

Tagging and Parsing Old Texts with New Techniques

Exploring Transformer Architectures for Ancient Greek

Daniel Clemeth

Informatics: Language Technology
60 credits

Department of Informatics
The Faculty of Mathematics and Natural Sciences

Spring 2022



Copyright, 2022, by the author.
The text face is Minion Pro by Robert Slimbach.
Ornamental illustrations by John Flaxman.

University of Oslo. Norway.

TAGGING AND PARSING
OLD TEXTS WITH
NEW TECHNIQUES

Exploring Transformer Architectures for Ancient Greek

BY
DANIEL CLEMETH

SUPERVISORS:
JAN TORE LØNNING
DAG HAUG

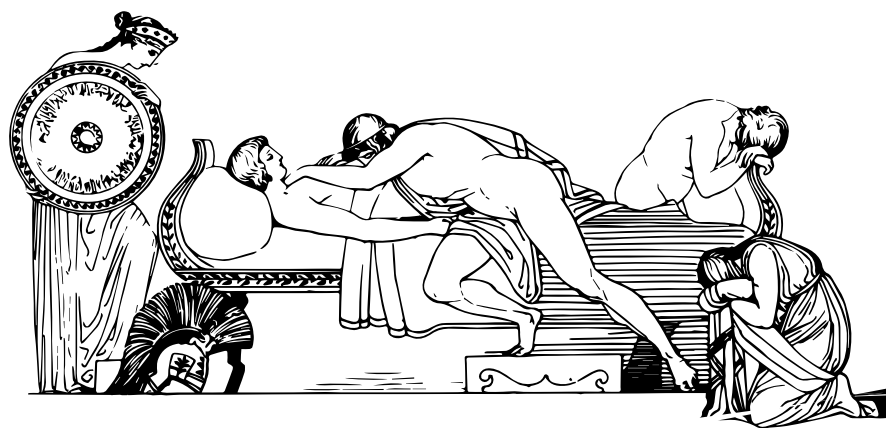
MMXXII



ABSTRACT

Despite the fact that substantial, language-specific treebanks have been available for many years, the best-performing parsers consistently have to relegate the Ancient Greek collections to the bottom third when it comes to parsing accuracy (such as in the CONLL shared tasks). With recent developments such as the release of a monolingual BERT model for Ancient Greek, we see the potential to push the performance numbers up by combining contextual word embeddings with a number of high-performing strategies such as biaffine attention, joint modeling, and language-specific enhancements. We look at both dependency parsing and grammatical tagging. We evaluate on several datasets and show that our approach is successful, achieving state-of-the-art results on all metrics and datasets that we test for. We further probe the implications of our findings by identifying specific linguistic traits that we have been able to accommodate through the various methods, and contextualize the results with regard to multi-task learning and domain adaption. Specifically, we show that domain-specific training is crucial for performance on certain datasets, and that diacritics are essential to tagging accuracy in Ancient Greek in general, and propose a novel way of incorporating them into tokenizable text. We release our code and training details for reproducibility.¹

¹Our code is available in Appendix A.



ACKNOWLEDGEMENTS

I owe my unmitigated gratitude to Prof. Jan Tore Lønning and Prof. Dag Haug for supervising this thesis. Without your broad insights spanning everything from papyrological minutiae to the latest in BERTology, it would not have had a fraction of the direction, precision, or scope required to make it compelling. I have also been impressed by your ceaseless patience, your clear-cut advice, and your willingness to entertain even my most far-fetched statements over the past semesters. I must also extend my sincere thanks to the talented people in the Language Technology Group, particularly David Samuel, for helping me out when I had technical challenges. I am especially grateful to Dr. Alek Keersmaekers for all the pioneering work he has already done on this topic, and for providing me with his datasets. I am indebted to Sondre Wold for offering his unyielding encouragement and for sparring with me during our many late evenings working on our respective theses—it has been a pleasure. I must also thank Luca Lukas for being just as supportive and for providing many helpful judgements regarding linguistic topics. Lastly, but not anywhere near the bottom, I must praise the members of Λύκειον: Knut Olav Nordstoga Sandvik, for answering every question I have ever had regarding Ancient Greek grammar; Sverre Hertzberg, for inspiring me with philosophical perspectives on all of life's facets; and Diærv Svermer, for being a dear friend, guiding me every step of my way, and for proofreading.



CONTENTS

ABSTRACT	iii
ACKNOWLEDGEMENTS	v
CONTENTS	vii
LIST OF FIGURES	ix
LIST OF TABLES	xi
1 INTRODUCTION	1
1.1 Motivation	2
1.2 Outline	3
1.3 Formal remarks	4
2 BACKGROUND	5
2.1 Greek	5
2.2 Tagging	6
2.3 Parsing	9
2.4 Inflectional languages	12
2.5 Corpora	14
2.6 Related work	16
3 EXPERIMENTAL SETUP	21
3.1 Data	21
3.2 Transformer-based models	26
3.3 Tagger	27
3.4 Parser	28
3.5 Joint model	30
	vii

4	RESULTS & DISCUSSION	33
4.1	Benchmarks	33
4.2	Fine-tuning	36
4.3	Multi-task learning	41
4.4	Domain adaption	44
4.5	Error analysis	47
4.6	Comparison	48
5	CONCLUSION	51
5.1	Summary of contributions	51
5.2	Future work	52
6	BIBLIOGRAPHY	57
A	CODE & REPRODUCIBILITY	65
A.1	Fine-tuning	65
A.2	<i>Trankit</i>	66
B	ADDITIONAL FIGURES	67

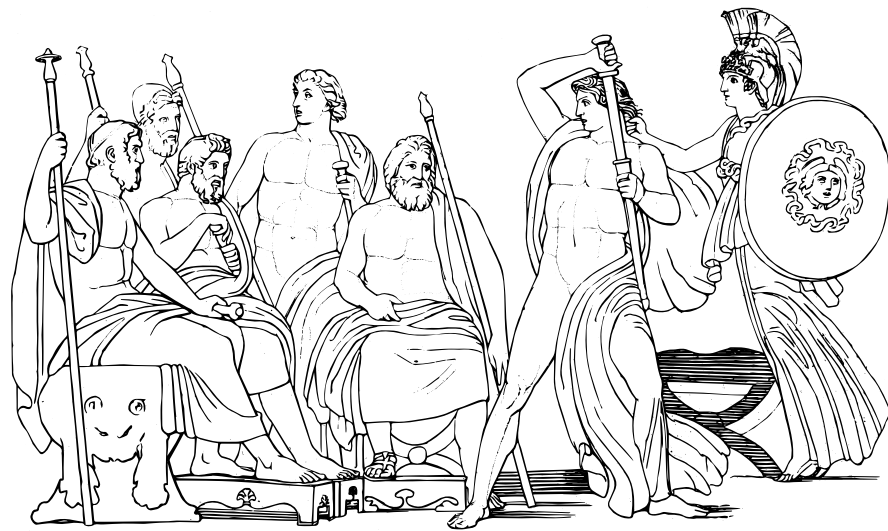
LIST OF FIGURES

2.1	Syntactic tree of the first line of Shakespeare’s Sonnet 18 according to the UD scheme, with UPOS tags	10
2.2	Syntactic analysis of Thucydides’ <i>Histories</i> 1.143.5 as annotated in AGDT	17
2.3	Syntactic analysis of Thucydides’ <i>Histories</i> 1.143.5 after conversion .	18
3.1	Model architecture for the joint tagger and parser	31
4.1	Validation loss values of the AGBERT-based models that reached the lowest validation loss on the papyrus dataset using different diacritic-expansion strategies	38
4.2	Confusion matrices of the case predictions by the best performing taggers on the papyrus dataset	39
4.3	Confusion matrices of the POS predictions by the best performing AGBERT-based taggers with and without expanded iotas on the papyrus dataset	40
4.4	Confusion matrices of the POS predictions by the best performing AGBERT-based taggers with expanded rough breathing marks and both strategies on the papyrus dataset	41
4.5	Charts comparing the accuracies of the joint models on all datasets	46
B.1	Confusion matrix of the XPOS predictions by the best performing AGBERT-based tagger with no diacritic expansion on the papyrus dataset	68
B.2	Confusion matrix of the XPOS predictions by the best performing AGBERT-based tagger with expanded iotas on the papyrus dataset .	69
B.3	Confusion matrix of the XPOS predictions by the best performing AGBERT-based tagger with expanded rough breathing marks on the papyrus dataset	70


B.4	Confusion matrix of the xpos predictions by the best performing AGBERT-based tagger with both diacritic expansion strategies on the papyrus dataset	71
B.5	Confusion matrix of the <i>deprel</i> predictions by the AGBTERT-based parser on PROIEL	72
B.6	Confusion matrix of the <i>deprel</i> predictions by the AGBTERT-based joint model on PROIEL	73

LIST OF TABLES

3.1	Overview of the experimental datasets and their split and tagset sizes	23
3.2	Beta Code equivalents of Greek Unicode characters	24
4.1	Accuracies of the parsing experiments by Keersmaekers (2020) on the papyrus dataset	35
4.2	Accuracies of <i>Stanza</i> and <i>Trankit</i> on PROIEL (UD)	35
4.3	Accuracies of <i>Stanza</i> and <i>Trankit</i> on AGDT (UD)	36
4.4	Optimized discriminative learning rates for the tagger, parser and joint model	37
4.5	Accuracies of the AGBERT-based models on the papyrus dataset with different diacritic-expansion strategies	38
4.6	Accuracies of the AGBERT-based models on the papyrus dataset . .	42
4.7	Accuracies of the AGBERT-based models on PROIEL (UD)	42
4.8	Accuracies of the AGBERT-based models on the AGDT (UD)	42
4.9	Number of correct <i>deprel</i> predictions per relation type by the AGBERT-based parser and joint model on PROIEL	43
4.10	Accuracies of the joint models on the papyrus dataset	45
4.11	Accuracies of the joint models on PROIEL (UD)	45
4.12	Accuracies of the joint models on AGDT (UD)	45
4.13	Our results on the papyrus dataset compared to Keersmaekers (2020)	49
4.14	Our results on PROIEL (UD) compared to <i>Stanza</i> and <i>Trankit</i>	49
4.15	Our results on AGDT (UD) compared to <i>Stanza</i> and <i>Trankit</i>	49
A.1	Configuration settings and hyperparameter spaces for our experiments	65



1 INTRODUCTION

ATURAL LANGUAGE PROCESSING, like most human endeavours, is not a fully unified field as of yet. While there exist substantial efforts to build more and more massively multilingual models that generalize across languages, headway is being made gradually, and when it comes to those languages for which the existing technologies are not yet that great, the wait can be too much, too soon. After all, there is still progress to be made in single-language applications. This is particularly true for smaller languages where researchers have restricted access to resources, and not to mention, fewer teammates. The big dogs have a head start, while the also-rans are still trying to catch up.

For historical languages, the picture is dimmer in some respects, and brighter in others. It can be hard to convince a research community, whether it be linguists, historians, or engineers, to pay attention to one's own little corner of the Earth. Indeed, the situation is especially dire for those languages which are, in fact, in use by small communities today, but which nonetheless are restricted from partaking in recent developments in language technology, simply because no one has the motivation, knowledge, or opportunity to do the work on their behalf. So-called *digital classicists* are lucky, however, in that there seems to be broad sympathy for the importance of preserving historical languages, and are, as such, often given a considerable amount of attention. Furthermore, classical languages are not newcomers on the digital scene; rather, the first digitized collection of Latin church texts saw its inception already in the 1940s—the *Index Thomisticus*.

This thesis is about grammatical tagging and syntactic parsing of Ancient Greek. It was conceived with one goal in mind: to improve on past achievements. If the reader has either read the thesis subtitle, or attentively noted the Roman numerals on the title page, it should already be clear that we will be working with transformer models. Our main perspective is: What makes Greek unique? That is, if we cannot solve these tasks for every language at once, what can we do to tailor existing solutions to our current needs?

These questions bring us to our main motivation, namely, the reason for why tagging and parsing is so important for this language in particular. As implied, Ancient Greek is a historical language, which means that it has no native speakers

that are still alive. This further entails that the only access we have to it is through extant original fragments and passed-down literature. In order to understand as much about it as possible, we therefore need to make use of everything that we have. One way in which we can contribute to this, is by way of treebanks. Through the systematic annotation of ancient texts, we can perform quantitative experiments to test our hypotheses regarding the natures of the languages of the past.¹ Automated morphological and syntactic analyses allow us to annotate texts much faster, either directly and fully automated, or as an assistive tool for human annotators.

This work is aimed at two audiences. Ostensibly, it is written for computational linguists; we presuppose knowledge about transformer models, and do not spend much time on explaining most technical aspects. On the other hand, we often operate with examples in Ancient Greek, and try to keep a broad perspective going from one part to the next. We have to navigate through a thick and unpredictable terrain of various data formats, linguistic peculiarities, and numeric tables, but all in all, we have tried our best at making it as traversable as possible. Our hope is that our second audience, the philologists and language researchers who wish to do quantitative research on Ancient Greek or similar languages, or otherwise have an interest in these topics, can also use this work as a sort of map to the current state of affairs, or even, make use of the tools that have come out of it.²

In the rest of this chapter, we will provide further motivation for our topic, and present an outline of the thesis structure. Finally, we make some formal remarks before we are ready to move on to the background chapter.

1.1 MOTIVATION

Before we go any further, we would like to highlight some of the aspects that separate languages like Ancient Greek from most other languages. The argument is quite straightforward: Historical languages are culturally significant, typologically interesting, and computationally challenging. We will now try to elaborate on these three claims.

Cultural significance. As we have already alluded to, Ancient Greek and classical languages in general attract our attention for a number of reasons. First and foremost, they are an important part of our cultural heritage, and the works that they have provided the words for continue to be a source of inspiration for all sorts of crowds. They offer unique insight into our past, and display a number of distinct qualities that make them worthy of preservation. Among the works that are written in pre-

¹For more on treebanks and linguistic research, we refer to Haug (2015) and Taylor (2020).

²Our code is available in Appendix A.

modern Greek we find the New Testament, Plato’s dialogues, Thucydides’ account of the Peloponnesian War, and Homer’s epics—to name only a very small subset—and all of these, and more, are part of our experimental data.

Typological interest. Ancient Greek is a language that is almost 3,000 years old, and that has close connections with the common ancestor of all Indo-European languages. It exhibits several now mostly lost linguistic qualities within its language family such as highly free word order, complex morphology, and a fascinating aspectual system. Furthermore, it is one of the oldest written languages, and thereby comprises one of the longest continuous textual traditions on Earth.

