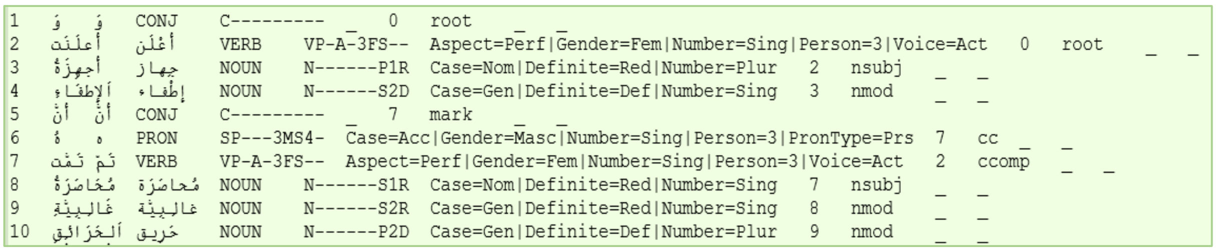

```
1   وَ  وَ  CONJ   C---------   _    0   root   _   _
2   أعلَنَت  أَعْلَن   VERB    VP-A-3FS--   Aspect=Perf|Gender=Fem|Number=Sing|Person=3|Voice=Act   0   root   _   _
3   أجهِزَةُ  جِهاز   NOUN    N------P1R   Case=Nom|Definite=Red|Number=Plur   2   nsubj   _   _
4   اَلإِطفَاءِ  إِطْفاء   NOUN    N------S2D   Case=Gen|Definite=Def|Number=Sing   3   nmod   _   _
5   أَنَّ  أَنَّ  CONJ   C---------   _    7   mark   _   _
6   هُ   ه   PRON   SP---3MS4-   Case=Acc|Gender=Masc|Number=Sing|Person=3|PronType=Prs   7   cc   _   _
7   تَمَّ تَ  تَمّ   VERB   VP-A-3FS--   Aspect=Perf|Gender=Fem|Number=Sing|Person=3|Voice=Act   2   ccomp   _   _
8   مُحَاصَرَةُ  مُحاصَرَة  NOUN    N------S1R   Case=Nom|Definite=Red|Number=Sing   7   nsubj   _   _
9   غَالِبِيَّةِ  غالِبِيَّة  NOUN    N------S2R   Case=Gen|Definite=Red|Number=Sing   8   nmod   _   _
10  اَلحَرَائِقِ  حَرِيق   NOUN    N------P2D   Case=Gen|Definite=Def|Number=Plur   9   nmod   _   _
```

Figure (2.4): Example of CoNLL data format representation

## 2.3 Rule-based Parsing

A framework for using the Natural Language Toolkit (NLTK) recursive-descent parser is developed in (Shatnawi & Belkhouche, 2012). The framework supports the construction of a treebank for the Holy Quran. The proposed model succeeds in parsing different Quranic chapters (Suras) in addition to Modern Standard Arabic (MSA) sentences. NLTK is an open source suite of libraries and programs which can be integrated within the Python environment and then used to perform different statistical and rule-based natural language processing tasks such as parsing. The recursive-descent parser is tested and evaluated in building parse trees of Arabic sentences in general and Quranic sentences in particular.

Othman, Shaalan, & Rafea, (2006) develop an efficient bottom-up chart parser. It is able to satisfy syntactic constraints reducing parsing ambiguity for Analyzing Modern Standard Arabic (MSA) sentences. Lexical semantic features are also used to disambiguate the sentence structure. They acquired the Arabic grammar rules of *irab* 'اعراب' and the effects of applying these rules to the constituents of the

Arabic sentence. The grammar rules encode the syntactic and the semantic constrains that help in resolving the ambiguity of parsing Arabic sentences. Unification Based Grammar (UBG) formalism (Covington, 1994) is used to write the Arabic grammar rules in the proposed chart parser. The grammar is implemented in Prolog. There are two types of features in the lexicon: syntactic features that eliminate syntactic ambiguity and lexical features that eliminate lexical ambiguity. The values of these features are stored in the lexicon and can be modified during the sentence analysis. For example, a verb has the following form: verb (Stem, Voice, Tense, [Subject Gender, Object Gender], Number, End case, Transitivity, [Subject rationality, Object Rationality], Infinitive).

The lexical features are: [Subject Rationality, Object rationality]: [rational/irrational, rational/irrational].this feature helps in determining that an agent is the proagent for a verb but not its subject by comparing the rationality feature of this verb with the agent feature. Infinitive: [infinitive form] this feature is needed when the morphology decides that the verb is in the passive voice, since they do not store the passive form of the verb in the lexicon.

A framework is proposed in (Al Daoud & Basata, 2009) to automate the parsing 'اعراب' of Arabic language sentences. The proposed system is divided into two separated phases which are lexical analysis and syntax analysis. Lexical phase analyses the words, finds its originals and roots, separates it from prefixes and suffixes, and assigns the filtered words to special tokens. Syntax analysis receives all the tokens and finds the best grammar for the given sequence of the tokens by using Context Free Grammar CFG. The system assumed that the entered sentences are correct lexically and grammatically. A recursive parser was used a top-down parser built from a set of mutually-recursive procedures where each such procedure usually implements one of the production rules of the grammar.

Some efforts like (Daimi, 2001) describe a technique for locating and identifying Syntactic Ambiguity in Single-Parse Arabic Sentences (SASPAS) by analyze each sentence and verifies the conditions that govern the existence of certain types of syntactic ambiguities in Arabic sentences, the input of the system is the Arabic sentence while the output is syntax ambiguity identification. SASPAS is integrated with the syntactic parser, which is based on Definite Clause Grammar (DCG) formalism. The ambiguity cases covered by SASPAS are classified according to the constituents being modified.

In (Alqrainy, Muaidi, & Alkoffash, 2012) a set of experiments are conducted on a dataset contains 150 Arabic sentence. The system achieved an average accuracy of 95%.They check whether the syntax of an Arabic sentence is grammatically correct or not by constructing new efficient Context-Free Grammar that makes Top-Down technique much valuable. NLTK parser that uses Top-Down technique to check whether the syntax of an Arabic sentence is grammatically correct also discussed. The testing corpus has been annotated (tagged) using AMT tagger that produced the three main general tags (N: noun, V: verb, P: particle).

## 2.4 Statistical Parsing

The best-known Arabic statistical parser was developed by (Bikel D. M., 2004), they built a multi-lingual parsing engine that is capable of instantiating a wide variety of generative, statistical parsing models (Bikel, 2002). They perform experiments that show that the true discriminative power of lexicalization appears to lie in the fact that unlexicalized syntactic structures are generated conditioning on the head word and its part of speech. In (Genabith, Tounsi, & Attia, 2009) they try to automatically enrich the output of Bikel’s parser with more abstract and “deep” dependency information (in the form of Lexical Functional Grammar LFG f-structure equations to trees) and achieve a dependency f-score of 77%, using the A3 Arabic annotation algorithm (Tounsi, Attia, & Genabith, 2009), extending the approach of (Cahill, Burke, O’Donovan, Genabith, & Way, 2004) originally developed for English. Compared to similar results for English, Arabic are somewhat disappointing. The most likely reason is the explosion in the size of the phrasal category set with 22 ATB phrasal categories as opposed to 150 (masked) categories (fusing ATB phrasal and functional tags) to be learnt by Bikel’s parser.

Establishing significantly higher parsing baselines is achieved (Christopher D. Manning & Green) over three statistical constituency parsers: Stanford parser, Bikel parser, and Berkeley parser. They show that Arabic parsing performance is not poor, but remains much lower than English. They identify sources of syntactic ambiguity understudied in the existing parsing literature. They show that although the Penn

Arabic Treebank is similar to other treebanks in gross statistical terms, annotation consistency remains problematic. They develop a human interpretable grammar that is competitive with a latent variable PCFG. Moreover, they build better models for three different parsers. In addition, they showed that in application settings, the absence of gold segmentation lowers parsing performance by 2–5% F1, where F1 is a ParseEval metric for evaluating constituency-based parsing and computed from the gold and the parse trees.

## 2.5 Arabic Dependency Parsing

Work in (Daoud, 2009) Morphological and syntactic processing of Arabic text is performed in a single and joint framework using a rule-based model. The approach helps in highly accurate analysis of sentences. The analysis produces a semantic net like structure expressed by means of Universal Networking Language (UNL). EnCo is a rule-based programming language specialized for the writing of enconverters (parsers). EnCo is oriented towards the production of dependency graphs. It analyses a sentence by establishing links between individual words and specifying the type of link in each case. Each link connects a word (the "head") with one of its "dependents" (an argument or modifier). A head can have many dependents, but each dependent can have only one head.

In (Marton, Habash, & Rambow, June 2010) improving Arabic Dependency Parsing with Lexical and Inflectional Morphological Features of Modern Standard Arabic (MSA),showing an improvement over form-based features, and explore the contribution of lexical and inflectional morphology features to dependency parsing of Arabic, a morphologically rich language with complex agreement patterns. Using controlled experiments.

Results in (Marton, Habash, Rambow, & Alkuhlani, 2013) showed that assignment features, specifically CASE and STATE, are very helpful in MSA.

## 2.6 MaltParser (A Transition-based Dependency Parser)

We use a data-driven dependency parser 'MaltParser' for our research. A brief description about this parser (drawn from the original papers) is presented. MaltParser (Nivre et al., 2007b) (Chanev, et al., 2007b) implements the transition-based approach to dependency parsing, which has two essential components:

- A transition system for mapping sentences to dependency trees
- A classifier for predicting the next transition for every possible system configuration

Given these two components, dependency parsing can be realized as deterministic search through the transition system, guided by the classifier. With this technique, parsing can be performed in linear time for projective dependency trees and quadratic time for arbitrary (possibly non-projective) trees (Nivre, 2008).

### 2.6.1 Transition Systems

MaltParser comes with a number of built-in transition systems. We describe the arc-eager projective system first described in (Nivre, 2003). Other systems are minor variations of these two (Nivre, 2008). The arc-eager algorithm builds a labeled dependency graph in one left-to-right pass over the input. A configuration in the arc-eager projective system contains a stack holding partially processed tokens, an input buffer containing the remaining tokens, and a set of arcs representing the partially built dependency tree. There are four possible transitions (where top is the token on top of the stack and next is the next token in the input buffer):

- LEFT-ARC (r): Add an arc labeled r from next to top; pop the stack.
- RIGHT-ARC (r): Add an arc labeled r from top to next; push next onto the stack.
- REDUCE: Pop the stack.
- SHIFT: Push next onto the stack.

Consider an example English sentence “Economic news had little effect on financial markets.” (Ambati, 2011). Figure 2.5 shows the dependency tree for the above mentioned sentence. Figure 2.6 shows the sequence of steps for parsing the example sentence described in Figure 2.5 using arc-eager algorithm. Although this system can only derive projective dependency trees, the fact that the trees are labeled allows non-projective dependencies to be captured using the pseudo-projective parsing technique proposed in Nivre and Nilsson (2005) (Nivre & Nilsson, 2005). This is a way of dealing with non-projective structures in a projective data-driven parser. Training data is projectivized by a minimal transformation, lifting nonprojective arcs one step at a time, and extending the arc label of lifted arcs using the encoding scheme called

HEAD by (Nivre & Nilsson, 2005), which means that a lifted arc is assigned the label $r \wedge h$, where $r$ is the original label and $h$ is the label of the original head in the non-projective dependency graph.

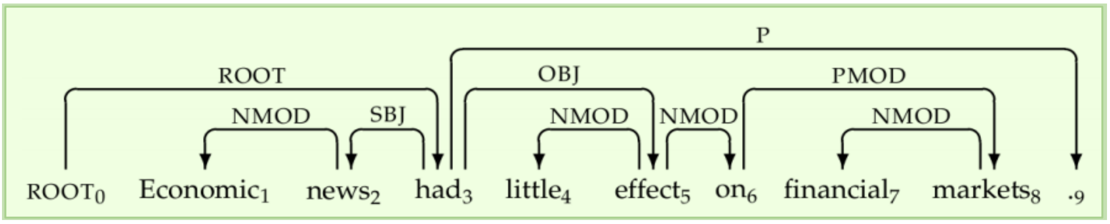
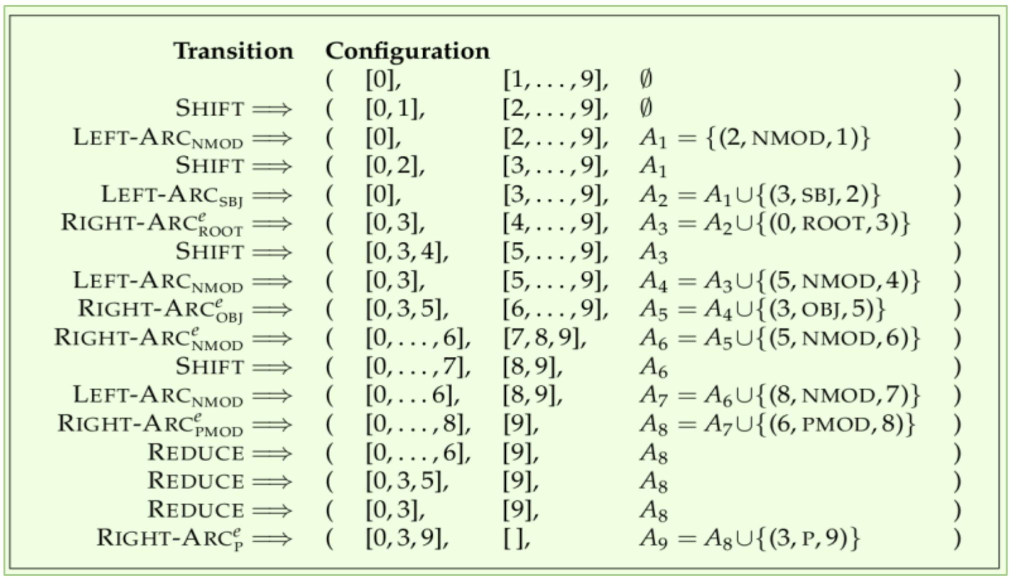


Figure (2.5): Dependency graph for an English sentence from the Penn Treebank

Non-projective dependencies can be recovered by applying an inverse transformation to the output of the parser, using a left-to-right, top-down, breadth-first search, guided by the extended arc labels $r \wedge h$ assigned by the parser.

MaltParser also provides an option for a non-projective transition system based on the method described by (Covington, 2001). This system uses a similar type of configuration of arc-eager described above, but adds a second temporary stack. Unlike the arc-eager, this allows the derivation of arbitrary non-projective dependency trees. There are again four possible transitions:



| Transition | Configuration | | | | |
|---|---|---|---|---|---|
| | ( | $[0]$, | $[1,\dots,9]$, | $\emptyset$ | ) |
| SHIFT $\Longrightarrow$ | ( | $[0,1]$, | $[2,\dots,9]$, | $\emptyset$ | ) |
| LEFT-ARC$_{\text{NMOD}}$ $\Longrightarrow$ | ( | $[0]$, | $[2,\dots,9]$, | $A_1 = \{(2,\text{NMOD},1)\}$ | ) |
| SHIFT $\Longrightarrow$ | ( | $[0,2]$, | $[3,\dots,9]$, | $A_1$ | ) |
| LEFT-ARC$_{\text{SBJ}}$ $\Longrightarrow$ | ( | $[0]$, | $[3,\dots,9]$, | $A_2 = A_1 \cup \{(3,\text{SBJ},2)\}$ | ) |
| RIGHT-ARC$^e_{\text{ROOT}}$ $\Longrightarrow$ | ( | $[0,3]$, | $[4,\dots,9]$, | $A_3 = A_2 \cup \{(0,\text{ROOT},3)\}$ | ) |
| SHIFT $\Longrightarrow$ | ( | $[0,3,4]$, | $[5,\dots,9]$, | $A_3$ | ) |
| LEFT-ARC$_{\text{NMOD}}$ $\Longrightarrow$ | ( | $[0,3]$, | $[5,\dots,9]$, | $A_4 = A_3 \cup \{(5,\text{NMOD},4)\}$ | ) |
| RIGHT-ARC$^e_{\text{OBJ}}$ $\Longrightarrow$ | ( | $[0,3,5]$, | $[6,\dots,9]$, | $A_5 = A_4 \cup \{(3,\text{OBJ},5)\}$ | ) |
| RIGHT-ARC$^e_{\text{NMOD}}$ $\Longrightarrow$ | ( | $[0,\dots,6]$, | $[7,8,9]$, | $A_6 = A_5 \cup \{(5,\text{NMOD},6)\}$ | ) |
| SHIFT $\Longrightarrow$ | ( | $[0,\dots,7]$, | $[8,9]$, | $A_6$ | ) |
| LEFT-ARC$_{\text{NMOD}}$ $\Longrightarrow$ | ( | $[0,\dots 6]$, | $[8,9]$, | $A_7 = A_6 \cup \{(8,\text{NMOD},7)\}$ | ) |
| RIGHT-ARC$^e_{\text{PMOD}}$ $\Longrightarrow$ | ( | $[0,\dots,8]$, | $[9]$, | $A_8 = A_7 \cup \{(6,\text{PMOD},8)\}$ | ) |
| REDUCE $\Longrightarrow$ | ( | $[0,\dots,6]$, | $[9]$, | $A_8$ | ) |
| REDUCE $\Longrightarrow$ | ( | $[0,3,5]$, | $[9]$, | $A_8$ | ) |
| REDUCE $\Longrightarrow$ | ( | $[0,3]$, | $[9]$, | $A_8$ | ) |
| RIGHT-ARC$^e_{\text{P}}$ $\Longrightarrow$ | ( | $[0,3,9]$, | $[\,]$, | $A_9 = A_8 \cup \{(3,\text{P},9)\}$ | ) |

Figure (2.6): Arc-eager transition sequence for the English sentence in Figure 2.4.

- LEFT-ARC (r): Add an arc labeled r from next to top; push top onto the second stack.

- RIGHT-ARC (r): Add an arc labeled r from top to next; push top onto the second stack.
- NO-ARC: Push top onto the second stack.
- SHIFT: Empty the second stack by pushing every word back onto the stack; then push next onto the stack.

### 2.6.2 Classifiers

Classifiers can be induced from treebank data using a wide variety of different machine learning methods. MaltParser uses support vector machines with a polynomial kernel, as implemented in the *LIBSVM* package (Chang and Lin, 2001). In addition to this, MaltParser also provides an option to use external *LIBSVM* packages. The task of the classifier is to map a high-dimensional feature vector representation of a parser configuration to the optimal transition out of that configuration.

