

Thesis for the Degree of Masters of Arts

Natural Language Processing Model for Maltese Syntax

Master in Language Technology

Greta Attard

Under the supervision of

Gerlof Bouma

(Språkbanken, University of Gothenburg)



UNIVERSITY OF
GOTHENBURG

Department of Philosophy, Linguistics and Theory of Science

University of Gothenburg

SE-405 30, Gothenburg, Sweden

Gothenburg, 2021

Contents

Acknowledgements	i
Abstract	ii
1 Introduction	1
1.1 Maltese Language History	1
1.2 Thesis Goals	1
1.3 Terminology	2
2 Literature Review	3
2.1 Natural Language Processing	3
2.2 Syntax and Linguistic Features for Computational Linguistics	4
2.3 Part-of-speech Tagger	5
2.3.1 Part-of-speech	5
2.4 About Maltese	7
2.4.1 Nominal Morphology	9
2.5 Dependency Parser	11
2.5.1 Word Order	11
2.5.2 Sentence Structure	13
2.5.3 Dependency Relations	15
2.6 Universal Dependencies	16
2.6.1 Annotation Principles	17
2.7 What is <i>spaCy</i> ?	23
2.7.1 Tokeniser Algorithm	25
2.7.2 Part-of-speech tagger Algorithm	26
2.7.3 Dependency Parser Algorithm	26
2.8 Conclusion	27
3 Methodology	28
3.1 Corpus Data	28

3.1.1	Considered corpus: <i>MLRS</i>	28
3.1.2	Chosen corpus: <i>MUDTv1</i>	29
3.2	Design of the Model	33
3.2.1	Preparing the Pipeline	33
3.3	Training	38
3.4	Testing	39
4	Results	40
4.1	Results from inputting unseen text to the training set	40
4.2	Results from test set	45
5	Discussion	46
5.1	Results after training	46
5.2	Comparing the performance of the different language models vs Maltese language Model	46
5.3	Maltese model output visualisation	49
5.4	Working with unseen text	54
5.5	Conclusion	58
6	Conclusion	60
7	References	62

Acknowledgements

This masters thesis is a product of work that was finalised with the help of these people.

Firstly, I would like to thank my thesis advisor Gerlof Bouma, for his work, efforts and guidance in this research which I am extremely grateful for. I would also like to thank Slavomír Čéplö and Dr Albert Gatt who have provided me with the stepping stones for the thesis.

I must express my very profound gratitude to my parents for providing me with unfailing support and continuous encouragement throughout my years of study and support thereafter.

Finally, an individual that deserves special attention is my fiance who helped me through all the obstacles that I had by literally sitting next to me holding my hand to aid me through those anxious moments when I thought I would not be able to make it.

This accomplishment would not have been possible without any of these people. Thank you!

Greta Attard

Abstract

The objective of this thesis is to create a Natural Language Processing Model for the Maltese Language. The ultimate goal is that the model would be able to recognise syntactical features, that is the linguistic features and the relationship of a sequence of words, in Maltese. The performance and accuracy of the Maltese model is compared with the models of languages that have great influence on the Maltese language. The results outputted by the dependency parser were linguistically analysed to provide in depth analysis of the results outputted during training and testing. The model is tested on unseen text to provide a further understanding of the level of accuracy of the machine learning algorithm.

For this syntax annotator, the model created is trained on manually annotated data and then the output is syntax data that is processed by the dependency parser and part-of-speech tagger. This model is made using the Python package *spaCy*. Since every language is unique, the linguistic rules are evaluated, to teach the model the rules of the language being researched. The MUDTv1 corpus developed by Slavomír Čéplö for his Phd Thesis is used to train this model. The results show that the Maltese syntax model had a 91% part-of-speech tag accuracy, 74% unlabelled attachment score and 66% labelled attachment score. The model is further tested on unseen non-annotated text, the tag accuracy is 75% and the tokeniser accuracy is 99%.

Keywords: natural language processing, syntax, *spaCy*, universal dependency, dependency parser, part-of-speech tagger, maltese nlp pipeline

1 Introduction

The focus of this thesis research is the creation of a Natural Language Model for Maltese syntax. Maltese is the national language of Malta ¹, and because of its rich history and the different colonies that governed the islands, this has left a great influence of the language as it is known today.

1.1 Maltese Language History

Maltese is a Semitic language with Romance influences, written in Latin script. History indicates that it is a descendant from Sicilian Arabic, a variety of Arabic that developed in Sicily around 831. Over the next 800 years, according to Azzopardi-Alexander and Borg (2013)'s research, Maltese evolved further from this variation of Arab to include features found in Romantic languages. These influences began as result of the invasion of the Normandy colony (1091), the colonisation of the Arabs (1225) and thereafter the re-introduction of Christianity (15th century), the governance of the Military Hospitaller Knights of St. John of Jerusalem (16th century), the French invasion (1798), the British Monarchy (1799), and becoming a Republic in 1974. Diglossia was apparent amongst citizens of Malta around the 19th century and this created a language question whereby there was a constant debate whether Maltese should be considered as the official language rather than the languages spoken at the time by aristocrats which were English and Italian. In 1934, Maltese was declared to be the official language alongside English. These became the primary languages of government, commerce, education and culture in Malta.

1.2 Thesis Goals

This work will create a further contribution to the research currently being done in academia at the University of Malta and on a Governmental level, specifically in the field of Natural Language Processing, to digitalise the Maltese language.

¹ My mother tongue

A tagged corpus is already existent, this is the Maltese Language Resource Server (*MLRS*) which has tokens annotated with their part-of-speech (POS), lemmas and their morphological roots. However, there is currently no syntactic annotation beyond POS in the *MLRS* Corpus. This peaked my interest in this research to bridge this gap. With the help of the existent syntactically annotated corpus created by Slavomír Čéplö (2018) for his PhD Thesis, Dr Čéplö provided his corpus for the completion of my research. The corpus was utilised to train a Maltese model created using the *SpaCy* (2018) platform.

1.3 Terminology

Throughout this study, the definitions of “sentence” and “word” will be understood to be the following:

Sentence: The word “sentence”, in the context of written language, is defined as the text organisation split up by punctuation; example, “The corpus is made up of a list of words.” “Sentences” will be used for the processing of the NLP Model, regardless of the number of words in it or its grammatical structure.

Word: The term “words”, as explained by (Čéplö, 2018), are separate tokens in a sentence that provide information and when joined together make up a sentence. Example: “corpus”; “sentences”; “books”.

Making use of the term “word” is an ambiguous concept since it ranges from phonological, morphological, syntactical and orthographical linguistics (as will be explained in Section 2.7.3, due to the use of Universal Dependencies). “Words” are usually analysed for their syntactic form, and not in the way they are spelt. This may affect the results because Maltese is a language with very rich morphology and commonly joins clitics to words. An example of this can be seen through the word *kielha* “he-ate-it(fem)”, where the clitic is *ha* which refers to the her and the verb *kiel* is conjugated as “he ate”. Used in a sentence: *Huwa kielha t-tuffieħa* “He ate the apple”; since *tuffieħa* is the feminine form therefore the verb takes on the clitic to refer to “it” in the feminine form. Clitics are morphemes that are bound phonologically to another word (Hartmann & Stork, 1972).

2 Literature Review

This chapter details the analysis done prior to carrying out this research. It will provide a better understanding of the computational linguistic research required to make a syntax model in Maltese.

Throughout this section there will be examples in Maltese and then following with a translation in English. A clearer explanation of the translation will be provided in some cases with the use of Leipzig Glossary Rules (Bickel, Comrie, & Haspelmath, 2008).

2.1 Natural Language Processing

The idea of computers acquiring the intelligence to understand and be able to handle natural language is one that has been in the pipeline for a long time according to Jurafsky and Martin (2009) in their book “Speech and Language Processing”. The research field of Natural Language Processing (NLP) sprouted from this aim and it is supported by different sectors of scientific research. These sectors; Artificial Intelligence (AI); Computational Linguistics and Linguistics, have contributed greatly in enriching this field of studies to its present state. As described by (Kral, 2015, p. 3) in his work such applications of NLP are machine translation, dialogue management, semantic analysis and speech recognition amongst others. Natural Language Processing is like a jigsaw puzzle whereby different linguistic features are merged together through an NLP pipeline for the productions of the NLP applications mentioned. Because of this, it is important to have a good understanding of the linguistic rules of a language.

For this thesis focus will be made on the syntactic features of Maltese. This will enable the creation of a syntax model which can then be appended into the existing language pipeline for Maltese that includes the other linguistic features models: morphology; semantics; pragmatics; phonetics and phonology. The research for this thesis may be appended to previous projects, such as Maltese Language Resource Server ² to enhance the capacity

² <https://mlrs.research.um.edu.mt/>

and capability of the project. This study will therefore be able to improve the results of applications such as text generation or summarisation and question answering.

- Text generation, possibly better known as Natural Language Generation, is a sub-field of NLP whereby with an extensive database an application can generate text in a language about a certain topic (Zhang & Sun, 2009). A syntax model would be required to analyse the words in the database to create cohesive sentences with correct syntactical relations.
- Text summarisation is an approach implemented to produce a piece of text that brings out the essence of a long text in a shorter form (Sarbo, Farkas, & van Breemen, 2011). A syntax model will help this application by analysing the text provided and be able to determine, through token frequency, how it is best to summarise the text.
- Question answering can automatically identify and understand utterances. This is one of the most common uses of speech recognition (Sinha, 2010). A syntax model would be able to generate the right answer from the converted analogue request to a coherent response.

Syntax is therefore an essential part of these applications due to the nature of the study relating to syntactic relations.

2.2 Syntax and Linguistic Features for Computational Linguistics

This study will focus on syntax, an area of scientific language research that studies the principles and processes to create a sentence structure through sets of rules (Chomsky & Lightfoot, 2002, p. 11) that may be either language specific or else understood universally by all languages. These rules and principles concern concepts like word order, temporal case, gender, quantifiers and part-of-speech, of the words in sentences.

This study shall incorporate the syntactical linguistic research to the field of NLP through the use of a Dependency Parser and a Part-of-speech Tagger, which together will facilitate the research for this thesis. To make grammatically correct sentences it is important to

have a hierarchical view of how the words depend on each other. This is represented in the form of a tree-like structure which helps provide a visual representation of the direction of the dependencies. This is also a good aid for future analysis of the semantic interpretation of the text at hand (Gamallo Otero, 2008). To further support the design of this syntax model, a part-of-speech tagger will be able to identify the diverse part-of-speech types. The output provided by this tagger is essential input for the dependency parser, therefore both models are required for a complete syntax model.

2.3 Part-of-speech Tagger

The function of this tagger is to annotate words in a corpus with their part of speech i.e verb, noun etc... . These tags are not only limited to words but also to punctuation. Therefore, part of the tagging process includes tokenisation (required to be done before this annotation process), which is a function that splits the corpus into tokens and punctuation to be able to disambiguate the long string of characters that a computer would be analysing (Grefenstette, 1999, p. 117-133).

A part-of-speech tagger takes a string of tokenised text as input, i.e. a list of words, which is then processed and outputs the best tag for each word in the corpus according to how the tagger was trained. Part of the processing to choose the perfect tag also includes the analysis of neighbouring tags of the tokens for context. This will help the tagger decide the best tag for ambiguous words like “book”, which is a word that according to the sentence context can either be a VERB or a NOUN (example from (Jurafsky & Martin, 2009, Chapter 8.3)).

2.3.1 *Part-of-speech*

Semantic knowledge is not needed to be able to tag sentences with their part-of-speech. If a sentence was in English-like gibberish, a native speaker of the English language would still be able to identify the part-of-speech of that sentence. For example:

The yinkish dripner blorked quastofically into the nindin with the pidibs. (7) (Carnie, 2013)

This example (Carnie, 2013, chapter 1.1) shows that by just looking at the sentence, it is possible to identify that “yinkish” is an adjective, “dripner” a noun, “blorked” a verb, “quastofically” an adverb, and “nindin” and “pidibs” both nouns. So, even without contextual meaning, a POS-tagger should be able to detect what part-of-speech these words have because of the dependency on word position and their affixes.

Part-of-speech is partially determined by the word’s position in the sentence and its morphology. This knowledge is then transferred into the field of NLP through a part-of-speech tagger, which is an algorithm that can assign a part-of-speech tag to a token and give a syntactic class marker (Jurafsky & Martin, 2009, Chapter 8.3).

Below is the output expected to be produced from a part-of-speech tagger:

- (1) Il-kelb iswed telaq mid-dar.
 the-dog-3SG black left-3SG.M-PAST from-the-house
 “The black dog left home”
 Il-/DEF kelb/NOUN iswed/ADJ telaq/VERB mid-/PREP_DEF dar/NOUN ³

There are various approaches to go about creating a part-of-speech tagger, the most well known being the rule-based and the HMM taggers. The rule-based taggers make use of a written dictionary that includes a list of words and their identified part-of-speech to be able to tag the tokens in a corpus. The tagger would search through the dictionary for the word being annotated and a list of possible matching tags would be outputted. Then, this list is compared to the another set of rules with specified language constraints to rule out the incorrect POS tags.

Hidden Markov Model (HMM) taggers make use of a method of probabilistic tagging to pick the correct POS tag for the word. This model tries to determine the optimum sequences of tags that agree with the given sequence of words and works out two kinds of probabilities, which are: how likely a word can be given to a certain tag according to the tags of the words around it and how likely that word has that specific tag. This tagger is first trained on a manually annotated corpora; after it makes use of an algorithm to

³ Annotation explained further in Section 3.1.2

be able to decipher which tag is most appropriate for an unseen word from a different corpus. Apart from these, there are various other machine learning techniques which are decision maximum entropy and other log-linear models; decision trees; memory-based learning and transformation-based learning (Jurafsky & Martin, 2009, p. 169).

2.4 About Maltese

After analysing the linguistic features of syntax in Maltese, understanding the features of the language such as the special characters found in the alphabet and even grammatical features was the next step in the background research for this thesis.