Computational challenges. The corpora we have to work with contain texts that stem from widely different locations and periods in history, and furthermore display significant dialectal diversity. As such, they present a valuable opportunity to examine language models in light of domain adaption and language change. Given the amount of available data, we are also faced with similar challenges as other low-resource languages, having to look to ideas such as transfer learning and language-specific modeling in order to make progress.

With these points made, we believe that our honorable intentions have been made clear and that we are ready to move on to the main part. But first, some more remarks regarding our overall plan. Our aim is, as mentioned, to improve results for tagging and parsing. Our expectation is that we will be able to do so by making use of language-specific transformer models combined with high-performing task-specific architectures. We wish to investigate how such systems compare to other, less graecocentric approaches, and evaluate our proposed solutions against them. We aim to highlight the challenges we face, and hand over ample suggestions for future researchers based on our experiences.

1.2 OUTLINE

CHAPTER 1 is this introduction.

CHAPTER 2 is the background chapter, where we disclose more details about the Greek language, make the case for the tasks of tagging and parsing, and present our angle of inquiry and our hypotheses for improvement. We subsequently provide an overview of the available data, and summarize relevant work on the topic.

CHAPTER 3 describes our experimental setup. We present our datasets, and introduce our model architectures. We also motivate and explain our additional ideas and techniques which are to be part of the experimentation.

CHAPTER 4 is where we present the results of our experimentation. We evaluate how the different approaches affect final performance, and compare our results with state-of-the-art benchmarks.

CHAPTER 5 concludes and summarizes our findings. We discuss the main lessons, and sketch out specific directions for future work.

1.3 FORMAL REMARKS

Whenever we refer directly to a URL, it can also be found in the bibliography along with information about the author(s) or hosting organization, the page title, and the visitation date.

2 BACKGROUND

FOR WHAT IS TO COME, we find it necessary to provide background on some key topics. Firstly, we must elucidate a few details regarding the Ancient Greek language, and situate it in history. Furthermore, we offer a run-through and summary of the tagging and parsing tasks, especially with respect to inflectional languages and the additional concerns that come with their particularities. Subsequently, we present the data that we will make use of, and finally, we provide a summary of the most relevant scholarship from the past.

2.1 GREEK

The introductory chapter and the thesis subtitle make use of the term *Ancient Greek* somewhat haphazardly, without specifying exactly which language we are referring to by this description. In order to provide a satisfying clarification, it makes the most sense to relate this term, which in actuality refers to a particular period, to the history of the Greek language as a whole.

The Greek language and varieties of it have been spoken on the Greek peninsula and throughout large parts of the Mediterranean world for more than four millennia, of which a temporal span of 3,200–3,400 years include written records. It is a part of the Indo-European language family, where it constitutes its own branch that is estimated to have split off from Proto-Indo-European and other IE-languages around the late 3rd millennium BC (Bakker, 2010, p. 173). Its history starts with its presumed ancestor, *Proto-Greek*, and continues until it reaches *Modern Greek*, its contemporary representative. Needless to say, this vast stretch of time has provided ample opportunity for the language to undergo considerable change, and as such, it is useful to divide the lifespan of the Greek language into several periods. We will not elaborate on all details of this variation here, but provide some brief background on the periods which become relevant at various points throughout this thesis, particularly during the discussion of *domain adaption* in Section 4.4.

Ancient Greek typically refers to the linguistic period of the Greek language between c. 800 and 300 BC; that is, from some time after the fall of the Mycenaean

civilization (from which our earliest records of written Greek originate) until its evolution into *Koine Greek* (the language of the New Testament and a *lingua franca* in the areas of the former Alexandrian Empire). The term Ancient Greek, then, encompasses a substantial body of textual history containing high diachronic and dialectal variation.

The data employed and discussed in this thesis never stems from earlier than the Homeric epics (c. 800 BC), but does, however, include texts from later periods, namely Koine literature like the New Testament, papyri and other non-literary texts spanning several centuries up until c. 700 AD, and in certain contexts even Modern Greek, as in, for instance, sections 2.6.2 and 3.2 when discussing Greek BERT models. As such, the linguistic basis of this thesis really goes beyond what we can rightly call Ancient Greek, but since this period is the common thread which gives relevance to the others, it still deserves to be placed as the central theme.

2.2 TAGGING

Tagging is the task of assigning the correct tag from a set of tags to each token in a sequence: a process also known as *token classification*. Tagsets vary depending on the task; for named entity recognition and other chunking tasks, for instance, so-called *inside–outside–beginning* (IOB) tags are used, allowing spans of tokens to be labeled accordingly (in this case, PERSON, LOC, ORG etc.), and there are other approaches as well, such as for semantic role labeling or clause identification.

The type of tagging that this thesis is concerned with, is the one often interchangeably called *part-of-speech tagging* or *grammatical tagging*, where the goal is to label words with some kind of grammatical or morphological information.

2.2.1 Tagsets and granularity

By name, part-of-speech tagging is about selecting the appropriate part of speech, e.g., disambiguating between verbs, nouns, adjectives, and so on. However, tagsets typically also include more fine-grained grammatical information such as various morphological features—e.g., person, case, number, and so on—or other properties such as possessivity and reflexivity. Therefore, *grammatical tagging* may be more appropriate as a general term, and will be used throughout, if not simply referred to as *tagging*. A further narrowing can then be made within grammatical tagging with the term *morphological tagging*, which refers only to morphological features, disregarding coarse-grained grammatical concepts such as part of speech, for which the aforementioned term *part-of-speech (POS) tagging* is reserved.

For analytic languages, where the amount of inflection on each word is low and mostly word order conveys final meaning, the task can be as simple as disam-

biguating between a few homonyms and deciding the part of speech. For instance, in English, which is a mostly analytic language, the word *needs* may be analyzed either as a verb or a noun. However, the analysis may also be more morphologically detailed, like the following:

- (1) needs.IND.PRES.3SG
- (2) needs.PL

This sort of full-fledged analysis including mood, tense, person and number is not very common for the purposes of English-language NLP as it is largely excessive: Usually it is sufficient to know only the part of speech in order to disambiguate the different usages. Had the example been the form *were*, we could quickly have fallen into poignant discussions about English grammar, and the list of options would grow with many more analyses. For this particular language, however, while it is true that *were* is ambiguous in its analysis as either 1st or 2nd person singular, 1st through 3rd person plural, and even the dubious subjunctive in all persons and numbers in the past tense, this sort of detailed differentiation is typically not made.

Instead of committing to some arbitrary level of granularity, appropriate tagsets are selected according to need. The Brown Corpus (Francis and Kučera, 1979), a POS-tagged selection of English texts across several genres, has 87 different tags, where nouns are discerned according to various other lexical features, such as possessivity, adverbiality and countability. When the tags as so contain more information than just pure part of speech, they are typically called *fine-grained part-of-speech tags*.

In some cases, the part-of-speech tags and the morphological features are not conflated into fine-grained part-of-speech tags, but are kept separate, as is the case for *Universal Dependencies* (Nivre et al., 2020). In the UD scheme, there is a distinction between *universal POS tags* (UPOS) and *universal features* (UFEATS). The former is simply lexical categories such as adj, noun, pron and verb, while the latter includes several other morphological options used to varying degrees in existing treebanks, such as gender, animacy, tense and voice, to name a few.¹ Universal Dependencies and its tagsets become the topic later in Section 3.1.2.

2.2.2 Taggers

In examples (1) and (2) in Section 2.2.1, the task of disambiguating these forms can come down to context. Any English-speaking human can easily decide between the two options in the example above given a sentence where it is used. Consider the following sentence:

- (3) The needs of the many outweigh the needs of the few.

¹A complete list can be found here: <https://universaldependencies.org/u/feat>

Here, the word type (*needs*) occurs twice. We can without trouble decide that the usage in question is that of the plural noun. One hint to this is that the form both times is preceded by the definite article *the*. It would be extremely unlikely and plausibly impossible to see a verb in this context, e.g., “the thinks”. A reasonable approach to grammatical tagging is therefore to take this knowledge into account, for instance, by analyzing n -grams over a corpus and learning probability distributions for each word and tag given their contexts. While this is an instance of a statistical model, rule-based approaches have also been quite common, and surprisingly effective in some situations. We will now look at the principles behind both, and mention briefly their relevance to Ancient Greek.

Rule-based

The earliest approaches to part-of-speech and grammatical tagging, as in all of early NLP, were rule-based. These taggers simply decide tags based on a set of predefined rules. It is worth noting that regularity and detectable patterns often accompany morphological richness, and because of this, rule-based computational approaches to Ancient Greek linguistic analysis have in fact been on the scene from quite early on.² Expectedly, rule-based taggers in general have not been competitive for a long time, and a survey of such systems would be malpositioned here. There is, however, one rule-based approach for Ancient Greek which is worth mentioning, namely *Morpheus* (Crane, 1991). This system does not attempt to analyze morphology in context, and is not a tagger in the sense of a token classifier as such, but rather provides exhaustive suggestions for what a word form can possibly be analyzed as morphologically. *Morpheus* is still in broad use today as a general reading assistance tool, and moreover is often used for supplying neural taggers with morphological alternatives so that they do not have to rely solely on statistical insights. To make it brief, given a word, *Morpheus* provides a list of possible morphological analyses according to a set of deterministic rules and hard-coded options. This tool will be relevant later in sections 4.1 and 4.6 when comparing our results with existing benchmarks.

Statistical and neural

In later times, of course, statistical and neural models have been in vogue. There are many different architectures which solve this task probabilistically, and there is no point in elaborating on these in depth here. We will instead refer to the survey of various taggers (Markov, maximum entropy and n -gram models) for Ancient Greek which has been done by Celano, Crane, and Majidi (2016). In summary, their best tagger achieves about 88% accuracy on the Ancient Greek Dependency

²See Packard (1973) for the first attempt.

Treebank (Bamman and Crane, 2011) after 10-fold cross validation. It is worth noting, however, that when it comes to morphologically rich languages, purely statistical methods that do not incorporate character or subword information may struggle due to the high probability of rare or unknown word forms (Keersmaekers, 2020, pp. 11–12). As such, statistical taggers, including the ones in the survey, still benefit from incorporating rule-based morphological analyses such as those provided by *Morpheus*.

2.3 PARSING

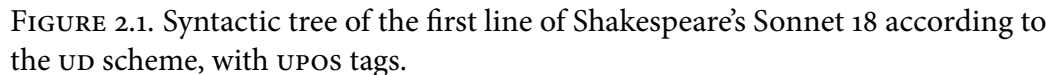
Syntactic parsing—hereby referred to as just *parsing*—is the task of analyzing a sentence according to a set of grammatical principles. There are multiple ways to represent such analyses, and a common way is to draw them as syntactic trees. While linguistic theories about the syntax of natural language are many and varying, syntactic analysis at some level or another has proven useful for NLP tasks in the past (Dozat and Manning, 2017, p. 1). Recently, however, the field seems to be moving away from this idea, at least in certain areas of NLP.³ Nonetheless, as noted in the introduction, treebanks containing syntactically annotated language data are still useful for fundamental language research.

2.3.1 *Dependency structure*

The most common approach to syntactic parsing in NLP today is by way of dependency structures, where syntactically related words in a sentence are considered as pairs called *dependencies*, where the governing word is known as the *head*, and the one governed is called the *dependent*. This sort of relation between a head and dependent is called a *dependency relation*. Formally, this can be represented as a number of arcs (directed edges) from each head to its dependents, as part of a graph for the entire sentence (see Figure 2.1 for an example). In some schemes, the graph is also required to be a tree. Additionally, one word⁴ is selected as the root of the sentence, with its own, special head, governed by no other word. As opposed to constituency-based grammars, where syntactic categories are assigned to one or more words seen as a single unit (*constituents*), dependency structures have the benefit of encoding grammatical relations between individual words directly through labels signifying the *type* of dependency relation. In a representation like this, each token is associated with two labels: one for the index of its head (relative

³See Glavaš and Vulić (2021) for an investigation into the usefulness of syntactic parsing for language understanding tasks.

⁴Or several, depending on the situation and scheme.



2.3.2 Universal Dependencies

One motivation behind UD is to need for a common framework that is able to capture morphosyntactic similarities between languages (de Marneffe and Nivre, 2019, p. 209). This is useful for an array of reasons, including cross-linguistic evaluation, research into multilingual models, transfer learning to low-resource languages, and comparative language research (ibid., p. 210).

2.3.3 Metrics

10

former is defined as the percentage of tokens with a correctly assigned head, and the latter as the percentage of tokens with both a correct head, and correct relation type. Additionally, there is *label accuracy* (LA): the percentage of tokens with the correct relation type.

2.3.4 Parsers

Transition-based

In the past, *transition-based* parsers have been among the preferred solutions for tackling this task. Such parsers work by deciding on actions or *transitions* given a list of the words in a sentence (the *buffer*), a *stack* of undecided words (which have not yet been assigned a head or have dependents which are also undecided), and previous transitions, and they are typically trained using supervised machine learning. A transition can for instance be LEFTARC(NSUBJ), which creates an arc from the topmost word on the stack to the one under it, with the NSUBJ relation. As the experimentation in this thesis makes no direct use of such parsers, this rudimentary explanation will suffice for our purposes.

Graph-based

In recent years, however, there has been a turn toward *graph-based* parsing.⁵ Graph-based parsers work differently, in that they are not based on transitions, but rather arc (or edge) scores for all possible dependencies. The goal, then, is to find the optimal graph given the scores. What constitutes an “optimal graph” depends on one’s notion of a valid parse, and the method by which one assigns the scores.

Valid parses. Typically, one restriction is that the resulting graph should be a tree, i.e., only have one root node, and no cycles. In this case, the optimal graph is the maximum spanning tree (MST) over the graph of all scores. In order to ensure this property, it is possible to use algorithms such as the *Chu–Liu/Edmonds’ algorithm* (Y.-J. Chu and T.-H. Liu, 1965; Edmonds, 1967), which, after selecting the highest scores (emanating from the root), recursively eliminates whatever cycles may be part of the parse until the optimal tree is found.