Features are very crucial for any classifier. The features used in MaltParser are all symbolic and extracted from the following fields of the CoNLL data representation (Buchholz and Marsi, 2006): FORM, LEMMA, CPOSTAG, POSTAG, FEATS, and DEPREL as presented in chapter (3) and chapter (4). Symbolic features are converted to numerical features using the standard technique of binarization. Features of the type DEPREL have a special status in that they are extracted during parsing from the partially built dependency graph and may therefore contain errors, whereas all the other features have gold standard values during both training and parsing. Once we have the list of all possible features, then getting the best feature set is the next important step. General procedure for feature optimization in Malt is, base model is defined and using forward and backward feature selection algorithms, language-specific feature selection is done.

## 2.7 MaltEval: An Evaluation and Visualization Tool for Dependency Parsing

(Nivre & Nilsson, 2007) presents a freely available evaluation tool for dependency parsing, MaltEval. MaltEval is a software tool written in Java. It is flexible and extensible, and provides functionality for both quantitative evaluation and visualization Figure 2.3 of dependency structure. The quantitative evaluation is compatible with other standard evaluation software for dependency structure which does not produce visualization of dependency structure, and can output more details as well as new types of evaluation metrics. It support CoNLL format.

### 2.7.1 The Metric Values

The currently available values for Metric are shown below, where two different values can be used for the first three:

**LAS (BothRight)** A token is counted as a hit if both the head and the dependency label are the same as in the gold-standard data. This is the default value.

**LA (LabelRight)** A token is counted as a hit if the dependency label is the same as in the gold-standard data.

**UAS (HeadRight)** A token is counted as a hit if the head is the same as in the gold-standard data.

**AnyRight** A token is counted as a hit if either the head or the dependency label (or both) is the same as in the gold-standard data.

**BothWrong** A token is counted as a hit if neither the head nor the dependency label are the same as in the gold-standard data.

**LabelWrong** A token is counted as a hit if the dependency label is **not** the same as in the gold-standard data.

**HeadWrong** A token is counted as a hit if the head is **not** the same as in the gold-standard data.

**AnyWrong** A token is counted as a hit if either the head or the dependency label (or both) is **not** the same as in the gold-standard data.

**self** This is a special type of metric that is dependent on the selected GroupByvalues.

### 2.7.2 Evaluation of Dependency Parsing Accuracy

We use the metric values to know how many decision did not get right (Jurafsky & Manning, 2012). For example the Figure 2.7 with a sentence 'she saw the video lecture'. Each word in the sentence is given an odd number (ID), starting with zero (ROOT). In the verbal sentence verb is the root, this figure is a correct dependencies, which is called gold. Figure 2.8 (a) represent the sentence in gold, as we see each word in column3 has an odd number in column 1, each word has heading word number in column 2, column 4 show labeled dependencies. When we parse the sentence the result in figure 2.8 (b) get a parsed result

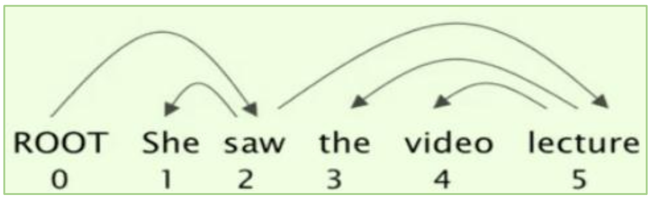


Figure (2.7): Dependency parsing of the sentence‘she saw the video lecture’

| Gold | | | |
|---|---|---|---|
| 1 | 2 | She | nsubj |
| 2 | 0 | saw | root |
| 3 | 5 | the | det |
| 4 | 5 | video | nn |
| 5 | 2 | lecture | dobj |

| Parsed | | | |
|---|---|---|---|
| 1 | 2 | She | nsubj |
| 2 | 0 | saw | root |
| 3 | 4 | the | det |
| 4 | 5 | video | nsubj |
| 5 | 2 | lecture | ccomp |

Figure (2.8): dependecies of the statement in Figure (2.7) (a) Gold representation (b) Parsed representation

Due to result Figure 2.8 (b) computing the accuracy

$$\text{Acc=}\ \frac{\text{\# correct deps}}{\text{\# of deps}} \qquad (2.1)$$

When we ignore the labels on the dependencies, all words heading number is correct except the third, where ‘the’ is dependent of the ‘lecture’, but in result it dependent to ‘video’, we have five dependencies, so the UAS with respect to equation 2.2 is 4/5=80%

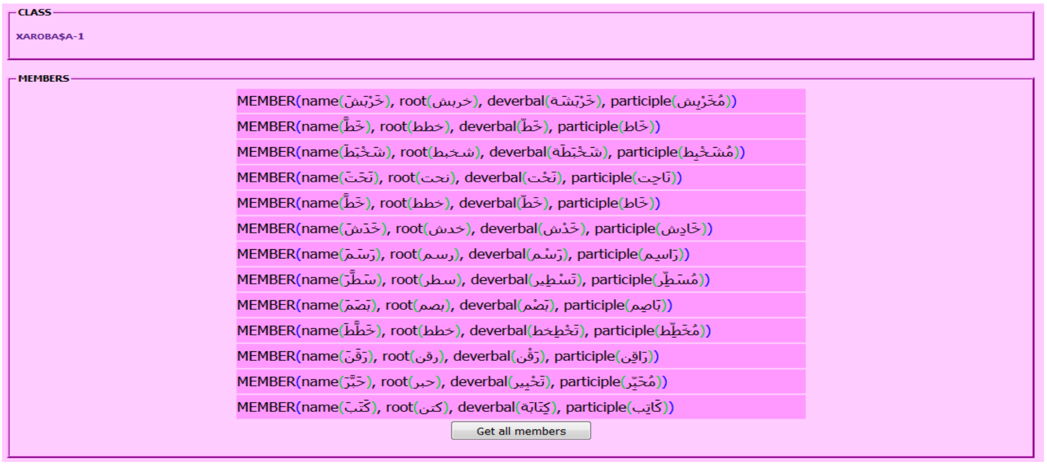
$$\text{UAS=}\ \frac{\text{\# of correct heading}}{\text{\# of heading}} \qquad (2.2)$$

The fourth and fifth words are right choose in grammar, but wrong in functionality (nsubj, ccomp) instead of (nn, dobj). Two words right over whole the sentence, so the LAS with respect to equation 2.3 is 2/5=40%

$$\text{LAS=}\ \frac{\text{\# correct heading and labels}}{\text{\# of heading and labels}} \qquad (2.3)$$

## 2.8 Arabic VerbNet

Arabic VerbNet proposes a scheme for classifying verbs of Modern Standard Arabic (MSA) in a manner similar to (Levin, 1993) with the purpose of building a verb lexicon based on VerbNet (Schuler K. K., 2005).

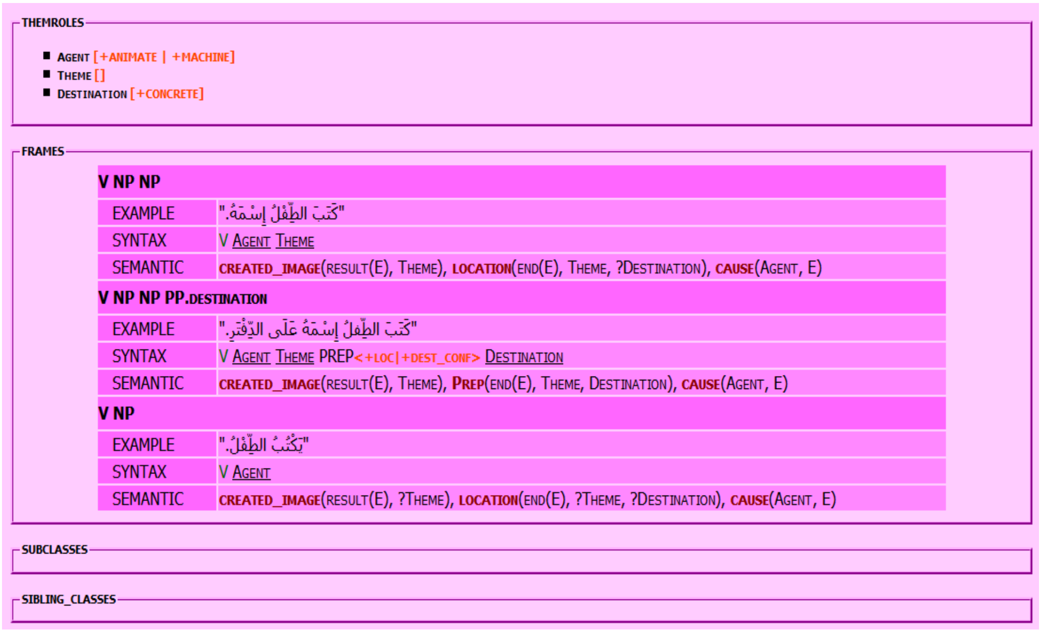


CLASS

XAROBA$A-1

MEMBERS

MEMBER(name(خَرْبَشَ), root(خربش), deverbal(خَرْبَشَة), participle(مُخَرْبِش))
MEMBER(name(خَطَّ), root(خطط), deverbal(خَطّ), participle(خَاط))
MEMBER(name(شَخْبَطَ), root(شخبط), deverbal(شَخْبَطَة), participle(مُشَخْبِط))
MEMBER(name(نَحَتَ), root(نحت), deverbal(نَحْت), participle(نَاحِت))
MEMBER(name(خَطَّ), root(خطط), deverbal(خَطّ), participle(خَاط))
MEMBER(name(خَدَشَ), root(خدش), deverbal(خَدْش), participle(خَادِش))
MEMBER(name(رَسَمَ), root(رسم), deverbal(رَسْم), participle(رَاسِم))
MEMBER(name(سَطَّرَ), root(سطر), deverbal(تَسْطِير), participle(مُسَطِّر))
MEMBER(name(بَصَمَ), root(بصم), deverbal(بَصْم), participle(بَاصِم))
MEMBER(name(خَطَّطَ), root(خطط), deverbal(تَخْطِيط), participle(مُخَطِّط))
MEMBER(name(رَقَنَ), root(رقن), deverbal(رَقْن), participle(رَاقِن))
MEMBER(name(حَبَّرَ), root(حبر), deverbal(تَحْبِير), participle(مُحَبِّر))
MEMBER(name(كَتَبَ), root(كتن), deverbal(كِتَابَة), participle(كَاتِب))



THEMROLES

- AGENT [+ANIMATE | +MACHINE]
- THEME []
- DESTINATION [+CONCRETE]

FRAMES

| V NP NP | |
|---|---|
| EXAMPLE | "كَتَبَ الطِّفْلُ إِسْمَهُ." |
| SYNTAX | V AGENT THEME |
| SEMANTIC | CREATED_IMAGE(RESULT(E), THEME), LOCATION(END(E), THEME, ?DESTINATION), CAUSE(AGENT, E) |
| **V NP NP PP.DESTINATION** | |
| EXAMPLE | "كَتَبَ الطِّفْلُ إِسْمَهُ عَلَى الدَّفْتَرِ." |
| SYNTAX | V AGENT THEME PREP<+LOC\|+DEST_CONF> DESTINATION |
| SEMANTIC | CREATED_IMAGE(RESULT(E), THEME), PREP(END(E), THEME, DESTINATION), CAUSE(AGENT, E) |
| **V NP** | |
| EXAMPLE | "يَكْتُبُ الطِّفْلُ." |
| SYNTAX | V AGENT |
| SEMANTIC | CREATED_IMAGE(RESULT(E), ?THEME), LOCATION(END(E), ?THEME, ?DESTINATION), CAUSE(AGENT, E) |

SUBCLASSES

SIBLING_CLASSES

Figure (2.9): A snapshot of the AVN class xaroba$a-1[1]

Arabic VerbNet (AVN) is a verb lexicon for Arabic with about 8000 verb entries in 326 classes and sibling classes and 230 subclasses with about 1368 frames. The information provided for each class is generally the same as in English VerbNet

(1) http://ling.uni-konstanz.de/pages/home/mousser/files/Arabic_VerbNet.php.

Figure (2.9). The resource consists of verb entries, a list of thematic roles assigned by the verbs, many verbs impose selectional restrictions to constrain the type of thematic roles. In sum, there are 36 selectional restrictions, a number of frames with descriptions and example sentences, shallow syntactic structures and their semantic structure adopts the entire set of semantic predicates used in English VerbNet. However, new semantic predicates are added to describe Arabic-specific meaning aspects. Each class can have subclasses, which have the same structure as their parent class and can also have subclasses (Mousser, 2010).

The features, which we intend in our project and extract them from Arabic VerbNet documents, are class identification of the verb, types of themroles and types of their restrictions, and the semantic predicate values of the semantic structure in the frame.

## 2.9 Dependency Parsing (MaltParser) with Morphological Features

In (Marton, Habash, & Rambow, June 2010) morphological features (both inflectional and lexical) are added to dependency parsing of Arabic of Modern Standard Arabic (MSA), showing an improvement over form-based features, they explore the contribution of lexical and inflectional morphology features to dependency parsing of Arabic, a morphologically rich language with complex agreement patterns. Using controlled experiments. They distinguish between two types of inflectional features: form-based features and functional features. Functional morphology is at odds with the “surface” (form-based) morphology; a well-known example of this is the “broken” (irregular) plurals of nominal. ROOT, PATTERN, LEMMA, and LMM[1] in their discussion as lexical features nominal lexemes can be further classified into two groups: denoting rational (i.e., human) entities, or irrational (i.e., non-human) entities.

Inflectional features and rationality interact with syntax in two ways. In agreement relations, two words in a specific syntactic configuration have coordinated values for specific sets of features. MSA has standard (i.e., matching value) agreement for subject– verb pairs on PERSON, GENDER, and NUMBER, and for noun–adjective

subject– verb pairs on PERSON, GENDER, and NUMBER, and for noun–adjective

(1) LEMMA: the canonical form or dictionary form of a set of words, for example fly, flies, flew and flying all has the lemma fly. LMM is an undiacritized lemma.

pairs on NUMBER, GENDER, CASE[1], and DET. They achieve 1.69% increase in LAS with inflectional features and 1.77% increase in LAS with lexical features as a best results, many combinations are used with different POS tagset of different corpus.

There are, however, three very common cases of exceptional agreement: Verbs preceding subjects are always singular, adjectives of irrational plural nouns are always feminine singular, and verbs whose subjects are irrational plural are also always feminine singular.

Most available Arabic NLP tools and resources model morphology using form-based ("surface") inflectional features, and do not mark rationality; this includes the Penn Arabic Treebank (PATB) (Maamouri, Bies, Buckwalter, & Mekki, 2004).

It is found in previous results (Marton, Habash, Rambow, & Alkuhlani, 2013), assignment features, specifically CASE and STATE, are very helpful in MSA, where Arabic nouns inflect for state, which has three values: definite, indefinite and construct. The definite state is the nominal form that appears most typically with the definite article and direct addressing with the vocative particle '**يا**' for example, **الكتاب** 'the book'. The indefinite state is used to mark an unspecified instance of a noun, e.g., **كتاب** 'a book'. The construct state indicates that the noun is the head of an Idafa construction, i.e., it is the first word (**مضاف**) that is possessed by the noun phrase that follows it, e.g., the word '**كتاب**' in '**كتاب الطالب**' the book of the student'. For some nouns, like '**كتاب**' the definite and construct state forms are identical.

## 2.10 Dependency Parsing (Maltparser) with Semantic Features

The need for richer information invoked several efforts in the direction of annotating higher order linguistic information in treebanks. It was felt that semantics can be leveraged for syntactic disambiguation and thus semantic annotation was performed in syntactic treebanks to complement the morpho-syntactic annotations (Kingsbury, Palmer, & Marcus, 2002) (Fujita, Bond, Oepen, & Tanaka, 2007) and (MacKinlay,

(1) All Arabic nominal inflect for case, which has three values in Arabic: nominative '**مرفوع**', accusative '**منصوب**', or genitive '**مجرور**'.

Dridan, McCarthy, & Baldwin, 2012) illustrated that semantic annotation delivers a significant improvement for constituency parsing, confirming the hypothesis that semantics can assist syntactic analysis. As well as dependency parsing (Ambati, Jain, & Sangal, 2008) (Ambati, Gade, GSK, & Husain, 2009) (Øvrelid & Nivre, 2007). Also for dependency parsing (Agirre, Bengoetxea, Gojenola, & Nivre, 2011) utilized the English WordNet semantic classes and improved accuracies parsing.

Exploration of lexical databases and ontologies for accessing semantic information are useful for dependency parsing (Jain, Jain, Tammewar, Bhat, & Sharma, 2013).

Our research uses semantic information in dependency parsing on Arabic Verbal Sentence utilizing the information from semantic features available in Lexical Arabic VerbNet to complement the morpho-syntactic information already available.

### 2.10.1 Dependency Parsing with Semantic Annotation Features

The need for richer information invoked several efforts in the direction of annotating higher order linguistic information in treebanks. It was felt that semantics can be leveraged for syntactic disambiguation and thus semantic annotation was performed in syntactic treebanks to complement the morpho-syntactic annotations (Kingsbury, Palmer, & Marcus, 2002) (Fujita, Bond, Oepen, & Tanaka, 2007) and (MacKinlay, Dridan, McCarthy, & Baldwin, 2012) illustrated that semantic annotation delivers a significant improvement in for constituency parsing, confirming the hypothesis that semantics can assist syntactic analysis.

On the basic of dependency parsing (Ambati, Jain, & Sangal, 2008) (Ambati, Gade, GSK, & Husain, 2009). In (Øvrelid & Nivre, 2007) they achieve the best reported results for dependency parsing of Swedish. They show that the addition of linguistically motivated features targeting specific error types may lead to substantial improvements. Such as the addition of information on animacy for nominal elements causes an improvement in overall results (p<.0002). The subject and object functions are the dependency relations whose assignment improves the most when animacy information is added. In (Ambati, Jain, & Sangal, 2008) illustrated that mere animacy (human, non-human and inanimate) of a nominal significantly improves the accuracy of the parser The final performance obtained for parsing Hindi are 69.64% and 88.67% for LAS and UAS respectively on a Treebank as small as 1200 sentences. In

(Ambati, Gade, GSK, & Husain, 2009) they study on extending such information with finer semantic distinctions like time, place, and abstract reconfirmed the substantial role of semantics in syntactic parsing. They achieved an increase of 1.65% and 2.01% in labeled attachment score (LAS) and labeled accuracy (LA) respectively over state-of-the-art data driven dependency parser.