The Maltese alphabet knows its origin from the Latin Alphabet with the inclusion of some letters, having diacritic marks and digraphs (Azzopardi-Alexander & Borg, 2013, p.xii). There are 30 letters that are made up of 24 consonants and 6 vowels, ‘a e i ie o u’:

a b ċ d e f ġ għ h ħ i ie j k l m n o p q r s t u w x ż z

The special characters that are unique to the Maltese language as follows (Čéplö, 2018):

- ċ (U+010B) / Ċ (U+010A) which is pronounced as /tʃ/ (in English it is the sound that occurs in the “ch” of “church”);
- ġ (U+0121) / Ġ (U+0120) which is pronounced as /dʒ/ (in English it is the sound that occurs in the “j” of “jump”);
- għ (U+0067 & U+0127) / Għ (U+0047 & U+0127) which is pronounced as /ʔ/ (in English it is the sound that occurs in the “a” of “cat”);
- h (U+0068) / H (U+0046) which is silent;
- ħ (U+0127) / Ħ (U+0126) which is pronounced as /h/ (in English it is the sound that occurs in the “h” of “hat”);
- ie (U+0069 & U+0065) / Ie (U+0049 & U+0065) which is pronounced as /ɛ/ (in English it is the sound that occurs in the “ee” of “free”);

- \dot{z} (U+017C) / \dot{Z} (U+017B) which is pronounced as /z/ (in English it is the sound that occurs in the “z” of “zebra”);

Some of the consonants are split into coronal and sonorant consonants. *Konsonanti xemxin* “coronal consonants” are the following letters \acute{c} d n r s t x \dot{z} z and *konsonanti likwidi* “sonorant consonants” are the following letters l m n r . These consonants behave differently according to the surrounding words. When a determiner is attached to a word starting with a coronal consonant the determiner changes to reflect the coronal consonant, example: *il-* “the” + *tuffieħa* “apple” → *it-tuffieħa* “the apple”. The sonorant consonant affects the way verbs are conjugated in the plural form in the present tense, example *Jiena nilgħab* “I play”; *Intom tilagħbu* “You play” where there is an inclusion of an epenthesis because of the letter “*l*” (Fabri, 2009).

Camilleri (2013) describes that in Maltese the determiner is represented as a definite article such as “*il-*”. This article changes form according to where it is placed within the sentence, that is, if it is placed in front of a vowel it becomes “*l-*” as in *l-iskola* “the school”. Another way this article changes is when it is followed by a coronal consonant such as *il-* “the” + *sigġu* “chair” → *is-sigġu* “the chair”. This determiner is also able to combine itself with prepositions to become one word such as, *ġo* + *il-* “in” + *forn* “over” → *ġol-forn* “inside the oven”. This ability disregards any other grammatical features such as gender or quantitative but can be used with nouns and adjectives (*l-isbaħ* “the prettiest”).

When it comes to cardinal numerals, the determiner is also required because in Maltese one would say the *l-ewwel* “first”, *it-tieni* “the second” and so on. However, when counting normally the situation is different. From 1 - 10, no article is needed but from then onwards the article is required to precede the noun that indicates the quantification; let’s take an example: *ħdax-il mejda* “eleven tables”. The different ways the determiner can change will be evaluated during training of the model.

Some words function as a noun phrase when used on their own and refer to the self; these are the pronouns, i.e. *Jiena* → I; *inti* → you; *huwa* → he/it; *hija* → she/it; *aħna*

→ we; *intom* → you (pl); *huma* → they. However, these are only used when emphasis needs to be made on the person, mainly because Maltese is a pro-drop language (refer to Section 2.5.2). They are enclitic and show possession when combined with nouns such as *omm* “mother” + *tiegħi* “mine” → *ommi* “my mother” . They can also be combined with an object which is then attached to verbs such as *rajna* “we saw” + *lilhom* “to them” → *rajn-ihom* “we saw them’ which is called pronominalisation.

2.4.1 Nominal Morphology

In this field of study, nominal morphology pays close attention to how words are transformed from one form to another, for example a noun changing from singular to the plural form. Nominal morphology plays an important part in part of speech research because the transformation of the word effects the syntax of that word and in turn the annotations of it and this was also part of the research required to gain a better understanding of the Maltese language. Below are features of nominal morphology:

Case

(Vaux & Cooper, 1999, p.93-103) explains **case** as a highlight of the subject and the predicate of the sentence. Case refers to a phenomenon that the form of a (nominal) word changes depending on its grammatical function which is not the case for Maltese. However, (Vaux & Cooper, 1999, p.93-103) identifies that case marking can also be shown through “*inalienable possession*” whereby the root of the words changes in order to show possession such as *ommu* “his mother” (*omm* is the root and *u* shows possession in reference to him). The latter is an example of a possessive enclitic pronoun which are pronouns appended to the end of the noun to show the possession of a subject or an object in a sentence (Ravishankar, Tyers, & Gatt, 2017, p.175-182).

Gender

Gender is presented usually through categorisation of the word as either masculine or feminine, which for Maltese is reflected through a demonstrative pronoun meaning “that”, where *dak* is in the masculine form and *dik* is in feminine form. Example: *dak is-sigġu*

(that-3SG.M the-chair) “that chair”; *dik il-medja* (that-3SG.F the-table) “that table”; *dik is-sodda* (that-3SG.F the-bed) “that bed”; *dak il-ħalib* (that-3SG.M the-milk) “that milk”. (Vaux & Cooper, 1999, p.93-103)

Number

In Maltese, a noun can be converted into plural either in the regular form (*Il-Plural Sħiħ*) by following a definite pattern, or in the irregular form (*Il-Plural Miksur*). (Vaux & Cooper, 1999, p.93-103) explain in their work how identifying quantity is one of the simplest features detectable through these forms. Table 1 gives examples of how words are made plural. Nouns in the regular form follow a pattern whilst nouns in the irregular form by definition do not follow such a pattern. There are exceptions to all rules and there are nouns that can be pluralised with both the regular and irregular form, therefore forming part of both families as can be seen in Table 1.

Regular Form			
<i>a</i> → <i>i</i>	karta “paper”	→	karti “papers”
add <i>ijiet</i>	sigġu “chair”	→	sigġijiet “chairs”
add <i>ien</i>	wied “valley”	→	widien “valleys” ³
add <i>at</i>	xemgħa “candle”	→	xemgħat “candles”
add <i>in</i>	ħalliel “thief”	→	ħallelin “thieves”
add <i>a</i>	għalliem “teacher”	→	għalliema “teachers”
add <i>s</i>	plejer “player”	→	plejers “players” ⁴
Irregular Form			
	sodda “bed”	→	sodod “beds”
	raġel “man”	→	irġiel “men”
	kċina “kitchen”	→	kċejjen “kitchens”
Exceptions:			
	Regular		Irregular
bieb “door”	bibien “doors”		bwieb “doors”
triq “road”	triqat “roads”		toroq “roads”
bolla “stamp”	bolli “stamps”		bollol “stamps”

Table 1

Representing Nominal Morphology

³ At the beginning of the word there is the vowel *ie*, in Maltese, this vowel cannot appear twice in a word and thus when the plural suffix is added the first *ie* is changed into *i*

⁴ Words adopted from the English language

2.5 Dependency Parser

The syntax model will have a dependency parser which will be trained on the chosen corpus and will be capable of handling the dependency syntax and grammar rules. (Jurafsky & Martin, 2009). Figure 1 illustrates a dependency visualisation where the sentence structure is made dependant on the syntax relation of the words in the sentence which is represented through directed labelled arcs. The token marked as root is always the word from where the relations start. In Figure 1, the verb *prefer* is the root of the sentence connecting the subject to its direct object. (Jurafsky & Martin, 2009, chapter 15).

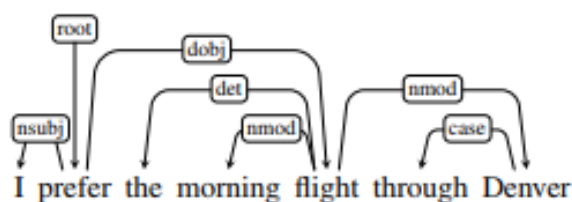


Figure 1

Dependency relation (Jurafsky & Martin, 2009)

2.5.1 Word Order

The sentence order holds a lot of information about the token relation and their dependencies. This linguistic concept is important for the dependency parser because the way the word order reflects the dependency relation of the sentence. Since certain languages are able to change word order and still keep the same contextual meaning that it holds, this is crucial information that needs to be understood and gathered by the parser. Some may say that the sentence may look “displaced”, but that does not mean it is incorrect (Jurafsky & Martin, 2009, chapter 15),

Below are the examples of the word order rules for sentences in Maltese:

SVO:

- (2) It-tifel xehet il-ġebbla.
 the-boy-SG throw-3M.SG-PST the-stone-SG
 “The boy threw the stone.”

SOV:

- (3) It-tifel il-ġebbla xeġet.
 the-boy-SG the-stone-SG throw-3M.SG-PST
 “The boy the stone threw.”

VOS:

- (4) Xeġet il-ġebbla t-tifel.⁶
 throw-3M.SG-PST the-stone-SG the-boy-SG
 “Threw the stone the boy.”

OVS:

- (5) Il-ġebbla xeġet it-tifel.
 the-stone-SG throw-3M.SG-PST the-boy-SG
 “The stone threw the boy.”

OSV:

- (6) Il-ġebbla t-tifel xeġet.
 the-stone-SG the-boy-SG throw-3M.SG-PST
 “The stone the boy threw.”

(Examples (2)–(6) taken from Spiteri (1998))

In the examples above, the word order varies across all of the sentences, however, there is an equivalent dependency structure analysis showing that the sentence meaning remained consistent and coherent. In examples (2)–(6), the root of all the sentences is the verb (i.e. *xeġet* “to throw”) which has the nouns that are dependant on it. The agent of the sentence (subject) is the person doing the action, and the patient (object) is the person receiving the action. Taking example (SVO) the agent, the boy, is doing the action, to throw, and the patient is the stone. The only word order that is not allowed in Maltese is “VSO” since sentence with this structure are incoherent. The placement of the object and the subject of the sentences at the different word orders can be identified because

⁶ In Maltese words that finish with a vowel cannot be followed by a word that starts with a vowel, therefore in this example, the determiner drops the “i” to accommodate for this exception.

of the morphological marker which is commonly found in languages with free word order (Comrie, 1989, chapter 4). In examples (2)–(6), the morphological marker is the word *xehet* “to throw” which is conjugated in the indicative mood, second person, singular, past tense.

Spiteri explains that although sentences are dealing mainly with grammar they also deal with expressions. This is because humans dialogue with each other and most of the time they do not speak in full sentences. Even though the context is irrelevant for a dependency parser, it does not negate the importance that word order is very significant in syntax. This is because the model would be able to recognise how the language behaves when the word order changes and how this effects the relation between the words.

2.5.2 *Sentence Structure*

A sentence is a group of words that conveys a statement and each element has its own function. The main job of the parser is to analyse the syntactic structure in a sentence by analysing the syntactical rules of the language and also the surrounding tokens in a sentence. This information, accompanied by that outputted from the POS tagger will help the parser analyse the syntactic structure of the sentence.

Maltese is a pro-drop language where it is common that sentences might have absent pronouns. Despite the absence of the pronoun, the reference of the sentence (i.e. to who or to what), is still understood. Example:

(7) *Jien mort il-baħar*

I go-1SG.M the-sea-SG “I went to the beach”

can be said as

(8) *Mort il-baħar*

go-1SG.M the-sea-SG “(I) went to the beach”

the pronoun, *Jien* “I” , is absent because it is understood that the person is referring to the self because of the way the verb is conjugated in the first person.

Maltese is also considered a null-subject language whereby grammatical features are referenced within the verb making the subject redundant, as demonstrated by the following examples:

	<i>It-tifel</i>	<i>xtara</i>	<i>rigal</i>	<i>lil ħuh</i>	(Spiteri, 1998)
	The-boy	buy-3SG.PST	gift-SG	to brother-POSS	
(9)	The boy	bought	a gift	for his brother	
	↑	↑	↑	↑	
	subject	verb	object	indirect object	

If an individual knew the context, this sentence (9) could be reduced into something shorter such as:

(10)	Xtra-hu-lu
	(He) buy-3M.SG-PST him-3SG-DAT.SG
	“He bought it for him”

This (10) is a classic example of a constituency test, which in this case is pronominalisation. A constituent is a number of words that when produced in a hierarchical structure provide the same meaning. Pronominalisation is a test that sees whether there was a substitution of a phrase into a pronoun. Since the structure of the sentence was not changed, the pronoun (in the case of example (10) is *hu* “him” in the verb *xtrahulu*), is a constituent since it replaced the subject and object. (Osborne, 2018, p.1-41)

A sentence structure can be split up and divided into its individual functions so that a dependency structure analysis can be made. It helps establish the relationship of the words through segmentation of the sentence structure.

2.5.3 *Dependency Relations*

Nivre et al.(2016) described dependency grammar as the relations between the tokens defined by the dependency parser through the annotated labels and their dependency arcs. Figure 2 illustrates a list of relations between sentence tokens.

Clausal Argument Relations	Description
NSUBJ	Nominal subject
DOBJ	Direct object
IOBJ	Indirect object
CCOMP	Clausal complement
XCOMP	Open clausal complement
Nominal Modifier Relations	Description
NMOD	Nominal modifier
AMOD	Adjectival modifier
NUMMOD	Numeric modifier
APPOS	Appositional modifier
DET	Determiner
CASE	Prepositions, postpositions and other case markers
Other Notable Relations	Description
CONJ	Conjunct
CC	Coordinating conjunction

Figure 2

Dependency relation taken from Jurafsky and Martin (2009)

When speaking about tokens, punctuation is also considered to be a token that delivers specific context to the sentence itself. Some rules need to be followed to produce a result by a dependency parser:

- The same single root node for every token in a sentence which has a unique arc connected to it.
- Each node has one single incoming arc excluding the root node which has multiple outgoing arcs.

The idea of projectivity is how to deal with the drawings of the arcs in the dependency tree which may be projective and non-projective. The difference between the two is that if the parse tree is projective the arcs do not cross edges. Therefore, a projective dependency can have all the words predefined in a linear order and all edges drawn on the plane with arcs overlapping each other because there is no immediate path between one node and its modifier that cannot be done without overlapping another arc.

Let's take the example in Figure 3, where the arcs that cross each other labelled *flight* and its modifier *was* is a non-projective arc. This is considered so because there is no way for the word *flight* to reach its modifier *was* without crossing the arc of the word *morning* that connects it to the head of the sentence *cancelled*. Non-projective dependencies are more commonly found in languages that have a more flexible word order such as Maltese.

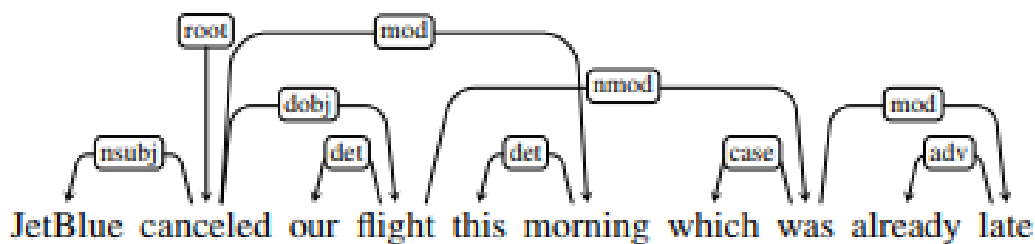


Figure 3

Dependency relation without projectivity (Jurafsky & Martin, 2009)

For the purpose of a dependency parser, different algorithms need to be utilised in order to deal with project and non-projective dependency trees. Transition-based dependency parsing is used for projective dependency parsing since it is a parsing process that is modeled as a sequence of transitions (explained in further detail in Section 2.7.3).