Scores. There are several ways of scoring a tree, and we will go further into the specifics of how scores are assigned during the explanation of the parser architecture used in our experimentation in Section 3.4. Nonetheless, any approach that is able to assign scores to edges will be a reasonable bid. For now, in order to make the premise

⁵The last transition-based parser with state-of-the-art performance was by Kuncoro et al. (2017).

clear, it is sufficient to show how parsing scores can be represented, and introduce some notation. We use lowercase italic letters for scalars, lowercase boldface letters for vectors, and uppercase boldface letters for matrices and higher-order tensors. Given a sentence of length N , the dependency arc scores can be represented in a word-by-word matrix

$$\mathbf{S}^{(\text{arc})} \in \mathbb{R}^{N \times (N+1)} . \quad (2.1)$$

An additional column ($N + 1$) is prepended to represent the special *root* head which can only have outward-pointing arcs, i.e., to the root of the sentence. The most likely head of a word i will then be given by the highest score in the corresponding row $\mathbf{s}_i^{(\text{arc})}$. Finally, $s_{ij}^{(\text{arc})}$ indicates the score for a potential dependency arc between word i and j (or the *root* node, if $j = 1$). These arc scores can then be optimized relative to the entire graph, according to some optimization objective, in our case the sum of the *cross-entropy loss* given the gold head $y_i^{(\text{arc})} \in [1 \dots N + 1]$ and its predicted score after softmax

$$\hat{y}_i^{(\text{arc})} = \sigma \left(\mathbf{s}_i^{(\text{arc})} \right)_{y_i^{(\text{arc})}} \quad (2.2)$$

for each word i , where σ is the softmax function returning a vector of probabilities. Since the gold value of a gold label will always be 1, this makes out to be simply the negative logarithm of each predicted score after softmax

$$\ell_{\text{CE}}(\hat{y}_i) = -\log \hat{y}_i . \quad (2.3)$$

For labeling, the same representation holds, but since the number of possible dependency relation labels have to be accounted for as well, the label score tensor $\mathbf{S}^{(\text{label})}$ will for each possible arc $\mathbf{s}_{ij}^{(\text{label})}$ hold a vector of scores for the number of labels L , giving the dimensionality

$$\mathbf{S}^{(\text{label})} \in \mathbb{R}^{N \times (N+1) \times L} . \quad (2.4)$$

Finally, the label scores can be optimized over the gold dependency arcs. We can calculate the sum of the cross-entropy loss ℓ_{CE} for each word i given the gold head $y_i^{(\text{arc})}$ and the gold label $y_i^{(\text{label})} \in [1 \dots L]$ of each, and the predicted scores of the respective gold labels after softmax

$$\hat{y}_i^{(\text{label})} = \sigma \left(\mathbf{s}_{i y_i^{(\text{arc})}}^{(\text{label})} \right)_{y_i^{(\text{label})}} . \quad (2.5)$$

2.4 INFLECTIONAL LANGUAGES

Now that the tasks of tagging and parsing have been rudimentarily explained, we will look at the case for inflectional languages such as Ancient Greek. When it comes to tagging these types of languages, the tagsets (in the case of fine-grained

POS tags) and/or feature sets (in the case of separate grammatical features) are often larger in size compared to those of analytic languages. As a practical example, the PROIEL treebank (Haug and Jøhndal, 2008)—which contains trees for several ancient Indo-European languages, including Ancient Greek, and all of them highly inflecting—has 26 fine-grained POS tags, in addition to morphological features including (with the number of options in parentheses): person (4), number (4), tense (10), mood (13), voice (4), gender (8), case (13), degree (4) and inflection (2) (Eckhoff et al., 2017, pp. 39, 41). As several of these features are combined in most parts of speech, it goes without saying that the complexity is quite significant.

One reason for this granularity at the morphological level is that it is often required in order to make important distinctions in the syntactic analysis. Vice versa, knowledge about syntax is often the key factor to predicting correct tags and features. Both of these facts arise due to the highly free word order in these types of languages. To illustrate, the words *μεγάλα* [*megála*], *τεκμήρια* [*tekméria*], *ὃ* [*hò*], and *ἔργα* [*érğa*] can all be neuters in the nominative, vocative or accusative. Here are the words in a context, with the correct analyses:

- (4) *μεγάλα δ' ἔγωγε ὑμῖν τεκμήρια παρέξομαι*
 great.N.ACC.PL but I you.DAT.PL evidence.N.ACC.PL give.FUT.MID.1SG
τούτων, οὐ λόγους ἀλλ' ὃ ὑμεῖς
 this.M.GEN.PL not word.M.ACC.PL but which.N.ACC.SG you.NOM.PL
τιμᾶτε, ἔργα.
 honor.IND.PRES.2PL action.N.ACC.PL
 “I will give you great proofs of this, not only words, but that which you honor,
 actions.”
 Plat. Apol. 32A

Going by locality alone, such as an *n*-gram model would, a tagger may well correctly recognize that most of these are accusatives, given their proximity to the verbs, *παρέξομαι* [*paréksomai*] and *τιμᾶτε* [*timâte*], and the presence of unambiguous nominative subjects, *ἔγωγε* [*égōge*] and *ὑμεῖς* [*humeís*]. For *érğa*, however, it is a harder case. The object role of *timâte* should be filled by *hò*, since *humeís* can only be the subject. Syntactic knowledge that *érğa* is in apposition with *hò*, which is coordinated with *λόγους* [*lógous*], which is in further apposition with *tekméria*, echoing *érğa* back as the object of *paréksomai*, may, then, be necessary in order to assign the correct tag as an accusative neuter.

A proposition, then, is to perform the tagging and parsing simultaneously in order to make more well-informed decisions at both levels of analysis. This has been suggested, among others, by Lee, Naradowsky, and Smith (2011), who dub the issue a “chicken-and-egg problem,” and later work also concludes that this may be an important step to bettering the performance of tagging and parsing systems for Ancient Greek (Keersmaekers, 2020, p. 25).

Keersmaekers (2020) explores this technique further in practice with the joint MATE tagger/parser (Bohnet et al., 2013), and finds that while it can make useful assumptions based on syntactic insight that the separate models cannot (Keersmaekers, 2020, p. 35), the model in fact does not perform better overall than standalone POS taggers. Furthermore, it is said of the joint tagger/parser in question, that its “low accuracy seems to be primarily caused by its tagging model that is unsuitable to analyze Ancient Greek” (ibid., p. 31), more specifically its treatment of morphological traits as one block instead of individual attributes. It is further suggested that resolving this in another joint model will enable performance improvements (ibid., p. 36). This all hits the point of the example in (4), where syntactic awareness in combination with fine-grained morphological information may potentially hold the key to a correct analysis.

Thereby, the point at issue is whether a tagger/parser system that performs the two analyses jointly, and/or is fully capable of exploiting the rich morphological features available, has the potential to improve performance in both tasks for Ancient Greek. This question will be investigated and addressed further in Section 4.3.

2.5 CORPORA

In order to perform any computational work, digitized language data is a prerequisite. Several corpora have been developed for Ancient Greek, often in projects alongside other classical languages, for the purposes of language research. What follows is an overview of the available data, including both more or less unannotated collections, and fully syntactically and morphologically annotated treebanks.

2.5.1 Collections

Perhaps the most well-known Greek language corpus in the digital age is the *The-saurus Linguae Graecae* (Pantelia, 2001), which aims to keep a full, digitized collection of all surviving Ancient Greek literature. At 110 million words, it is by far the largest collection of digitized pre-modern Greek.⁶

Another project aimed at digitization of classical literature is the *Perseus Digital Library* (Crane, 1995), which stands at about 13.5 million words of Ancient Greek.⁷ The content is similar to that of the TLG, but not as extensive.

Yet another significant venture is the *First1KGreek* project,⁸ whose goal is to collect Greek works from the 1,000 years after Homer, with a special focus on

⁶Numbers are available here: <http://stephanus.tlg.uci.edu/tlg.php>

⁷Numbers are available here: <https://www.perseus.tufts.edu/hopper/collections>

⁸More information about the *First1KGreek* project can be found here: <https://opengreekandlatin.github.io/First1KGreek>

texts not already contained in the *Perseus* library, and which boasts a considerable 25.2 million words as of the present date.

Lastly, there is the *Papyri.info* project,⁹ which maintains a digital corpus that combines text from the *Duke Databank of Documentary Papyri*¹⁰ with material from various other papyrological research projects. The textual material consists of almost the entirety of the non-literary Greek papyrus corpus, which numbers at about 4.6 million tokens (Keersmaekers, 2020, p. 80).

2.5.2 Treebanks

The above-mentioned collections are mostly unannotated,¹¹ meaning that they only contain raw, digitized text, and no linguistic analysis whatsoever. There are, however, several projects that aim to build treebanks over much of the same data, containing both morphological information and syntactic trees for large amounts of text, typically for the purpose of linguistic research. The treebanks mentioned here are all dependency treebanks.

The *Pragmatic Resources of Old Indo-European Languages* project (PROIEL) (Haug and Jøhndal, 2008) is a treebank, or rather family of treebanks of old Indo-European Languages, namely Ancient Greek, Latin, Gothic, Armenian and Old Church Slavonic, and has complete, annotated dependency trees with morphological tags. The Greek part numbers at about 280,000 tokens (Eckhoff et al., 2017, p. 32).

The *Ancient Greek Dependency Treebank* (AGDT) (Bamman and Crane, 2011) encompasses a significant collection of both prose and poetry, and contains (as of version 2.1) about 560,000 tokens of Ancient Greek that have been lexically, syntactically and morphologically annotated.

Other substantial treebank projects include the *Gorman* treebank (Gorman, 2020) which is based on Greek prose texts from the *Perseus* project, at about 550,000 tokens, and the *Pedalion Treebanks* (Keersmaekers et al., 2019), both poetry and prose, at about 300,000 tokens.

Lastly, there are a few smaller, but nonetheless valuable projects worth mentioning. These are the *Sematia* project (Vierros and Henriksson, 2017), which contains about 6,000 tokens of annotated documentary papyri, the Harrington treebanks (Harrington, 2020) at about 18,000 tokens of Greek prose, as well as a smaller project of annotating Aphthonius' *Progymnasmata* (Yordanova, 2018), at about 7,000 tokens.

⁹More information about *Papyri.info* can be found at <https://papyri.info>, and the data is available at <https://github.com/papyri/idp.data>.

¹⁰More information about the DDBDP can be found here: <https://papyri.info/docs/ddbdp>.

¹¹*Perseus* includes automatically predicted morphological analyses with most of its data, and TLG has lemma information.

2.6 RELATED WORK

2.6.1 *Automatic annotation of the Greek papyrus corpus*

As mentioned in Section 2.5.1, in addition to the portion of literary Greek that makes up the majority of the available data, we also have a significant amount of *documentary papyri* on our hands. The most comprehensive collection is the aforementioned *Papyri.info* project, which maintains roughly 50,000 texts from transcribed papyri and other inscriptions spanning from the 4th century BC until the 8th century AD (Vannini, 2018, p. 1).

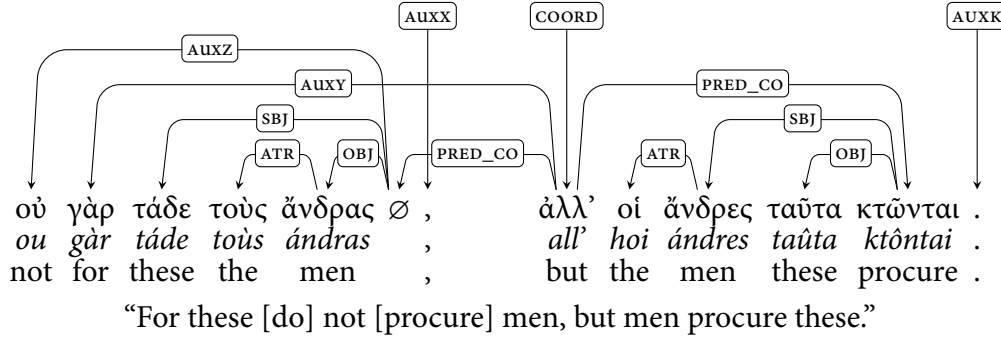
As argued by Keersmaekers (2020, p. 2), the papyrological data is particularly interesting for linguistic research, as it is produced by a more “sociologically diverse” set of authors and in a style that is of a more ordinary nature than traditional, literary texts. It also has a lot to offer to those interested in diachronic change, theoretical and computational linguists alike, due to its substantial temporal breadth.

Among the corpora mentioned in the previous section (2.5), the largest of them is unfortunately not available for any kind of processing. This means that the amount of data for pre-modern Greek at our disposal (in the broad sense, excluding papyri) in actuality totals at about 38.5 million tokens. The addition of the papyrus data, then, with its roughly 4.6 million tokens, is a significant contribution to the total (c. 12%).

These factors, among others, speak for the value of the work that was done by Alek Keersmaekers in the dissertation entitled “A Computational Approach to the Greek Papyri: Developing a Corpus to Study Variation and Change in the Post-Classical Greek Complement System” (2020). This project explores and addresses many of the challenges and concerns associated with processing a historical corpus such as the Greek one. Many of these will be relevant also to this thesis, as it largely builds on, and could not exist without, the work laid out in the dissertation.

Among the aims of the project was to build an automatically (morphologically, syntactically and semantically) annotated corpus of the papyrus data. In order to do so, the experimentation makes use of a number of tagging and parsing systems which require data for training, validation, and testing. This is procured by compiling all of the treebanks mentioned in Section 2.5.2. This naturally comes with homogenization issues. The treebanks follow different formats and annotation styles, and were therefore automatically converted to the same uniform AGDT format (ibid., p. 44). Furthermore, some of the annotation inconsistencies were ironed out using written rules, automatic anomaly detection and manual correction (ibid., p. 47).

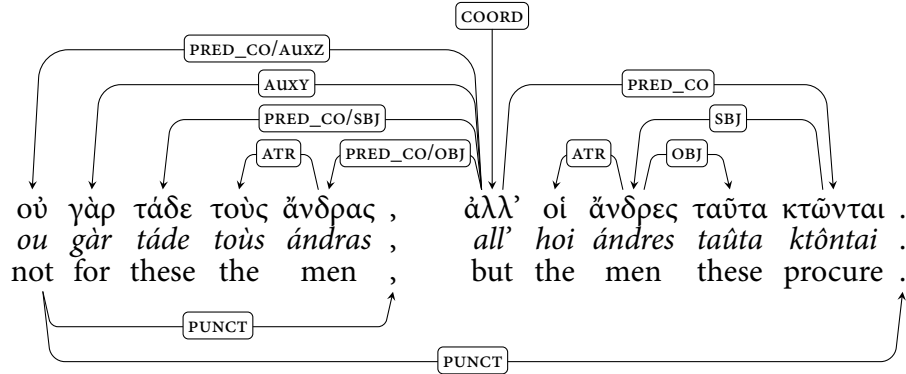
Moreover, there is the issue of *elliptic structures*. It is a common occurrence that words are elided within a sentence. For instance, a verb can be implied instead of

FIGURE 2.2. Syntactic analysis of Thucydides’ *Histories* 1.143.5 as annotated in AGDT.

repeated if it is used again with a different subject and/or object, or commonly, the copula verb is left to be understood. Figure 2.2 shows an example of such a sentence in the AGDT format, where an empty node is inserted for the elided element. Here, two conjunct clauses share a predicate, κτῶνται [*ktôntai*], which receives the relation label PRED_CO (the suffix “_CO” indicates that it is part of a coordination). In the first clause, τάδε [*táde*] is the subject (SBJ) in the nominative, and τοὺς ἄνδρας [*toùs ándras*] is the object (OBJ) in the accusative. In the second clause, they have switched places: οἱ ἄνδρες [*hoi ándres*] is the nominative of *toùs ándras*, and ταῦτα [*taûta*] is an accusative determiner that is coreferent with *táde*. The ellipsis occurs in the first clause, where the analysis includes an empty node to which the dependents of the understood predicate is attached. As mentioned, the dataset is originally converted to AGDT, but because the parsers used in the experimentation are unable to insert empty nodes, the issue of elliptical structures had to be resolved in some other way. The chosen solution was to remove every empty node, and assign each node’s head and relation to its dependents, with the original relations of the dependents appended. It is worth noting that this explodes the tagset significantly, from 33 to 276 different relation types. The same sentence after conversion is shown in Figure 2.3.