### 2.10.2 Dependency Parsing with Semantic Lexical-Based Features

(Agirre, Bengoetxea, Gojenola, & Nivre, 2011) tested a combined parsing/word sense disambiguation model based in WordNet which do not obtain improvements in parsing. They successfully introduce WordNet classes in a dependency parser, obtaining improvements on the full PTB using gold POS tags, trying different combinations of semantic classes, their experiments show that selecting the adequate combination of semantic features on development data is key for success.

On WordNet Semantic Classes and Dependency Parsing, studying the effect of semantic classes in three dependency parsers, using two types of constituency-to-dependency conversions of the English Penn Treebank. Overall, the improvements are small and not significant using automatic POS tags, contrary to previously published results (Agirre, Bengoetxea, Gojenola, & Nivre, 2011) (Bengoetxea, Agirre, Nivre, Zhang, & Gojenola, 2014) using gold POS tags.

### 2.10.3 Dependency Parsing with Semantic Ontology-based Features

Exploring Semantic Information in Hindi WordNet for Hindi Dependency Parsing, utilizing the information from concept ontologies available in Hindi WordNet to complement the morpho-syntactic information already available. They found concept ontology available in HWN quite resourceful in furnishing features, which can essentially break syntactic ambiguity, resulting in better accuracies for parsing. They perform experiments over datasets of different sizes. They achieve an improvement of 1.1% (LAS) when training on 1,000 sentences and 0.2% (LAS) on 13,371 sentences over the baseline. The improvements are statistically significant at $p<0.01$. The higher improvements on 1,000 sentences suggest that the semantic information could address the data sparsity problem (Jain, Jain, Tammewar, Bhat, & Sharma, 2013).

Two state-of-the-art statistical parsers are trained (Charniak, 2000) (Bikel D. M., 2004) on semantically-enriched input, where content words had been substituted with their semantic classes. This was done trying to overcome the limitations of lexicalized approaches to parsing (Magerman, 1995) (Collins, 1996) (Charniak, 1997) (Collins M. , 2003), where related words, like scissors and knife cannot be generalized, but both knife and scissors with the semantic class TOOL. This simple method allowed incorporating lexical semantic information into the parser. They tested the parsers in both a full parsing and a prepositional phrase PP attachment context. The experiments showed that semantic classes gave significant improvement relative to the baseline, demonstrating that a simplistic approach to incorporating lexical semantics into a parser significantly improves its performance. The work presented the first results over both WordNet and the Penn Treebank to show that semantic processing helps parsing. For the Bikel parser, they achieved a maximal error reduction rate over the baseline parser of 6.9% and 20.5%, for parsing and PP-attachment respectively, using an unsupervised WSD strategy. This demonstrates that word sense information can indeed enhance the performance of syntactic disambiguation. (Agirre, Baldwin, & Martinez, 2008).

Our work improves dependency parsing of verbal Arabic sentences by: Collecting Arabic language Treebank ATP corpus. Then, we extracted and used a dataset out of the corpus with existing morpho-syntactic features and we added semantic features of the entity or tokens of the verb and its modifiers based on Arabic VerbNet. These added semantic features contain: semantic predicates, thematic roles, and semantic classes. We build a parsing model based on these additions that is a data-driven and a history based dependency parser for Arabic verbal sentences. The parser utilizes specific real world aspects of both the lexical and semantic based Arabic VerbNet that achieves parsing improvement over existing parser with 2% increase in accuracy.

## 2.11 Summary

This chapter presented the foundation of our research. It is divided into two related parts. The first part is the state of the art which presents dependency parsing approaches, projective and non-projective dependencies, and brief review of MaltParser including its essential components which are the transition system and the classifier. In this part, we described the evaluation of dependency parsing and related accuracy measures and also the Arabic VerbNet is presented which is the source of our semantic lexical-based features. The second part is a review of various related works which include: inflectional features and semantic features with MaltParser dependency parsing, and related works also included adding the semantic features annotated-based, lexical-based, and ontology-based works in dependency parsing with MaltParser.  Next chapter display the research architecture and the needed stages used to build, the stages are data preparation, parsing model, and results evaluation. We present the steps and tasks are needed to accomplish each stage.

# Chapter3

# Dependency Parsing of Verbal Arabic Sentences

# Chapter 3
# Dependency Parsing of Verbal Arabic Sentences

The dependency parsing community has for the last few years shown considerable interest in parsing morphologically rich languages with flexible word order such as Arabic. This is partly due to the increasing availability of dependency treebanks for such languages, but it is also motivated by the observation that the performance obtained for these languages has not been very high (Ambati, 2011). More efforts apply to the parsers to narrow this gap. Some efforts apply to data while others apply to the parsers themselves as present in previous chapter.

Our approach intends to enhance parsing by applying extended lexical-based semantic features to existing morpho-syntactic features in the data and adapting feature model of the parser. The prepared modified data decreases the sparsity of the data which leads to notable dependency parsing enhancement. The general architecture of our approach is shown in Figure 3.1.

The architecture shows the three essential stages of the proposed approach, namely data preparation, parsing model and results evaluation. The input to the approach is an MSA Arabic wirenews dataset. First, a .CoNLL format preparation is performed on the dataset which already has morpho-syntactic features. Then we extract the semantic features from Arabic VerbNet and add them to the prepared data. The modified data is then input to the parsing model as training data. The parser learns from the training data which has morpho-syntactic features as well as semantic features formatted in what is called FEATS column.

In the parsing model and during the parsing task, we use test data for parsing where the HEAD and DEPRAL are removed (they are stored in the seventh and eighth columns of the test data) must be removed. They are replaced by new HEAD numeric values and DEPRAL values such subject, object, root, etc. produces by the parsing task.

The third stage in our deterministic dependency parsing is evaluating the results. To this end, we use MaltEval (an evaluation and visualization tool for dependency parsing) to compute the accuracy scores of the results. These scores are: Unlabeled

Attachment Score (UAS), Labeled Attachment Score (LAS) and Label Accuracy (L). These scores show the effectiveness and degree of improvement in the dependency parsing of the verbal sentences in the test data. Next we elaborate on these three stages starting with data preparation.

## 3.1 Data Preparation

The prepared data is the input of our parsing model. To prepare CoNLL format data for our research we add lemmatization values of the words in LEMMA column, and due to LEMMA verb value we extract the semantic features from Arabic VerbNet.

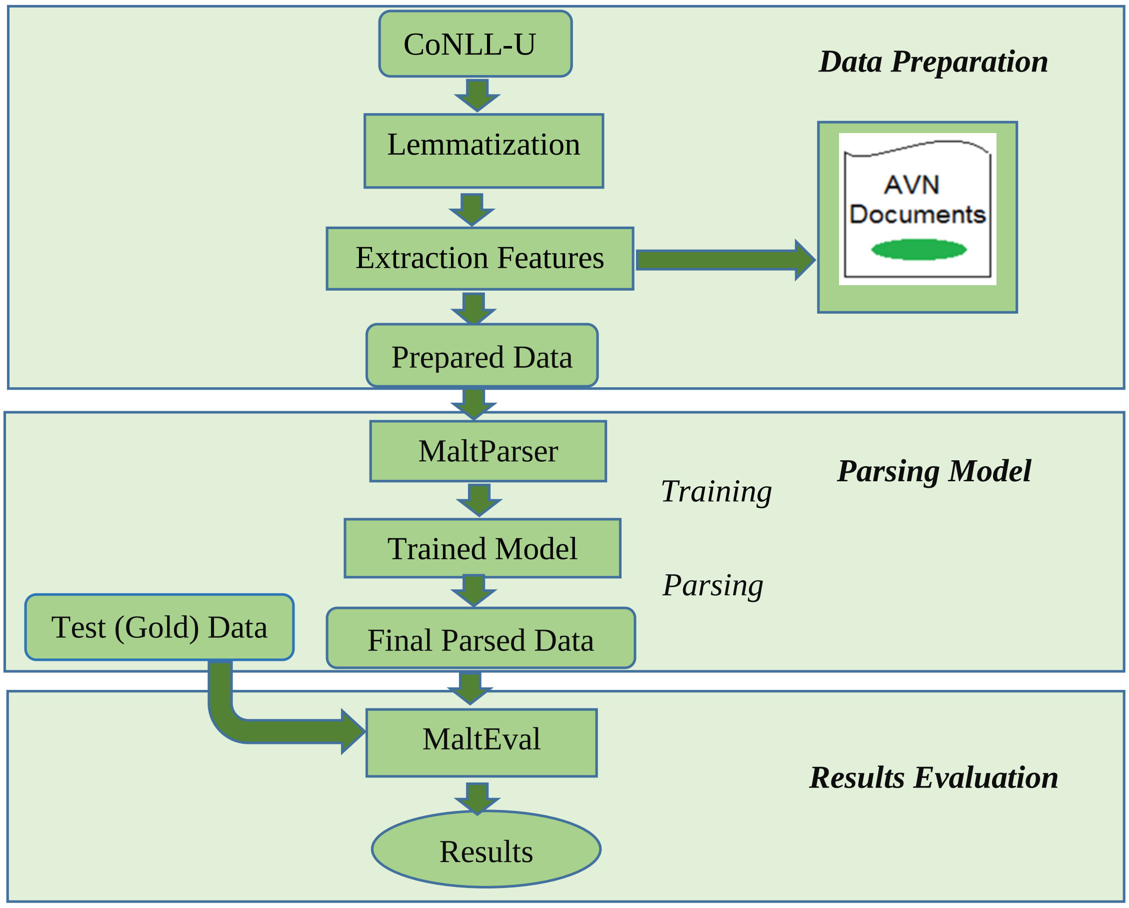


Figure (3.1): Dependency Parsing Architecture

The extracted semantic features are append to existing morpho-syntatic features in FEATS column in the CoNLL format data.

## 3.2 CoNLL Format Representation

We use a revised version of the CoNLL-X format called CoNLL-U (Annodoc & brat, 2014). Annotations are encoded in plain files (UTF-8, using only the LF character as line break) with three types of line:

- Word lines containing the annotation of a word/token in 10 fields separated by single tab characters;
- Blank lines marking sentence boundaries.
- Comment lines starting with a hash (#).

CoNLL format is the standard format being used in CoNLL Shared Tasks on dependency parsing. This is a ten-column format. A short description of these columns is presented in Table 3.1.

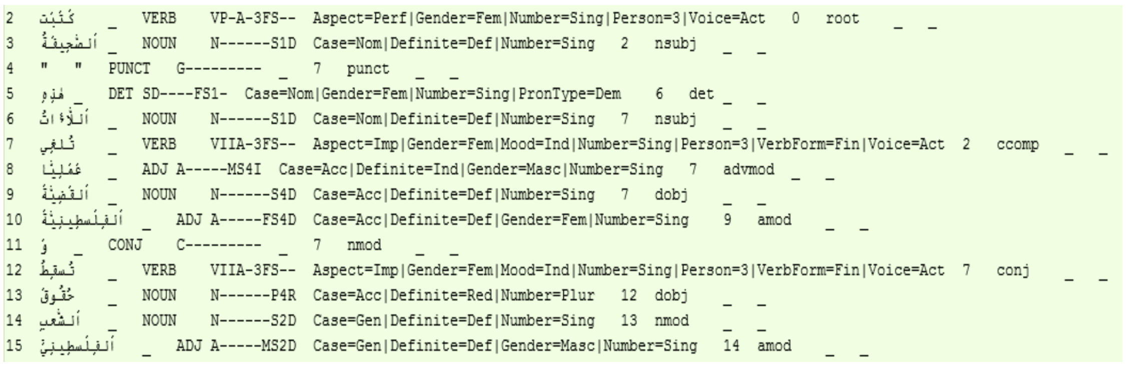
Table (3.1): Columns in CoNLL format

| Field number: | Field name: | Description: |
|---|---|---|
| **1** | ID | Token counter, starting at  for each new sentence |
| **2** | FORM | Word form or punctuation symbol |
| **3** | LEMMA | Lemma or stem of word form, or an underscore if not available |
| **4** | CPOSTAG | Coarse-grained POS tag |
| **5** | POSTAG | Fine-grained POS tag |
| **6** | FEATS | Unordered set of syntactic and/or morphological features, separated by a vertical bar (\|) |
| **7** | HEAD | Head of the current token, which is either a value of ID or zero (‘0’) |
| **8** | DEPRAL | Dependency relation to the HEAD |
| **9** | PHEAD | Projective head of the current token, which is either a value of ID or zero (‘0’) |
| **10** | PDEPRAL | Dependency relation to the PHEAD |

Out of the ten columns, the columns ID, FORM, CPOSTAG, POSTAG, HEAD and DEPREL are mandatory and the rest are optional. In case of optional columns, if the information is not available, an underscore is present. Except FEATS column, all other columns are fixed. But, in FEATS column we can have any useful information other than the information encoded in the fixed columns.

Each row/line represents a node in the sentence. A blank line separates each sentence. The format can handle UTF (LF)[8].

Consider the first row in the CoNLL format presented below in Figure 3.2. '2' is the ID of the node in the sentence. 'كتبت' is the word, '_' is the LEMMA form of the word 'VERB', and 'VP-A-3FS-- are the coarse-grained POS tag and fine-grained POS tags respectively. As this node is the root of the sentence. HEAD and DEPREL are '0' and 'root' respectively. All these are fixed columns. FEATS column is used to represent the extra information in the form of gender, number, person, case, and suffix.



```
2   كَتَبَت       _    VERB    VP-A-3FS--   Aspect=Perf|Gender=Fem|Number=Sing|Person=3|Voice=Act   0   root    _   _
3   اَلصَّحِيفَةُ   _    NOUN    N------S1D   Case=Nom|Definite=Def|Number=Sing   2   nsubj   _   _
4   "   "   PUNCT   G---------   _   7   punct   _   _
5   هَذِهِ  _   DET SD----FS1-   Case=Nom|Gender=Fem|Number=Sing|PronType=Dem    6   det _   _
6   اَلأَوْرَاقُ   _    NOUN    N------S1D   Case=Nom|Definite=Def|Number=Sing   7   nsubj   _   _
7   تُلغِي      _    VERB    VIIA-3FS--   Aspect=Imp|Gender=Fem|Mood=Ind|Number=Sing|Person=3|VerbForm=Fin|Voice=Act  2   ccomp   _   _
8   عَمَلِيًّا     _    ADJ A-----MS4I   Case=Acc|Definite=Ind|Gender=Masc|Number=Sing   7   advmod  _   _
9   اَلقَضِيَّةَ   _    NOUN    N------S4D   Case=Acc|Definite=Def|Number=Sing   7   dobj    _   _
10  اَلفِلَسطِينِيَّةَ   _    ADJ A-----FS4D   Case=Acc|Definite=Def|Gender=Fem|Number=Sing    9   amod    _   _
11  وَ   _   CONJ    C---------   _   7   nmod    _   _
12  تُسقِطُ     _    VERB    VIIA-3FS--   Aspect=Imp|Gender=Fem|Mood=Ind|Number=Sing|Person=3|VerbForm=Fin|Voice=Act  7   conj    _   _
13  حُقُوقَ     _    NOUN    N------P4R   Case=Acc|Definite=Red|Number=Plur   12  dobj    _   _
14  اَلشَّعبِ    _    NOUN    N------S2D   Case=Gen|Definite=Def|Number=Sing   13  nmod    _   _
15  اَلفِلَسطِينِيِّ   _    ADJ A-----MS2D   Case=Gen|Definite=Def|Gender=Masc|Number=Sing   14  amod    _   _
```

Figure (3.2): The CoNLL format practical example

### 3.2.1 Lemmatization Using MADAMIRA

MADAMIRA is the combination and refinement of two valuable tools in Arabic NLP: MADA and AMIRA. MADA is a system for Morphological Analysis and Disambiguation for Arabic (Pasha, et al., 2015). The primary purpose of MADA is to, given raw Arabic text, derive as much linguistic information as possible about each word in the text, thereby reducing or eliminating any ambiguity surrounding the word. MADA does this by using ALMOR (an Arabic lexeme-based morphology analyzer) to generate every possible interpretation (or analysis) of each input word. MADA then applies a number of language models to determine which analysis is the most

[8] In DOS/Windows text files, a line break, also known as newline, is a combination of two characters: a Carriage Return (CR) followed by a Line Feed (LF). In Unix text files a line break is a single character: the Line Feed (LF)

probable for each word, given the word's context. MADA also includes TOKAN, a general tokenizer for MADA-disambiguated text. TOKAN uses the information generated by the MADA component to tokenize each word according to a highly-customizable scheme. AMIRA is a system for tokenization, part-of-speech tagging, Base Phrase Chunking (BPC) and Named Entity Recognition (NER). These components can be used in Arabic parsing. MADAMIRA is designed with the goals of being a functional replacement for MADA and AMIRA, being platform-independent, and providing the ability to process Arabic text at a much faster rate than the older tools. In addition. MADAMIRA is designed to process MSA. The third column in our CoNLL format dataset was filled in its LEMMA values by using MADAMIRA tool. Depending on LEMMA values and coarse-grained POS tag 'VERB' we extract the needed features from Arabic VerbNet.

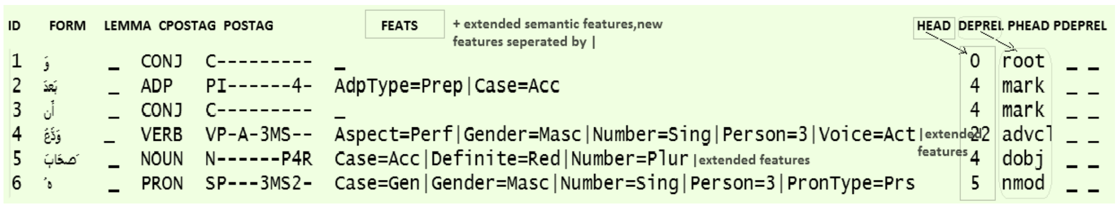
We can use MADAMIRA in two ways, the first by invoking it from our code and perform the required configuration setting. The second by passing data input using the command prompt command.

### 3.2.2 Extracting Semantic Features from Arabic VerbNet

The semantic features have been extracted from Arabic VerbNet (Section 8.2) and are added in FEATS column in CoNLL data format. They are as follows:

- **Class Id**

The main elements of AVN are the verb classes. A verb class is an XML document with a DTD (data type definition) describing the elements, attributes and values allowed to appear. The root element (AVNCLASS) names the class itself. The child elements are the members, thematic roles and the frames. A member is constituted of the verb itself, the root the deverbal and the participle. Each frame has an example sentence, a syntactic structure and a semantic structure. The rest of the class children are the subclasse(s) and the sibling classe(s). The last two elements can be empty in cases where classes do not possess them. The candidate choice of the class depends on finding the verb LEMMA form in its members; the same LEMMA verb may be included in many classes.