2.6 Universal Dependencies

When trying to process natural language automatically one may be presented with some obstacles which are; each language has a diverse annotation scheme, and this makes evaluating languages and understanding the annotation of the language that much harder. The Universal Dependencies project was created to develop and regulate the type of treebank annotation made in languages universally (Nivre et al.2016) . Čéplö (2018), for the *MUDT_{v1}* corpus, used the Universal Dependency scheme as a guideline to create the Maltese annotation scheme (reference is made further in Section 3.1).

According to Kolachina and Ranta (2016), the UD files correspond to a particular language including information about the parts-of-speech together with grammatical functions. The analysis, of the annotation scheme, specifically the universal dependency scheme will provide a better understanding to its importance when comparing languages

syntactically to each other. Nivre et al. (2016) explored the syntax and parsing across different levels of languages. This can be clearly seen in their example that discusses an argument for Universal Dependency on the basis of different styles of dependency analysis:

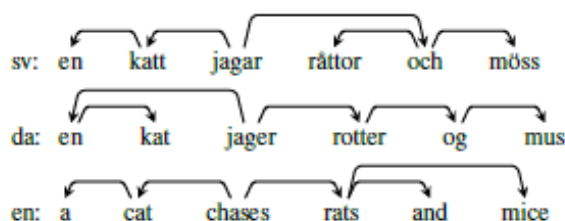


Figure 4

Different annotation for parallel sentences (Nivre et al. 2016, p. 1659)

Figure 4 shows three sentences in Swedish, Danish and English which hold the same meaning, that being “a cat chases rats and mice”. They are annotated according to the guidelines of the treebanks of their respective languages. The syntax structure of the sentences is the same and have identical word order. However, the manner in which the words depend on each other in the sentences differ from one language to another. As can be seen in Figure 4, there are instances where “cat” depends on the determiner “a” as with the case in English; however in Danish, the determiner “en” depends on “kat”. This poses a problem for when one wants to train a parser in several languages as it will be trained on one type of annotation for a specific language thus producing a high error rate. The Universal Dependencies deals with this problem by trying to make cross-linguistically consistent grammatical annotation that will complement the existing language-specific schemes (Nivre et al. 2016).

2.6.1 Annotation Principles

The annotation in the Universal Dependency is syntactic and is based on “dependency” and “lexicalism” which means that words are the smallest units of grammatical annotation. To get “a cross-linguistically consistent and transparent annotation” (Nivre et al. 2016, p. 1659), one would need to make sure that there are the same annotation structures and parallelism across the languages. For the purpose of this research, the need

to understand how the Universal Dependencies annotation scheme works came from the eventual understanding of how our corpus is annotated and how to analyse the model output. Universal Dependency annotation holds the following information:

Word Segmentation: Since UD is all based on syntax this annotation level takes into consideration how words are split when words have clitics or contractions. Contractions, such as *isn't = is not*, from the English language, are non-existent in the Maltese language. Therefore for Maltese, focus is when words have clitics such as:

- (11) naghtihomlok
 give-1SG.PRES-them-to-you-SG
 “I give them to you”

which is split into you *nagħti-hom-lok* mapped to *nagħti-huma-lilek* “(I) give-them-to-you”.

Morphology: Words hold information for lemma, part-of-speech tags, and morphological features. The details of this information are seen in Table 2 and 3 respectively.

Lexical	Inflectional	
	(Nominal)	(Verbal)
PronType	Gender	VerbForm
NumType	Animacy	Mood
Poss	Number	Tense
Reflex	Case	Aspect
	Definite	Voice
	Degree	Person
		Negative

Table 2

Morphological features for the Universal Dependency (Nivre et al.2016)

Open class words		Closed class words		Other	
ADJ	adjective	ADP	preposition/postposition	PUNCT	punctuation
ADV	adverb	AUX	auxiliary	SYM	symbol
INTJ	interjection	CONJ	coordinating conjunction	X	unspecified POS
NOUN	noun	DET	determiner		
PROPN	proper noun	NUM	numeral		
VERB	verb	PART	particle		
		PRON	pronoun		
		SCONJ	subordinating conjunction		

Table 3

Part-of-speech tags in the Universal Dependency (Nivre et al.2016)

Table 2 shows the morphological features where each feature is attached to some tokens. Table 3 shows 17 different types of part-of-speech tags that may be used for all languages and arranged accordingly, however, not all classes are used in all languages.

Syntax:

It is important to note that there is a relation between morphology and syntax; where morphology deals with how forms of words are built (starting from morphemes and adding affixes) and syntax deals with how these morphologically correct words can be placed coherently into phrases or sentences (Julien, 2007). The organisation of the grammatical relations holds three types of structures:

- Nominal: This clusters together nouns and adjectives, example the noun phrase

(12) *Aħna ġħandħa žiemel*

we have-1PL.PRES horse-SG.

“We have (a) horse”

where the phrase starts with a pronoun and a finishes noun. Within this phrase there needs to be agreement throughout the gender, noun class and case. Such agreement can be seen across the affix of a noun, the determiners and adjective (Kapust, 1998). Nominals are beneficial to identify what type of phrase a phrase may be and thereafter the sentence structure would be identified which is helpful for the dependency parser.

- **Clauses:** A group of words which hold the subject of the sentence plus the verb phrase. A clause is a statement that is independent such as *Jien norqod* “I sleep” which can also be referred to as a sentence. However, a sentence is able to have more than one clause since it may have an independent clause and one or more subordinate clauses (Kroeger, 2005). This information will definitely be helpful for the dependency parser too and is directly related with the word order of sentences.
- **Modifiers:** Phrases or words that can modify the meaning of the structure and how the words depend on each other, like adjectives and adverbs. An example of this is:

(13) *Għandi ċavetta ħadra*
 have -1M.SG-PRES key -SG green
 “I have (a) green key”

In this sentence, the adjective is *aħdar* “green” which describes the noun. Without this adjective, the sentence is still grammatically and structurally correct; therefore the adjective is the modifier in the sentence as it modified the noun key (Kroeger, 2005). This is especially helpful for part-of-speech tagging since the tag of the modifier word will help in understanding the specific placement of that word.

Part of the annotation principles the Universal Dependency deals with is the relation taxonomy of the words in a sentence. Table 4 (De Marneffe et al., 2014) shows the grammatical syntactical relations. The rows of the table are the functional categories of the root, also known as head, of the sentence whilst the columns are linked to the structural categories of the dependent words of the sentences. The labels within the

table all directly refer to the words holding that label, example “nsubj” refers to nominal subject such as the word *bieb* “door” in the sentence: *Il-bieb huwa aħmar* “The door is red”.

	Nominals	Clauses	Modifier words	Function Words
Core arguments	nsubj obj iobj	csubj ccomp xcomp		
Non-core dependents	obl vocative expl dislocated	advcl	advmod* discourse	aux cop mark
Nominal dependents	nmod appos nummod	acl	amod	det clf case
Coordination	MWE	Loose	Special	Other
conj cc	fixed flat compound	list parataxis	orphan goeswith reparandum	punct root dep

Table 4
Universal Dependency taxonomy relations (De Marneffe et al., 2014) taken from <http://universaldependencies.org/u/dep/>

The relation between words is a major part of the study of syntax and thus of relevance for this research.

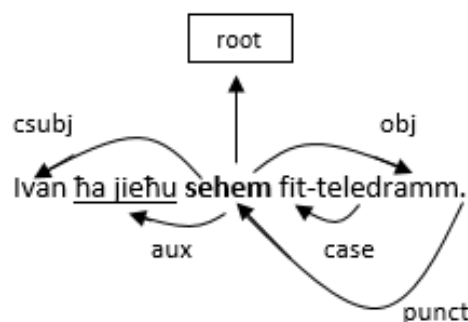


Figure 5
Maltese: Dependency relation in a sentence

Figure 5 illustrates the relation between content words. With the rules of Universal De-

pendency the sentences in Figure 5 and Figure 6 (which hold the same contextual meaning, this being *Ivan ħa jieh̄u sehem fit-teledramm* “Ivan will participate in the television drama”), are parallel to each other syntactically. This is because the same annotation scheme was performed on both sentences.

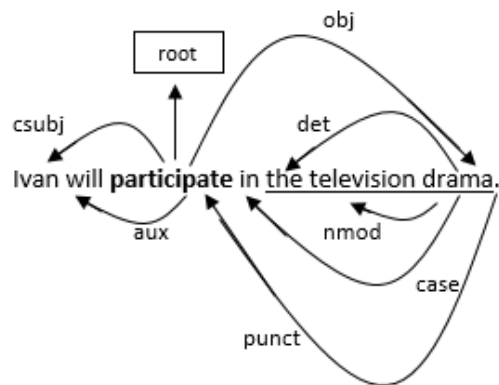


Figure 6

English: Dependency relation in a sentence (translation of Figure 5)

Apart from the word by word relation within a sentence there also exists the relationship between clauses in the sentence itself. The dependency relation of clauses is called coordination and subordination. The coordination of clauses is the joining of two or more elements with the use of conjunctions, such as: *u* “and” or *imma* “but”. The head of the sentence is the leftmost conjunct and the other words in the clauses depend on it (Figure 7).

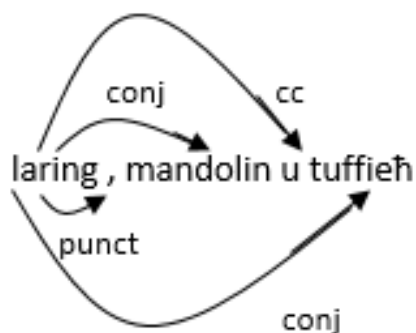


Figure 7

Coordination relation in a clause. Translation: “oranges, tangerines and apples”

**Figure 8**

Subordination relation in a clause. Translation: “She left after the meal”

- (14) Hi telqet wara l-ikla
 she leave-2F.SG-PST after the meal
 “She left after the meal”

Figure 8 and (14) , demonstrates subordination which is how thoughts are hierarchically organized. This means that a sentence has a main clause and a dependant clause, better known as a subordinate clause, and are joined together with a subordinate conjunction, these being *wara* “after”, *qabel* “before”, *għax* “because” etc.. ⁷

2.7 What is *spaCy*?

spaCy ⁸ is a freely available Python package which can build Natural Language Processing applications and it is developed by *Explosion AI* ⁹. This particular platform was chosen to pursue this project because it was developed specifically for natural language understanding. It is also capable of handling new languages to build models. The algorithms for *spaCy* feature neural models to contribute to the tasks of tagging and parsing.

spaCy does not have an existent model for Maltese and therefore a new model was created for this research. This added model will support the Maltese language through the building of a custom language subclass that includes custom information that would be able to train the model. ¹⁰

⁷ Figure 7 till 8 were inspired by the examples in (Nivre et al.2016)

⁸ <https://spacy.io/>

⁹ <https://explosion.ai/>

¹⁰ <https://spacy.io/usage/adding-languages>

then processed in the model by using the default algorithms for the part-of-speech tagger (labelled as *tagger* in Figure 10, discussed in 2.7.2) and the dependency parser algorithm (labelled as *parser* in Figure 10, discussed in 2.7.3). The component labelled as *ner* is referring the Entity Recogniser algorithm, which is ignored for this case of this research as it is not applicable.

2.7.1 *Tokeniser Algorithm*

Tokenisation is the process of splitting words found within a corpus into their smallest units called tokens. This process is the first step performed when creating any NLP pipeline since it breaks data into chunks and into a numerical data structure. One of the most common ways to perform this tokenisation process is splitting at white spaces within a string of text as the “delimiter” of words.

Let’s take the sentence:

- (15) Wara x-xogħol mort id-dar
 after the-work.SG went-1SG.PST the-house
 “After work I went home”

If this sentence had to be tokenised, the sentence would be split into four tokens like so:
 [‘Wara’, ‘x-xogħol’, ‘mort’, ‘id-dar’, ‘.’]

If complete control over the tokeniser is required then at often times regular expressions are used. This is used to match a sequence of characters through the description of a finite-state automaton (Mitkov, 2003) which is a crucial part of the morphological parsing (Jurafsky & Martin, 2009, chapter 2).

The tokeniser algorithm used by *spaCy* (explained further in Section 3.2.1[4]) gives the liberty to specify special tokens that do not need to be split up, which deemed to be a necessary feature for the tokenisation of Maltese. This is because, as can be seen in Example (15) the token *x-xogħol* should be considered as one token and therefore this

would need to be specified to the tokeniser. If not the tokeniser would split the token as [*'x', '-', 'xogħol'*] since by default the tokeniser would split at the punctuation. Therefore, it was specified to the *spaCy* tokeniser that instances such as *x-xogħol* are to be deemed as one token (Jurafsky & Martin, 2009, chapter 2).

2.7.2 Part-of-speech tagger Algorithm

The “Vocab” object has elements relating to the part-of-speech tags retrieved from the training corpus, it specifically holds the mapping of the Maltese tags to the Universal Dependency scheme tags. This mapping is done to standardise the tags and facilitate the creation of the model with *spaCy*. The Maltese tags are those used in the training corpus, in this case the *MUDTv1* (to be discussed further in Section 3.1).

The algorithm for the part-of-speech tagger is a tagger that maps annotations and is done according to the relevant assigned tag map (as described in Section 3.2.1). The basis of rule-based morphology is where the root form of a word can be changed through affixes, however it does not implicitly change the tags of the words. The steps taken for annotating unseen text is that the part-of-speech tagger will confer with the file that contains the tags which are mapped to the tokens obtained from the corpus used during training. The tagger will annotate the token with a tag that was previously assigned to that same word during training. If the word was not in the corpus the model was trained on then the algorithm will look at the surrounding word tags and make an assumption of the correct tag to assign to the token according the similar scenarios the tagger has seen before.

2.7.3 Dependency Parser Algorithm

The Dependency Parser algorithm is the shift-reduce dependency parsing through greedy decoding. The greedy shift-reduce algorithm works by looping through the length of the sentence twice and within each iteration, the features of the sentence are extracted (Jurafsky & Martin, 2009). These elements are mapped to the sentence and their relevant tokens to keep the relations intact. Mapping is a unique feature found within a transition-

based dependency parser such as this one, since it makes use of the logic found in a finite-state transducer to map the tokens to their heads. During the mapping task, the sentences are weighted according to their length which is represented as an array for efficient extraction.