What results is a comprehensive dataset of almost all available Greek data from the relevant periods, literary and documentary, containing both morphological and syntactic information in a uniform format. It is worth noting that the data was selected with performance on papyrus data in mind, and as such, some parts of the available material was omitted in order to “avoid too large diachronic and genre differences with the papyri” (ibid., p. 45). The dataset is split into training, validation and test partitions, where the main genre components are prose, prose/papyri, and papyri, respectively. A complete overview of the particular texts in each set can be found in the dissertation (ibid., pp. 46–47).

The actual findings of the experimentation in the thesis will be the topic later during the comparisons in Section 4.6.



“For these [do] not [procure] men, but men procure these.”

FIGURE 2.3. Syntactic analysis of Thucydides’ *Histories* 1.143.5 after conversion. Note that the new format also analyzes punctuation differently.

2.6.2 Ancient Greek BERT

For the better part of a semidecade, word embedding strategies across the board have been outcompeted by those produced by transformer-based models such as BERT (Devlin et al., 2019). Pretraining contextualized encoder representations on large amounts of language data and fine-tuning them to specific tasks has turned out to be a reliable way of pushing the state of the art in most, if not all areas of NLP.

Of course, the lingering notion that “BERT is better,” while often the case, is no guarantee. The performance of a system which incorporates BERT embeddings has been shown to be very much affected by one parameter: the amount of data. Multilingual models such as mBERT (ibid.) and XLM-ROBERTa (Conneau et al., 2020) perform comparably to—and often outperform—smaller, monolingual models in the target language (Wu and Dredze, 2020, p. 127). Monolingual models do have an advantage, however, as long as there is *enough* training data, and several experiments with language-specific models for low- to medium-resource languages have shown this (Koutsikakis et al., 2020; Virtanen et al., 2019).


Singh, Rutten, and Lefever (2021) show that this also is the case for Ancient Greek, which is arguably a low-resource language by itself, in their training of a specialized BERT model. As mentioned in Section 2.6.1, the amount of actually available Ancient Greek data is restricted to the few open-source projects that provide it, and as such, the authors have had to look elsewhere for improvements. The model, called *Ancient Greek BERT*, is therefore first initialized with weights from the Modern Greek model, GREEK-BERT (Koutsikakis et al., 2020), which is trained on several gigabytes worth of Modern Greek data from the internet, namely the

Greek parts of *Wikipedia*,¹² *Europarl* (Koehn, 2005) and *OSCAR* (Suárez, Romary, and Sagot, 2020). Its training is then resumed for the *masked language modeling* objective (Devlin et al., 2019, pp. 4171–4172) on parts of the available Ancient Greek data, namely *PROIEL* (Haug and Jøhndal, 2008), *AGDT* (Bamman and Crane, 2011), the Gorman treebanks (Gorman, 2020), the Greek part of the *Perseus Digital Library* (Crane, 1995), and the *First1KGreek* project (Open Greek and Latin, 2022).

After pretraining, the model is fine-tuned and evaluated on morphological tagging, where it outperforms the so-called *RNNTagger* by Schmid (2019) on their test set of Byzantine Greek. It has to be noted that several of the mistakes made by the *RNNTagger* can be attributed to a mismatch in the available tagset (Singh, Rutten, and Lefever, 2021, p. 135). As such, the evaluations are not directly comparable. The tagger architecture consists of an LSTM-CRF encoder-decoder which employs fine-tuned character embeddings stacked with the token embeddings from BERT (ibid., p. 134).

¹²Recent dumps can be found here: <https://dumps.wikimedia.org/elwiki/>

3 EXPERIMENTAL SETUP

HE GOAL of the following experimentation is to investigate how and whether tagging and parsing performance for Ancient Greek can be improved. We do so by combining several techniques that have yielded state-of-the-art results in the same tasks for other languages, and other tasks for Ancient Greek. In summary, we fine-tune a number of BERT models. We also explore how variation in the architecture, training data, and data formats affect system performance. Finally, we experiment with multi-task learning to see whether our system can be improved by allowing interaction between the tagging and parsing components, as per the ideas presented in Section 2.4.

3.1 DATA

As with all forms of supervised machine learning, we rely on labeled data. In Chapter 2, we mentioned that tagsets and formalism for tagging and parsing in any language are quite prone to variation due to the numerous ways of thinking about grammar. In Ancient Greek computational linguistics, there is likewise no consensus or central annotation project to serve as the one standard to end them all, and each project has its merits and deficits. We will discuss these topics more in depth in Section 5.2.2. For now, we will also note that there are discrepancies even when it comes to such things as orthography, encodings, and formats. In this section we therefore summarize the available data, and comment on the most important aspects that will be relevant during the experimentation.

3.1.1 Datasets

We focus on three datasets: PROIEL (Haug and Jøhndal, 2008), AGDT (Bamman and Crane, 2011), and the dataset developed as part of the papyrus annotation project by Keersmaekers (2020), which we will generally refer to as the *papyrus dataset*. These datasets all follow their own formats and annotation schemes. While the papyrus dataset is in part a conversion of several treebanks into the same AGDT scheme, it

does contain further manipulations which makes it incompatible with the original AGDT data. The dataset is laid out in further detail in Section 2.6.1.

With these datasets, we have a rather diverse selection of texts from different periods and genres to evaluate our systems on. AGDT contains classical prose and poetry from Homer (c. 8th century BC) until Plutarch (2nd century AD); PROIEL contains prose works from three distinct linguistic periods, namely Herodotus’ *Histories* (classical Greek, 5th century BC), the Greek New Testament (Koine Greek, 1st century AD) and Sphrantzes’ *Chronicles* (Medieval Greek, 15th century AD); and the papyrus dataset contains more or less everything from Herodotus until 8th century documentary papyri.

In addition to their original formats, PROIEL and AGDT have also been automatically converted to UD.¹ One benefit from this is that it makes comparison with other tagger/parser systems an easier job, since the UD treebanks are commonly used as an evaluation benchmark for these tasks. Moreover, (basic) UD is easier to annotate computationally speaking, since it does not include empty nodes, as was the topic in Section 2.6.1. Instead, it promotes one token to the role of the elided element, and labels it with either the original relation or the special orphan relation depending on the situation. For more information on the handling of ellipsis in UD, we refer to Marneffe et al. (2021, p. 282). One downside to the converted datasets, however, is that the resulting evaluations may not be as constructive as those from more carefully procured datasets, due to inaccuracies or discrepancies in the automatic conversions. For instance, the UD conversion of AGDT does not include sentences with elliptical structures at all, which is a rather common syntactic phenomenon that is best left in order for the evaluation to be linguistically reasonable. Such annotation issues are part of the topic in Section 5.2.2. It is also central to note that the UD version of PROIEL does not contain the Medieval Greek part, since it is intended to be a treebank for Ancient Greek proper. For all of these reasons, we only make use of the UD conversions of AGDT and PROIEL in our experimentation. We use the same training, validation and testing splits as in the official UD treebanks. More information about the contents of the splits can be found in the relevant documentation (see footnote 1).

Table 3.1 gives an overview of the datasets, their training, validation, and test split sizes, and tagset sizes for the labels that are relevant to our experimentation, which are to be explained in Section 3.1.2.

¹The converted treebanks are available at https://github.com/UniversalDependencies/UD_Ancient_Greek-PROIEL and https://github.com/UniversalDependencies/UD_Ancient_Greek-Perseus, respectively.

Dataset	Train	Val.	Test	(U)POS	XPOS	<i>feats</i>	<i>deprel</i>
Papyrus	894,134	40,373	71,280	6	18	678	276
PROIEL (UD)	187,033	13,652	13,314	14	23	811	33
AGDT (UD)	159,895	22,135	20,959	14	810	639	25

TABLE 3.1. Overview of the experimental datasets and their split and tagset sizes. Train, validation and test sizes are counted in number of tokens. The tagset size counts only include tags which occur in the training sets, as only these can be learnt by our models.

3.1.2 Formats and tagsets

Treebanks that are part of the UD project are stored in the CONLL-U format,² which organizes sentences into lists of tokens, with associated labels across several columns. Of these, five are relevant to our experimentation. These are the UPOS, XPOS, *feats*, *head* and *deprel* columns. While *head*, which indicates the index of the head of the token, needs to be an integer (in basic UD), tagsets for the remaining columns can vary from treebank to treebank.

In UD proper, UPOS and *feats* have to be among the aforementioned, standardized universal POS tags and universal features (Section 2.2.1), while the XPOS column (“x” for “extended”) is reserved for language-specific categories that can be used at the treebank creators’ discretion. In the UD conversion of AGDT, the XPOS column corresponds to the original AGDT *postags*, which conflate POS tags and morphological features into nine-character strings where each character encodes a field. This is the reason for the large tagset size, as shown in Table 3.1. In the UD version of PROIEL, the column is rather used for fine-grained POS tags (Eckhoff et al., 2017, p. 39) like in the original treebank.

In the papyrus dataset, which is also available in the CONLL-U format, the columns are all used slightly differently. In this dataset, the UPOS column is reserved for coarse-grained grammatical categories such as *nominal*, *verbal*, and *adverbial*, while the XPOS column denotes slightly more specific categories such as *finite* (for verbals), *proper* (for nominals) and *coordinator* (for adverbials). The *feats* column more or less matches the features that are part of the UD UFeats, but there are some slight variations, such as aspect not being included. The dataset also includes semantic features for verbs and nouns, but these are left out in our experimentation. Since the UPOS column in this case does not adhere to UD universal POS tags, we refer to this label type as simply POS in the context of the papyrus dataset.

The *feats* column consists of a collection of key-value pairs denoting the morpho-

²More information about CONLL-U can be found here: <https://universaldependencies.org/format>.

the letter separately may allow the system to make more use of the information (ibid., p. 56). In practice, normalizers, when faced with Greek Unicode, often remove accents in order to reduce the character set, with the downside that some information is lost. This is, for instance, the case with the tokenizer that comes with Ancient Greek BERT. By converting to Beta Code, however, this information can be kept, even without having a considerably larger character set. Because of this discrepancy, we attempt to incorporate some accents in Unicode after normalization, as will be explained in Section 3.1.4.

3.1.4 Preprocessing

Due to the amount of normalization that has already been done in the papyrus dataset, there are not many discrepancies which have to be resolved. Most of those that persist have to do with odd punctuation characters or diacritical marks that are either the result of human error or inaccuracies during normalization.⁴ Because the BERT tokenizer has a limited vocabulary, it is necessary to manually replace these with consistent characters in order to avoid too many unknown tokens or having to add and train new tokens for the different variants. The normalization strategy which makes the papyrus dataset, PROIEL and AGDT compatible with the relevant BERT tokenizers is made available in Appendix A.

For keeping diacritics, we apply a simple strategy of appending an *iota* (ι) to those vowels which have one under it (*iota adscript*), and prepending a *heta* (ἥ) to words where any vowel in the word has rough breathing with no consonants preceding it. Since this character is not originally part of the tokenizer vocabulary, we add it manually, the idea being that it can be fine-tuned to embed the information that the aspiration marks hold.

We do not attempt to incorporate tonal accents into the Unicode data. This is because these accents very often occur, and would have to be indicated, inside words. This would, in turn, interfere with the tokenizer by corrupting the subwords it would have been likely to have recognized from pretraining, and thereby hurt performance. The advantage of expanding iotas and breathing marks is that they can be included in the word without making a large impact on the tokenization. Since the *heta* is a new token, and only occurs at the start of a word, it is always tokenized as its own subword. Meanwhile the *iota*—while it is likely to disrupt the subwords to some degree—is fairly likely to have occurred in adscripted positions during pretraining, since the orthography of Greek is not always completely consistent. Nonetheless, it is also probable that the effects of the added information are more valuable to the fine-tuning than they are detrimental to the pretraining.

⁴One common problem is quotation marks, which can be written in many ways (", ", ' ', etc.), and especially apostrophes, which are often confused with the smooth breathing mark. For those interested, we recommend using U+2019 RIGHT SINGLE QUOTATION MARK for Greek apostrophes.

3.2 TRANSFORMER-BASED MODELS

Considering the promising results by Singh, Rutten, and Lefever (2021), we wish to examine the potential of using specialized pretrained models for Ancient Greek tagging and parsing. For purposes of comparison, we make use of three different transformer-based models, which are all variants of BERT (Devlin et al., 2019). These are as follows:

- Ancient Greek BERT (Singh, Rutten, and Lefever, 2021).⁵ Initialized with weights from GREEK-BERT and pretrained further on Ancient Greek data. It has 12 encoder layers and attention heads with a hidden size of 768 and a 35,000 subword vocabulary. The tokenizer is identical to the one that comes with GREEK-BERT. We refer to this model as “AGBERT” for short. More background is available in Section 2.6.2.
- GREEK-BERT (Koutsikakis et al., 2020).⁶ Trained on Greek internet data. It has the same architecture and vocabulary as AGBERT. More details are once again available in Section 2.6.2.
- mBERT (Devlin et al., 2019).⁷ Trained on the 104 languages that have the largest Wikipedia projects (which includes Modern Greek). It has 12 encoder layers and attention heads with a hidden size of 768 and a 119,547 subword vocabulary. Unlike the two others, this model is case sensitive.

These models have been selected since each one is trained on some form of Greek data, but from various domains and distributed differently. In comparing them, we hope to see the effects of increasing the amount of in-domain data, and thereby assess what further steps can be taken in order to improve task performance for our target language in the future, and validate the need for monolingual models for low- to medium-resource languages.