+ extended semantic features,new features seperated by |

| ID | FORM | LEMMA | CPOSTAG | POSTAG | FEATS | HEAD | DEPREL | PHEAD | PDEPREL |
|---|---|---|---|---|---|---|---|---|---|
| 1 | وَ | _ | CONJ | C--------- | _ | 0 | root | _ | _ |
| 2 | بَعدَ | _ | ADP | PI------4- | AdpType=Prep\|Case=Acc | 4 | mark | _ | _ |
| 3 | أن | _ | CONJ | C--------- | _ | 4 | mark | _ | _ |
| 4 | وَدَّعَ | _ | VERB | VP-A-3MS-- | Aspect=Perf\|Gender=Masc\|Number=Sing\|Person=3\|Voice=Act \|extended features | 2 | advcl | _ | _ |
| 5 | صحابَ | _ | NOUN | N------P4R | Case=Acc\|Definite=Red\|Number=Plur \|extended features | 4 | dobj | _ | _ |
| 6 | هُ | _ | PRON | SP---3MS2- | Case=Gen\|Gender=Masc\|Number=Sing\|Person=3\|PronType=Prs | 5 | nmod | _ | _ |

Figure (3.3): CoNLL-X dataset columns

• **Thematic Roles and Restrictions**

AVN adopted the entire set of thematic roles used in English VerbNet, however, it expanded the scope of some them to encompass more cases than those for which they were originally designed. According to (Schuler, 2005), these roles are: agent, patient, theme, experiencer, actor, cause, recipient, beneficiary, instrument, location, destination, source, asset, extent, stimulus, attribute, proposition, topic, time, material, and product. As mentioned above, many verbs impose selectional restrictions to constrain the type of thematic roles. In sum, there are 36 selectional restrictions (semantic types) taken from EuroWordNet Interlingua (Vossen, 1998) (Vossen, 2004). Figure 3.4 shows the hierarchy of these restrictions, which, are used in combination with the operators [+/-], where the positive value describes the case where the restriction applies for a role and the negative value when the opposite property applies. Thus, the restriction [+Solid] describes the physical property of a solid object [-Solid] describes the physical property of a liquid object. Furthermore, selectional restrictions can be combined using the logical operators [AND/OR]. Section 4.3 illustrates the thematic roles and their restrictions added to the dataset used in our model.

• **Semantic structures**

The last important information provided by the frames is the semantic structure. Basic predicates like cause, state, use or instrument are conjoined in association with information on thematic roles as well as temporal information to convey the core semantic meaning of each frame. The compositional nature has the advantage of allowing verb sense extension or modification contributed by morphological morphemes or by different types of arguments (subcategorized and unsubcategorized).

### 3.2.3 Adding the Extracted Semantic Features to the Data

An important stage in our parsing architecture is data preparation, where we add the semantic features which are extracted from ArabicVerbNet. As notify in figure 3.3.

The UML class diagram in figure 3.5 illustrates the design of data preparation. It consist of:

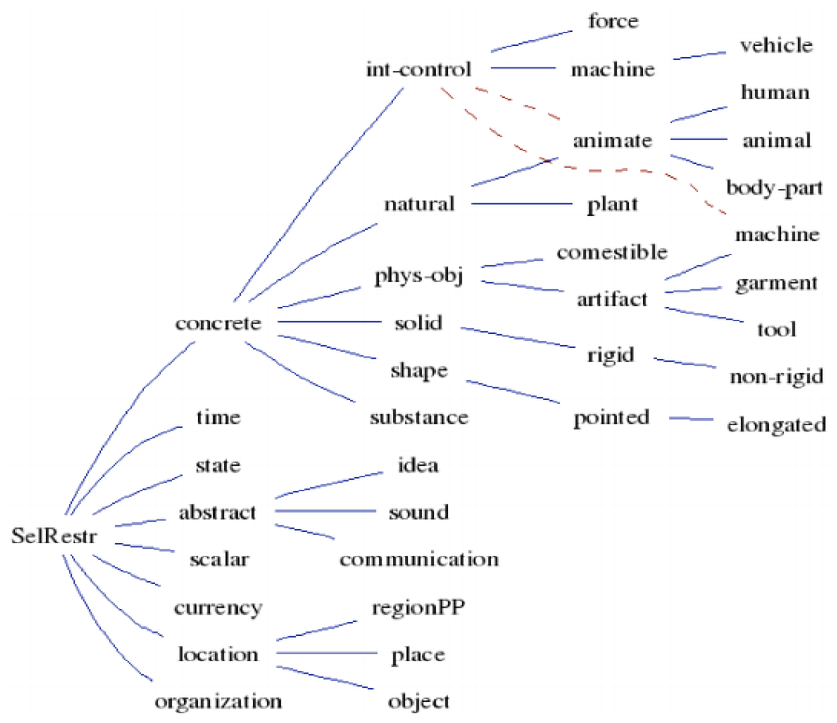


Figure (3.4): Selectional restrictions on thematic roles[9].

• **VerbReader Class**

VerbReader is used to extract the features from Arabic VerNet and add them to the FEATS column in CoNLL format dataset, dataset used is a newest version portion of PADT as shown in Section 2.2. Due to LEMMA values of the verb, the iteration is performed on the XML AVN documents or classes to decide if this verb is a member of the class or not, and if that VerbReader extracts all the needed features such as class id, themroles, and semantic roles.

---

(9) This figure was taken from the official website of English VerbNet http://verbs.colorado.edu/~mpalmer/projects/VerbNet.html

• **MemberThemeRoles**

MemberThemeRoles is invoked when the features are drown as a string, At the time VerbReader finds the LEMMA of the verb in AVN documents and extracts the needed features, MemberThemeRoles is already invoked for drawing the features as a string. The string is extended to existing morpho-syntactic features in FEATS column and VERB's records (Figure 3.3).

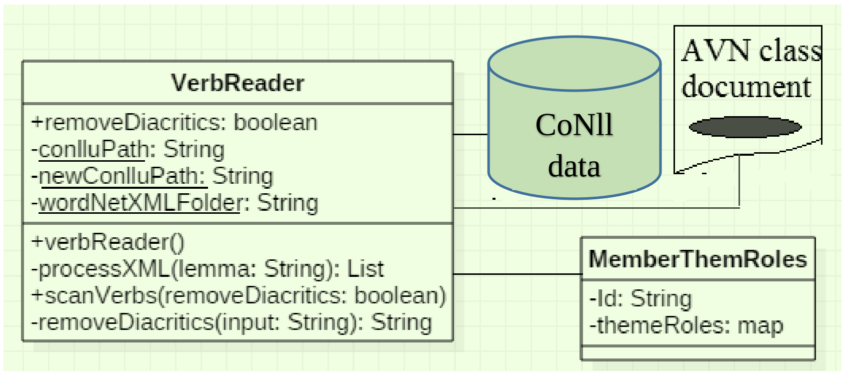

VerbReader
+removeDiacritics: boolean
-conlluPath: String
-newConlluPath: String
-wordNetXMLFolder: String
+verbReader()
-processXML(lemma: String): List
+scanVerbs(removeDiacritics: boolean)
-removeDiacritics(input: String): String
CoNll data
AVN class document
MemberThemRoles
-Id: String
-themeRoles: map

Figure (3.5): Class diagram for data preperation

## 3.3 Parsing Model

The parser now takes prepared data in CoNLL-U format as input, for our experiments. The verb already have 7 morphological features in FEATS column namely aspect, gender, number, person, mood, verb form, or voice Appendix E. We consider additional different which is extended semantic features that have been extracted from Arabic VerbNet. We create a parsing model based on the **convert**, **learn** and **parse** modes respectively as shown in Figure 3.6 .The final parsed data is the output of the model.

### 3.3.1 MaltParser General Parameters

Any deterministic parsing algorithm compatible with the MaltParser architecture can be implemented in the MaltParser package.

MaltParser 1.9.0 contains three families of parsing algorithms: Nivre, Covington and Stack. We experimented with all the algorithms, the stacklazy Stack algorithm gave better performance over others consistently. It provides option for one learning algorithm which is *libsvm* learning method. Finally, we tested the performance by adapting the CoNLL shared task 2007 (Nivre, et al., 2007b). SVM classifier settings used by the same parser for various languages (Hall, et al., 2007).

### 3.3.2 MaltParser Feature Model

One of the advantages of the transition-based approach to dependency parsing is that it enables rich history-based feature models for predicting the next transition, and MaltParser provides an expressive specification language for defining feature models of arbitrary complexity. Features are defined relative to tokens in the main data structures for a given parsing algorithm (Section 4.7), which normally include at least a stack holding partially processed tokens and a buffer holding remaining input tokens. The data structure used in our feature model is the Stack, where we use the stacklazy Stack algorithm. The actual feature values are normally linguistic attributes of one or more tokens. The following attributes (columns) are available in the CoNLL-X format assumed by the parser

1. FORM:Word form.
2. LEMMA: Lemma.
3. CPOSTAG: Coarse-grained part-of-speech tag.
4. POSTAG: Fine-grained part-of-speech tag.
5. FEATS: List of morphosyntactic features (e.g., case, number, tense, etc.)
6. DEPREL: Dependency relation to head.

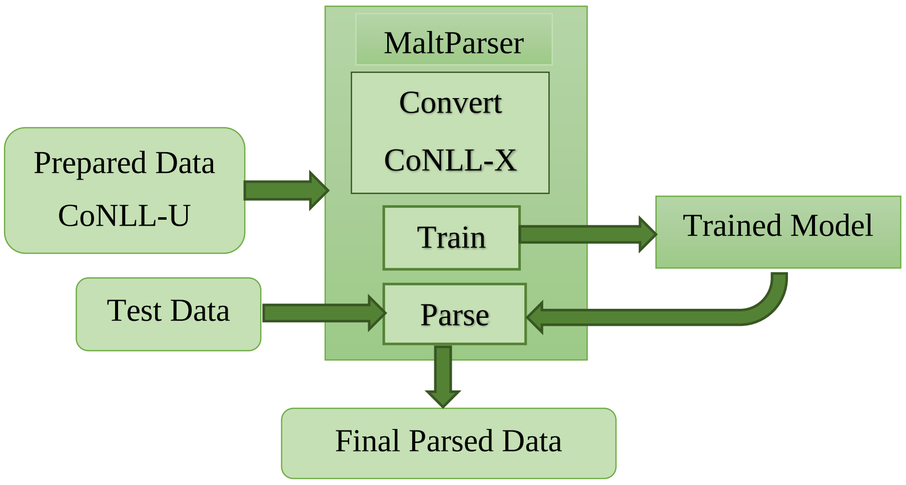


Figure (3.6): Parsing model

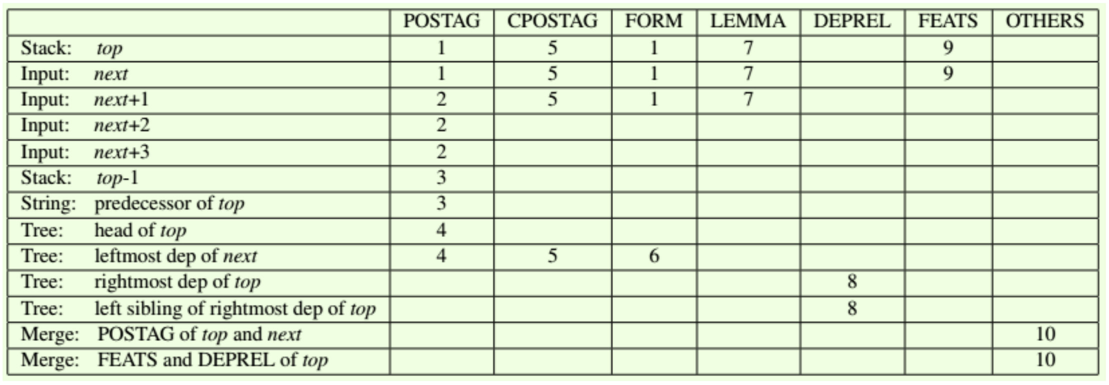
The attributes (columns) LEMMA and FEATS are not available in all data sets, and the CPOSTAG and POSTAG tags are sometimes identical. The DEPREL attribute is only available dynamically in the partially built dependency tree (Ballesteros & Nivre, 2012).

### 3.3.3 MaltParser Feature Model Selection

MaltParser uses history-based feature models for predicting the next action in the deterministic derivation of a dependency structure, which means that it uses features of the partially built dependency structure together with features of the (tagged) input string. Features that make use of the partially built dependency structure corresponds to the OUTPUT category of the data format, for example DEPREL in the CoNLL-X data format, and features of the input string corresponds to the INPUT category of the data format, for example CPOSTAG and FORM  (Hall, Nilsson, & Nivre, 2014).

We follow the feature model selection in (Nivre J., et al., 2007) which have taken wide variety of features and grouped them into 10 groups (indicated by numbers 1 – 10 in Table 3.3). In Table 3.3, first column lists different features and first row represents different columns in the CoNLL format. For this forward features selection technique is used to incrementally add different feature groups and analyzed their impact on parsing accuracy. We adapt feature model Appendix D for arc-eager specified as XML file but according to parsing algorithm data structure 'stack' and feature pool used for stacklazy algorithm as shown in next chapter.

Table (3.2): Feature pool used for arc-eager algorithm of Malt.

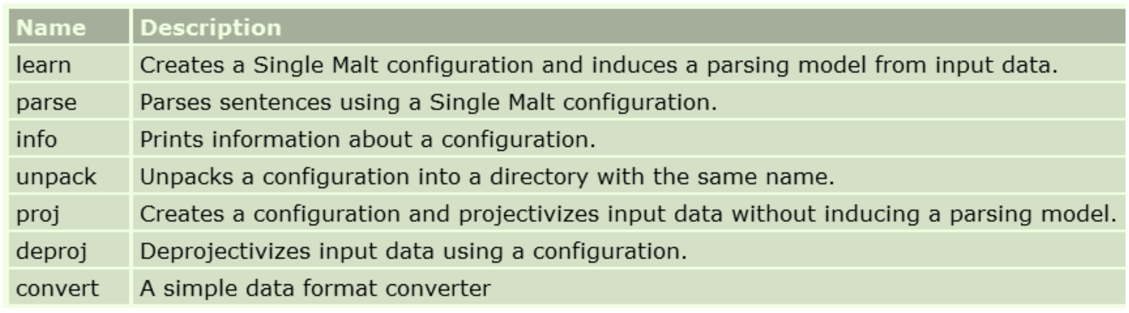


| | POSTAG | CPOSTAG | FORM | LEMMA | DEPREL | FEATS | OTHERS |
|---|---|---|---|---|---|---|---|
| Stack: *top* | 1 | 5 | 1 | 7 | | 9 | |
| Input: *next* | 1 | 5 | 1 | 7 | | 9 | |
| Input: *next*+1 | 2 | 5 | 1 | 7 | | | |
| Input: *next*+2 | 2 | | | | | | |
| Input: *next*+3 | 2 | | | | | | |
| Stack: *top*-1 | 3 | | | | | | |
| String: predecessor of *top* | 3 | | | | | | |
| Tree: head of *top* | 4 | | | | | | |
| Tree: leftmost dep of *next* | 4 | 5 | 6 | | | | |
| Tree: rightmost dep of *top* | | | | | 8 | | |
| Tree: left sibling of rightmost dep of *top* | | | | | 8 | | |
| Merge: POSTAG of *top* and *next* | | | | | | | 10 |
| Merge: FEATS and DEPREL of *top* | | | | | | | 10 |

### 3.3.4 MaltParser Tasks (Modes)

MaltParser have seven pre-defined flow charts that describe what tasks MaltPasrer should perform. These seven flow charts are shown in table 3.3. In Section 4.6 MaltParser configuration, creates a parsing model based on the **convert**, **learn** and **parse** modes respectively. Due to our parsing model configuration first we convert the data from CoNLL-U format to CoNLL-X, second the parser trains (learns) from the prepared data, after training task we have a trained model as shown in figure 3.6. Finally we parse the test data, the test data is an input to trained model. We get the final parsed data as output of the parsing model as shown in Figure 3.6.

Table (3.3): Seven flow chart of the MaltParser tasks



| Name | Description |
|---|---|
| learn | Creates a Single Malt configuration and induces a parsing model from input data. |
| parse | Parses sentences using a Single Malt configuration. |
| info | Prints information about a configuration. |
| unpack | Unpacks a configuration into a directory with the same name. |
| proj | Creates a configuration and projectivizes input data without inducing a parsing model. |
| deproj | Deprojectivizes input data using a configuration. |
| convert | A simple data format converter |

### 3.3.5 Test Data and Parsed Data

After training task we have a trained model, in other words the parser have learned from the prepared data. In parse task (mode) we use the test data as an input to the trained model, the parser should be executed in the parsing mode and get the result which is the final parsed data as shown in Figure 3.6. In conclusion the test (gold) data is an input to parsed model and the parsed data is the output of the trained model.

## 3.4 Evaluating Scores

The third and last stage in the dependency parsing approach as depicted in the architecture (Figure 3.1) is evaluating the results.

The final parsed data is the input of this stage, the results of score metric values are the output.

We use MaltEval tool, an evaluation and visualization tool for dependency parsing (Nilsson & Nivre, 2007). The metric values used to evaluate the results are LAS, UAS, and LA (see Section 2.7.2).

In LAS, default value (BothRight) a token is counted as a hit if both the head and the dependency label in the final parsed data are the same as in the gold-standard data. In LA (LabelRight) a token is counted as a hit if the dependency label in the final parsed data is the same as in the gold-standard data.

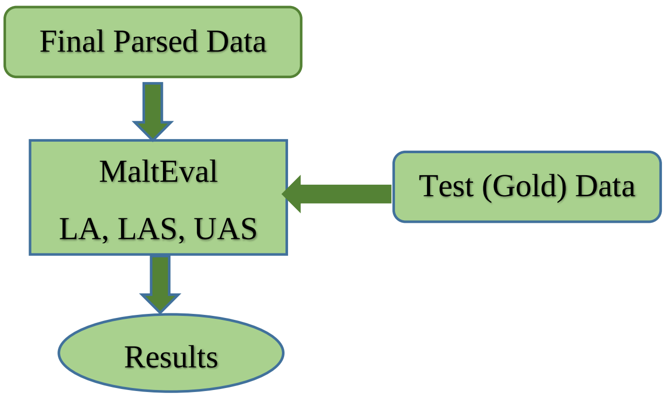


Figure (3.7): Results evaluation

In UAS (HeadRight) a token is counted as a hit if the head in the final parsed data is the same as in the gold-standard data. MaltEval has a module for viewing the content of the gold-standard and parsed files visually as shown in Figure 2.3. Parsed data and gold data (parsed data before testing), are inputs to MaltEval tool (for more detail, see Section 2.7.2.)