During parsing, apart from keeping track of the position of the token through an index, the model also keeps track of the evaluation process of the sentence and then stores the index of the token into another array. By doing so the parser would be able to keep track of the hierarchical structure of the sentences in the corpus. This algorithm is used profusely for the training and the testing of the corpus. It will ultimately be able to assign the dependency labels on the tokens provided to the model. ¹¹ ¹²

2.8 Conclusion

The purpose of this literature review was to provide a strong understanding of the research background. The Maltese Syntax model will be facilitated by using the *spaCy* open library and the knowledge acquired from the research made about syntactical linguistic features and how it co-relates to Computational Linguistics. It transpires that a Dependency Parser and a Part-of-Speech tagger is required to be able to give the adequate syntax results for the Maltese language. To be able to work with these algorithms one would need to have a basic understanding of the Maltese language therefore, time was taken to supply adequate knowledge about the language with reference to syntax. The dependency parser and part-of-speech tagger will give results about which is the best part-of-speech annotation for words inputted, and how they depend on each other in a sentence.

¹¹ <https://explosion.ai/blog/how-spaCy-works>

¹² <https://explosion.ai/blog/parsing-english-in-python>

3 Methodology

In this chapter, an explanation of the implementation of the Natural Language Processing Model for Maltese Syntax is given. This is facilitated by the *spaCy* open library that supports new languages. The model will include computational linguistic syntax features; these being a Dependency Parser and Part-of-Speech taggers, as was discussed in Section 2.

3.1 Corpus Data

The corpus chosen for the project to be executed was the *MUDTv1* (*bulbulistan corpus*¹³) alongside the *Korpus Malti* (*MLRS*)¹⁴. *MUDTv1*, follows the Maltese tagging scheme created by Čéplö which was inspired by the Universal Dependency scheme¹⁵ while *MLRS* is a corpus developed by Dr. Albert Gatt at the University of Malta, which influenced the *MUDTv1* corpus heavily.

3.1.1 Considered corpus: *MLRS*

The corpora that were available and adequate for my research were *MUDTv1* and *MLRS*. Both corpora were evaluated to see which deemed fit. The *Korpus Malti v3.0* also known as *Maltese Language Resource Server* (*MLRS*), was highly considered but not chosen. This corpus was developed mainly by Dr Albert Gatt and can be found on the Maltese Language Resource Server¹⁶. The corpus is made up of about 250 million tokens which are automatically generated annotations with their part-of-speech tags, lemmas and their morphological roots. It is comprised of diverse text types mainly from journalistic texts and parliamentary debates.

The reason why this corpus was not chosen for my work because it only had a small

¹³ bulbul.sk/bonito2

¹⁴ mlrs.research.um.edu.mt

¹⁵ https://github.com/UniversalDependencies/UD_Maltese-MUDT

¹⁶ <http://mlrs.research.um.edu.mt/>

selection of manually tagged tokens which totalled to about 28,000, where when compared, the number of tokens in *MUDTv1* is far larger. Apart from this, the focus of *MLRS* was more on morphology. Due to this focus in the corpus, it does not include any information about syntax. Because of this, *MUDTv1* deemed to be a better fit for this thesis. However, the manually tagged corpus from the *Korpus Malti v3.0* was used in the preparation of the model and during the evaluation stage.

3.1.2 Chosen corpus: *MUDTv1*

The *MUDTv1* corpus was adapted to focus on *Constituent order in Maltese: A quantitative analysis* (Čéplö (2018) PhD thesis). It has an annotation scheme based on the Universal Dependency Annotation scheme and holds certain tags specific to Maltese. This corpus is made up of texts that are freely obtainable on the web and to the general public. Table 5 demonstrates the type of texts that are in the corpus which were taken from newspaper articles, parliament speeches, academic texts, blogs and fiction literature.

Text type	Subtype	Sentence count
newspaper	news	239
	op-eds	240
	Subtotal	479
quasi-spoken	newspaper interviews	280
	parliament: debates and Q&A	294
	Subtotal	574
fiction	short stories	246
	novel chapters	251
	Subtotal	497
non-fiction	humanities	249
	science, encyclopedic and instructional	275
	Subtotal	524
Total		2074

Table 5

Number of sentences according to Text Types in MUDTv1 Corpus (Čéplö, 2018)

The type of data in the corpus determines that the words will be very formal and orthographically correct. The texts chosen by Čéplö met the criteria of having to be original

writings in Maltese by native speakers written in the first two decades of the 21st century i.e. between 2000 - 2020. After the corpus was compiled in Čéplö's research, there were a total of 44,162 tokens as part of 2,074 manually annotated sentences. These gathered tokens are compiled in CONLL-U file formats. These file formats are compatible with the Universal Dependency scheme mentioned in Section 2.6.1.

When searching through the available Maltese corpora accessible for research, *MUDTv1* deemed to be the best fit for my thesis. A corpus which focused mainly on syntax and provided enough information to perform dependency parsing which was of great importance to reach the aim of this research.

For model training, the *MUDTv1* corpus was split into training, testing and development sets, split into 70%, 20% and 10% respectively. This was performed to have a more effective model for machine learning. It gives the model the opportunity to train and develop on a limited amount of data, and thereafter be tested on similar but yet unseen data. The model is first trained on the training set, giving the model details about what kind of data it will see and provide statistical information about it. This trained model is then stored so that any testing or any pipeline processing will be able to be executed. The development set is used to be able to evaluate and optimise the model better and give an unbiased evaluation. This set is used to be able to look at the data, not to learn from it but rather to indirectly enhance the performance of the model. The test set is used to evaluate the model and provide the best final model fit according to the data seen before and during training. This ratio of splitting the corpus was done so that the model is exposed to many different examples and thus the training set is at its largest. The test set is then larger than the development set but smaller than the training set to get the most statistical power. (Jurafsky & Martin, 2009, Chapter 15)

In his PhD Thesis, Čéplö describes how after gathering the different texts to compile this corpus they were encoded into CoNLL-U format UTF-8 encoded files. Along with the tokens, more information is annotated that aids the model to be able to process the data and produce a syntax model.

The information included is:

- **ID** - This is the word index which is restarted at the start of each new sentence
- **FORM** - Identifies whether it is a word or a punctuation character
- **LEMMA** - The stem of the word
- **UPOS** - The token's part-of-speech that is universally recognized
- **XPOS** - The token's part-of-speech that is language specific
- **FEATS** - The morphological features
- **HEAD** - The dependency relation of the HEAD to the token
- **DEPREL** - The dependency relation of the token to the HEAD
- **DEPS** - Other secondary relationships
- **MISC** - Additional information

1	L-	DEF	DEF	2	det			
2	Assocjazzjoni	NOUN	NOUN	8	nsbj			
3	tal-	GEN_DEF	GEN_DEF	4	case:det			
4	Kunsilli	NOUN	NOUN	2	nmod:poss			
5	Lokali	ADJ	ADJ	4	amod			
6	kontra	PREP	PREP	8	case			
7	l-	DEF	DEF	8	det			
8	mod	NOUN	NOUN	0	root			
9	kif	PRON_INT	PRON_INT	10	advmod			
10	twaqqaf	VERB	VERB	8	acl			
11	il-	DEF	DEF	12	det			
12	MEUSAC	X_ABV	X_ABV	10	nsubjpass			
1	L-	DEF	DEF	2	det			
2	Assocjazzjoni	NOUN	NOUN	7	nsbj			
3	tal-	GEN_DEF	GEN_DEF	4	case:det			
4	Kunsilli	NOUN	NOUN	2	nmod:poss			
5	Lokali	ADJ	ADJ	4	amod			
6	ma	NEG	NEG	7	neg			
7	taqbilx	VERB	VERB	0	root			
8	mal-	PREP_DEF	PREP_DEF	9	case:det			
9	mod	NOUN	NOUN	7	nmod:obj			
10	kif	PRON_INT	PRON_INT	13	advmod			
11	il-	DEF	DEF	12	det			
12	Gvern	NOUN	NOUN	13	nsbj			
13	waqqaf	VERB	VERB	9	acl			
14	mill-	PREP_DEF	PREP_DEF	15	case:det			
15	gdid	ADJ	ADJ	13	nmod:advmod			
16	il-	DEF	DEF	17	det			
17	Kumitat	NOUN	NOUN	13	doobj			
18	ta'	GEN	GEN	19	case			
19	Azzjoni	NOUN	NOUN	17	nmod:poss			
20	bejn	PREP	PREP	21	case			
21	Malta	NOUN_PROP	NOUN_PROP	19	nmod			
22	u	CONJ_CORD	CONJ_CORD	21	cc			
23	l-	DEF	DEF	24	det			
24	Unjoni	NOUN	NOUN	21	conj			
25	Ewropea	ADJ	ADJ	24	amod			

Figure 11

Extract from a CoNLL-U file compiled by for the purposes of MUDTv1

Figure 11 demonstrates how a CoNLL-U format looks. The data represented by an _ (underscore) are unavailable parameters, these being *LEMMA*, *FEATS*, *HEAD*, *DEPS*

and *MISC*. This information was not required for the research conducted by Čéplö. The files are formatted where each token of the sentence is split into words per line.

The column that should hold the UPOS tags actually holds the Maltese specific POS tags, this is because in the research done by Čéplö 2018 it was argued that the mapping between the UD annotation scheme and the Maltese specific POS tags was relatively straight forward. In fact, the mapping of these tags is done later on within the *spaCy* Maltese model as described in Section 3.2.1[1].

Each line will include a token and its Part-of-Speech tag, which is found in Table 6 and its dependency relation. Table 6 shows the POS tags specific to the Maltese language.

<i>ADJ</i>	adjective	<i>ADV</i>	adverb
<i>COMP</i>	complementiser	<i>DEF</i>	definite determiner
<i>CONJ-CORD</i>	coordinating conjunction	<i>FOC</i>	focus part
<i>CONJ-SUB</i>	subordinating conjunction	<i>FUT</i>	future part
<i>GEN-DEF</i>	genitive part w' DEF	<i>GEN</i>	genitive part
<i>GEN-PRON</i>	genitive part w' p-noun	<i>INT</i>	interjection
<i>LIL-PRON</i>	oblique part w' p-noun	<i>HEMM</i>	existential marker
<i>LIL-DEF</i>	oblique part	<i>KIEN</i>	auxiliary
<i>LIL</i>	oblique part	<i>NEG</i>	verbal negator
<i>NOUN</i>	noun	<i>NOUN-PROP</i>	proper noun
<i>NUM-CRD</i>	cardinal numeral	<i>NUM-FRC</i>	fraction
<i>NUM-WHD</i>	the number one (<i>wieħed</i>)	<i>PART-ACT</i>	active part
<i>PART-PASS</i>	passive part	<i>PREP</i>	preposition
<i>PREP-DEF</i>	preposition w' article	<i>PREP-PRON</i>	preposition w' p-noun
<i>PROG</i>	progressive particle	<i>PRON-DEM</i>	demonstrative pronoun
<i>PRON-DEM-DEF</i>	demonstrative p-noun w' article	<i>PRON-INDEF</i>	indefinite p-noun
<i>PRON-INT</i>	interrogative p-noun	<i>PRON-PERS</i>	personal p-noun
<i>PRON-PRES-NEG</i>	negated personal p-noun	<i>QUAN</i>	quantifier
<i>PRON-REC</i>	reciprocal p-noun	<i>PRON-REF</i>	reflexive p-noun
<i>VERB</i>	verb	<i>VERB-PSEU</i>	pseudo verb
<i>X-ABV</i>	abbreviation	<i>X-BOR</i>	bordel
<i>X-DIG</i>	digit	<i>X-ENG</i>	English word
<i>X-FOR</i>	other foreign words	<i>X-PUN</i>	punctuation

Table 6

*POS Tags for MLRS and MUDTv1 *(w' = with; part = particle; p-noun = pronoun) (Gatt & Čéplö, 2013)*

These annotation tags were gathered by Čéplö for his research with the help of the research done by Borg and Azzopardi-Alexander (Azzopardi-Alexander & Borg, 2013) and also the Universal Dependency annotation scheme. The focus of Čéplö's research was mainly on Semantics, Morphology and Syntax. Semantics helped to make the distinction

between Nouns and Verbs; Morphology helped to make the distinction between various types such as GEN and LIL whereby GEN refers to genitive participle example: *ta* ‘of’; and Syntax helped to make the distinction to establish tagging types such as PRON-INDEF. (Čéplö, 2018).

3.2 Design of the Model

3.2.1 Preparing the Pipeline

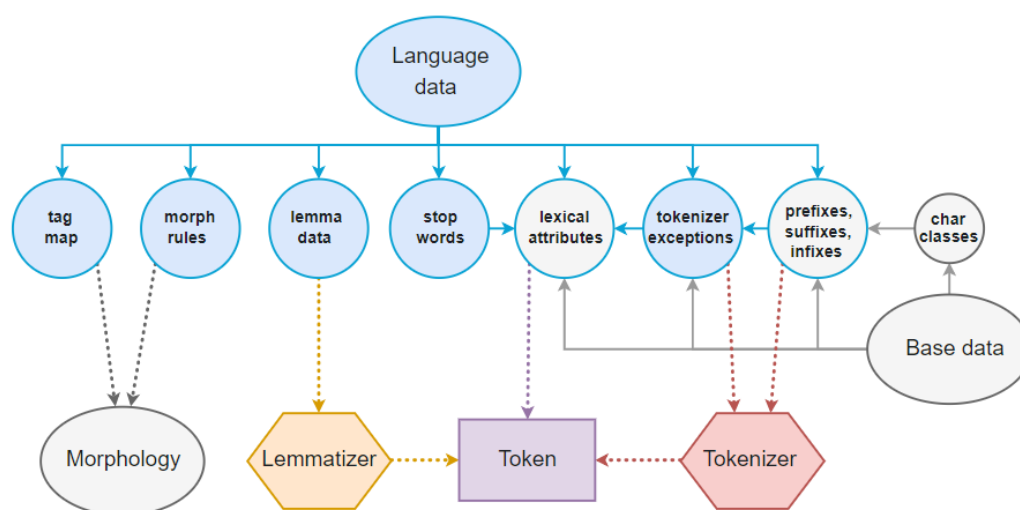


Figure 12

The process needed to build the model (SpaCy, 2018)

As can be seen in Figure 12, there are several components and pre-processing functions required to be able to build this model properly and effectively. These different features have been created for the sole purpose of this research to deal with the exceptions and special cases that are entirely specific to the language. Apart from this the model will have gained knowledge and learned certain traits of the language through training. These features are saved into a *spaCy* directory under an ISO Code for the language which is (*mt*).

Below is a description of the items, that were created for the research to facilitate the pre-processing information required for this Maltese Syntax model and any future use of the model:

1. Tag Map:

This includes the part-of-speech tags that are attached to the treebanks. A tag map, maps the tag of a certain language scheme to its relevant equivalent token in the UD scheme. Therefore for the case of this model, the Maltese language specific tags found in Table 6 were mapped to the Universal Dependency tags detailed in Table 3. The tags understood by *spaCy* are the general ones established by the Universal Dependencies. To get over this obstacle, the characteristics of all the POS tags of the corpus are sorted side by side in the form of a dictionary, created by Čéplö, where the *MUDT* tags are mapped to the recognisable Universal Dependency tags.