The experimental setup is identical across models, with the same data,⁸ classifier architectures, and hyperparameter values. We optimize the hyperparameters based on the performance of Ancient Greek BERT, which include: number of epochs, batch size, discriminative learning rates, number of warmup steps, and dropout probabilities. We also observe the results of whether subscript iotas are expanded

⁵<https://huggingface.co/pranaydeeps/Ancient-Greek-BERT>

⁶<https://huggingface.co/bert-base-greek-uncased-v1>

⁷<https://huggingface.co/bert-base-multilingual-cased>

⁸Since mBERT is multilingual, it tends to produce smaller subword tokens than the other models. Some of the sentences in the AGDT and the papyrus dataset are therefore tokenized into more subwords than the model can handle (512). For this reason, the sentences are discarded, but in total the number is in the single digits.

or not, and whether breathing marks are prepended or not. We use *Adam* (Kingma and Ba, 2014) for optimization. Further specifics about the hyperparameters and their optimization can be found in Section 4.2.

We tokenize all data using each model’s bundled tokenizer and extract the last hidden states to get subword embeddings, and then produce token representations by averaging the subword embeddings that correspond to each token. Alternative ways of constructing token representations include using only the first or last subword, or max pooling. Kondratyuk and Straka (2019, p. 2781) report that using either strategy makes no difference during evaluation for the same tasks, but since related work handles it this way (Glavaš and Vulić, 2021; Nguyen et al., 2021), we also opt for this procedure. It is furthermore tempting to hypothesize that morphologically rich languages such as Ancient Greek may have more to gain from taking every subword into account, but we cannot claim this offhandedly. Ács, Kádár, and Kornai (2021) provide testimony, however, that averaging or running LSTMs over the subwords are well-performing strategies for POS tagging, and particularly so for the most inflectional language they test for, which is Finnish. We discuss this and other such options for subword pooling further in Section 5.2.1.

3.3 TAGGER

Given the formats we work with, our tagging task involves predicting three different types of labels for each token: POS, XPOS, and *feats*.⁹ The tagger therefore consists of three classifiers—one per label type. Each classifier is an affine transformation with learned weights and bias from the stacked token representations $\mathbf{R} \in \mathbb{R}^{N \times H}$ to the label outputs, where N is the number of tokens in a sentence and H is the hidden size of the transformer:

$$\mathbf{S}^{(\text{POS})} = \mathbf{RW}^{(\text{POS})} + \mathbf{b}^{(\text{POS})} \quad (3.1)$$

$$\mathbf{S}^{(\text{XPOS})} = \mathbf{RW}^{(\text{XPOS})} + \mathbf{b}^{(\text{XPOS})} \quad (3.2)$$

$$\mathbf{S}^{(\text{feats})} = \mathbf{RW}^{(\text{feats})} + \mathbf{b}^{(\text{feats})} \quad (3.3)$$

We apply a dropout to the last layer outputs from BERT before averaging the respective subword embeddings of each word to get their respective token representations $(\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_N) = \mathbf{R}$. The classifiers are trained jointly by calculating the loss as the sum of the cross-entropy loss of the softmaxed predicted values of the gold labels $\hat{y}_i^{(\text{POS})}$, $\hat{y}_i^{(\text{XPOS})}$, $\hat{y}_i^{(\text{feats})}$ of each word i for each classifier:

$$\mathcal{L}^{(\text{tagger})} = \sum_{i=1}^N \left(\ell_{\text{CE}} \left(\hat{y}_i^{(\text{POS})} \right) + \ell_{\text{CE}} \left(\hat{y}_i^{(\text{XPOS})} \right) + \ell_{\text{CE}} \left(\hat{y}_i^{(\text{feats})} \right) \right) \quad (3.4)$$

⁹See Section 3.1.2 for an explanation of what each label type is used for in the different datasets.

We apply discriminative learning rates, with one value for the BERT parameters, and another for the tagger weights and biases.

3.4 PARSE

Following Glavaš and Vulić (2021), we implement a simplified version of the graph-based biaffine parser introduced by Dozat and Manning (2017). In the original paper, the authors apply *biaffine attention* over dimension-reduced LSTM representations in order to produce arc and label scores. Glavaš and Vulić (2021), however, omit the dimension-reducing MLPs in order to keep the task-specific architecture shallow. The idea is to allow the high number of parameters available in the transformer to be exploited during fine-tuning, instead of adding complexity in the task heads themselves. The resulting parser architecture, as with the tagger, is therefore quite minimal, allowing the transformer parameters to do most of the work.

The idea of biaffine attention stems from the attention mechanism for machine translation introduced by Bahdanau, K. Chu, and Bengio (2014). The intuition is that by mapping an input and an output token to a score using an MLP, and training it jointly with an encoder-decoder system, the MLP can learn the relative alignment between words, allowing the decoder, which incorporates the alignment scores, to mimic something approximating cognitive attention—namely, knowing which input tokens are likely to be relevant when predicting the next output token. In the context of dependency parsing, the representations of the input and output tokens can be seen as candidates for a head and a dependent, and the score is the probability that there should be an arc between them.

Dozat and Manning (2017) replace the alignment MLP with a simple, biaffine transformation, which takes two token representations as input; one for the potential dependent, and one for the potential head. As mentioned, they also reduce the dimensionality and concentrate the information of the original LSTM embeddings, which they do by running them through two other MLPs, where one produces representations of the tokens as heads, and the other representations of the tokens as dependents. For predicting arc scores, they further introduce a bias term over the head representations in order to model the prior probability of a word taking dependents. Likewise, they add two bias terms for label prediction: one to capture the prior probability of each label, and one to model both the likelihood of a label given a word, and the likelihood of a label given the word’s head.

For the reasons explained earlier, we instead skip the MLPs altogether, and feed the representations directly into the biaffine transformation. We add the bias by concatenating a vector of ones to the stacked representations and increasing the size of the parallel dimension in the weight matrix by one. Doing this, we end up with the dependent representation with bias $\mathbf{r}_i^{(\text{dep})} \in \mathbb{R}^{H+1}$, the head representation

$\mathbf{r}_j^{(\text{head})} \in \mathbb{R}^H$, and the weight matrix $\mathbf{W}^{(\text{arc})} \in \mathbb{R}^{(H+1) \times H}$. We can then calculate the score of a potential arc between words i and j as such:¹⁰

$$s_{ij}^{(\text{arc})} = \mathbf{r}_i'^{(\text{dep})} \mathbf{W}^{(\text{arc})} \mathbf{r}_j^{\text{T}(\text{head})} \quad (3.5)$$

For labels, we add another dimension to the weight matrix to accomodate for the number of relation labels L . Furthermore, we concatenate a bias constant vector to the head representations as well, giving us $\mathbf{r}_j'^{(\text{head})} \in \mathbb{R}^{H+1}$, and include the corresponding weights in the new weight tensor $\mathbf{W}^{(\text{label})} \in \mathbb{R}^{L \times (H+1) \times (H+1)}$:

$$\mathbf{s}_{ij}^{(\text{label})} = \mathbf{r}_i'^{(\text{dep})} \mathbf{W}^{(\text{label})} \mathbf{r}_j'^{\text{T}(\text{head})} \quad (3.6)$$

On the sentence level, the representations for heads and dependents differ slightly. Since only actual, present tokens in the sentence can be dependents, the stacked representations for dependents with bias can be given simply by $\mathbf{R}'^{(\text{dep})} \in \mathbb{R}^{N \times (H+1)}$. When it comes to heads, however, the special *root* node is also an option. In order to have a representation for it, we prepend the classification token from the transformer model to the dependent representations, which gives us $\mathbf{R}^{(\text{head})} \in \mathbb{R}^{(N+1) \times H}$ and $\mathbf{R}'^{(\text{head})} \in \mathbb{R}^{(N+1) \times (H+1)}$, with and without bias.

In summary, the full score tensors are calculated as such:

$$\mathbf{S}^{(\text{arc})} = \mathbf{R}'^{(\text{dep})} \mathbf{W}^{(\text{arc})} \mathbf{R}^{\text{T}(\text{head})} \quad (3.7)$$

$$\mathbf{S}^{(\text{label})} = \mathbf{R}'^{(\text{dep})} \mathbf{W}^{(\text{label})} \mathbf{R}'^{\text{T}(\text{head})} \quad (3.8)$$

After the multiplications they end up with the following dimensionalities:

$$\mathbf{S}^{(\text{arc})} \in \mathbb{R}^{N \times (N+1)} \quad (3.9)$$

$$\mathbf{S}^{(\text{label})} \in \mathbb{R}^{L \times N \times (N+1)} \quad (3.10)$$

We calculate the arc loss as the sum of the cross-entropy loss of the softmaxed predicted scores for the gold heads $\hat{y}_i^{(\text{arc})}$ of each word i . For the relation label loss, we mask the scores with the gold heads and calculate the cross-entropy loss of the softmaxed predicted scores of the gold relation labels $\hat{y}_i^{(\text{label})}$ for each word. The total loss is the sum of the arc and label loss.

$$\mathcal{L}^{(\text{parser})} = \sum_{i=1}^N (\ell_{\text{CE}}(\hat{y}_i^{(\text{arc})}) + \ell_{\text{CE}}(\hat{y}_i^{(\text{label})})) \quad (3.11)$$

As in the previous task, we apply a dropout to the BERT outputs before getting the averaged token representations. At prediction, we permute $\mathbf{S}^{(\text{label})}$ to make the label dimension the last one, and greedily select the maximum values in the final

¹⁰For a clarification on the notation, we refer back to Section 2.3.4.

dimensions of both score tensors to get the predicted head of each token and the relation labels for each possible dependency. Since we do not use an MST decoding algorithm, this does not guarantee a tree structure, but as X. Zhang, Cheng, and Lapata (2017, p. 668) note, the predicted graphs usually turn out to be trees either way. Finally, we mask the relation label predictions with the predicted heads to get labels for each token. During training, we once again employ different learning rates for the BERT and task head parameters, respectively.

3.5 JOINT MODEL

Following Kondratyuk and Straka (2019) and Nguyen et al. (2021), we devise our joint model as a multi-task learning system, where a single BERT model is fine-tuned together with all of the aforementioned task heads simultaneously. This is known as the *tightly joint* or *share-tight* (Zhou et al., 2020) strategy, where all of the system parameters, excluding the decoders themselves, are shared. The entire model is then optimized on the joint loss, which is given by $\mathcal{L}^{(\text{joint})} = \mathcal{L}^{(\text{tagger})} + \mathcal{L}^{(\text{parser})}$.

In order to provide an overview, Figure 3.1 shows an illustration of the model architecture of the joint tagger and parser. We list the task heads again here and refer to their associated equations, to compare with the illustration.

- Eq. 3.1 – $\text{Affine}^{(\text{pos})}$
- Eq. 3.2 – $\text{Affine}^{(\text{xpos})}$
- Eq. 3.3 – $\text{Affine}^{(\text{feats})}$
- Eq. 3.7 – $\text{Biaffine}^{(\text{arc})}$
- Eq. 3.8 – $\text{Biaffine}^{(\text{label})}$

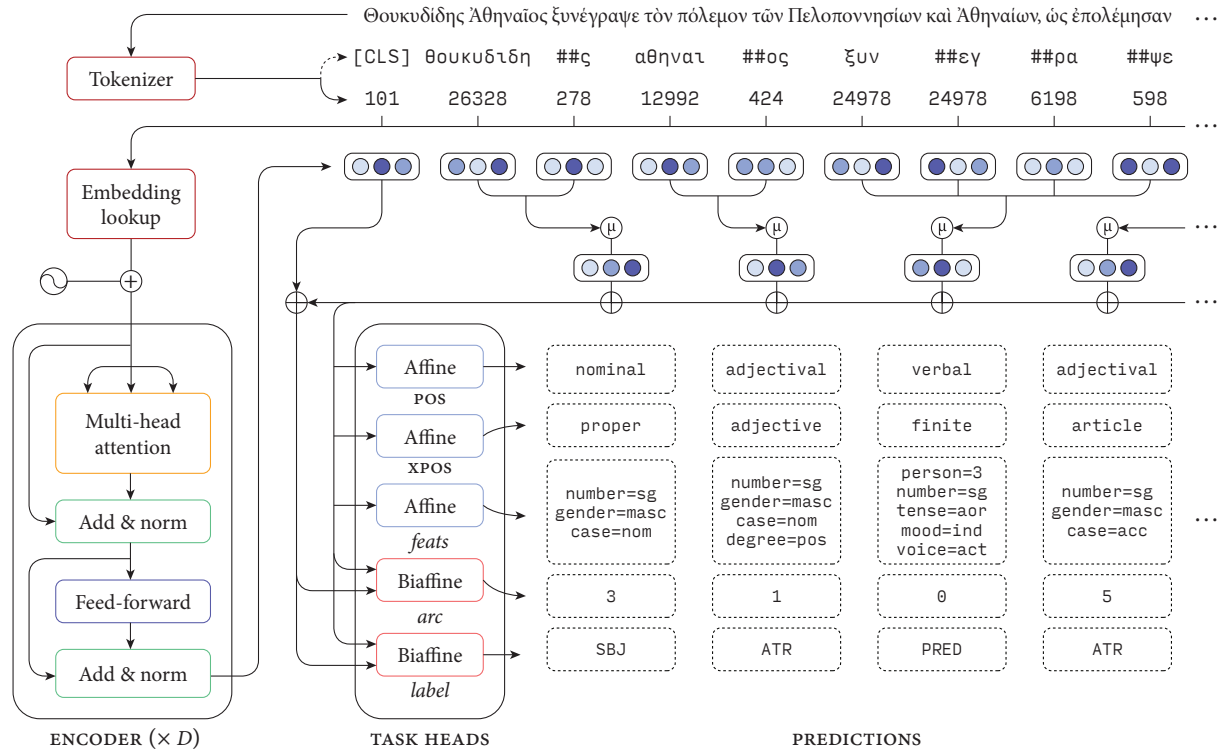



FIGURE 3.1. Model architecture for the joint tagger and parser. D is the number of encoder layers. Rounded rectangles with colored circles represent embeddings of length H (the hidden size of the transformer). The circle with a curved line (\odot) represents the learned positional embeddings that are added to each token embedding. A circle with a plus sign (\oplus) denotes addition, crossed circles (\oplus) denote concatenation, and circles with a μ (μ) denote averaging.

4 RESULTS & DISCUSSION

N THIS CHAPTER, we present the results of the experimentation that was described in Chapter 3, and compare them to existing state-of-the-art benchmarks. We evaluate the results with respect to the hypotheses put forth regarding what can lead to improvements in Ancient Greek tagging and parsing, namely: joint modeling (Section 2.4), expanding Greek diacritics into Unicode (Section 3.1.4), and using specialized pretrained models (Section 2.6.2).