## 3.5 Summary

We introduced the basic concepts, tools, and stages needed to perform the proposed dependency parsing. We presented the parsing approach through an architecture. The dependency parsing stages based on the architecture are: preparing data, parsing model, and results evaluation. Each stage includes several steps. In data preparation phase, we talked about extracting the features from AVN, defining semantic features and performed lemmatization to obtain LEMMA values using MADAMIRA tool and adding semantic features to data. In parsing model we presented the parsing tasks which are the convert task, the train task and the parse task. During the parsing process, we set parser parameters such as feature model, algorithm, and other features. Finally, we presented the scores used to evaluate the results which are LA, LAS, and UAS. Next chapter presents the implementation of the dependency parsing approach as reflected in the stages of the architecture.

# Chapter 4

# Implementation

# Chapter 4

# Implementation

This chapter presents the implementation of the dependency parsing approach of Arabic verbal sentences as described in Chapter 3, particularly, it presents the realization of the architecture of the approach shown in Figure 3.1. First, we describe the dataset type and division, then we proposes a scheme for classifying verbs of Modern Standard Arabic (MSA) in Arabic VerbNet. Based on data preparation stage we propose the features which we intend and extract for the approach from Arabic VerbNet documents. The features are class identification of the verb, types of themroles and types of their restrictions, and the semantic predicate values of the semantic structure in the frame. We extract the semantic features from Arabic VerbNet documents due to LEMMA value in our data which is filled by using MADAMIRA tool and java code. After data preparation we create our parsing model and conduct our experiments, we set the MaltParser with its parameters provides options give us the best results. The contributing experiment is the data with LEMMA and semantic features, the second without semantic features, finally the baseline where the Lemmatization and semantic features are both absent .Finally in the last stage we evaluate the experiments' results using MaltEval tool.

## 4.1 The Dataset

The data used in the experiments are based on a subset of the HamleDT 3.0, which is a collection of linguistic textual data in multiple languages. The data is divided into two parts train data and test data (Diab, Habash, Rambow, & Roth, 2013). The train data contains different stories in sport, politics, weather, etc. collected from F.B. wirenews and comprises over eleven thousands tokens of data they are used in our experiments while annotated analytically and are provided with the disambiguated morphological information. We use .CoNLL format which is the standard format being used in CoNLL Shared Tasks on dependency parsing (Nivre J. , et al., 2007). It is a ten-column format as described in Section 3.1.1.As shown in Figure 4.1 the test data contains 6 news items with 1564 tokens and are divided as shown in table 4.1.

The train data set is the data, which we prepare and add the lemmatization and needed semantic features to it.

Table (0.1): Dataset Division

| Division | #Docs | #Statements | #Words |
|---|---|---|---|
| **Train** | 43 | 332 | 10,900 |
| **Test** | 6 | 45 | 1564 |

## 4.2 Lemmatization

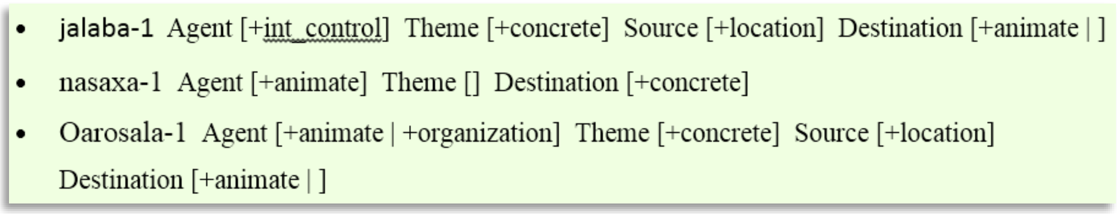
LEMMA is the canonical form or dictionary form of a set of words. The LEMMA of the words (يُمَثِّلُونَ ، تَمثُلُ ، يُمَثِّلُ) in our dataset is 'مَثَّل'. We invoke MADAMIRA tool Section 3.1.2 programmatically to fill the third column of the data which is belong LEMMA values. In our work we consider only LEMMA of the verbs. LEMMA values of verbs are the input to our program. One form of the verbs in the Arabic VerbNet is LEMMA forms. So depending on LEMMA verbs' values, we extract all the features of the specific verbs. Our data contains 823 verbs, but 300 LEMMA. The LEMMA of the verb "أشار" for instance, is duplicated 11 times, otherwise the verb "قال"is duplicated 21 times.

## 4.3 Semantic Features

When we find a verb in .xml Arabic VerbNet documents' *member* elements, immediately three types of semantic features are extracted from the document (classes), which are classID, thematic roles, and restrictions belong thematic roles, notice that AVN folder contain 335 xml documents.

It is obvious that the same verb could be found in different documents or AVN classes (polysemy), so we choose manually the suitable feature groups that agree with the verb in the *context*. For example the LEMMA 'نَقَل' has three options of the semantic feature groups (Figure 4.1). All options or semantic features groups are drawn in FEATS column at this verb record in the form such as in Figure 4.2, but we remain just one choice and delete the others manually, in our example the second option is our choice, notice that each feature is separated by each other using '|' regard to CoNLL data format. The feature group include 4 semantic features, in our group choice, which are the classID is nasaxa-1, and we have three thematic roles (Agent, Destination, Theme) with their restrictions (animate, concrete, no restriction) respectively. If the thematic role has no restriction it is written in the form *thm=thematic-role*, so the semantic features drawn in training data as a string such as

|CId=nasaxa-1|Agent=animate|Destination=concrete|thm=Theme



- jalaba-1 Agent [+int_control] Theme [+concrete] Source [+location] Destination [+animate | ]
- nasaxa-1 Agent [+animate] Theme [] Destination [+concrete]
- Oarosala-1 Agent [+animate | +organization] Theme [+concrete] Source [+location] Destination [+animate | ]

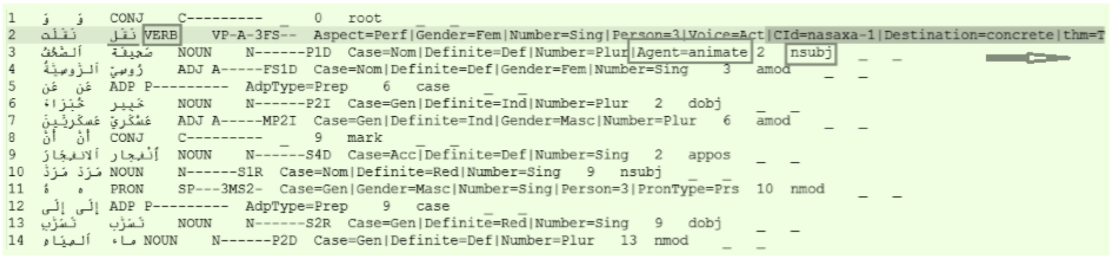
Figure (0.1): Four options for the semantic features group of the verb 'نَقَلَ

The selected option should be compatible with the meaning of the sentence and agree with the *context*. Manually we choose the suitable option and remove the others. The following example demonstrate this point. The sentence,

**'و نقلت الصحف الروسية عن خبراء عسكريين أن الانفجار مرده الى تسرب المياه'**

in Figure 4.2 the verb '**نقل**' should has the second option above where the semantic feature group is ClassId=nasaxa-1, thematic role Destination=concrete, thematic role Agent=animate, and thematic role Theme. When the thematic role is *agent*, we cut it and its restriction value and paste them at the row, where DEPRAL value is subject in FEATS column as shown in Figure 4.2. On the other hand, many verbs could found in the same semantic feature group such as **حصل ،فاز ،حظي**.

|CId=faAza-1|Agent=animate|Agent=organization|thm=Theme|Source=concrete.



```
1   وَ    وَ    CONJ   C---------   _   0   root   _   _
2   نَقَلَت   نَقَل VERB   VP-A-3FS--   Aspect=Perf|Gender=Fem|Number=Sing|Person=3|Voice=Act|CId=nasaxa-1|Destination=concrete|thm=T
3   اَلصُّحُفُ   صَحِيفَة   NOUN   N------P1D   Case=Nom|Definite=Def|Number=Plur|Agent=animate 2   nsubj   _   _
4   اَلرُّوسِيَّةُ رُوسِيّ   ADJ A-----FS1D   Case=Nom|Definite=Def|Gender=Fem|Number=Sing   3   amod   _   _
5   عَن   عَن   ADP P---------   AdpType=Prep   6   case   _   _
6   خُبَرَاءَ   خَبِير   NOUN   N------P2I   Case=Gen|Definite=Ind|Number=Plur   2   dobj   _   _
7   عَسْكَرِيِّينَ عَسْكَرِيّ   ADJ A-----MP2I   Case=Gen|Definite=Ind|Gender=Masc|Number=Plur   6   amod   _   _
8   أَنْ   أَنْ   CONJ   C---------   _   9   mark   _   _
9   اَلانفِجَارَ إِنفِجَار   NOUN   N------S4D   Case=Acc|Definite=Def|Number=Sing   2   appos   _   _
10  مَرَدُّ مَرَدّ NOUN   N------S1R   Case=Nom|Definite=Red|Number=Sing   9   nsubj   _   _
11  هُ   ه   PRON   SP---3MS2-   Case=Gen|Gender=Masc|Number=Sing|Person=3|PronType=Prs   10   nmod   _   _
12  إِلَى إِلَى ADP P---------   AdpType=Prep   9   case   _   _
13  تَسَرُّبِ   تَسَرُّب   NOUN   N------S2R   Case=Gen|Definite=Red|Number=Sing   9   dobj   _   _
14  اَلمِيَاهِ   مَاء NOUN   N------P2D   Case=Gen|Definite=Def|Number=Plur   13   nmod   _   _
```

Figure (0.2): Practical example

We have added semantic feature in FEATS column, the 6th column, as notify in Figure 4.2. The semantic features extracted from Arabic VerbNet are added to the existing morpho-syntactic features in FEATS column. The semantic features are classId, thematic roles, and their restrictions. Many verbs impose selectional restrictions to constrain the type of thematic roles. In sum, there are 36 selectional restrictions (semantic types) as demonstrate in chapter 3 Section 3.1.3.

### 4.3.1 Thematic Roles

There are 21 thematic roles.in Arabic VerbNet, we pass and use all of them in preparing the training data (Section 3.1). The thematic roles with an example of a sentence contains the verb of this semantic role from the prepared data is presented below

***Agent:*** encodes an active and volitionally acting ‘doer’ or instigator of the action denoted by the verb, which is generally a human or an animate subject. In our experiments, we pass 4 types of restriction of the agent role, which are animate, machine, human, and organization. This role is the only one, which we cut from verb record in FEATS column and paste it in the record, which is the DEPRAL column, is subject.

***Patient:*** entail a change of state of the participant for example,

’***أطلق*** النار عليه فأرداه“ .

‘He ***shot*** and killed him’.

***Theme:*** describes an entity in a certain position or location or an entity moved as a result of the action denoted by the verb, for example.

"***تدخل*** أفراد الشرطة".

‘Police officers ***intervene***’.

***Experience:*** the living entity that experiences the action or event denoted by the verb. It is assigned by transitive or intransitive psych verbs, verbs of perception and verbs involving bodily-reflexes.

"الحرائق لم تعد***تهدد*** المناطق الماهولة".

‘Fire no longer ***threatening*** populated areas’.

***Actor:*** generally describes agent-like participants of verbs that do not imply a change of state. These verbs are mainly verbs of communication and involves explicit or implicit coequal participants: actor1 and actor2, for example.

"الرئيس الأمريكي ***سيشارك*** في قمة الثمانية".

'US President ***will participate*** in the eight summit'.

***Cause:*** it is used to describe an agent-like participant causing a living entity to experience an emotional state. It is mainly used for psych verbs and verbs involving the body. The cause can be a concrete entity (human, organization or force) or an abstract entity (e.g. sadness, hunger, happiness).for example.

"أحد العدائين الثلاثة الذين ***هزموا*** عرين هذا العام".

'One of the three runners who ***defeated*** den this year'.

***Recipient:*** the target of a transfer of concrete or abstract entities. It is found with verbs of possession transfer, information transfer and a variety of verbs involving the body, for example.

"***أشار*** الى أن خبراء آثار تمكنوا من الكشف على الردم",

'He ***pointed*** out that the effects of experts were able to detect the backfill'.

***Beneficiary:*** the entity benefiting from the action denoted by the verb. In Arabic the benefactive is in most cases assigned to a prepositional phrase headed by**"لـ أو من أجل"** for example,

"أنهار جميلة***تشكل*** الحوض الرئيسي لـنهر اوب" .

'Beautiful rivers ***form*** the main basin of Ob River'.

***Instrument:*** the object or the force by which the action or the event denoted by the verb is carried out. It is introduced mostly by verbs involving a change of state. However, some classes of verbs allow the instrument to be assigned to the subject (subject instrument alternation), for example,

"مطار هذه المدينة***أقفل*** منذ سنوات".

'This city's airport ***was closed*** years ago'.

***Location:*** generally an unspecified place [84], destination or source. It is introduced by prepositional phrases only, for example,

”حادث اطلاق النار الذي ***وقع*** مساء الجمعة“.

‘Shooting accident, which ***occurred*** on Friday evening’.

***Destination:*** the end point of a directed or undirected motion, for example,

”***تدخل*** افراد الشرطة للقبض عليه“.

‘Police officers ***intervene*** to catch him’.

***Source:*** the starting point of a directed or undirected motion, for example,

”ان السلطات ***لا تزال*** ترجح احتمال اصطدام الغواصة بسفينة أخرى“.

‘Authorities are ***still*** outweigh the possibility of submarine collisions another ship’.

***Asset:*** the sum of money used in a transaction (in the broad sense). It is used for verbs of obtaining and getting as well as verbs of creation, for example,

”***حصلت*** اليونان على مساعدة عدد من الدول الأجنبية“ .

‘Greece ***got*** to help a number of foreign countries’.

***Extent:*** the degree of change of an entity measured in percent or in other values, for example,

”***تقدم*** المنتخب السعودي بسرعة في بداية المباراة“.

‘Saudi team ***progress*** quickly in the beginning of the game’.

***Stimulus:*** used for external entities or events that attract the attention of the experiencer and elicit a response of some kind, for example

”***سيُشاهد*** فريق النصر بثوب جديد مختلف عن المواسم الماضية“.

‘Victory team ***will be seen*** with new custome that different with the past seasons’.

***Attribute*:** the property of an entity–typically a patient– such as the price, the temperature or the mass that undergoes a change as a result of the action described by the verb. It is found with transitive and intransitive verbs of calibratable change of state and a variety of verbs of change of state. Verbs of these classes set restrictions on the nature of the attribute. Thus, a verb like ‘double’ imposes the restriction [+/-SCALAR] to its attribute, for example,

”الا أننا نأمل أن ***يزيد*** هذا الرقم سريعا“.

‘But we hope that this figure ***increases*** rapidly’.

***Proposition:*** the abstract or concrete object of conception and knowledge verbs, for example,

**"لا توجد بيننا دولة تستطيع أن *تقف* بمفردها في وجه التجمعات الاقتصادية الكبرى".**

‘No country can ***stand*** alone in the face of major economic groupings’.

***Topic:*** describes the topic or the message in communication verbs and verbs of information transfer like ‘explain’ and ‘say’. For example,

***و أضاف*** "ان المشكلة الوحيدة هي أن بعض البلدان....” .

And ***added*** "The only problem is that some countries ....”

***Time:*** a class specific role describing the time, for example,

”لكن اعادة الهيكلية ***ستستمر***“.

‘But structural ***re continue’***.

**Material:** the raw material as the starting point of change of state verbs. This role is assigned by verbs of transformation and creation, for example,

”بانها ***تنتج*** 37 مليون برميل في اليوم“ .

‘It ***produces*** 37 million barrels per day’.

***Product:*** the end state or the product of change of state verbs assigned mainly by verbs of transformation and creation, for example,

”***ستشكل*** اختبارا للنوايا السليمة“ .

‘***Will be a test*** of the intentions of sound’.

In our dataset we pass 108 unique verbs that are not found in Arabic VerbNet. such as ‘**اعتزم**’, ‘**اعتبر**’, ‘**اعتاد**’ ‘accustomed’, ‘considered’, ‘intended’. Appendix A contains a list of these verbs.

## 4.4 Featuring Data

To add the semantic features that have been extracted from AVN to our training data, we use Java codes under eclipse IDE. Data is read from .CoNLL record by record as an input. When we find VERB record, the LEMMA is picked. We make an iteration on .xml documents of AVN to fetch the value for this LEMMA in these document members. At the time the LEMMA value of the verb is found in the .xml documents or classes, we directly extract all the semantic features belong this LEMMA from the xml documents and copy them to the training data. The features are class or document identification (CId), and thematic roles and their restrictions. These features are invoked as MemberThemeRoles object at the same time they are founded in xml AVN documents and are added to FEATS column to extend the existing morpho-syntactic features, Appendix E.

The same LEMMA of the verb could be found in many documents, as mentioned in Section 4.3, so we could find many semantic features group or options

After that, we manually choose the suitable features option that agree with the context. The UML diagram in Figure 3.5 in Section 3.1.4 illustrates the steps are needed to accomplish this task. The VerbReader class has 4 methods. The *Verbreader()*, the *processXML()*, which takes a LEMMA as input and returns the features in a list as an output. the *removeDiacritic()* method removes the short diacritics from the LEMMA to compare it later if this LEMMA is found in xml documents of AVN, and the *scanVerb()* method which reads the training data, it scans the data record by record when it reads VERB record, it passes its LEMMA to processXML() method after removing the diacritics, and it invokes the MemberThemRoles object to draw the feature group string, if the verb is found in AVN documents.

The MemberThemeRole class has the semantic features group. It is invoked when LEMMA is found in xml AVN documents to list these features as an object, which is represented as a string of semantic features in the training data file.

## 4.5 Experiments Setting

In Figure 4.2, data has ten columns. The third is the LEMMA, the fourth is the coarse-grained POS tag, while the sixth is FEATS column. In FEATS column, there are existing morpho-syntactic features. We extend these features with the semantic features, which are class identification and the thematic roles of the verb. Again the verb in its LEMMA form, according to the context.