An example of the dictionary looks like this:

(16) {"NOUN_PROP": {POS: PROPN},
 "PREP_DEF": {POS: DET},
 "NUM_CRD": {POS: NUM},}

Example (16) defines certain tags that are specific to the Maltese language: where the tag on the left is language specific and is mapped to the tag set by the generic UD annotation.

- The *NOUN_PROP* tag refers to Proper Nouns *PROPN* such as *Daniel*.
- *Det* refers to Determiner, in this case specifically the tag is *Prep_Def* (prepositional determiner) such as *mill-* “from the”. Here the determiner is one word and therefore has one specific tag.
- *NUM_CRD* is a tag used for cardinal numbers that is mapped to the POS that is used for numbers.

2. Stop Words

Figure 12 shows that the model requires a list of stop words which is fed into the lexical attributes element, in order to make up the Maltese language data. This list holds the words that have a high frequency within the corpus. Usually these are words which, while they are vital to carry the ideas of the text might not have any contextual importance, such as “the” or *u* “and”. However, this is not the case

of all common words in a text, for example if a document has the word “*chukker*” as the most common word, this would indicate that the document has a greater probability to be about the sport of polo, and therefore it is a significant word in the text. Due to the latter a stop list might have a negative affect on the output if not filtered since the most frequent words listed in the stop list will be removed from the original corpus and hence the document loses contextual meaning or else phrase searches will return null results (Jurafsky & Martin, 2009, Chapter 19).

Stop lists will be used during the pre-processing of the model after tokenisation takes place. The value behind this list is that insignificant words are removed since with them included in the corpus creates a disorganised frequency count which in turn may cause bloated probabilities of the stop words themselves.

$$TF_{ij} = \frac{f_{ij}}{\max_k f_{kj}} \times IDF_i = \log_2(N/n_i)$$

Figure 13

Equation for TF-IDF (Rajaraman & Ullman, 2011)

The technique that was used to develop the stop list (as this was non-existent) was the term frequency-inverse document (TF-IDF) ranking, which ranks the word according to the number of times it has appeared in the individual documents against its popularity across the corpus. The concept of the technique works in the same way of stop words, however, the ranking given to the prevalent words are low and therefore the words with a lower score hold the highest frequency in the corpus. TF-IDF was chosen as it is a measure of uniqueness and is a better measurement than calculating the density of keywords. The calculation set by TF-IDF can compare the unique or common words across all the documents in the corpus not just in that individual documents itself and therefore gives a more accurate score (Rajaraman & Ullman, 2011, p. 8).

To keep this TF-IDF ranking unbiased from the *MUDTv1* corpus, the *MLRS* corpus was used to calculate this ranking. Using this corpus gave a better probability to how the stop words list of the Maltese language would look. After the calculations,

it returned a list of 240 tokens comprised of punctuation, prepositions (*fuq* “on”), oblique articles (*lill-* “to”), existential marker (*hemm* “there”), determiner (*il-* “the”) and conjunctions (*imma* “but”). This filtering out of the stop words will not be creating ambiguity as the main content of the document is extracted.

3. Lexical Attributes

Indicating the lexical attributes will provide more information on the words that may be found in the corpus. The function will be able to recognise what words need to be uppercase or lowercase. With this lexical rule, the model will identify that a proper noun needs to start with an uppercase character, such as my name “*Greta*” but a noun, such as *mejda* “table” unless found at the beginning of a sentence, needs to be lowercase. It is important to define this distinction as certain languages have diverse features. In return, the model will somewhat have a better understanding of the semantic meaning of the words in the corpus.

4. Tokeniser exceptions

Tokenisation on the *MUDTv1* corpus is performed by splitting the sentences in the corpus at white spaces and ignoring punctuation characters. By getting rid of punctuation, importance and focus is given to the words. However, this may not always be the case, because sometimes punctuation is important to a word for contextual or grammatical purposes. Let’s take the word *ma*, it is used in a negative connotation, example: *ma jidhirx* “does not show”, but *ma’* (with an apostrophe) means “with”. Another example of the importance of punctuation can be seen with these words: *it-tifla* “the girl”; *il-mejda* “the table”. Here determiner “it-” is attached to a noun with the use of a hyphen. Manning, Raghavan, and Schutze (2008) explain that to be able to still tokenise the corpus, a tag is provided to the determiner to hold on to the hyphen.

To achieve this pre-processing process, special-case rules were created for this research where punctuation was treated as affixes between the interaction of the tokeniser and the *spaCy* open library. The tokeniser exceptions is able to have

special cases defined such as in the example below:

(17) Example: “ċ-”: [{ORTH: "ċ-", LEMMA: "ċ-"}]

Whereby the prefixes, suffixes and infixes define the rules to when tokens containing punctuation are to be left intact such as the determiner in Example (17).

5. Norm Exceptions

This is an attribute that ties hand in hand with the tokeniser. It enables the normalisation of words in a model whereby words with dissimilar spellings are set to one common spelling. Therefore, it avoids word ambiguities both in orthography and in punctuation. Taking an example from English the word “*gonna*” will be normalised to “*I am going to*”. Examples such as the latter are not present in the Maltese language, however synonyms were placed here to be normalised. Normalising synonyms was done to make the relation between the words with the same definition as close as possible, example the location “*Vittoriosa*” is equivalent to “*Birgu*”.

The normalisation of characters also occurs with the different styles of quotation marks and currency symbols. By normalising these characters, the model understands that they carry the same definition, and therefore can be based on a standardised symbols.

6. Character Classes

This introduces a character set where the special characters in a language are listed and then string matched with the tokens found in the corpus. The non-Latin characters in Maltese (*ċ, ġ, ħ, għ, ie, ż*) were introduced to the model in both upper and lower case for the model to be able to tell the characters apart and recognise that they are part of the language. Apart from these characters, there are also the duo letters *ie* and *għ*, which are considered individually as one letter. This was set to be part of the for symbols subset which includes units (eg. km),

currency (eg. \$) and diverse quotation marks.

7. Punctuation

This feature is able to recognise which are the punctuation marks that are to be given special attention to. The main punctuation marks, just like in all other languages are found in the Maltese language, however some provide further information to the generic use of that particular punctuation. As detailed above, the - , is attached to an article to form a determiner, i.e. *il-* “the”. This will improve the work performed by tokeniser and therefore provide more information to the character class. It is important to help the tokeniser recognize what are the infixes, suffixes, prefixes. In the example of the determiner, this was introduced to the tokeniser as a prefix to a word and this was completed through string matching with the use of the regex rules. The same concept was applied to words finishing in an apostrophe such as *ma'* “with,” whereby the ' was set as a suffix and therefore kept the punctuation mark as one with the word.

3.3 Training

To be able to handle the Universal Dependency scheme the corpus, created by Čéplö (2018), is in a CoNLL-U format that is able to interpret the data according to the UD Scheme and then convert into JSON format (store syntax in a Javascript object notation) to facilitate it being read by *spaCy*.

During the training of the Maltese syntax model, the model is fed 70% of the training data and 10% development data, as a collection and iterated through all of it 30 times. It created 30 models that finally outputted a final version. This number of iterations was kept at the default value to train the model as it is deemed enough by the *spaCy* open-library package. This amount provided enough information to confirm that the values outputted cannot be improved. Through each iteration, information concerning Parser Loss, Named Entity Recognition Loss (NER), Unlabelled attachment score (UAS), NER Precision on Development data, NER Recall on Development data, NER F1 Measure on

development data, Tag Accuracy on development data and finally tokenisation accuracy on development data was provided. The results will be discussed in Section 4.

3.4 Testing

The model was tested to check whether it gives the expected results. These are that the relation between the tokens are significantly correct and hold the right part-of-speech tag. The model is also fed unseen corpora, to identify it has actually understood the language fed and that it can annotate and produce required results. The model was evaluated twice, with the *MUDTv1* test set and also the *MLRS* corpora.

The *MUDTv1* test set saw the initial evaluation made, the model then outputted the relation between the entities through a visualisation of the dependency parser and scoring values including information about the Labelled Accuracy Score, Unlabelled Accuracy Score, Words per second tested, Total amount of words in the test set, part-of-speech Tagger Accuracy and the time taken to iterate within the test set. The visualisations previously mentioned were exported with the use of another package found within *spaCy*, known as “*displaCy*”. The latter can visualise the annotated results and make linguistic analysis or debugging easier. With the case of the parser visualisation, it will display the tokens, their syntactic dependencies and also the part-of-speech tags of each token.

The second type of evaluation was performed with the manually annotated portion of *MLRS* corpus, where the model was fed the non-annotated tokens that were extracted from the *MLRS* corpus for this research. The *MLRS* corpus only provided the part-of-speech tags as syntactical information and therefore this evaluation could only be performed on the behaviour of the part-of-speech tagger. The *spaCy* model annotated the tokens itself according to how it was trained prior in this case on the *MUDTv1* corpus. The resulting annotations of the tokens was then compared to that of the original corpora. This comparison made analysis and evaluation of the model accuracy performance better and more reliable. This final process was able to identify that the Maltese syntax model is to able make syntactical analysis on a corpus provided.

4 Results

This section will discuss the results and outputs provided from the model after training and testing. Analysis was made on the performance of the model through the evaluation of providing the model unseen text and comparing the *spaCy* annotated corpus to the golden standard corpus.

The results outputted by the model are related to the efficiency of the model which was then used to make statistical tests for evaluation. The model was trained over 30 times during training. This was the default amount of iterations set by *spaCy* for training the model, however this was tested to evaluate why 30 iterations would deliver an optimal model. The value of iterations was changed to test and it was concluded, that the more times the model was trained over the same corpus, the more the model would improve its performance and learn from the data being inputted. For the case of the Maltese syntax model, results remained constant after the twenty fifth (25th) iteration and so the model was in its most optimal form.

4.1 Results from inputting unseen text to the training set

Statistical results were outputted on every iteration determining the optimisation performed during training over the development set. The results provided were the time taken in seconds to perform the phase; parser loss; tag loss; unlabelled attachment score; labelled attachment score; tag percentage accuracy; token accuracy.

Figure 14 shows the values of the parser loss per iteration during training, which is the value of the prediction of the relation between the words. With each iteration the model improved its ranking and in return, the way sentences were being parsed (Collins & Koo, 2005, p.25-70). The values on the x-axis are the number of iterations that occurred during the training phase and the values on the y-axis are the parser loss values. The parser loss value decreased with every iteration made compared to the initial ranking, falling to 5896 errors performed on the 30th iteration. The amount of parsing errors made decreased and thus each subsequent model created with each iteration was better at parsing the corpus.

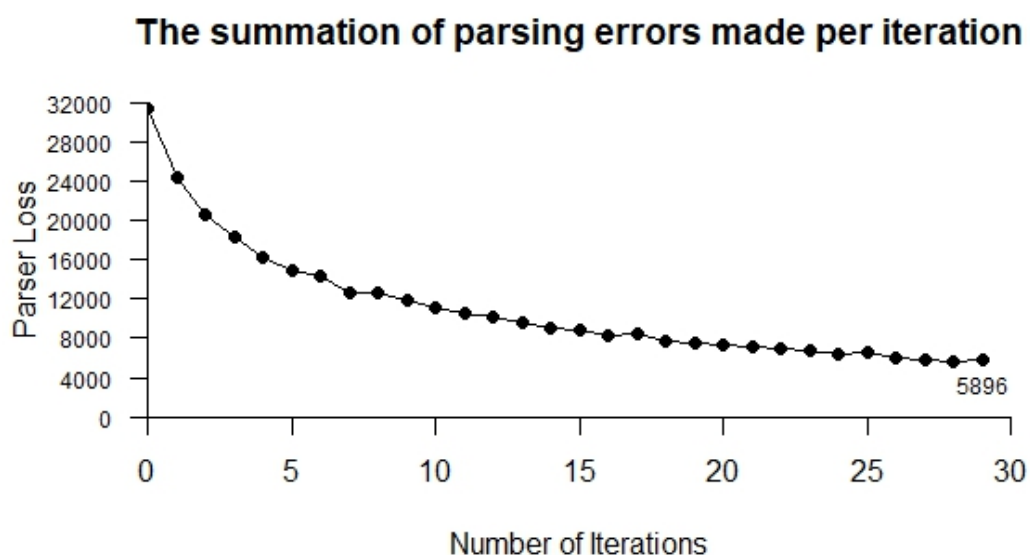


Figure 14

Parser Loss per iteration on the development set during training (Graph made with R)

The tagger loss value, which is the prediction of the relevant tag to be annotated to a certain token, shows the performance of the part-of-speech tagger improving with each subsequent model and the amount of annotated tag errors decreased as shown in Figure 15. The graph shows that from the first model iteration to the next the value of errors decreased, indicating that more words in the corpus were being correctly annotated. By the 30th iteration, the tagger loss value is at its lowest level of 168. This tagger loss value shows the amount of words annotated with the incorrect POS tag during training.

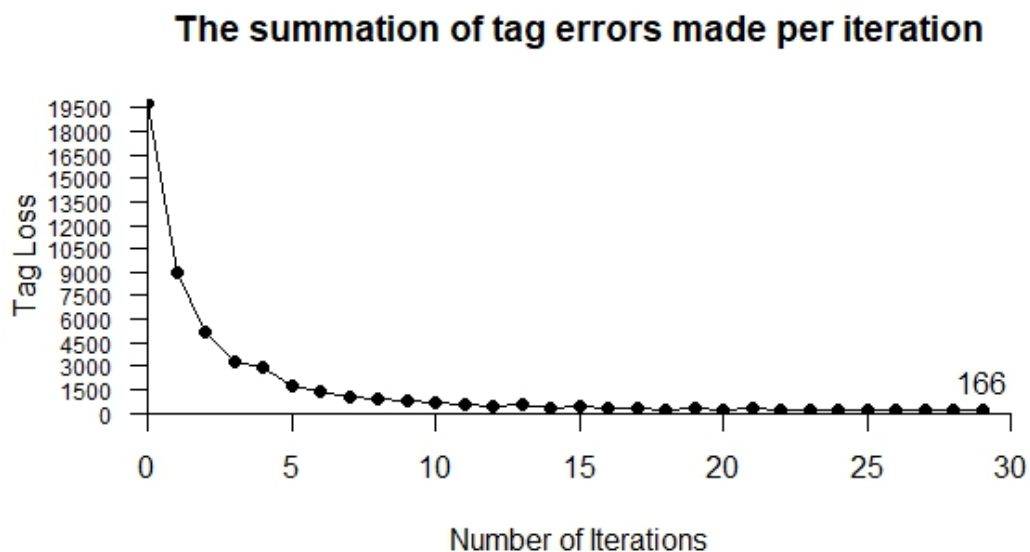


Figure 15
Tag Loss per iteration on the development set during training (Graph made with R)

The unlabelled attachment score (UAS) and labelled attachment score (LAS) are scores that judge how the dependency parser will affect how other Natural Language Processing tools will perform. LAS is the score of how accurate the dependency labels are on the arc whilst UAS is the score of how accurate the relationship between the head of the sentences and the dependency arcs are (Green, 2011). In summary, these scores look at how accurately the dependency parser labels dependency relations.