4.1 BENCHMARKS

Our benchmarks are comprised of systems with previous state-of-the-art results on our experimental datasets. For the papyrus dataset, we can only compare our results with those that were achieved by Keersmaekers (2020), as this thesis is the first to evaluate on this dataset since its release.¹ For the UD datasets, we measure against two major NLP tools that share the current state-of-the-art performances on our relevant metrics between each other: *Stanza* (Qi, Y. Zhang, et al., 2020) and *Trankit* (Nguyen et al., 2021).

Stanza. *Stanza* is a multilingual toolkit that offers pipelines, processors, and pretrained models for 66 languages on various NLP tasks, including Ancient Greek tagging and parsing. Underlying both the tagging and parsing components are bidirectional LSTM networks trained on UD treebanks with *word2vec* (Zeman et al., 2018) and *fastText* embeddings (Bojanowski et al., 2017). The tagger and parser architectures are similar to ours (including biaffine attention), but with some further augmentations (Qi, Y. Zhang, et al., 2020, pp. 102–103). We report the official numbers as of version 1.3 on UD version 2.8.²

¹While it is theoretically possible to train the other systems on the papyrus data, it is not immediately feasible since the data is not fully UD-compliant, which is a superficial requirement of the training procedures of the toolkits. For instance, the papyrus dataset contains sentences with multiple nodes attached to the root, which breaks the in-built evaluation scripts that are run during training.

²Numbers can be found here: <https://stanfordnlp.github.io/stanza/performance>

Trankit. *Trankit* is another multilingual toolkit. As opposed to *Stanza*, it is transformer-based, making use of XLM-ROBERTa (Y. Liu et al., 2019). Like its competitor, *Trankit* takes a pipeline approach with pretrained models, including support for Ancient Greek tagging and parsing. Again, the tagger and parser architectures are comparable to ours, with a few differences. Notably, *Trankit* implements *adapters*—smaller neural networks inserted inside the encoders—keeping the transformer layers frozen and updating only the adapter and task head weights. The performance was evaluated with version 1.0.0, which supports the large, 24-layer version of XLM-ROBERTa, as opposed to the base model used in earlier versions. In order to have comparable results, we trained three customized *Trankit* pipelines on each UD dataset (as of version 2.8) using default parameters, and report the average accuracies after evaluating each using the official evaluation script.³

4.1.1 *Papyrus dataset*

In Section 2.6.1 we reviewed some of the technical details and motivation behind the papyrus annotation project by Keersmaekers (2020), and we discussed the so-called *papyrus dataset*, which is a large, homogenized dataset of more or less all available morphologically and syntactically annotated Ancient Greek, including papyri. We will now summarize some of the findings presented in this research with regards to tagging and parsing.

For grammatical tagging Keersmaekers (ibid.) found the best results using the HMM-based *RFTagger* (Schmid and Laws, 2008) supplied with a morphological lexicon based on *Morpheus* with a few manual additions (Keersmaekers, 2020, p. 28). It is important to note that the tagging experimentation was not trained and evaluated on the papyrus dataset, but instead another, smaller subset consisting of only papyri (ibid., p. 27). As such, the numbers are not directly comparable to our own experimentation—which uses the the papyrus dataset for both tagging and parsing—but we state them here nonetheless. The reported accuracy is 94.7%, which is the percentage of predictions (excluding punctuation and textual gaps⁴) that have both part-of-speech *and* morphological information correct. This definition corresponds to the metric “*all tags*” in the coming tables.

Syntactic parsing, however, is evaluated on the same test set as our own models. For this task Keersmaekers (ibid.) utilizes the *Stanford Graph-based Neural Dependency Parser* (Dozat, Qi, and Manning, 2017), which is once again just a slight variation of the same architecture that we employ. The main difference is that it uses LSTM embeddings, and that it incorporates part-of-speech information directly in

³See Appendix A.2 for more information about the *Trankit* fine-tuning process.

⁴The papyrus data contains sentences where certain tokens are missing due to damage or other corruptions in the archaeological documents. In the missing places, special “gap” tokens are inserted. These are given a “GAP” label in the POS, XPOS and *deprel* columns.

Strategy	UAS	LAS	LA
Unicode	84.3	78.4	84.4
Beta Code	84.8	79.3	84.9
Beta Code + morphological features	85.9	79.5	85.4

TABLE 4.1. Accuracies of the parsing experiments by Keersmaekers (2020) on the papyrus dataset.

them. More specifically, the input embeddings to the parser are made up of pre-trained *word2vec* embeddings and trainable word-level and character embeddings summed together, concatenated with summed UPOS and XPOS embeddings.

The resulting accuracies (excluding punctuation and gaps) are summarized in Table 4.1. As the table shows, the performance increases when the data are fed to the parser in Beta Code, as opposed to Unicode. We have already discussed encodings in Section 3.1.3, where we also explain that our model is only able to make use of Unicode-encoded Greek. For this reason, we report both numbers, since the Unicode comparison is more direct. Furthermore, even better results are reported when the inputs to the original UPOS embeddings are replaced with the morphological features column (*feats*) instead. It has to be noted that Keersmaekers (2020) does make a number of changes to the dataset (such as rethinking coordination and reducing elliptical structures) which further improve parser performance, but these modified datasets are only provided in Beta Code, and have as such not been evaluated in our own experimentation. Furthermore, these modifications are really only changes to the underlying data, and should not necessarily be of importance to general performance of the system.

4.1.2 PROIEL (UD)

Table 4.2 shows the performance of *Stanza* and *Trankit* on the UD version of PROIEL, with the highest values in bold. *Stanza* is marginally better at most tagging metrics, while *Trankit* severely outperforms its competitor in parsing. For the record, PROIEL does not contain punctuation.

	UPOS	XPOS	<i>feats</i>	<i>all tags</i>	UAS	LAS
<i>Stanza</i>	97.38	97.74	92.20	91.03	81.27	77.38
<i>Trankit</i>	97.54	97.66	92.18	90.91	87.48	83.79

TABLE 4.2. Accuracies of *Stanza* and *Trankit* on PROIEL (UD).

	UPOS	XPOS	<i>feats</i>	<i>all tags</i>	UAS	LAS
<i>Stanza</i>	92.36	84.95	91.03	84.75	79.26	73.65
<i>Trankit</i>	93.04	84.99	90.07	84.13	81.27	75.82

TABLE 4.3. Accuracies of *Stanza* and *Trankit* on AGDT (UD).

4.1.3 AGDT (UD)

The performances of the two benchmarks on UD-converted AGDT are shown in Table 4.3. In general, accuracies are much lower for AGDT than for PROIEL in both systems. Once again, *Stanza* seems to be better at morphological tagging, while *Trankit* is the clear winner when it comes to parsing. It is to be noted that these scores include punctuation, which is likely to inflate the results to some degree, but is nonetheless the convention when evaluating on UD.

4.2 FINE-TUNING

We trained all models for 50 epochs, with early stopping if there were 5 epochs after which the validation loss had not decreased. Most models converged after around 10–15 epochs, but those based on mBERT usually took longer, whereof one reached the epoch limit without converging. All runs were instantiated with a batch size of 8, and we used set RNG seeds during all experiments for reproducibility.⁵ Furthermore, we fixed the optimizer, *Adam* (Kingma and Ba, 2014), to linearly decay the learning rate after 1000 warmup steps in order to try to prevent avoid early overfitting. Similarly to Dozat and Manning (2017) we also set the dropout probability on the last hidden state of BERT to 0.3. Initial experiments also showed that this was more beneficial compared to less regularization. The models were trained on GPUS, and were implemented in *PyTorch* version 1.9.0 (Paszke et al., 2019) using version 4.12.3 of the transformers module (Wolf et al., 2020). All hyperparameter optimization was done based on the performance of AGBERT on the validation data of the papyrus dataset; we did not optimize for different pretrained models or datasets. All UD experiments use the official treebank data as of version 2.8 (Zeman et al., 2021).

4.2.1 Learning rates

We found that the learning rates for both the tagger and parser need to be quite low for the models not to immediately diverge. Initial experiments based on the validation loss showed that a stable value could be found at around 3×10^{-6} for all

⁵See Appendix A.1 for more information about our configuration.

	BERT	Tag heads	Parse heads
Tagger	3×10^{-6}	8×10^{-5}	–
Parser	2×10^{-6}	–	1×10^{-6}
Joint model	3×10^{-6}	8×10^{-5}	1×10^{-6}

TABLE 4.4. Optimized discriminative learning rates for the tagger, parser and joint model.

model parameters. For parsing specifically, 2×10^{-6} gave slightly better results. We therefore opted for these learning rates for the BERT parameters of all tagging and parsing models. For the joint model, we also stayed with 3×10^{-6} .

Applying discriminative learning rates adjusted to each task head showed further improvements. From a search space between 10^{-5} and 10^{-4} , we found that the tagger had the lowest validation loss with classifier learning rates of 8×10^{-5} . For the parser, the lower 10^{-6} proved most efficient, after exploring values from between 6×10^{-7} and 6×10^{-6} . For the joint model, we took the most effective learning rates from the individual parsing and tagging experiments and applied them to the parsing and tagging components separately. Table 4.4 shows the chosen discriminative learning rates of each model.

4.2.2 Expanding diacritics

In Section 3.1.4, we explained two strategies for keeping some of the information that is stripped away when Greek letters are normalized intact. The first was to append iotas to vowels that have subscripted iotas, and the second was to prepend a *heta* to those words where any vowel had rough breathing with no consonants preceding it. In order to evaluate the effectiveness of these strategies, we compare the performance of models that were trained with and without them, using the same hyperparameters as explained earlier.

Figure 4.1 shows the validation loss of the various strategies on the papyrus dataset using AGBERT. In tagging, a small benefit can be traced from the consistently lower validation loss. In parsing, however, the advantage is less consistent. In all cases the lowest validation loss occurred when applying both strategies, followed by expanding iotas, expanding rough breathing marks, and lastly, leaving them out.

Once we evaluate the different strategies on the papyrus test set, we can see that the effect is noticeable, albeit somewhat inconsistent. Table 4.5 shows the average results of the AGBERT-based tagger and parser trained separately over three runs per strategy. For the tagger we can see a slight benefit from both strategies on all metrics, with the largest improvement in morphological tagging (*feats*), where the accuracy increases by 0.17 percentage points by expanding iotas, which, in turn,

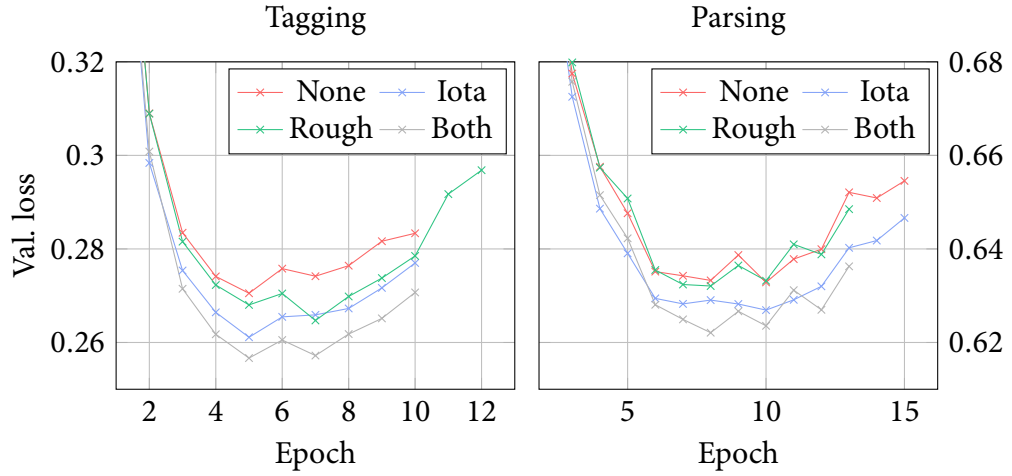


FIGURE 4.1. Validation loss values of the AGBERT-based models that reached the lowest validation loss on the papyrus dataset using different diacritic-expansion strategies.

results in a 0.19 point improvement in the *all tags* metric combined with the increase in POS and XPOS tagging. Notably, the expanded iotas help most with morphological features (+0.17), while expanding the rough breathing marks helps most with POS and XPOS tagging (+0.06 and +0.08, respectively), as do the iotas (+0.03 and +0.01, respectively). While these numbers may seem small, given the large size of the test set (71,280 tokens), even a 0.01 point increase implies that some 71 additional tokens are correctly analyzed.

If we look at the confusion matrices of the best performing taggers (with and without expanded iotas) on morphological features we can inspect the differences closer. There is one grammatical category that is particularly likely to be affected

	Tagger				Parser		
	POS	XPOS	<i>feats</i>	<i>all tags</i>	UAS	LAS	LA
No expanding	98.78	98.02	95.02	94.31	86.89	80.75	86.97
Expanded iota	98.81	98.03	95.19	94.50	86.89	80.76	87.05
Expanded rough	98.84	98.10	95.13	94.46	86.75	80.55	86.91
Both	98.87	98.11	95.14	94.49	86.76	80.58	86.95

TABLE 4.5. Accuracies of the AGBERT-based tagger and parser on the papyrus dataset with different diacritic-expansion strategies. Numbers are the average over three runs.