After preparing data (Section 4.2 and 4.4), the data is in CoNLL-U format so we use the MaltParser input and output format commands to convert to CoNLL-X format, which is the input and output format of the MaltParser. We remove the comments lines in data, which start at ‘#’ sign.

Training data comprises over 10,100 tokens as mentioned in section 4.1. In the training data all columns exist, while in testing data the output column for head and dependency relation are missing. Simply, Maltparser should fill these columns in parse task (mode) of the parsing model.

We use Maltparser commands to create a configuration of the parsing model based on the **convert**, **learn** and **parse** modes or tasks respectively, we use the Stack algorithm, stacklazy, and according to data structure used by the algorithm and the features pool allowed, we adapt the feature model xml file for our parsing model.

## 4.6 Parsing Model Tasks

As discussed and illustrated in Chapter 3, the parsing model performs three tasks *convert*, *learn*, and *parse*. The commands used to perform these tasks are summarized in Appendix C. Next we describe these three tasks.

### 4.6.1 Convert data from CoNLL-U to CoNLL-X

We create a parsing model named train.mco from the training data. The parsing model gets its name from the configuration name. The configuration name is a name of our own choice. The purpose of the configuration is to gather information about all settings and files into one file. During learning, the configuration is created and stored in a configuration file with the file suffix .mco. This configuration file can later be reused whenever the trained model is used to parse new data. MaltParser input data is CoNLL-X format, we use a data convert input command format to convert the prepared data from CoNLL-U data format to CoNLL-X data format, see Appendix C

### 4.6.2 Training (Learning) Mode

Learn is the processing mode train (as opposed to parse). We use a default learning method (*LIBSVM*). Transition system is Arc-Eager, the parser informs us about the learning time. MaltParser 1.9.0 only knows the Single Malt configuration (singlemalt). Sometimes it is useful to get information about a configuration, for instance, to know which settings have been used when creating the configuration and so change the parameters of the parsing model in which lead to best results. Appendix C.

### 4.6.3 Parsing (Testing) Mode

Now, we have created a parsing model that we can use for parsing new sentences from the Arabic language. Unparsed sentence are formatted according to the format that was used during training (except that the output columns for head and dependency relation are missing). In our case, the first eight columns of the CoNLL-X data format represent tokens. The same name of the configuration file is used. The parser should be executed in the parsing mode to get the final parsed data Figure 3.6 for evaluating the accuracy matrix.

## 4.7 Parsing Algorithm

In our experiments, we use the Stack algorithm, namely stacklazy. The Stack algorithms are described in (Nivre, 2008). The Stack algorithms are similar to Nivre algorithm (the default) in that it uses a stack and a buffer but differ in that they add arcs between the two top nodes on the stack and that they guarantee that the output is a tree without post-processing. The Lazy (stacklazy) Stack algorithm in addition make use of a swap transition, which makes it possible to derive arbitrary non-projective dependency trees. The stacklazy algorithm postpones swapping as long as possible.

The Stack algorithms use three data structures: *a stack*, stack of partially processed tokens, *a list input*, which is a prefix of the buffer containing all nodes that have been on stack, *a list lookahead*, which is a suffix of the buffer containing all nodes that have not been on Stack. This is useful information when we adapt the feature model used in (Nivre, et al., 2007) as describe next.

## 4.8 Feature Model

The feature model specification must be specified in an XML file according to the format in Appendix D. We tested the performance by adapting the CoNLL shared task 2007 (Nivre et al., 2007) SVM settings used by the same parser for various languages (Hall et al., 2007).

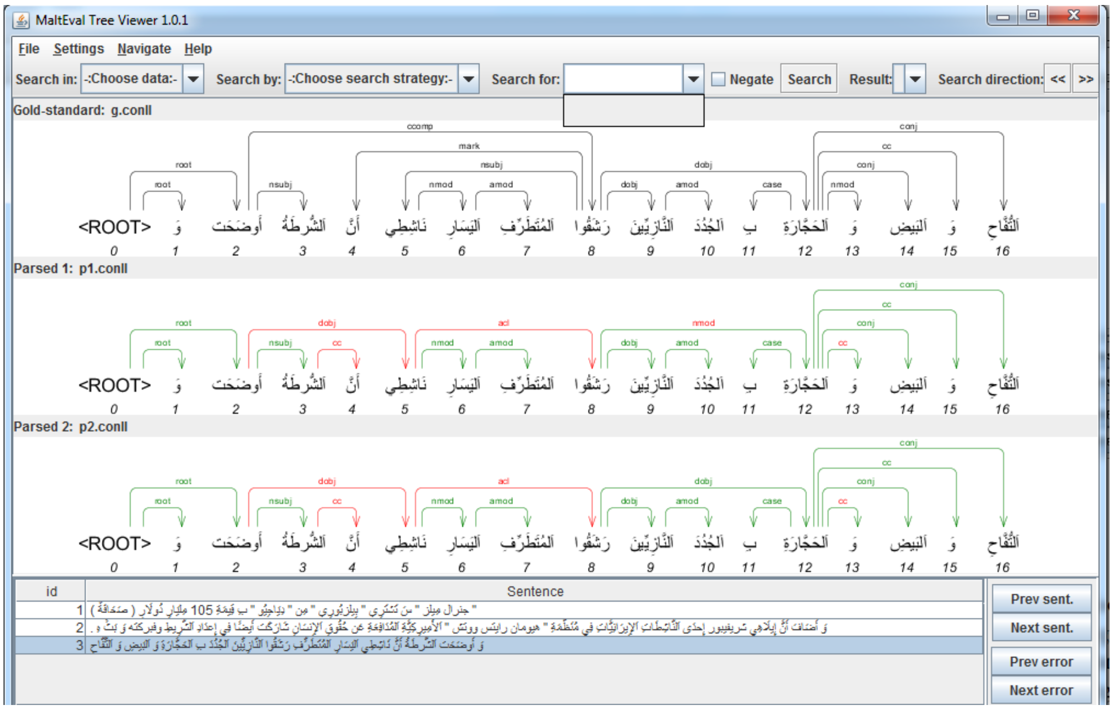
## 4.9 Computing and Viewing the Scores of the Results

To this end, we compute the scores of the results using MaltEval tool, an evaluation and visualization tool for dependency parsing (Nilsson & Nivre, 2007). MaltEval is able to read a number of Treebank formats. In our experiments the Treebank is in CoNLL format (Appendix B). There are two required flags that specify the gold-standard file(s) and the file(s) that we want to compute:

-g <gold-standard file, files or directory (tab|xml|conll)>

-s <file, files or directory with parser output (tab|xml|conll)>

The commands used to perform these metrics are shown in Appendix C


MaltEval Tree Viewer 1.0.1
File Settings Navigate Help
Search in: -:Choose data:-
Search by: -:Choose search strategy:-
Search for:
Negate
Search
Result:
Search direction: << >>
Prev sent.
Next sent.
Prev error
Next error

Gold-standard: g.conll

Parsed 1: p1.conll

Parsed 2: p2.conll

<ROOT> وَ أوضَحَت اَلشُّرطَةُ أَنَّ نَاشِطِي اَليَسَارِ اَلمُتَطَرِّفِ رَشَقُوا اَلنَّازِيِّينَ اَلجُدُدَ بِ اَلحِجَارَةِ وَ اَلبَيضِ وَ اَلتُّفَّاحِ
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
root
nsubj
ccomp
mark
nmod
amod
dobj
case
conj
cc
acl

| id | Sentence |
|---|---|
| 1 | " جنرال مِيلز " مِن تَشتَرِي " بِيلزبُورِي " مِن " دِيَاجِيُو " بِ قِيمَةِ 105 مِليَار دُولَار ( صَفقَة ) |
| 2 | وَ أَضَافَ أَنَّ إِيلَاهِي شَرِيفِيُور إِحدَى النَّاشِطَاتِ الإِيرَانِيَّاتِ فِي مُنَظَّمَةِ " هيومان رايتس ووتش " الأَمِيرِكِيَّةِ المُدَافِعَةِ عَن حُقُوقِ الإِنسَانِ شَارَكَت أَيضًا فِي إِعدَادِ الشَّرِيطِ وفبركته وَ بَثّ و . |
| 3 | وَ أَوضَحَت الشُّرطَةُ أَنَّ نَاشِطِي اليَسَارِ المُتَطَرِّفِ رَشَقُوا النَّازِيِّينَ الجُدُدَ بِ الحِجَارَةِ وَ البَيضِ وَ التُّفَّاحِ |

Figure (0.3): Visualizing many parsed dependency files, and highlighting errors

MaltEval has a module for viewing the content of the gold-standard and parsed files visually. The visual mode is enabled by setting the flag -v to 1 (default is 0). We perform this command for our sentences. Figure 4.3 shows an example for a sentence. The figure illustrates how a window containing a visual representation of three files (gold.conll, parsed1.conll, parsed2.conll) could look like. Gold file is the test data interred to trained model (Section 3.2.5). Parsed1 file is the final passed data results from the second experiment where the LEMMA value is exist and no semantic features added yet, as shown in Section 4.10. Parsed2 file is the final passed data results from the third experiment where LEMMA and semantic features are exist. Green and red arcs and labels in the dependency trees of the two parsed files indicate whether the arcs and labels were correct or incorrect compared to the gold-standard, depending on arcs and labels of the sentence in the Figure 4.3 and refer to equation 2.3 in Section 2.7.2 the LAS computation for Parsed1 and Parsed2 are 75% and 78% respectively, which means increasing in accuracy. In addition, the bottom of the window shows a scroll list containing the sentences. By selecting another sentence, all sub windows above are updated so that the dependency trees for all files of that sentence are shown in Figure 4.3.

## 4.10 Experiments Results

We conduct three experiments on the proposed dependency parsing approach that reflect the required aspects of the approach which are baseline (on the data before being prepared), after lemmatization, and after adding LEMMAS and semantic features. Next we presents these three experiments.

***Experiment 1:*** Experiment 1 uses a baseline of our model. All attributes (columns) is presented in the data, but the LEMMA value is underscore (before lemmatization). This results in a Labeled Attachment Score (LAS) of 76.2% and an Unlabeled Attachment Score (UAS) of 82.1%.

***Experiment 2:*** Experiment 2 uses all attributes in the data, but the LEMMA value is filled by using MADAMIRA. This results in a (LAS) of 72.2% and an (UAS) of 78.8%.

Table (0.2): LAS and UAS of experiments 1-3

| Scores / Exp. # | LA | UAS | LAS |
|---|---|---|---|
| Experiment 1 | 85.4 | 82.1 | 76.2 |
| Experiment 2 | 85.4 | 78.8 | 72.2 |
| Experiment 3 | 87.4 | 77.5 | 71.5 |

***Experiment 3:*** Experiment 3 uses all attributes in the data used in experiment 2. In addition, FEATS attribute is extended to complement morph syntax feats by adding sematic features, which we have extracted from AVN. This results in a (LAS) of 71.5% and an (UAS) of 77.5%. The results are summarized in Table (4.2).

## 4.11 Summary

To sum up, this chapter presented the implementation of the dependency parsing for Arabic verbal sentences. We show the exact semantic features used and the description of the code used to extract and add these features. We presented the main method used for processing data, and choosing the MaltParser parameters, features model, and algorithm to obtain the best results. We presented MaltEval tree viewer, which provided visual view to the content of the parsed and unparsed statement by the command tree, and provide an empirical example of computing the score. Finally we introduced the experiments' and the UAS, LAS. LA scores' values. In next chapter we present the discussion and analyze the experiments' results.

# Chapter 5

# Results and Discussion

# Chapter 5
# Results and Discussion

We evaluate the performance of our experiments using a 12.5k-token manually annotated and validated sample of data (370 sentences) derived from the Arabic dependency Treebank. We divide the data into 11k and 1.5k for training and testing respectively. We conduct three experiments over ten columns trained data, the baseline where the LEMMA column has no value (underscore) the second experiment with LEMMA values, and the third, which is our contribution work, the LEMMA column and semantic features exist.

We conducted the parsing mode of the experiments individually for each the experiment over the same configuration file and the same features model file. We are able to achieve an accuracy of 77.5% LAS and 71.5% UAS, with 2% increasing in LA (LabelRight), token is counted as a hit if the dependency label is the same as in the gold-standard data.

The results improved a lot, but in some cases the features are helpful and they improve the results on some dependency relations, but the results are worse for other dependencies, and the overall effect could be a decrease in performance as shown in UAS. The results satisfy the dependency parsing of Arabic verbal sentences improvement we intend as we expected.

We can explain the results in Table 4.2 due to the impact of several factors including:

## 5.1 Impact of Arabic VerbNet

Good results visualize when the verbs in dataset have semantic features in Arabic VerbNet. We pass 108 unique verbs not found in .XML file Arabic VerbNet (Appendix1). So there is no semantic features added to these verb.

Also some verbs such as '**ألمح**' 'hinted' the Experience is animate in its thematic roles, but in some situations in our dataset we have example such as

'ألمحت البحرية الروسية أن الغواصة اصطدمت بغواصة أخرى'

'The Russian navy has hinted that the submarine collided with another submarine water flooded a number of rooms', the agent is 'The Russian navy'. The parser should learn more for this new example. In the other hand the verb ‘غمر’ ‘flooded’ the agent is animate, but we have different role in our dataset such as ‘المياه غمرت عددا من الغرف’ ‘Water flooded a number of rooms’**.** Another example the verbs ‘صعق، شهد’ ‘dumbfounded’ ‘witnessed’, where the experience is animate, but this is not as shown in the sentences

‘صعق التيار الكهربائي رجلا’,’ شهد الموقع أخطر حادث نووي ’

‘Power dumbfounded a man’ ‘The site witnessed the most serious nuclear accident’. This limit the accuracy at the time the parser has actually animate or even human agent in the test data.

As demonstrated in Section 4.2 even we have the same repeated LEMMA of the verbs, but it is not necessary reduce the sparsity of the data, for example the (نَشَرَ) in the sentence

‘برقية التهنئة التي نَشَرَتها وكالة الانباء الفلسطنية’

‘A congratulatory telegram published by the Palestinian news agency’ in this example it classify in the Oa$aAEa-1 class. While we have four classes for the same LEMMA ‘Ead~ada-1, Oa$aAEa-1, qaTaEa-1,and TawaY-1’ or ‘عدّدَ، أشَاعَ، قَطَعَ، طوي’, where the verb ‘نَشَرَ’ ‘published’ in their members, that means the parser may learn 4 times for the same verb .the context is the decision to solve this trick or point. Due to the context our choice is ‘Oa$aAEa-1’ and therefore the extended semantic features are:

|CId=Oa$aAEa 1|thm=Agent|thm=Theme|Location=location as they appear in the modified data. However, this group could not be the same for the same LEMMA along the data.

## 5.2 Impact of the Dataset

We get better results by experiments in many cases such as the train data topic alike the test data where most of verbs repeated in both.

Some verb appear n passive form, so the *Agent* thematic roles remain in VERB record in FEATS column because the subject does not appear in the data.

In our dataset, many verbs are repeated several times. For example the verb ‘قال’ ‘said’ is repeated 21 times, the verb ‘أضاف’ ’added’ 19, the verb ‘أعلن’ ’announced’

31, the verb'أثار' 'motivated' 7, while the verb 'كان' 'been' is repeated 62 times, and so on. This is a good remark demonstrates good results of the parser's classifier, by decreasing the sparsity. We note also the stories at the same topic during the dataset have the similar verbs, in other words most verbs in sport stories for instance will be repeated in the train and test data in the sport stories and so their thematic roles, such classifying in this manner is helpful for the classifier.

It is worth mentioning that there are verbs having the opposite meaning for example 'خسر، فاز أو حصل' 'won, got, or lost' in the sentences '

'حصلت السعودية على اول نقطتين لها في البطولة /خسرت مباراتها الأولى امام الجزائر

'Saudi Arabia got the first two points in the championship / lost the first match against Algeria' but have the same semantic feature group, which is

|CId=faAza-1|Agent=animate|Agent=organization|thm=Theme|Source=concrete .

## 5.3 Impact of Human Sense Disambiguation

Another point which should be discussed in this situation that we use our semantic knowledge to accomplish the choice of the semantic features extracted from the Arabic VerbNet and choose the features group agree with a context. For example, the verb 'تقف' 'stand' in the sentence

'لا توجد بيننا دولة تستطيع أن تَقِف بمفردها في وجه التجمعات الاقتصادية الكبرى'

'No country can stand alone in the face of major economic groupings' in training data have many semantic features groups, the semantic feature group choice is

|CId=kar~asa-1|Agent=animate|thm=Theme|thm=Proposition.

Manually we remove the semantic feature groups which do not agree with the context and keep the suitable group. We assume just one features group for the each verb. Note that the dataset are small and that we need more training data to create a useful parsing model.

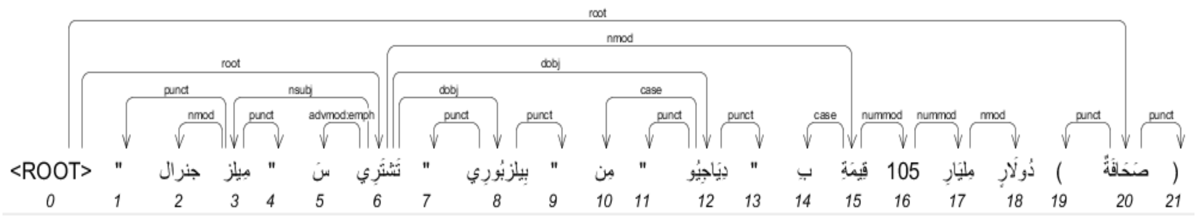
## 5.4 Parser

A good adapting for feature model xml file and parsers' parameters such as algorithms leads to better results. Also while the parser learnt more the accuracy increases. Notice that we use a deterministic data driven, history based dependency parser. The parser is guided by word knowledge, with probabilistic grammar and semantic rules, even we conduct our experiments depending on choosing the parameters gets the best results, but it is good to apply our architecture to other model to compare if our choosing model is efficient.