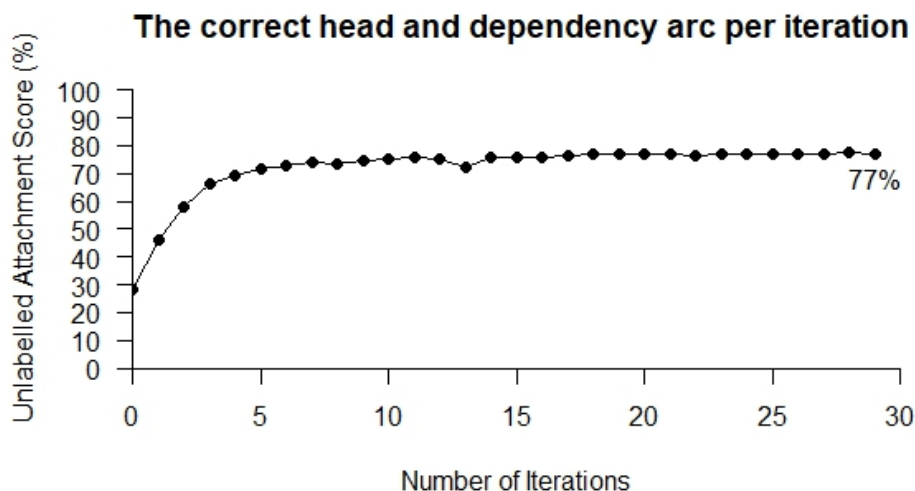


Figure 16
UAS per iteration on the development set during training (Graph made with R)

The unlabelled attachment score (UAS) gives a better judgement of how the dependency parser affects the output given. In Figure 16, the y-axis is the UAS whilst the x-axis are the number of iterations during the training phase. The score gets greater as the number of iterations increase and stabilise thereafter showing that the dependency parser was able to label dependency relations. The UAS score was valued at 77% by the final iteration, showing that with each subsequent model the relationship between the head and the dependency arc became more accurate as is expected. This score will be further evaluated during the testing of the model with the test set.

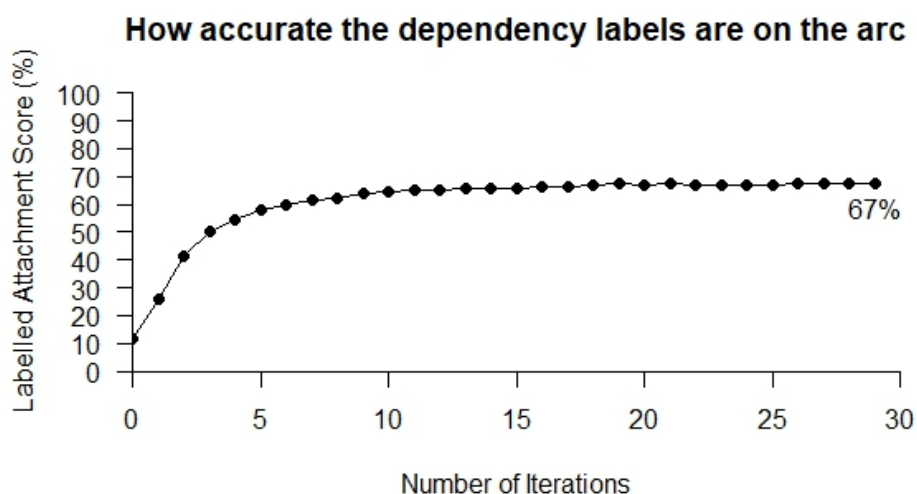


Figure 17

Labelled Attachment Score per iteration on the development set during training (Graph made with R)

Figure 17 shows the LAS obtained during training of the model. On the x-axis is the number of iterations during the training phase whilst the y-axis shows the score of how the subsequent models fared when it come to to annotating the right dependency labels on the arc. The LAS score was 67% during the final model iteration.

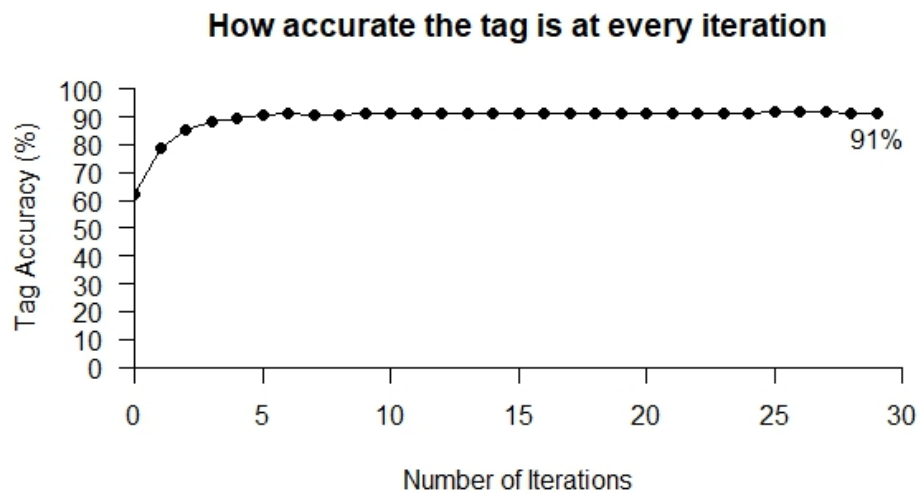


Figure 18

Tag Accuracy Percentage per iteration on the development set during training (Graph made with R)

The graph in Figure 18, shows how accurate the models managed to tag the corpus with each iteration. This value is the percentage of the tags learned by the model, that agree with that found in the corpus (*MuDTv1*). On the y-axis is the values of tag accuracy percentage and on the x-axis is the number of iterations. The average tag accuracy percentage is 91% which means that even though a good number of tokens in the corpus were correctly annotated there was still a 9% error rate.

4.2 Results from test set

Table 7 shows the evaluation results outputted after the testing of the model was completed. The test set was made up of 10,667 tokens, which is 20% of the whole corpus, and it took approximately 2.18 seconds to test this portion of the corpus. When the corpus was tested it had scored 66% as a Labelled Attachment Score and 74% Unlabelled Attachment Score. During testing, 4886 tokens were evaluated per second and the tag accuracy was that of 91%. These results are close and similar to the results outputted for the final model during training.

Words/s	Words per second	4,886
UAS	Unlabelled Attachment Score	74.10
LAS	Labelled Attachment Score	66.01
Words	Total amount of words in test set	10,667
POS	Part-of-speech Tagger	90.64
Time	Time taken to test the test set	2.18s
Training	Words in Training Set	30,914
Development	Words in Development Set	4,416

Table 7

The results outputted after testing the corpus with the trained model

5 Discussion

5.1 Results after training

During the training of the model the UAS was 77%, demonstrating that there is a high level of correct relations predicted between the head of the reference parse and its dependency arc but still in need of improvement. On the same lines of observation the parser loss was evaluated to have decreased during each epoch of training the model. When comparing the last iteration to the first one, as is shown in the line graph (Figure 15), one can see that errors from one epoch to the next decreased drastically. Therefore the model became more efficient as more annotations were correctly chosen. With each iteration the tagger's performance improved too and become more accurate providing a 91% accuracy score at the end of the training of the model.

After testing, the Maltese syntax model, these values kept consistent as the unlabelled attachment score, scored at 74% accuracy during testing and a mean of 77% during training. The part-of-speech tagger gave a 91% accuracy, regarding annotated tags. These values are deemed to be very similar, and there was no distinct contrast between the training and testing of the corpus with the Maltese syntax model. The testing of the model took 2.18 seconds and there were 10,667 words in total.

5.2 Comparing the performance of the different language models vs Maltese language Model

In order to confirm how accurately the model was performing, it was compared to other languages. The language models that the Maltese syntax model was compared to was English and Italian. These languages were selected because historically Maltese was highly influenced by these two languages along with Arabic however the language model for this language was not available in the *spaCy* library. Due to this, all the models were executed on the same library but with different NLP pipeline (according to the respective language). Figure 19 shows the values visually for a better understanding of

the differences and similarities of these models.

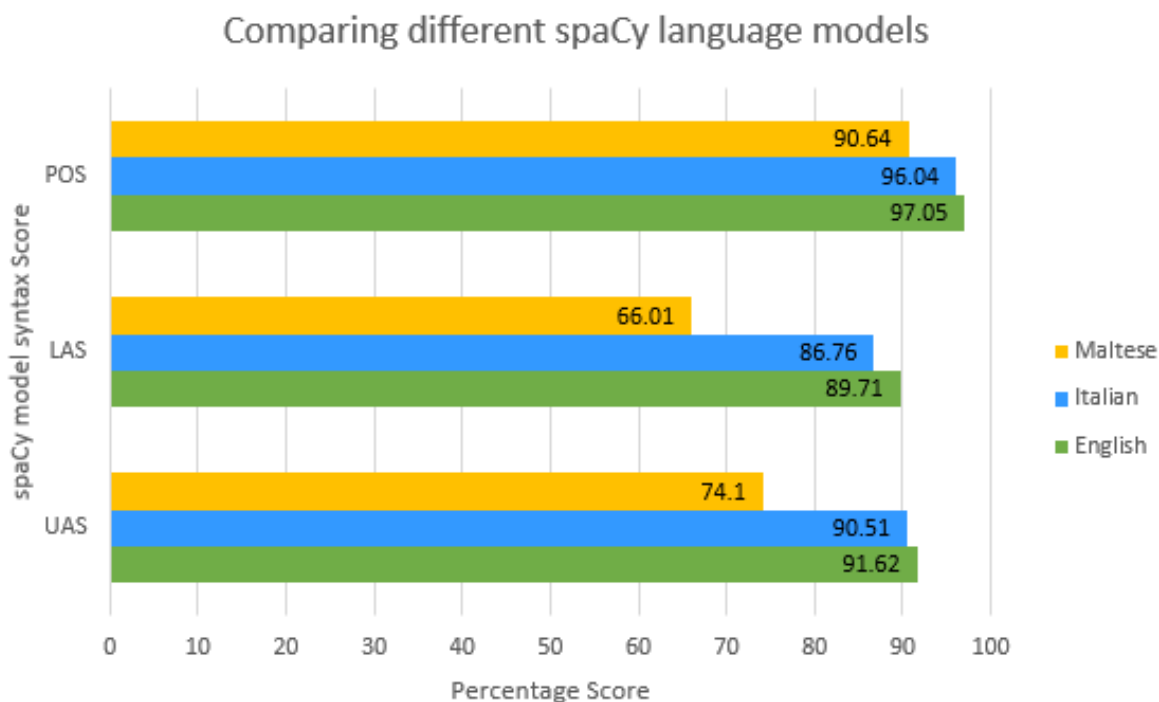


Figure 19

Comparison of the language models with the open-source library spaCy

The part-of-speech accuracy percentage for the Maltese syntax model gave a 91% accuracy whilst the Italian and English models gave 96% and 97% respectively. The latter accuracy scores are the expected values to be produced by a tagger. A tagger is not expected to produce a 100% tag accuracy since this will indicate that there is a high probability of the tagger overfitting and modelling noise, neither of which are desired. When the Maltese model was compared to the Italian and English models, there was a 5-6% difference in percentage accuracy. For the POS tagger, there was also a 9% error rate made by the Maltese syntax model whilst the Italian and English model had a 4% and 3% error rate. This shows that the Maltese syntax model has a lot of room for improvement to be on par with the other language models that are considered as the golden standard since it performed far worse. This might have happened because of the size of the corpus. The *MUDTv1* corpus has 44,162 tokens whilst the size of the Italian corpus and the English corpus are far larger, in that, the models are most likely trained on several of millions of tokens. The Maltese model would have had a better chance to learn more about the

Maltese language and in turn improve these scores, if the training corpus had more data and therefore was larger.

In Figure 19 there is a 15% difference in the scores of the labels. It is evident, that although many of the dependency arcs were labelled correctly, there is still a lot more improvement to be done as other models hold better scores.

Inspired from the work of Kolachina and Ranta (2016) on Universal Dependencies described in Section 2.6 the opportunity was taken to research the UD annotation scheme across different languages as suggested in Nivre et al.(2016) works so as to explore syntax and parsing over different languages and different models. As an example, the following sentence, taken from the *MUDTv1* corpus was used:

- (18) X' inhu l-Unicode?
 what the unicode?
 What (is) the Unicode?

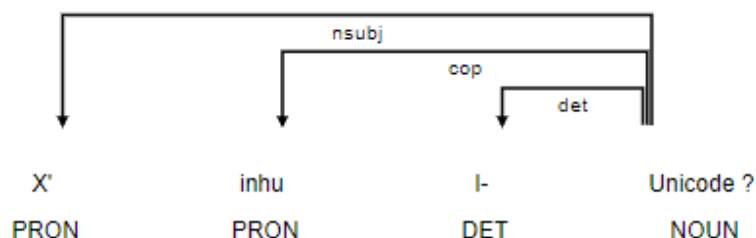
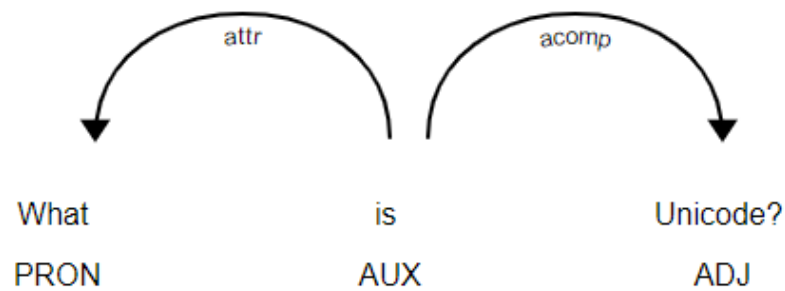


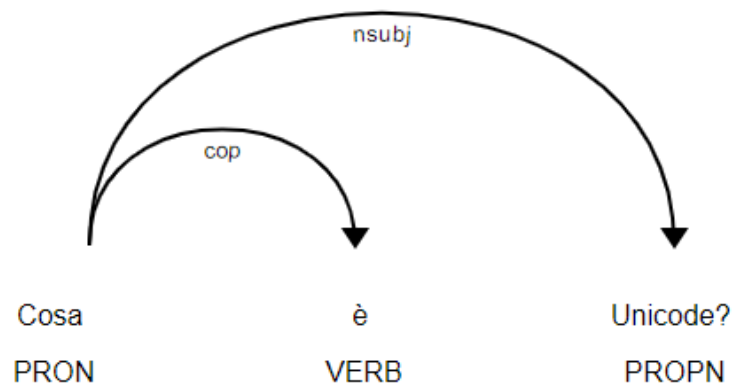
Figure 20

The relation between words in this sentence (produced by “displaCy”)

Taking the same sentence translated and parsed through the respective models for English and Italian, produces the following dependency relations:

**Figure 21**

The relation between words in the sentence (English) (produced by “displaCy”)

**Figure 22**

The relation between words in the sentence (Italian) (produced by “displaCy”)

The relation between the words is diverse across all languages. In Maltese, the root of the sentence was the object, in Italian it was the subject and in English it was the AUX verb. This further proves the importance of having Universal Dependencies in order to naturalise this posed parsing problem. In fact, the dependency relations of the language models that were not Maltese were trained similarly and thus had similar results.