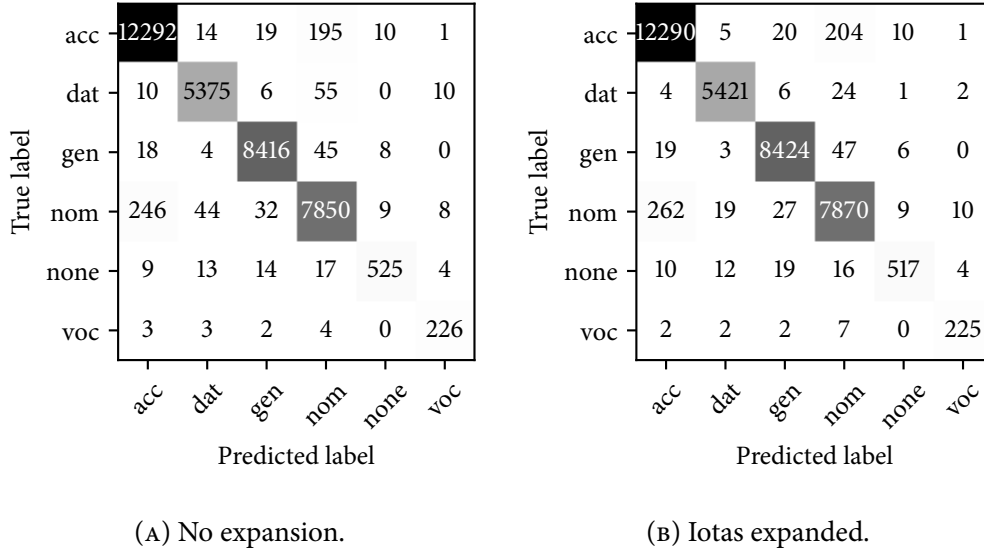


FIGURE 4.2. Confusion matrices of the case predictions by the best performing taggers on the papyrus dataset.

by the expansion of iotas; namely, the dative. The dative is often characterized only by a subscripted iota, such as ἀνθρώπω [*ánthrōpō*] (M/F.NOM/ACC.DU) vs. ἀνθρώπῳ [*ánthrōpōi*] (M/F.DAT.SG) or σελήνῃ [*selénē*] (F.NOM.SG) vs. σελήνῃ [*selénēi*] (F.DAT.SG). As Figure 4.2 shows, the dative is correctly recognized in 46 more cases when iotas are expanded. A closer inspection shows that of the 81 tokens that were not correctly recognized as dative by the tagger without expansion, 48 have, or agree with words that have, final iota subscripts. After expansion, this number is down to 3 (out of a total of 37), showing that except for one token, the improvement can be directly traced to this phenomenon. Notably, the tagger also becomes better at recognizing genitives and nominatives (8 and 20 additional tokens, respectively). Furthermore, we can look at the matrices for POS and XPOS. Regular POS predictions are shown in Figure 4.3, while we refer to figures B.1 and B.2 in Appendix B for the XPOS predictions. As the figures show, the POS category adjectival is recognized correctly in 63 more instances than before, the XPOS category adjective, 44 more times, and also verbal, 19. Similar results are seen for the XPOS categories common and coordinator as well. Somewhat surprisingly, the performance deteriorates in the nominal POS category and the personal and proper XPOS categories. This is most likely due to the tagger’s newfound knack for predicting nominal-like words as adjectives, as the figures also show.

Likewise, we review the error rates of the best performing tagger with rough breathing marks expanded, as well as using both strategies. As became apparent

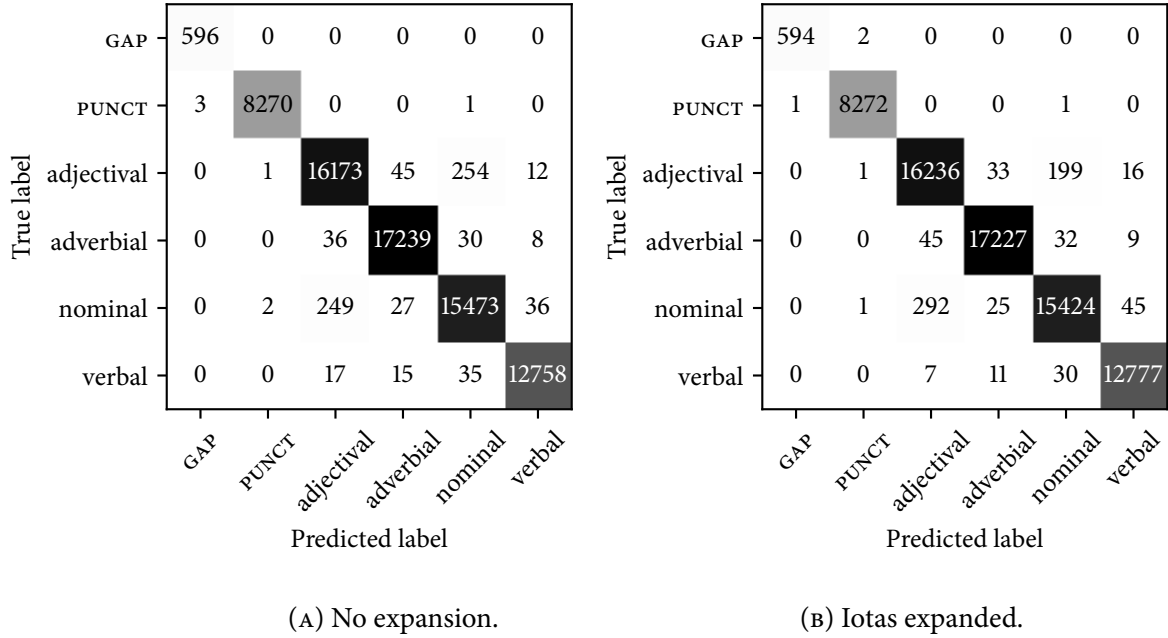
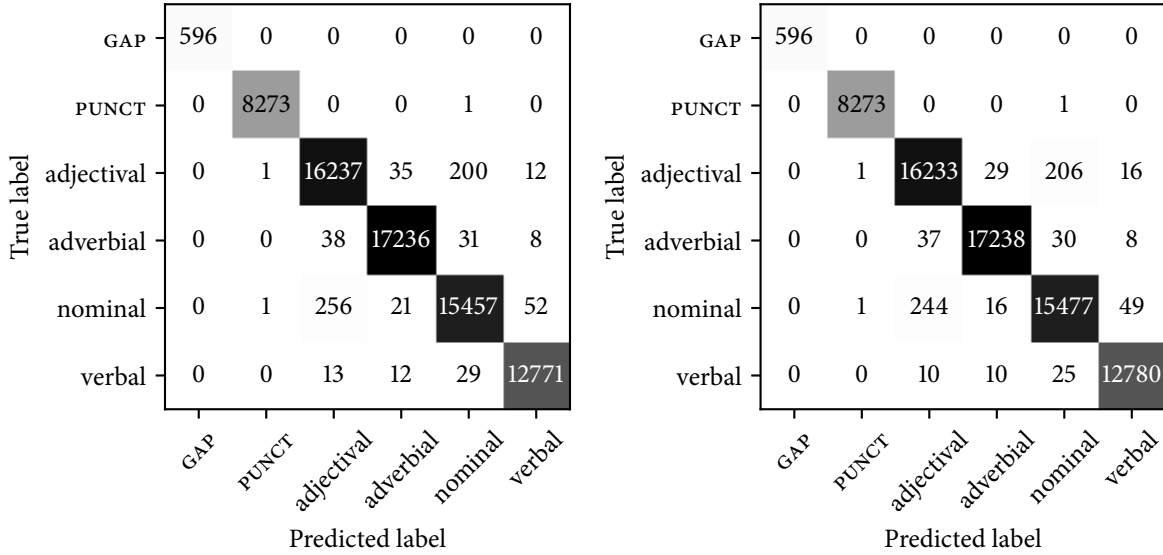


FIGURE 4.3. Confusion matrices of the POS predictions by the best performing AGBERT-based taggers with and without expanded iotas on the papyrus dataset.

in Table 4.5, improvement is seen in all tagging metrics when expanding rough breathing marks, while applying both strategies is best with POS and XPOS, and falls somewhere in between when it comes to *feats*. Figure 4.4 shows the confusion matrix of the POS predictions by the tagger with expanded rough breathing marks, and we once again refer to Appendix B and figures B.3 and B.4 for the XPOS matrices. The numbers show that, like with expanded iotas, the tagger becomes better discerning adjectives from nominals. Furthermore, it is not as prone to overpredicting adjectives at the expense of nominals as the tagger with expanded iotas was, and in fact improves accuracy on nominals too. Finally, applying both strategies makes it possible to reap all the benefits, while avoiding the problems with the individual strategies that were just highlighted. Except for adjectives, where it is beat by the tagger with only expanded rough breathing by a single token, it outcompetes all other options.

The parsers, meanwhile, as Table 4.5 shows, get a slight advantage on all metrics by incorporating iotas, with a 0.08 point increase in label accuracy and a 0.01 increase in LAS. Expanding the rough breathing marks, however, is somewhat detrimental, with a 0.20 point decrease in LAS compared to no expansion, and applying both strategies falls somewhere in between. We hypothesize that the expanded diacritics may interfere too much with the syntactic expectations of



(A) Rough breathing marks expanded.

(B) Both strategies.

FIGURE 4.4. Confusion matrices of the POS predictions by the best performing AGBERT-based taggers expanded rough breathing marks and both strategies on the papyrus dataset.

the model that it has derived from pretraining, given that it has been trained on sentences without diacritics, and that its unsupervised learning of syntax during pretraining is most likely the main component to its performance. While adding new tokens, as the previous experimentation provides basis for arguing, may easily be fine-tuned to improve tagging, it seems that it is not as feasible when it comes to syntax.

The experiments with diacritic expansion strategies show that the models are able to make use of the information that the diacritics provide, once incorporated into tokenizable text. While the results are not always completely predictable—especially when it comes to parsing—we nonetheless include both strategies in all further experiments.

4.3 MULTI-TASK LEARNING

In Section 2.4, we hypothesized that interaction between the tagging and parsing components is especially important for a highly inflected language with free word order like Ancient Greek. To explore whether this is the case, we compare the

	POS	XPOS	<i>feats</i>	<i>all tags</i>	UAS	LAS	LA
Tagger	98.87	98.11	95.14	94.49	–	–	–
Parser	–	–	–	–	86.76	80.58	86.95
Joint	98.60	97.91	94.85	94.18	86.75	80.63	87.03
Δ	–0.27	–0.20	–0.29	–0.31	–0.01	+0.05	+0.08

TABLE 4.6. Accuracies of the AGBERT-based models on the papyrus dataset.

	UPOS	XPOS	<i>feats</i>	<i>all tags</i>	UAS	LAS	LA
Tagger	98.11	98.29	93.22	92.13	–	–	–
Parser	–	–	–	–	86.76	84.05	90.27
Joint	98.15	98.33	92.76	91.69	86.95	84.57	90.90
Δ	+0.04	+0.04	–0.46	–0.44	+0.19	+0.52	+0.63

TABLE 4.7. Accuracies of the AGBERT-based models on PROIEL (UD)

	UPOS	XPOS	<i>feats</i>	<i>all tags</i>	UAS	LAS	LA
Tagger	94.49	88.62	93.05	87.95	–	–	–
Parser	–	–	–	–	84.30	79.84	89.12
Joint	94.56	88.43	92.71	87.59	83.10	78.61	88.58
Δ	+0.07	–0.19	–0.34	–0.36	–1.20	–1.23	–0.54

TABLE 4.8. Accuracies of the AGBERT-based models on the AGDT (UD).

performance of separately trained taggers and parsers with that of joint models that perform both tasks simultaneously. Tables 4.6, 4.7, and 4.8 show the performance of AGBERT-based taggers, parsers, and joint models on the three datasets. The results are the average over three runs. Following our observations in Section 4.2.2, all models were trained using both diacritic expansion strategies.

The largest improvement in parsing is seen in PROIEL (Table 4.7), which has a 0.19 point boost in UAS, 0.52 in LAS, and 0.63 in label accuracy when tagging and parsing is done jointly as opposed to separately. In fact, even POS and XPOS tagging is slightly improved in PROIEL, by 0.04 points for both metrics. The setbacks occur in *feats* and *all tags*, where there is a 0.46 and 0.44 point decrease in accuracy, respectively. As the test set of PROIEL is quite small (13,314 tokens), we must have a higher threshold for interpreting these numbers as very significant (0.01 points equals a little more than one token).

	acl	advcl	advmod	amod	appos	case	cc	ccomp	conj	cop	csbj	det	disc	disl
Parser	114	531	603	178	77	874	984	120	686	202	2	2217	722	3
Joint	122	534	617	187	79	878	989	113	676	205	1	2228	724	1
Δ	+8	+3	+14	+9	+2	+4	+5	-7	-10	+3	-1	+11	+2	-2

	iobj	mark	nmod	nsbj	:pass	num	obj	obl	:agen	orph	para	root	voc	xcomp
...	386	211	418	781	69	63	843	705	14	11	3	978	52	162
...	382	213	420	780	78	65	846	714	13	9	2	991	54	173
Δ	-4	+2	+2	-1	+9	+2	+3	+9	-1	-2	-1	+13	+2	+11

TABLE 4.9. Number of correct *deprel* predictions per relation type by the AG-BERT-based parser and joint model on PROIEL. Relation names have been slightly abbreviated, and unchanged results omitted. Relations subtypes (those starting with colons) have been detached and placed on the right-hand side of the original relation that they modify.

The effects of joint modeling on parsing can be observed more closely in Table 4.9. The full confusion matrices that the table and the coming discussion is based on can be seen in figures B.5 and B.6 in Appendix B. As we can see, the accuracy increases for 19 out of 32 relations, while it decreases for 8 and stays unchanged for the rest (these are omitted from the table). The five categories that see the largest relative improvement are *acl* (*adnominal clause*, +7.0%), *amod* (*adjectival modifier*, +5.1%), *nsbj:pass* (*passive nominal subject*, +13.0%), *vocative* (*vocative*, +3.8%), and *xcomp* (*open clausal complement*, +6.8%). Adnominal clauses are most often confused with adverbial clause modifiers (*advcl*), which is intuitive, given that the only thing that differentiates these two classes is the part of speech of the head. Complimentarily, adjectival modifiers, which are often mistaken for nominal modifiers (*nmod*), are discernable by their own part of speech. Similarly, the vocative relation more or less correlates entirely with case when it comes to nominals, and the *pos* tag when it comes to interjections. Moving on, for passive nominal subjects, it is central to be aware of the voice of the governing verb (namely, that it is passive) in order to separate the usage from regular nominal subjects *nsbj*. Finally, open clausal complements are often confused with regular clausal complements (*ccomp*). While the benefit of joint tagging is not as conveniently explained in this case, we hypothesize that it may be at least because the tagging contributes in disambiguating finite verbs from infinitives, given that, for instance, the aorist infinitive ending -σαι [-*sai*] is the same as the 2nd person singular middle primary tense ending, as well as the 3rd person singular aorist optative. Finite verbs are more likely to take the *ccomp* relation, and almost never *xcomp*. It may be that the parser in these instances, when coming across for instance λῦσαι [*lûsai*], which

is tokenized as ($\lambda\nu$, $\#\sigma\alpha\iota$), analyzes its function as that of a finite verb instead of an infinitive, both of which are morphologically valid analyses, and thereby misses out on the correct parse. Given the morphological information, however, it is able to make the distinction in favor of the correct syntactic structure.

For the other datasets, however, the trends are less exciting. On AGDT, the joint model only outperforms the separate models on POS tagging with 0.07 points, and otherwise falls 0.19–1.23 points below them. The papyrus dataset similarly degrades in tagging accuracy, but gets a slight increase (0.11 and 0.08 points) in parsing.