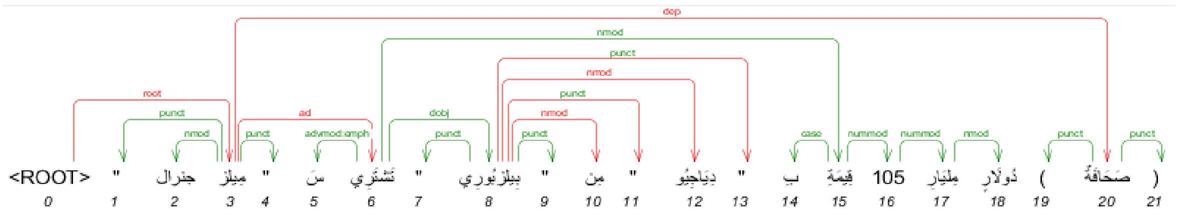
The dependency parsing tree example in Figure 5.1 illustrates a visual representation tree of the sentence

‘جنرال مِيلز" ستشتري "بيلزبُوري" من "دِيَاجِيُو" بقيمة 105 مليار دولار (صحافة)‘

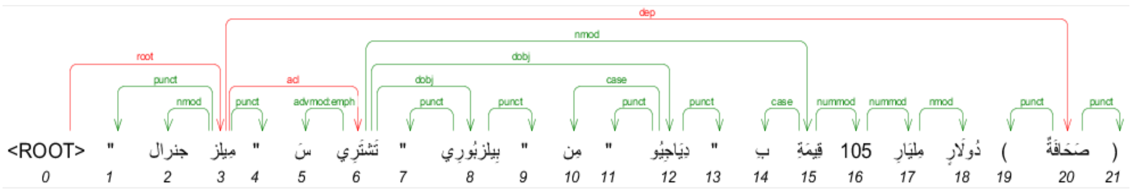
‘General Mills "will buy" Pillsbury "from" Diageo "worth $ 105 billion (Press)’



(a)



(b)



(c)

Figure (5.1): Practical example of the sentence ‘ *جنرال مِيلز" ستشتري "بيلزبُوري" من "دِيَاجِيُو" بقيمة 105 مليار دولار (صحافة)*’ (a) gold standared (b) parsed before semantic features addition (c) parsed after semantic features addition

The figure display the visual representation tree of the sentence picked from the (gold.conll, parsed1.conll, parsed2.conll) files respectively, where gold file is the original test data and the parsed1 file is the final parsed data result after adding only the lemmatization value, as presented in Section 4.9, and parsed2 file is the final parsed data after adding lemmatization and semantic features (Section 4.9). Green and red arcs and labels in the dependency trees of the two parsed files indicate whether the arcs and labels were correct or incorrect compared to the gold-standard (Section 2.7.2). for the LEMMA 'اشترى' 'bought' The example in (c) demonstrate the efficient of our model to reduce the dependencies labels in more directly than the parsed example in Figure 5.1b and determine the object, in brief the green arc and label object to the word (دِيَاجِيُو) for the LEMMA (اشترى).

This is the essential goal for our project to build a parsing model, which is more efficient, direct, and active to determine the verbal Arabic sentence dependencies such as subjects, and objects depending on the semantic features of the verb regardless the order structure of the sentence and grammatical restrictions and rules.

## 5.5 Summary

In summary this chapter present a discussion in view of the results. We introduced the factors could play a role in improving the results' scores. These factors could be due the features, dataset size, or even parsing model, where we use Maltparser. We displayed imperial example satisfy a goal. Next chapter conclusions and recommendations

# Chapter 6

# Conclusions and Recommendations

# Chapter 6
# Conclusions and Recommendations

Parsing morphologically rich, free word order languages (MoR-FWO) such as Arabic is a challenging task. In this research we presented a parsing approach incorporating semantic features and linguistic constraints from Arabic VerbNet to dependency parsing for Arabic. We performed a series of experiments exploring the effect of different semantic features in Arabic dependency parsing using a data-driven parser, namely MaltParser. With just 370 sentences, we were able to build a dependency parser with stat-of-the-art accuracy of 71.5% Labelled Attachment Score (LAS) and 77.5% Unlabeled Attachment Score (UAS).

We did a step by step analysis of the importance of different features to increase the performance like, feature model, Maltparser working behavior, and useful semantic features for sentence level parsing of Arabic. We were able to achieve an accuracy of 87.4% Labelled score. With more training data and more experiments, we hope to achieve even better accuracies.

During analysis, we found that labeled accuracy when adding semantic features are high compared to without these features by 2%. This shows the importance of semantic features as indicated in (Bharati, et al., 2008a) (Bharati, et al., 2009b). We plan to see their effect in complete sentence parsing using different parser model and different dataset format if needed also.

Incorporating semantic features that extracted from Arabic VerbNet to improving verbal Arabic sentence dependency parsing is the contribution of this thesis and is an initial work in this area. We are planning to improve our parsing approach by applying other lexical ontology-based and other data format. Other semantic features could add to modified parser as hierarchy relations between verbs and their dependencies in lexical ontologies. In addition, data format differ from system to system. We are planning to improve our parsing approach for such systems with more accuracy. We would also like to evaluate our model on the time taken for validation.

Given the basic nature of the semantic classes and thematic roles, we think there is room for incorporating new kinds of semantic information, such as Arabic WordNet base concepts, Wikipedia concepts, or similarity measures. Word sense disambiguation WSD algorithms also can appear to implement the suitable features group to fit the meaning due to the context (Agirre, Bengoetxea, Gojenola, & Nivre, 2011).

Finally, in this thesis, we presented our work on verbal Arabic sentence level dependency parsing and the effects of incorporating semantic features to enhance parsing.

As a final comment, we can say that semantic features are the basis is the basic for determining the effectiveness of the parsing process. Based on this we can determine the syntactical structures and then determine the short and long vowel or diacritics drawn, but not vice versa. When the morpho-syntactic used in parsing it is actually drawn due to the parsing process. If task is parsing in itself, there is no need to use morpho-syntactic. Other issues arise when we perform parsing tasks such as free order words. This issue is commonly clear in wirenews and big data behind the social media, which may have no restriction for the short and long vowel diacritic, and linguistic grammar. We are able to achieve an increasing in accuracy 2% Labelled score, eventhogh we conduct our experiments over general data and 12.5 k tokens.

## The Reference List

Agirre, E., Baldwin, T., & Martinez, D. (2008). *Improving Parsing and PP attachment Performance with Sense Information*. In Proc. of ACL-08: HLT, (pp. 317–325). Columbus, Ohio.

Agirre, E., Bengoetxea, K., Gojenola, K., & Nivre, J. (2011). *Improving Dependency Parsing with Semantic Classes*. in Proc.of the 49th Annual Meeting of the Association for Computational Linguistics, (pp. 699–703).

Al Daoud, E., & Basata, A. (2009). *A Framework to Automate the Parsing of Arabic Language Sentences*. the Int. Arab Journal of Information Technology.

Alqrainy , S. K., Muaidi, H., & Alkoffash , M. (2012). *Context-Free Grammar Analysis for Arabic Sentences*. International Journal of Computer Applications (0975 - 8887).

Ambati, B. R. (2011). *Hindi Dependency Parsing and Treebank Validation*. M.S. thesis, Dept. Comput. Sci & Eng. Hyderabad, India: International Institute of Information Technology.

Ambati, B. R., Gade, P., GSK, C., & Husain, S. (2009). *Effect of Minimal Semantics on Dependency Parsing*. Student Research Workshop, RANLP 2009, (pp. 1–5). Borovets, Bulgaria.

Ambati, B. R., Jain, S., & Sangal, R. (2008). *Two semantic features make all the differencein parsing accuracy*. In Proc. of ICON.

Annodoc , & brat. (2014). CoNLL-U Format. Retrieved from Universal Dependencies: http://universaldependencies.org/format.html

Attia, M. (2006). *An Ambiguity-Controlled Morphological Analyzer for Modern Standard Arabic Modelling Finite State Networks*. The Challenge of Arabic for NLP/MT. London: The British Computer Society.

Ballesteros, M., & Nivre, J. (2012). *MaltOptimizer: A System for MaltParser Optimization*.

Bengoetxea, K., Agirre, E., Nivre, J., Zhang, Y., & Gojenola, K. (2014). *On WordNet Semantic Classes and Dependency Parsing*. In Proc. of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers), (pp. 649–655).

Bharati, A., Chaitanya, V., & Sangal, R. (1995). Natural language processing. A paninian perspective. Prentice-Hall of India, (pp. 65-106) .

Bharati, A., Husain, S., Ambati, B., Jain, S., Sharma, D. M., & Sangal, R. (2008a). *Two semantic features make all the difference in parsing accuracy*. In Proceedings of the 6th International Conference On Natural Language Processing (ICON) (pp. 11–19). Pune, India: Macmillan Publishers India Ltd.

Bharati, A., Sharma, D. M., Husain, S., Bai, L., Begum, R., & Sangal, R. (2009b). *Anncorra: Treebanks for Indian Languages, Guidelines for Annotating Hindi Treebank (version 2.0).* http://ltrc.iiit.ac.in/MachineTrans/research/tb/DS-guidelines/DS-guidelinesver2-28-05-09.pdf.

Bhattacharya, P. (2010, June 5). Centre For Distance Engineering Education Programme. Retrieved from Natural Language Processing: http://www.cdeep.iitb.ac.in/previous_courses.php

Bikel, D. (2002). *Design of a Multi-lingual, Parallel-processing Statistical Parsing Engine*. Int. Conf. on Human Language Technology Research (HLT) (pp. 24–27). DOI: 10.3115/1289189.1289191.

Bikel, D. M. (2004). *Intricacies of Collins' parsing Model.* Computational Linguistics. ACM Computational Linguistics, 30(4), (pp. 479-511).

Buchholz, S., & Marsi, E. (2006). *CoNLL-X shared task on Multilingual Dependency Parsing*. Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL-X) (pp. 149–164). New York: Association for Computational Linguistics.

Cahill, A., Burke, M., O'Donovan, R., Genabith, J. v., & Way, A. (2004). *Long-Distance Dependency Resolution in Automatically Acquired Wide-Coverage PCFG-Based LFG Approximations*. ACL '04 Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics , (pp. 319-326).

Chanev, A., Nivre, J., Hall, J., Nilsson, J., Eryigit, G., Kubler, S., . . . Marsi, E. (2007b). *Maltparser: A Language-independent System for Data-driven Dependency Parsing*. Natural Language Engineering, 13(2), (pp. 95–135).

Chang, C.-C., & Lin, C.-J. (2001). *Libsvm: A library for Suppor Vector Machines*. Retrieved from http://www.csie.ntu.edu.tw/cjlin/libsvm

Charniak, E. (1997). *Statistical Parsing with a Context-free Grammar and Word Statistics*. In Proc. of the 15th Annual Conf. on Artificial Intelligence (AAAI-97), (pp. 598–603). Stanford, USA.

Charniak, E. (2000). *A maximum Entropy-based Parser*. In Proc. of the 1st Annual Meeting of the North American Chapter of Association for Computational Linguistics (NAACL2000). Seattle, USA.

Christopher D. Manning, & Green, S. (n.d.) (2010). *Better Arabic Parsing: Baselines, Evaluations, and Analysis.* In Proc. of the 23rd International Conference on Computational Linguistics. (pp. 394-402)

Collins, M. (2003). *Head-driven Statistical Models for Natural Language Parsing*. Computational Linguistics, 29(4), (pp.589–637).

Collins, M. J. (1996). *A new Statistical Parser Based on Lexical Dependencies*. In Proc. of the 34th Annual Meeting of the ACL, (pp. 184–91). USA.

Covington, M. A. (2001). *A fundamental Algorithm for Dependency Parsing*. In Proceedings of the 39th Annual ACM Southeast Conference, (pp. 95–102).

Daimi, K. (2001). *Identifying Syntactic Ambiguities in Single-Parse Arabic Sentence*. Computers and the Humanities3, (pp. 333–349).

Daoud, D. (2009). *Synchronized Morphological and Syntactic Disambiguation for Arabic*. research in Computing Science 41, © A. Gelbukh (Ed.), (pp. 73-86).

Diab, M., Habash, N., Rambow, O., & Roth, R. (2013). *LDC Arabic Treebanks and Associated Corpora: Data Divisions Manual*. Computational Learning Systems - CCLS Columbia University.

Fujita, S., Bond, F., Oepen, S., & Tanaka, T. (2007). *Exploiting Semantic Information for HPSG Parse Selection*. In ACL Workshop on Deep Linguistic Processing, Prague, Czech Republic, June. Association for Computational Linguistics, (pp. 25–32).

Gabbard , R., & Kulick, S. (2008). *Construct State Modification in the Arabic Treebank*. Association for Computational Linguistics (ACL -08): HLT (pp. 209–212). Columbus,: DOI: 10.3115/1557690.1557750.

Genabith, J. v., Tounsi, L., & Attia, M. (2009). *Parsing Arabic Using Treebank-based LFG Resources*. Proceedings of the LFG09 Conference. Dublin: CSLI Publications.

Green, S., Sathi, C., & Manning, C. D. (2009). *NP Subject Detection in Verb-Initial Arabic Clauses*.

Habash , N., & Roth, R. (2009). *CATiB: The Columbia Arabic Treebank. The Association for Computational Linguistics* (ACL-IJCNLP) 2009 Conf. Short Papers (pp. 221–224). Suntec, Singapore: DOI: 10.3115/1667583.1667651.

Habash, N. (2010). *Introduction to Arabic Natural Language Processing*. Morgan & Claypool Publishers.

Hajiˇc, J., Hladká, B., & Pajas, P. (2001). *The Prague Dependency Treebank: Annotation Structure and Support*. In Proceedings of the IRCS Workshop on Linguistic Databases (pp. 105–114). Philadelphia: University of Pennsylvania.

Hall, J. (2006). *MaltParser-An Architecture for Inductive Labeled Dependency Parsing*. M.S. thesis, Dept. Comput. Sci., Växjö Univ., Sweden.

Hall, J., Nilsson, J., & Nivre, J. (2014, November 22). User guide. Retrieved from MaltParser: http://www.maltparser.org/userguide.html

Hall, J., Nilsson, J., Nivre, J., Eryigit, G., Megyesi, B., Nilsson, M., & Saers, M. (2007). *Single malt or Blended, A study in Multilingual Parser Optimization*. Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007 (pp. 933–939). Prague: Czech Republic.

Hudson, R. (1984). Word grammar. Basil Blackwell, 108 Cowley Rd, Oxford,.

Jain, S., Jain, N., Tammewar, A., Bhat, R. A., & Sharma, D. M. (2013). *Exploring Semantic Information in Hindi WordNet for Hindi Dependency Parsing*. In Int. Joint Conf. on Natural Language Processing, (pp. 189–197).

Jurafsky, D., & Manning, C. (2012, April 28). *Natural Language Processing, Lecture Slides from the Stanford Coursera course* . Retrieved from Dependency Parsing: https://web.stanford.edu/~jurafsky/NLPCourseraSlides.html

Kingsbury, P., Palmer, M., & Marcus, M. (2002). *Adding Semantic Annotation to the Penn TreeBank*. In Proc. of the Human Language Technology Conf., (pp. 252–256).

Kulick, S., Gabbard, R., & Marcus, M. (2006). *Parsing the Arabic Treebank: Analysis and Improvements*. Proceedings of the TLT (pp. 31-42). Prague, Czech Republic: Institute of Formal and Applied Linguistic.

Levin, B. (1993). *English Verb Classes and Alternations*. The University of Chicago Press, Chicago and London.

Maamouri, M., Bies, A., Buckwalter, T., & Mekki, W. (2004). *The Penn Arabic Treebank: Building a Large-Scale Annotated Arabic Corpus. In Proc. of the* NEMLAR Conference on Arabic Language Resources and Tools, (pp. 102–109). Cairo.

MacKinlay, A., Dridan, R., McCarthy, D., & Baldwin, T. (2012). *The Effects of Semantic Annotations on Precision Parse Ranking*. First Joint Conference on Lexical and Computational Semantics (*SEM) (pp. 228–236). Montreal: Association for Computational Linguistics.

MacKinlay, A., Dridan, R., McCarthy, D., & Baldwin, T. (2012). *The Effects of Semantic Annotations on Precision Parse Ranking*. in Proceedings of the First Joint Conf. on Lexical and Computational Semantics Volume 1: Proceedings of the main conference and the shared task, and Volume 2:Proceedings of the Sixth International Workshop on Semantic Evaluation, (pp. 228–236).

Magerman, D. M. (1995). *Statistical Decision-tree Models for Parsing*. In Proc. of the 33rd Annual Meeting of the ACL, (pp. 276–83). USA.

Marton, Y., Habash, N., & Rambow, O. (June 2010). *Improving Arabic Dependency Parsing with Lexical and Inflectional Morphological Features*. Proc. of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages, (pp. 13–21). Los Angeles, California.

Marton, Y., Habash, N., Rambow, O., & Alkuhlani, S. (2013). *SPMRL'13 Shared Task System: The CADIM Arabic Dependency Parser*. in Proc. of the Fourth Workshop on Statistical Parsing of Morphologically Rich Languages, (pp. 86–90).

Mel'cuk, I. A. (1988). *Dependency syntax: Theory and practice*. State University, Press of New York.

Mousser, J. (2010, June 20). Jaouad Mousser. Retrieved from Arabic Verbet: http://ling.uni-konstanz.de/pages/home/mousser/

Nilsson, J., & Nivre, J. (2007). *MaltEval: An Evaluation and Visualization Tool for Dependency Parsing*.

Nivre, J. (2003). *An Efficient Algorithm for Projective Dependency Parsing.* Proceedings of the 8th International Workshop on Parsing Technologies (IWPT), (pp. 149–160).

Nivre, J. (2008). *Algorithms for Deterministic Incremental Dependency Parsing*. Computational Linguistics, 34(4).(pp. 513–553).

Nivre, J., & Nilsson, J. (2005). *Pseudo-projective Dependency Parsing*. In ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, (pp. 99–106). Ann Arbor, Michigan.

Nivre, J., Hall, J., Kubler, S., McDonald, R., Nilsson, J., Riedel, S., & Yuret, D. (2007). The CoNLL 2007 *Shared Task on Dependency Parsing*. Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007 (pp. 915–932). Prague: 2007 Association for Computational Linguistics.

Nivre, J., Nilsson, J., & Hall, J. (2005). *Talbanken05: A Swedish Treebank with Phrase Structure and Dependency Annotation*.

Othman, E., Shaalan, K., & Rafea, A. (2006). *A Chart Parser for Analyzing Modern Standard Arabic Sentence*. in Proc. of the 10th Conf. on Computational Natural Language Learning (CoNLL-X), (pp. 149–164). New York.