5.3 Maltese model output visualisation

The result outputted by the model shows the establishment of the relationship between the words in the sentence and the root of that sentence. Through the use of a built-in

dependency visualizer called *displaCy* visualisations were created based on 20% of the *MUDTv1* corpus (the test set).

Manual analysis was made on the resulting output, therefore providing a better understanding to the results obtained during training and testing of the corpus. Reference will be made to the results achieved above and the co-relation of the UAS and LAS scores since the visualisations will ultimately reflect these results.

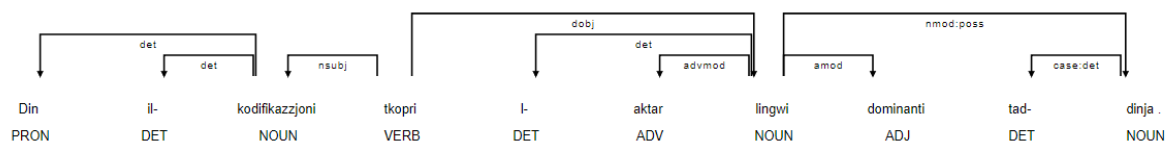


Figure 23

Example of sentence visualisation from the dependency parser

Figure 23 shows what the Dependency Parser Visualisation looks like. The sentence that is being parsed is:

(19) Din il-kodifikazzjoni tkopri l-aktar lingwi dominanti tad-dinja.

this (is -SG) the code -SG cover -2M.SG-PRES the most language -PL dominant of the word -SG

“This is the code (that) covers the most dominant (programming) languages in the world.”

The token tagged as verb in this sentence is (i.e. *tkopri* “covers”), is the root of the sentence since the dependency relation of the rest of the words depend on this specific verb. From there on, the model identifies which is the subject and the object of the sentence.

The subject, tagged as *nsubj*, is the verbal dependent on the noun *kodifikazzjoni* “code” and *dobj* is the verbal dependent on the noun *lingwi* “languages”. The determiner is dependent on the noun, these tagged as determiner and pronoun (referring to: *Din il-[kodifikazzjoni]* (This-SG.F the-code) “this is is the code”). In the object phrase related

to the root, is a phrase demonstrating possession *tad-dinja* “of-the-world” which has the case determiner *tad-* “of-the”. An adverbial relation is found on the noun *lingwi* “languages” with the adverb *aktar* “most”.

From the analysis that Čéplö (2018) performed in his work and also my research, the nominal modifier and adverbial modifier relation is quite commonly found in the Maltese language. It is a relation that is made up of an adverbial with either a prepositional or noun phrase. To explain this further, in the object of the same sentence (19), there is a relation between the nominal dependent and the adjective modifier: i.e. *dominanti* “dominant” is modifying the preceding noun; i.e. *l-aktar lingwi dominanti tad-dinja* (the-most language-PL. dominant of-the world-sg.) “the most dominant languages of the world”.

Further thorough analysis has made evident, and with reason, that when the parser was faced with text from another language, in this case, English text, it did not know how to parse it.

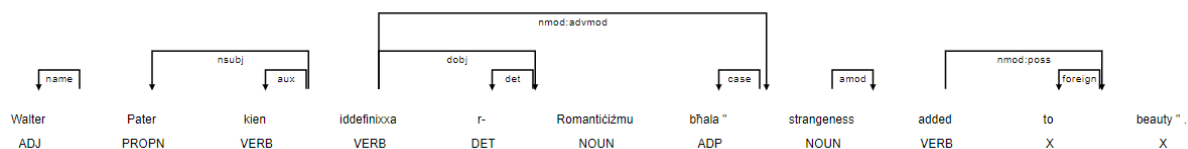


Figure 24

Example of sentence visualisation of Maltese and English, from the dependency parser

(20) Walter Pater kien iddefinixxa r-Romanticizmu bhala “strangeness added to beauty”
 Walter Pater have-3M.PRES define-2M.PST the romanticism as strangeness
 added to beauty.

Walter Pater has defined the romanticism as strangeness added to beauty

Figure 24 demonstrates how this occurs within example (20), whereby the parser performs well with the Maltese text but fails when the English words appear. Both the relations between the words and their heads are incorrect. The English phrase is correctly identified as an adverbial phrase due to the adverb *bhala* “like”. In fact the model tagged some of the

English words with an *X*. Therefore, the Maltese model was able to identify the foreign words to the language it knows. In this example (20), the root of the sentence is the verb *iddefinixxa* “defined” whereby from the noun subject *Walter Pater* was identified. The proper noun *Walter* was tagged as ADJ, however *Pater*, the surname was correctly tagged as a proper noun. Alternatively, the parser identified that *Walter* was contextually related to *Pater* and labelled the relation with name. It is assumed that model identified the Proper Noun and made the understanding that the word before it is describing that proper noun.

When the visualisations were introduced for this analysis, it was evident that *displaCy* did not consider the punctuation. In fact no relation was made or there was no specific tag annotation, but when an end-mark was found a split between the sentences was made. However, when the model was faced with documents that held titles, such as in Figure 25, which was most common in the documents that were news articles, it mixed up the relationship of the sentences in the corpus. The title of the document in the question is *Id-Djalett* “the dialect”.

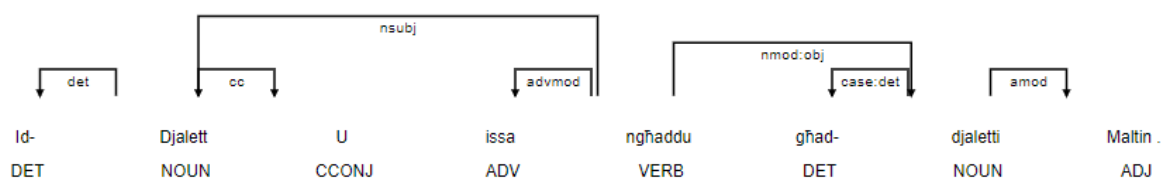


Figure 25

Example of sentence visualisation for punctuation and title

Usually titles are not accompanied by punctuation, for example full stops or question marks, and thus there is no distinct split between the sentences. This affected the relationship between the sentences found within the document following the title. Since the model didn’t find an end-mark it correctly evaluated the title and first sentence as a combined sentence. This can be further confirmed in Figure 26, where at the ending of the sentence, an end-mark was omitted in the original corpus.

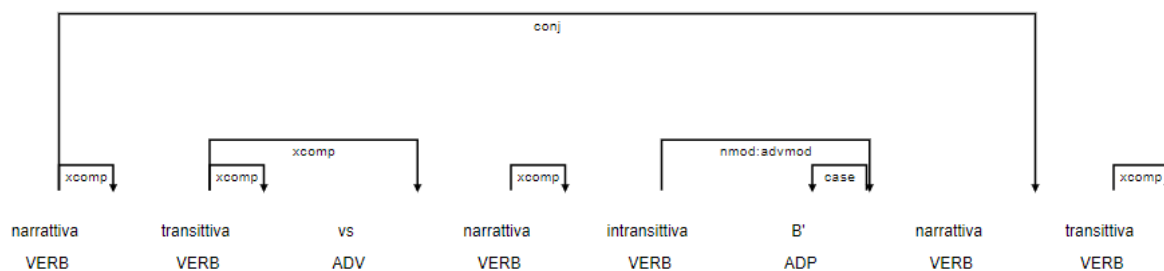


Figure 26

Example of sentence visualisation for omitting an end-mark

(21) narrattiva transittiva vs narrattiva intransittiva B' narrative transittiva nifhmu

narrative transitive-ADJ vs narrative intransitive-ADJ With narrative transitive-ADJ understand-3.M.PL

...transitive narrative vs intransitive narrative With transitive narrative we understand

Contextually, the sentence together do not make sense and neither does the relationship between them. Due to this inconsistency in the punctuation, linguistically it is not logical, since there is a relation between the noun *narrattiva* “narrative” and the same noun in the next sentence. This relation is set as an adverbial relationship between the words which of course does not conform as the word “*narrattiva*” is by no means modifying the word *narrattiva* the second time it appears.

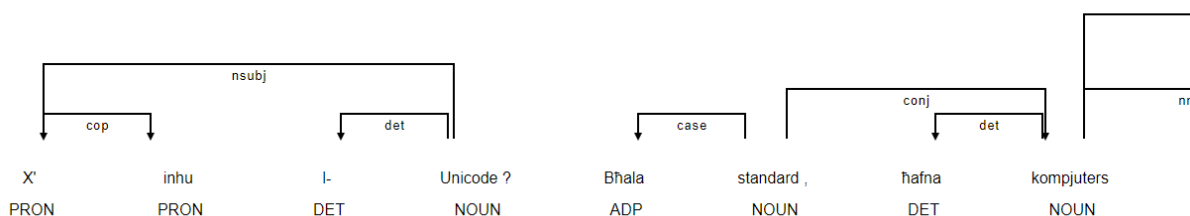


Figure 27

Example of sentence visualisation for end-mark and title

- (22) X'inhu l-Uncode? Bħala standard, ħafna kompjuters
what the unicode? As standard, a lot computers
What (is) the Unicode? As (a) standard, a lot (of) computers

Figure 27 demonstrates this punctuation issue quite clearly since the title “X'inhu l-Uncode? (*What is Unicode?*)” has an end-mark (question mark), now the model considers this title as a sentence on it's own and therefore correctly parses it and does not create a syntax relationship with the following sentence.

To summarise, it would seem that despite the limited corpus size and that the *spaCy* library has never seen a language such as Maltese it fared quite well. The model having made understandable mistakes in the parsing of the sentences, had a POS accuracy of 91% which is reflected in UAS which is 74% and LAS which is 66%. This was further supported whilst looking through the visualisation provided by *displaCy* to try and get an understanding of why such values might be so. It is a positive result of the study that a dependency parser and POS was successfully completed and can be implemented for other future projects or studies.

5.4 Working with unseen text

At this stage, the model has been trained and tested on the same corpus. To make sure that the model actually managed to learn Maltese and was able to parse sentences and tag them, the *MLRS* corpus was utilised for further analysis.

The manually annotated tokens of the *MLRS* corpus (28,000 tokens) had the annotations extracted from the corpus to have just the tokens. This will remove any form of bias for the model. Therefore, the *MLRS* corpus now looks like the way an article would traditionally look like. Once the non-annotated tokens were fed into the corpus, these were then compared to the actual *MLRS* corpus (with annotations). This comparison was made to identify if the model managed to correctly tag the words in the corpus.

When the model was tested on unseen text, the tokeniser made 0.76% mistakes, which

totalled to 214 tokens. Through manual analysis of these mistakes, it was found that the “-” created a great ambiguity. The model was able to identify the tokens that were determiners i.e. *il-* “the”, however certain words such as compound words that included a “-” provided problems. The first part of the compound was incorrectly labelled, for example, *viċi-prim ministru* “vice-prime minister” was split as *viċi-; prim; ministru*, whereas the compound word *viċi-prim* should have been split as word, punctuation, word (*viċi; -; prim ministru*). Part of why this might have happened is because the model is set to recognise determiners such as *għall-arti* “for-art”. In fact, *viċi-* was tagged (by *SpaCy* Maltese syntax model) as PREP_DEF (prepositional determiner) when it should have been an ADJ (adjective) (as tagged in the *MLRS* corpus). The dash also effected other instances, example “2006-2007” whereby the same ambiguity as before occurred. Another similar instance, however with foreign words, was with phrases such as “line-and-cue” and “one-stop”. In a similar manner, the tokeniser dealt with the first part of the phrase, “line-” as if it was a determiner tagged as “LIL-DEF” (a determiner classified as *lil* “to”).

The model is only able to understand Maltese, as determined before, and this remained consistent. Such mistakes made were for example, to show possession (in English) one would add an “ ’s ” at the end of the word like with “SME’s”. The tokeniser split the word after the apostrophe, as it would have done if the token was *ta’* “of”. The incorrectly tokenised “SME’ ” was tagged as a NOUN and an ADJECTIVE in different instances the word appeared in. In the *MLRS* corpus, it was tagged as X_ABV (abbreviation) a tag not known by the *spaCy* model. This same mistake was observed for the word *Imn’Alla* “from God”, which is expression used to say “Thank God”, as in “Thank God I was there to help”. This token was tagged as a INT (interjection) in the *MLRS* corpus. It’s interesting to note that the *spaCy* model, although it split up the word as “*Imn*” and “*Alla*”; it tagged them as PRON_INT (interrogative pronoun) and NOUN_PROP (proper noun) respectively. The model treated the first token, “*Imn*”, as if the sentence was a very specific exclamative statement:

- (23) *Imn'Alla għandha l-festi u s-sajf!*
 from god have-3PL. the-feast-PL. and the-summer-SG.!
 “Thank God we have feasts and summer!”

A similar instance to use an interrogative pronoun would have been,

- (24) *Min kellu jgħidlek li f'certi skejjel fl-Awstralja kellhom jerggħu jdahħlu t-test ta' l-ispelling u l-grammatika!*
 who have-2SG.PST say-2SG.PST-to-you.SG that in-certain school.PL in-Australia
 have-3PL.PST again introduce-3PL.PST) the-test of the-spelling and the-grammar
 “Who would have said that in certain schools in Australia they have to introduce
 again the spelling and grammar test”. (taken from *MLRS* corpus)

spaCy tagged the foreign words “test” and “spelling” as a NOUN. The preceding words were determiners, in this case *t-* and *l-*. Therefore, it is assumed, that the model tagged the words as NOUN (correct in English) because of the nature of the words surrounding it.

An interesting observation was how, the *spaCy* model, tokenised three full stops next to each other (ellipses) as one thing and tagged them as X_PUN (punctuation). When comparing this to the *MLRS* corpus these full stops were annotated separately.

Most of the comparative evaluation was done on the POS tags. From a total of 28,000 tokens, 7,099 were incorrectly tagged. This means that 75% of the *MLRS* corpus was correctly annotated by the *spaCy* model with the right tags, which were identical to the tags found in the *MLRS* manually annotated corpus.