Øvrelid, L., & Nivre, J. (2007). *When Word Order and Part-of-speech Tags are not Enough —Swedish Dependency Parsing with Rich Linguistic Features*. in Proc. of the Int. Conf. on Recent Advances in Natural Language Processing (RANLP), (pp. 447–451).

Pasha, A., Al-Badrashiny, M., Diab, M., Habash, N., Pooleery, M., Rambow, O., & Roth, R. (2015). MADAMIRA v2.0 User Manual. Center for Computational Learning Systems.

Pennsylvania, T. T. (2016, 9 21). Linguistic Data Consortium. Retrieved from Prague Arabic Dependency Treebank 1.0: https://catalog.ldc.upenn.edu/LDC2004T23

Prague, C. U. (2004). Prague Arabic Dependency Treebank. Retrieved from Prague Arabic Dependency Treebank 1.0: https://ufal.mff.cuni.cz/padt/PADT_1.0/docs/index.html

Schuler, K. K. (2005). *VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon*. Ph.D. thesis, University of Pennsylvania.

Sgall, P., Hajiˇcová, E., & Panevová, J. (1986). *The Meaning of the Sentence in Its Semantic and Pragmatic Aspects*. D. Reidel & Academia, 106.

Shatnawi, M., & Belkhouche, B. (2012). *Parse Trees of Arabic Sentences Using the Natural Language Toolkit*. in the Proc. of the Int. Arab Conf. on Information Technology.

Shieber, S. M. (1985). *Evidence Against the Context-freeness of Natural Language.* Linguistics and Philosophy, 8 (pp.334–343).

Tounsi, L., Attia, M., & Genabith, J. v. (2009). *Automatic Treebank-Based Acquisition of Arabic LFG Dependency Structures*. European Chapter of the Association for Computational, EACL, (pp 45-52).

Vossen, P. (1998). *EuroWordNet: A Multilingual Database with Lexical Semantic Networks*. Kluwer, Dordrecht, The Netherlands.

Vossen, P. (2004). *Eurowordnet: A multilingual Database of Autonomous and Language-specific Wordnets Connected Via an Inter-lingual-index*. International Journal of Lexicography, (pp. 161–173).

**المراجع العربية**

الأفغاني, س. (2003). *الموجز في قواعد اللغة العربية*. بيروت: دار الفكر للطباعة و النشر و التوزيع.

# Appendix A:  List of Verbs

Verbs not found in Arabic VerbNet documents members in LEMMA elements value (108 verbs)

| | | | | |
|---|---|---|---|---|
| أتى | استطرد | أقسم | تعادل | زار |
| أتاح | استعد | اكد | تعرض | سعى |
| اتسم | استغرق | التزم | تقرر | سيطر |
| اتفق | استفاد | التقى | تمخض | شدّد |
| اتهم | استقبل | الحق | تنكر | شنّ |
| اثر | استمسك | امتد | توجه | صادق |
| اجتاح | استهان | أمكن (يمكن) | تولى | صبا |
| أجرى | استوجب | انبغى | حدّد (حدد الرئيس في خطابه اليوم) | طالب |
| احتاج | اصطدم | انتخب | | عثر |
| احتجز | اضطر | انتزع | جدّد | فقد |
| احترم | اطاح | انتشر | جهد | قارب |
| احتفظ | اعتاد | انتمى | حضر | قرر |
| احتفل | اعتبر | انتهك | حق | لقي |
| احتل | اعتزم | انتهى | حقق | ليس |
| اختار | اعتقد | اندرج | حكم | مدّد (الوقت) |
| اختبر | اعتمد | انطلق | خاض | نفذ |
| ارتكب | اعتنق | انقطع | خسر | وجب |
| ازداد | اغتصب | انهار | خلف | وجه |
| استحق | أفرج | أوضح | درب | وطئ |
| استخدم | اقترح | تراس | دعا | ولى |
| استضاف | اقتلع | تسرب | رجج | يجتمع |
| استطاع | أقدم | تضمن | رعى | |

# Appendix B: CoNLL Format

The CoNLL data format adheres to the following rules:

• Data files contain sentences separated by a blank line.

• A sentence consists of one or tokens, each one starting on a new line.

• A token consists of ten fields described in the table below. Fields are separated by a single tab character. Space/blank characters are not allowed in within fields

• All data files will contain these ten fields, although only the ID, FORM, CPOSTAG, POSTAG, HEAD and DEPREL columns are guaranteed to contain non-dummy (i.e. non-underscore) values for all languages.

• Data files are UTF-8 encoded (Unicode). A more detailed description is found here: http://depparse.uvt.nl/depparse-wiki/DataFormat.

# Appendix C: Experiments' Commands

```
prompt> java -jar maltparser-1.9.0.jar -c train -m info
```

```
prompt> java -jar maltparser-1.9.0.jar  -c train  -i   AFP_ARB_20000715.0021.conllu -if conllx  -m convert   -o AFP_ARB_20000715.0021.conll  -of conllx
```

```
prompt> java -jar maltparser-1.9.0.jar   -I otraindata.conll   -c train -a stacklazy -F MyFeatureModel.xml   -grl root -gcs '~' -d POSTAG -T 1000 -gds T.TRANS,A.DEPREL -l libsvm -m learn
```

```
prompt> java -jar maltparser-1.9.0.jar -c train   -i  AFP_ARB_20000715.0035.conll   -o rersult.conll  -m parse
```

```
prompt> java -jar MaltEval.jar --Metric LAS;UAS;LA -s result.conll -g AFP_ARB_20000715.0035.conll
```

```
prompt> java -jar MaltEval.jar -v 1 -s parsed1.conll parsed2.conll -g gold.conll
```

# Appendix D:  Feature Model .XML File

```
InputColumn(FORM,Input[0])
InputColumn(FORM,Input[1])
InputColumn(FORM,Stack[0])
InputColumn(FORM,head(Stack[0]))
InputColumn(POSTAG,Input[0])
InputColumn(POSTAG,Input[1])
InputColumn(POSTAG,Input[2])
InputColumn(POSTAG,Input[3])
InputColumn(POSTAG,Stack[0])
InputColumn(POSTAG,Stack[1])
Merge(InputColumn(POSTAG,Input[0]),OutputColumn(DEPREL,ldep(Input[0])))
Merge(InputColumn(POSTAG,Stack[0]),InputColumn(POSTAG,Input[0]))
Merge(InputColumn(POSTAG,Stack[0]),OutputColumn(DEPREL,Stack[0]))
Merge3(InputColumn(POSTAG,Input[0]),InputColumn(POSTAG,Input[1]),InputColumn(POSTAG,Input[2]))
Merge3(InputColumn(POSTAG,Input[1]),InputColumn(POSTAG,Input[2]),InputColumn(POSTAG,Input[3]))
Merge3(InputColumn(POSTAG,Stack[0]),InputColumn(POSTAG,Input[0]),InputColumn(POSTAG,Input[1]))
Merge3(InputColumn(POSTAG,Stack[0]),OutputColumn(DEPREL,ldep(Stack[0])),OutputColumn(DEPREL,rdep(Stack[0])))
```

Merge3(InputColumn(POSTAG,Stack[1]),InputColumn(POSTAG,Stack[0]),InputColumn(POSTAG,Input[0]))

OutputColumn(DEPREL,Stack[0])

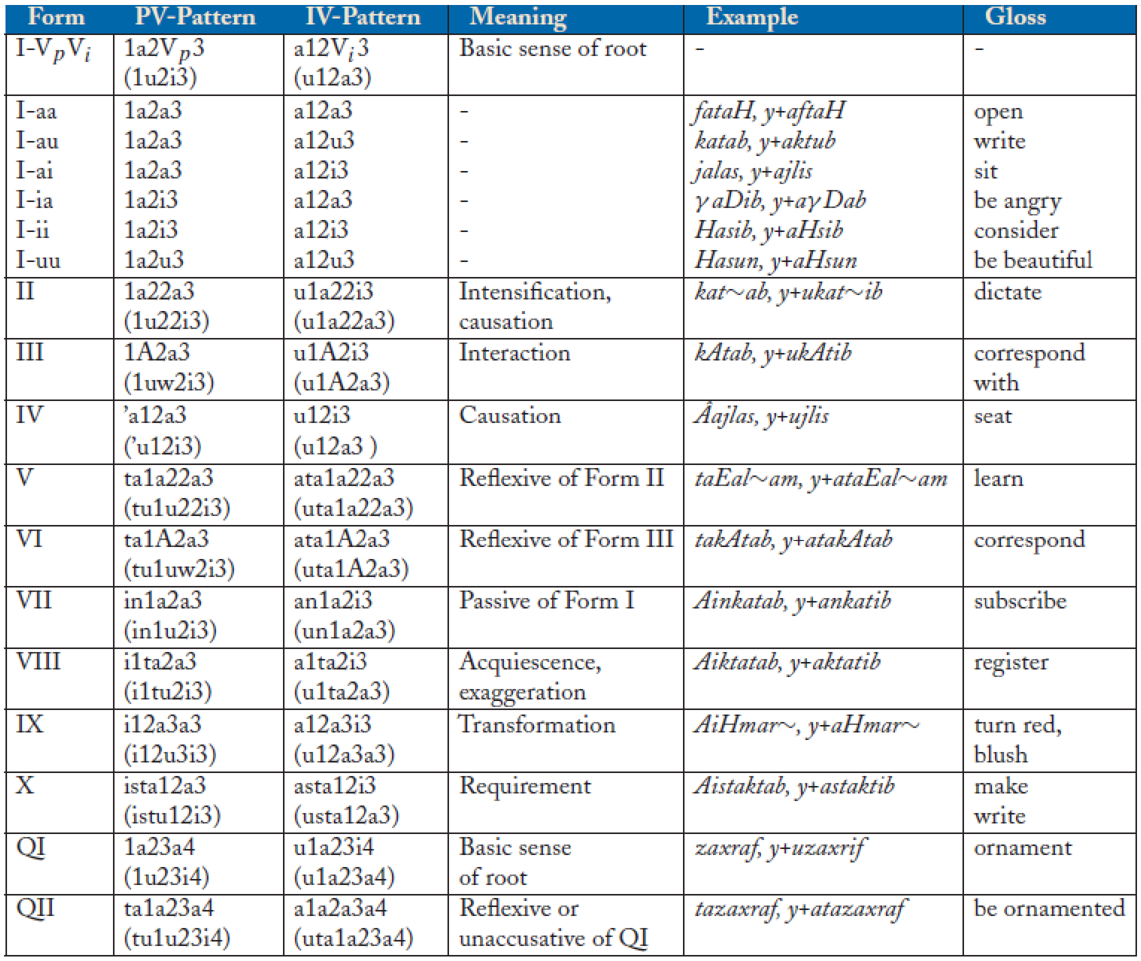
OutputColumn(DEPREL,ldep(Input[0]))

OutputColumn(DEPREL,ldep(Stack[0]))

OutputColumn(DEPREL,rdep(Stack[0]))

# Appendix E: Verbal Morphology

***Verbal Forms*** Arabic verbs have a limited number of patterns: ten basic triliteral patterns and two basic quadriliteral patterns. Verbal patterns are also called Forms (and given a Roman numeral )Figure below lists the different basic verbal patterns and their general meaning associations.

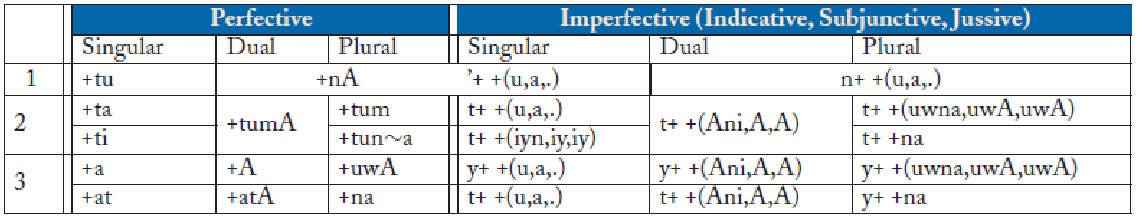


| Form | PV-Pattern | IV-Pattern | Meaning | Example | Gloss |
|---|---|---|---|---|---|
| I-$V_pV_i$ | 1a2$V_p$3<br>(1u2i3) | a12$V_i$3<br>(u12a3) | Basic sense of root | - | - |
| I-aa<br>I-au<br>I-ai<br>I-ia<br>I-ii<br>I-uu | 1a2a3<br>1a2a3<br>1a2a3<br>1a2i3<br>1a2i3<br>1a2u3 | a12a3<br>a12u3<br>a12i3<br>a12a3<br>a12i3<br>a12u3 | -<br>-<br>-<br>-<br>-<br>- | *fataH, y+aftaH*<br>*katab, y+aktub*<br>*jalas, y+ajlis*<br>*γaDib, y+aγDab*<br>*Hasib, y+aHsib*<br>*Hasun, y+aHsun* | open<br>write<br>sit<br>be angry<br>consider<br>be beautiful |
| II | 1a22a3<br>(1u22i3) | u1a22i3<br>(u1a22a3) | Intensification, causation | *kat~ab, y+ukat~ib* | dictate |
| III | 1A2a3<br>(1uw2i3) | u1A2i3<br>(u1A2a3) | Interaction | *kAtab, y+ukAtib* | correspond with |
| IV | ’a12a3<br>(’u12i3) | u12i3<br>(u12a3 ) | Causation | *Âajlas, y+ujlis* | seat |
| V | ta1a22a3<br>(tu1u22i3) | ata1a22a3<br>(uta1a22a3) | Reflexive of Form II | *taEal~am, y+ataEal~am* | learn |
| VI | ta1A2a3<br>(tu1uw2i3) | ata1A2a3<br>(uta1A2a3) | Reflexive of Form III | *takAtab, y+atakAtab* | correspond |
| VII | in1a2a3<br>(in1u2i3) | an1a2i3<br>(un1a2a3) | Passive of Form I | *Ainkatab, y+ankatib* | subscribe |
| VIII | i1ta2a3<br>(i1tu2i3) | a1ta2i3<br>(u1ta2a3) | Acquiescence, exaggeration | *Aiktatab, y+aktatib* | register |
| IX | i12a3a3<br>(i12u3i3) | a12a3i3<br>(u12a3a3) | Transformation | *AiHmar~, y+aHmar~* | turn red, blush |
| X | ista12a3<br>(istu12i3) | asta12i3<br>(usta12a3) | Requirement | *Aistaktab, y+astaktib* | make write |
| QI | 1a23a4<br>(1u23i4) | u1a23i4<br>(u1a23a4) | Basic sense of root | *zaxraf, y+uzaxrif* | ornament |
| QII | ta1a23a4<br>(tu1u23i4) | a1a2a3a4<br>(uta1a23a4) | Reflexive or unaccusative of QI | *tazaxraf, y+atazaxraf* | be ornamented |

**Figure Arabic** verb forms. Patterns for perfective (PV) and imperfective (IV) aspect are provided in the active and passive voice.Passive voice patterns are in parentheses.All patterns and examples are conjugated in the 3rd person masculine singular.Form I has six subtypes that vary in the perfective/imperfective stem vowel (marked as Vp and Vi , respectively); however, Form I has only one passive voice form per aspect (regardless of subtype).

***Verbal Subject, Aspect, Mood and Voice*** The verbal subject is specified using three features: person,gender and number. Person has three values: 1st (speaker, متكلم *mutakal~im*), 2nd (addressee, مخاطب *muxATab*) 3rd (other,غائب γ *Aˆyib*). Gender has two values: masculine 'مذكر' or feminine 'مؤنث'. And number has three values: singular, dual or plural. The verbal subject is indicated through affixations, whose form is constrained by verbal aspect and mood. See next Figure for a list of all the verbal subject affixes.

The subjects conjugated with the perfective aspect are only suffixes, while the subjects conjugated with the imperfective are circumfixes.



| | Perfective | | | Imperfective (Indicative, Subjunctive, Jussive) | | |
|---|---|---|---|---|---|---|
| | Singular | Dual | Plural | Singular | Dual | Plural |
| 1 | +tu | +nA | | '+ +(u,a,.) | n+ +(u,a,.) | |
| 2 | +ta | +tumA | +tum | t+ +(u,a,.) | t+ +(Ani,A,A) | t+ +(uwna,uwA,uwA) |
| | +ti | | +tun~a | t+ +(iyn,iy,iy) | | t+ +na |
| 3 | +a | +A | +uwA | y+ +(u,a,.) | y+ +(Ani,A,A) | y+ +(uwna,uwA,uwA) |
| | +at | +atA | +na | t+ +(u,a,.) | t+ +(Ani,A,A) | y+ +na |

**Figure** Arabic verb subject affixations.

Arabic has three common moods that only vary for the imperfective aspect: indicative (مرفوعmarfuwς), subjunctive (منصوب *manSuwb*), and jussive (مجزوم *majzuwm*). The perfective 'الماضي' aspect does not vary by mood, although the value of the mood feature with the perfective aspect is typically defaulted to *indicative*. The indicative mood is also the default mood with the imperfective 'المضارع' aspect indicating present or incomplete actions. The other moods are restricted in their use with particular verb particles.

## Appendix F: Arabic Verbs

العديد من النقاط أًخذت بعين الاعتبار مثل الأفعال المتعدية و اللازمة، الدلالات اللغوية للأفعال المختلفة و أوزانها، الأفعال المختلفة وأنواعها كالناقصة، و المدح و الذم، و التعجب، والصحيح و المعتل، و المجرد و المزيد، مصدر الفعل وهو ما تضمن أحرفه لفظاً أو تقديراً، دالاً على الحدث مجرداً من الزمن مثل: علِم علْماً، ناضل نضالاً في توظيف العلاقة بين الأفعال و الأسماء، و كذلك المشتقات الأخرى التي أصلها المصدر كاسم الفاعل و اسم المفعول.

و الى أي مدى يمكننا الاستفادة من قواعد اللغة العربية الموجودة في اللغة العربية الحديثة من ناحية وجوب التقديم و التأخير و ظهور الضمير و غيابه و الصفة المشبهه و الدلالات السيميائية المتعلقة بذلك.

Also several points are taken into account, such as Transitive and intransitive verbs Linguistic connotations for different acts and weights Various acts and types like inadequacy, praise , slander, the exclamation, the right ,ailing, and abstract and more The source of the act, which guarantees its letters rude or recognition, indicative of an abstract of time such event: flag note, fought struggle in the recruitment of the relationship between verbs and names, as well as other derivatives, which originated as the name of the source and the name of the actor effect. And to what extent we can take advantage of the Arabic grammar in modern Arabic language in terms of presentation and should be delayed and the emergence of conscience and character, and his absence and similitude semiotic connotations relating to there.