Figure 28 shows the amount of words incorrectly tagged when compared to the *MLRS* corpus. Taking the tag “NOUN”, this means that 1752 words were incorrectly tagged as “NOUN” when these tokens needed to be tagged with another POS tag. Such instances that certain words were mistakenly annotated as “NOUN” are for example *Nippruvahom*

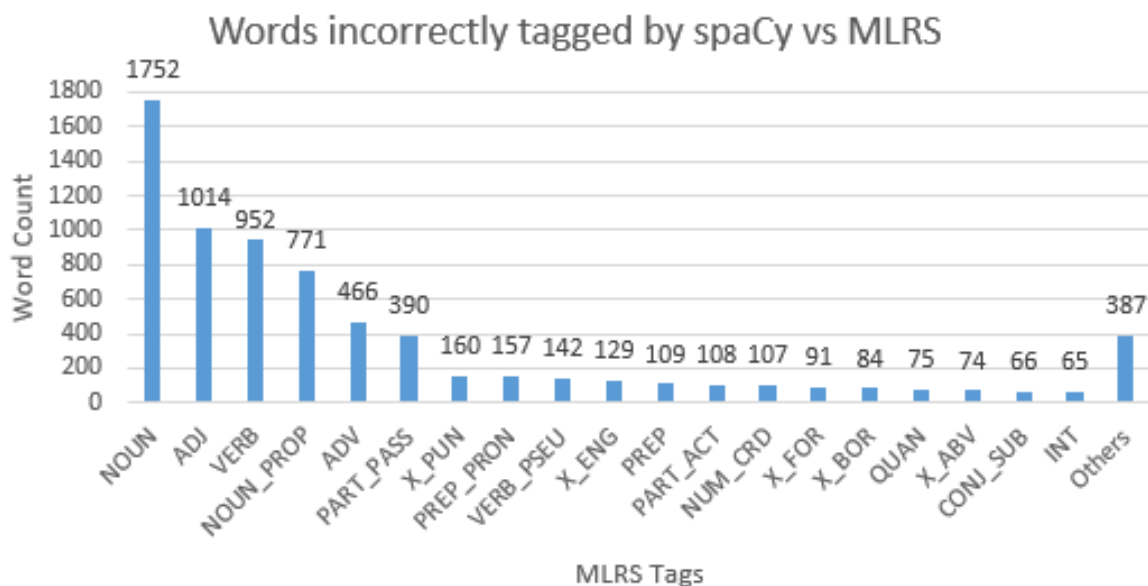


Figure 28
Comparing the performance of spaCy and MLRS

“we try them” which is a VERB and *għaqlija* “responsible” which is an ADJ (adjective). The latter might have been annotated as a “NOUN” because within the same document the words that finished with *-ija* were correctly annotated as “NOUN” (*tmexxija* “ruling”; *trobija* “upbringing”).

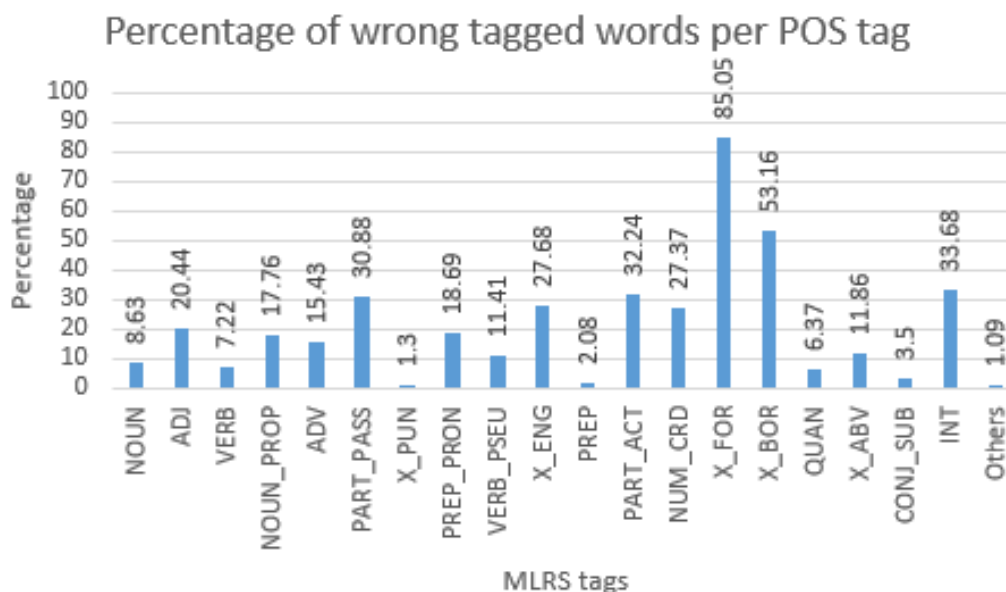


Figure 29
Comparing the performance of spaCy and MLRS

Figure 29 shows the percentage of all the incorrectly tagged words of a specific POS tag over the whole corpus. Therefore, even though from Figure 28, it seems that the most incorrect tagged words are “NOUN” (1752 words), this amount only makes up for 9% of the total amount of words tagged as “NOUN” by the Maltese spaCy model. In Figure 29, the category with the most incorrectly tagged words is X_FOR (foreign words) which makes up 85% of the amount of words in this annotation. However as can be seen from Figure 28 this only amounts to 91 tokens which is 1% of the total amount of incorrectly tagged words. When looking through the words that should have been tagged as X_FOR (words in Italian or English or even Latin and French), the model tried to annotate these words as if they were part of the Maltese corpus. Phrases such as “*punto e basta* (and that’s that) (Italian)” was tagged as PART_PASS (part participle); X_DIG (digit) and ADJ (adjective) respectively.

Unfortunately, the *MLRS* corpus does not provide any information about the dependency relation of the words. Therefore, the results outputted by the *spaCy* model could not be compared to the *MLRS* corpus.

During testing of the *MUDTv1* the tagger had 91% accuracy. When comparing these values to the 75% tag accuracy that is produced through testing the model on unseen non-annotated text there is a 16 percentage point difference in these results. This difference is assumed because the model has not encountered with the words found in the *MLRS* manually annotated corpus before and thus the error in accuracy increases. During the testing of the model on the unseen non-annotated text the tokeniser percentage accuracy was 99%.

5.5 Conclusion

After manually analysing the model and looking at the results outputted during training and testing, it can be concluded that the a syntax Maltese model has been created. Although both the part-of-speech tagger and the dependency parser fared well with a 91% tag accuracy and a 66% LAS; 74% UAS; there is still room for improvement and

testing to be done in future work. With the limited data currently available disposition for researchers, these results are sufficient as they are similar to other languages that were trained on the *spaCy* platform. This model was also tested on unseen text that was not annotated and produced a 75% tag accuracy and a 99% tokeniser accuracy.

6 Conclusion

The goal of this study was to broaden the digital resources for computational linguistic research on Maltese since there was a gap in the NLP Model for Maltese Syntax, which is syntactical research. The opportunity was taken to produce this using a library that is widely popular and provide research to enhance the digital representation of the Maltese language.

The model was created with the use of an open-source library *spaCy* and research was performed to be able to teach the library Maltese by including features, such as the unique Maltese characters, the diverse stop words and how to tokenise the corpus. *MuDTv1* was made up of CONLL-U files that are annotated with the Universal Dependency tags for Maltese and included the dependency relations. The model was tested on annotated and non-annotated corpora to better determine the performance of the Maltese syntax model.

The results show that the syntax model has a POS Tag Accuracy of 91%, an Unlabelled Attachment Score of 74% and a Labelled Attachment Score of 66%. Considering that the corpus is significantly small when compared to other languages, the model managed to get high scores. When the model was manually analysed through the visualisation provided by the library, it made further evaluation more accurate since the values are supported through linguistic analysis. In fact, the visualisation of the test set provided a more coherent idea of the performance of the model and this supports the annotated results.

Furthermore, unseen corpora were fed to the model to be able to evaluate the model further, and examine the accuracy of the machine learning. The POS Tag Accuracy was that of 75% and the tokeniser accuracy was that of 99%.

Future Work

The current model might be further enhanced through increasing the data in the corpus to give the model more data to learn from. The primary intention was to make use of the manually annotated data provided by the *MLRS* Corpus, which was described in Section

4 along with the *MuDTv1* corpus and thus creating a larger corpus. However, due to inaccurate file formats and omitting required information from the *MLRS* corpus as well as due to time constraints, adapting this corpus was not possible. There is a possibility that with more data, the above scores might have been improved and thus the model would have made more accurate tag annotation and sentence parsing.

The work on my thesis focused on Maltese Syntax, but maybe in time all of the linguistic features could be combined, these include, Morphology, Semantics and Phonology. This will therefore provide a more holistic view of the language and be able to do more research once the full Maltese model is complete on the *spaCy* platform.

A complete model with all the linguistic features could be used for many use cases, such as chatbot or voice activated assistants. A possible further avenue for the future work of this research is to make use of *RASA*¹⁷, which is a machine learning tool made for developers, in order to improve chatbots to more than just answering simple questions, but to be closer to human behaviour, be it text or voice enabled. Through this, the work will be delving into Natural Language Understanding and giving a chance to have a realtime automatic chatbot in Maltese. It can also be a stepping stone for the creation of a speech recognition system or even a spell checker.

The contribution that this research has provided to the field is that it is using an open source library, *spaCy* that is becoming increasingly popular in the field of NLP and therefore will be able to be implemented for a wide range of projects in the future. This work involved significant research to enhance the digital presence of the Maltese language. When using this research, projects making use of Natural Language Processing technology would be able to support this language and provide native speakers the accessibility of using it.

¹⁷ <https://rasa.com/>

7 References

- Azzopardi-Alexander, M., & Borg, A. (2013). *Maltese*. Routledge.
- Bickel, B., Comrie, B., & Haspelmath, M. (2008). The Leipzig glossing rules. conventions for interlinear morpheme by morpheme glosses. *Revised version of February*.
- Camilleri, J. J. (2013). *A computational grammar and lexicon for Maltese* (Master's thesis, Chalmers University of Technology, Gothenburg, Sweden). Retrieved from <https://odr.chalmers.se/bitstream/20.500.12380/185320/1/185320.pdf>
- Carnie, A. (2013). *Syntax: A generative introduction*. John Wiley & Sons.
- Čéplö, S. (2018). *Constituent order in Maltese: A quantitative analysis* (Doctoral dissertation, Charles University, Prague, Czech Republic). Retrieved from https://bulbul.sk/phd/Text/Slavomir_Ceplo-autoreferat.pdf
- Chomsky, N., & Lightfoot, D. (2002). *Syntactic structures*. Mouton de Gruyter. Retrieved from <https://books.google.com.mt/books?id=SNeHkMXHcd8C>
- Collins, M., & Koo, T. (2005). Discriminative reranking for natural language parsing. *Computational Linguistics*, 31(1), 25–70.
- Comrie, B. (1989). *Language universals and linguistic typology: Syntax and morphology*. University of Chicago press.
- De Marneffe, M.-C., Dozat, T., Silveira, N., Haverinen, K., Ginter, F., Nivre, J., & Manning, C. D. (2014). Universal Stanford dependencies: A cross-linguistic typology. In *Lrec* (Vol. 14, pp. 4585–4592).
- Fabri, R. (2009). Stem allomorphy in the Maltese verb. *Ilsienna—Our Language*, 1(2009), 1–20.
- Gamallo Otero, P. (2008). The meaning of syntactic dependencies. *Bop.unibe.ch, Bd. 35, Nr. 3*. Retrieved from bop.unibe.ch/linguistik-online/article/view/522/872
- Gatt, A., & Čéplö, S. (2013). Digital corpora and other electronic resources for Maltese. *Corpus Linguistics 2013*, –96.
- Green, N. (2011). Dependency parsing. *WDS 2011 Proceedings of Contributed Papers*,

137–142.

- Grefenstette, G. (1999). Tokenization. In H. van Halteren (Ed.), *Syntactic wordclass tagging* (pp. 117–133). Dordrecht: Springer Netherlands. Retrieved from https://doi.org/10.1007/978-94-015-9273-4_9 doi: 10.1007/978-94-015-9273-4_9
- Hartmann, R. R. K., & Stork, F. C. (1972). *Dictionary of language and linguistics*. Applied Science Publishers.
- Julien, M. (2007). On the relation between morphology and syntax. *The Oxford handbook of linguistic interfaces*, 209–238.
- Jurafsky, & Martin, D., James H. (2009). *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Pearson/Prentice Hall.
- Kapust, W. H. (1998). Universality in noun classification. *San Jose State University*.
- Kolachina, P., & Ranta, A. (2016). From abstract syntax to universal dependencies. *LiLT (Linguistic Issues in Language Technology)*, 13.
- Kral, P. (2015). *Lexical information, syntax and semantics for natural language processing* (Doctoral dissertation). Retrieved from <https://home.zcu.cz/~pkral/>
- Kroeger, P. R. (2005). *Analyzing grammar: An introduction*. Cambridge University Press.
- Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval* (Vol. 2). Cambridge University Press.
- Mitkov, R. (2003). *The Oxford handbook of computational linguistics*.
- Nivre, J., De Marneffe, M.C., Ginter, F., Goldberg, Y., Hajic, J., Manning, C.D., McDonald, R., Petrov, S., Pyysalo, S., Silveira, N., Tsarfaty, R., & Zeman, D. (2016). Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the tenth international conference on language resources and evaluation (LREC'16)* (pp. 1659–1666).
- Osborne, T. J. (2018). Tests for constituents: What they really reveal about the nature

- of syntactic structure. *Language Under Discussion*, 5(1), 1–41.
- Rajaraman, A., & Ullman, J. (2011). *Mining of massive datasets*. Cambridge University Press.
- Ravishankar, V., Tyers, F. M., & Gatt, A. (2017). A morphological analyser for Maltese. *Procedia Computer Science*, 117, 175–182.
- Sarbo, J. J., Farkas, J. I., & van Breemen, A. J. J. (2011). Text summarization. In *Knowledge in formation: A computational theory of interpretation* (pp. 151–164). Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from https://doi.org/10.1007/978-3-642-17089-8_9 doi: 10.1007/978-3-642-17089-8_9
- Sinha, P. (2010). Speech recognition. In *Speech processing in embedded systems* (pp. 143–155). Boston, MA: Springer US. Retrieved from https://doi.org/10.1007/978-0-387-75581-6_10 doi: 10.1007/978-0-387-75581-6_10
- Spacy*. (2018). Retrieved from <https://spacy.io/usage/adding-languages>
- Spiteri, A. (1998). *Lingwa u lingwistika*. Klabb Kotba Maltin.
- Vaux, B., & Cooper, J. (1999). *Introduction to linguistic field methods* (Vol. 1). Lincom Europa.
- Zhang, L., & Sun, J.-T. (2009). Text generation. In L. LIU & M. T. ÖZSU (Eds.), *Encyclopedia of database systems* (pp. 3048–3051). Boston, MA: Springer US. Retrieved from https://doi.org/10.1007/978-0-387-39940-9_416 doi: 10.1007/978-0-387-39940-9_416