The results are, as shown, varying, but there are nonetheless clear indications from the PROIEL results that informedness about tagging can have a direct impact on parsing performance. It is worth reporting that the exact same trends become apparent even when replacing AGBERT with GREEK-BERT or mBERT. In fact, the benefits of joint parsing are generally slightly more pronounced in the two latter cases. In conclusion, even though the joint models are not always the best performing, we base the remaining experiments on them in order to avoid too much complexity. While we could have reported the best results from the different models as they happen to occur without taking up too much extra space, for the purposes of tidyness and direct comparison we instead stick with the joint models, still applying both diacritic expansion strategies.

4.4 DOMAIN ADAPTION

After all hyperparameter values and additional strategies have been chosen for AGBERT-based models, we similarly evaluate the performance of fine-tuned GREEK-BERT and mBERT on all datasets with the same settings. Tables 4.10, 4.11, and 4.12 show how changing the pretrained models affects performance in joint models on the papyrus dataset. Except for the AGBERT numbers on the papyrus dataset, which are the averages over three runs, all numbers are from single runs.

As the numbers show, the AGBERT-based models clearly outcompete those based on GREEK-BERT and mBERT, particularly when it comes to parsing. All models are able to achieve decent results, but there is a visible correlation between domain-specific pretraining and increased accuracy in all metrics. To highlight the differences, the same numbers are illustrated as bar charts in Figure 4.5.

First of all, there is a considerable discrepancy between datasets. For the papyrus dataset, the difference in POS and XPOS tagging accuracy is more or less negligible across the pretrained transformer models, and even including morphological features and the *all tags* metric, the average difference across tagging metrics between AGBERT and mBERT only makes out to be 0.69 points. The actual gains that domain-specific pretraining provides are more evident in parsing, but even then, there is only an average difference of 2.57 percentage points between the models. Shifting

	POS	XPOS	<i>feats</i>	<i>all tags</i>	UAS	LAS	LA
AGBERT	98.60	97.91	94.85	94.18	86.75	80.63	87.03
GREEK-BERT	98.38	97.47	93.91	93.14	85.27	78.71	85.70
mBERT	98.41	97.60	93.75	93.03	84.34	77.51	84.86

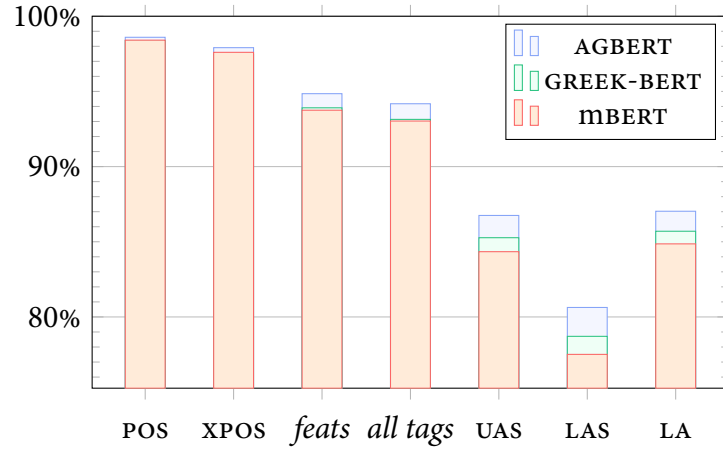
TABLE 4.10. Accuracies of the joint models on the papyrus dataset.

	UPOS	XPOS	<i>feats</i>	<i>all tags</i>	UAS	LAS	LA
AGBERT	98.15	98.33	92.76	91.69	87.95	84.57	90.90
GREEK-BERT	97.42	97.57	91.14	89.80	85.92	82.09	89.23
mBERT	96.77	97.14	89.76	88.10	83.45	78.77	87.30

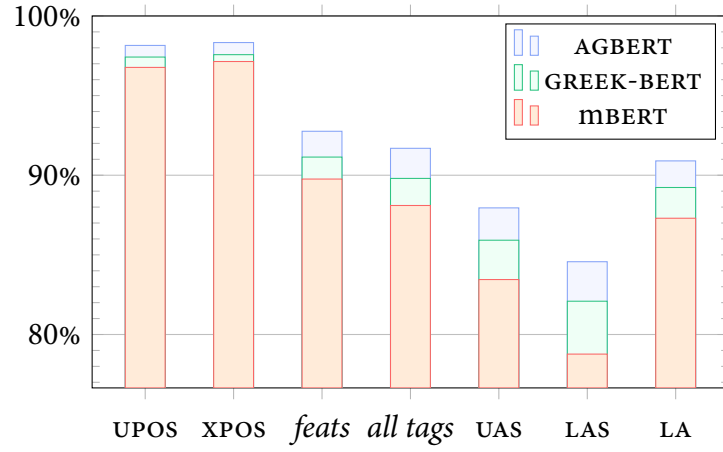
TABLE 4.11. Accuracies of the joint models on PROIEL (UD).

	UPOS	XPOS	<i>feats</i>	<i>all tags</i>	UAS	LAS	LA
AGBERT	94.56	88.43	92.71	87.59	83.10	78.61	88.58
GREEK-BERT	92.50	84.38	89.65	83.12	78.21	72.81	84.94
mBERT	92.07	83.33	88.79	82.01	75.30	69.41	82.60

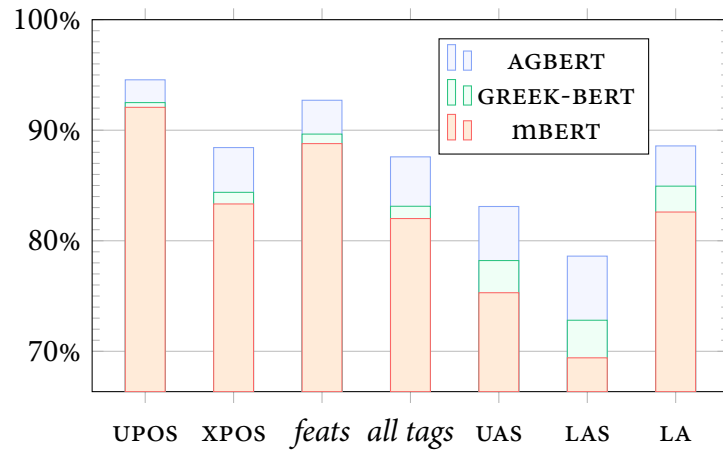
TABLE 4.12. Accuracies of the joint models on AGDT (UD).



(A) Papyrus dataset



(B) PROIEL (UD)



(C) AGDT (UD)

FIGURE 4.5. Charts comparing the accuracies of the joint models on all datasets.

the attention to PROIEL, we start to see larger differences also in tagging. There is now an average tagging accuracy difference of 2.29 points between AGBERT and mBERT, and a 4.64 point difference in parsing. The absolute largest boosts occur in AGDT, where there is an average 4.27 point accuracy increase in tagging, and an average 7.65 point increase in parsing.

One reason for why the papyrus dataset has much more consistent results independently from the pretrained models may be that the dataset is quite large—it has about 4–6 times more training data than both PROIEL and AGDT—allowing the fine-tuning to be highly impactful. Moreover, the manual effort to increase the consistency of the combined data may have resulted in a dataset that is easier to parse than the two UD datasets, which have only undergone automated conversion. Additionally, the genres represented in the test split of the papyrus dataset are especially homogeneous, given that almost half of it is papyri. Papyri are also usually simpler syntactically than for instance classical prose and, especially, poetry. Keersmaekers (2020, p. 43) points out, for instance, that only 13% of the papyrus sentences in the training partition contain non-projective arcs, while the number is 41% for the entire partition.

The data contained in PROIEL is also arguably easier to parse than what is contained in the AGDT. While Herodotus with his slightly more challenging classical prose is also a part of it, two thirds of PROIEL is New Testament Greek, which usually has a simpler style than earlier prose. The AGDT on the other hand contains archaic and classical poetry and prose in abundance, which often has a high degree of stylistic intricacy contributing to syntactic complexity. Conversely, some conditions regarding the UD-conversion of the AGDT should work in favor of it being easier to parse, namely, that it contains no elliptical structures. Nonetheless, it is here that the AGBERT really gets to shine.

Even just using GREEK-BERT is also a significant improvement over the multilingual mBERT, showing that domain adaption between Modern and Ancient Greek is a fruitful avenue of exploration, and that the decision by Singh, Rutten, and Lefever (2021) to initialize AGBERT from GREEK-BERT’s weights was well-motivated. In summary, using a specialized BERT model for Ancient Greek ends up being the decisively best option, particularly when it comes to syntactic parsing and possibly when handling difficult genre components.

4.5 ERROR ANALYSIS

In addition to our remarks throughout the previous sections, we refer to Appendix B for additional figures highlighting the behavior of our systems for purposes of error analysis. As noted in Section 3.1.1, many errors can likely be attributed to inconsistent or incomplete conversions of the UD datasets. For the papyrus dataset, we will refer

to Keersmaekers (2020, p. 64), who after manually analyzing 500 parsing errors discovered that almost half could be owed to either consistency issues, annotation errors, bugs, or ambiguity.

4.6 COMPARISON

We now revisit and summarize the results that have been presented so far. Tables 4.13, 4.14, and 4.15 show the performance of our joint models compared to the current state-of-the-art benchmarks on the different datasets. Our own results are the average over three runs. It is worth mentioning that our accuracies, unlike how it is done in the official UD scripts, are calculated *including* relation subtypes (such as `nsubj:pass`). As *Stanza* and *Trankit* are evaluated using the official UD scripts which only take into account the left-hand side of such labels, our results would actually be a bit higher with the exact same evaluation, but the difference is most likely negligible.

It was stated in Section 4.1.1 that Keersmaekers’ tagging results are based on another test set, and are therefore not directly comparable to our own. We can nonetheless see that the performance is similar, but since we test on more data, our results are likely to be more consistent. When it comes to parsing, we see a 0.9 percentage point increase in UAS, a 1.1 point increase in LAS, and a 1.6 point increase in LA compared to the best benchmark model on the same dataset. The reasonable LA score also shows that removing empty nodes and replacing their function with composite tags is not an overtly bad strategy.

When it comes to PROIEL, we see consistent improvements over the benchmarks across all metrics. Tagging is generally improved with around 0.6 points, with the largest jump in the *all tags* metrics at 0.66 points. Parsing is similarly improved, with a 0.47 point boost in UAS, and 0.78 in LAS, surpassing the benchmarks by a reasonable margin.

The largest gains are seen in AGDT, where our systems experience a 3.44 point increase in XPOS tagging, which correspond to the original, fully morphological AGDT *postags* that we explained in Section 3.1.2, compared to the best benchmark. Likewise, POS and *feats* increase with 1.52 and 1.68 points, respectively, leading to a total increase in the *all tags* metric of 2.84 points. If we compare the performance to that which was obtained on PROIEL, we can see that numbers are starting to come closer than before. While there used to be an over 1 percentage point difference in morphological feature tagging between PROIEL and AGDT in the benchmarks, this number is now down to 0.05 points using our model. Likewise, we see that the former 4.5 point difference in POS tagging is down to 3.59, and in parsing, from a 6.21 point difference down to 4.85 in UAS, and from 7.97 to 5.96 in LAS. This may be an indication that there are differences between the datasets which make the

parsing of AGDT harder (e.g., more poetry), and that more domain-specific training is required in order to handle this increased complexity. The xPOS results are still lagging somewhat behind, most likely due to them being more complex compared to the other datasets, since they conflate both POS and morphological features into a single tag.

	POS	xPOS	<i>feats</i>	<i>all tags</i>	UAS	LAS	LA
Unicode	–	–	–	(94.7)	84.3	78.4	84.4
Beta Code	–	–	–	–	84.8	79.3	84.9
Beta Code + morph. features	–	–	–	–	85.9	79.5	85.4
Joint AGBERT	98.6	97.9	94.8	94.2	86.8	80.6	87.0
Δ					+0.9	+1.1	+1.6

TABLE 4.13. Our results on the papyrus dataset compared to Keersmaekers (2020) with different encoding strategies.


	UPOS	xPOS	<i>feats</i>	<i>all tags</i>	UAS	LAS
<i>Stanza</i>	97.38	97.74	92.20	91.03	81.27	77.38
<i>Trankit</i>	97.54	97.66	92.18	90.91	87.48	83.79
Joint AGBERT	98.15	98.33	92.76	91.69	87.95	84.57
Δ	+0.61	+0.59	+0.56	+0.66	+0.47	+0.78

TABLE 4.14. Our results on PROIEL (UD) compared to *Stanza* and *Trankit*.

	UPOS	xPOS	<i>feats</i>	<i>all tags</i>	UAS	LAS
<i>Stanza</i>	92.36	84.95	91.03	84.75	79.26	73.65
<i>Trankit</i>	93.04	84.99	90.07	84.13	81.27	75.82
Joint AGBERT	94.56	88.43	92.71	87.59	83.10	78.61
Δ	+1.52	+3.44	+1.68	+2.84	+1.83	+2.79

TABLE 4.15. Our results on AGDT (UD) compared to *Stanza* and *Trankit*.

5 CONCLUSION

UR EXPERIMENTS HAVE SHOWN that leveraging a language-specific BERT model is highly advantageous for Ancient Greek tagging and parsing. Our joint models with diacritic expansion strategies achieve state-of-the-art results in all of the datasets that we evaluate on. In this chapter, we summarize our specific contributions and present our ideas for future work, highlighting certain takeaways from our findings with regard to three perspectives: transformer models, the Ancient Greek language, and the tasks of tagging and parsing.

5.1 SUMMARY OF CONTRIBUTIONS

Our main goal was to improve the results for Ancient Greek tagging and parsing. We did this by combining a specialized Ancient Greek BERT model (Singh, Rutten, and Lefever, 2021) with high-performing task-specific architectures (Dozat and Manning, 2017; Glavaš and Vulić, 2021), and assessed the effectiveness by evaluating our system on three different datasets (Bamman and Crane, 2011; Haug and Jøhndal, 2008; Keersmaekers, 2020). Our secondary goal was to test past propositions that joint modeling can be a particularly effective strategy for Ancient Greek tagging and parsing due to the inflectional nature of the language. We have shown that in some datasets, there is a significant benefit from this approach, and we have demonstrated the specific ways in which these improvements unfold with linguistic justifications. In addition to achieving our two principal goals, we recognize the following contributions:

- We publish the code for a tagger and parser system that can be trained separately and jointly with any BERT model, and with a high number of hyperparameter options out of the box. We additionally provide language-specific methods for Ancient Greek, namely, a preprocessing scheme that makes the three datasets compatible with the AGBERT tokenizer, and methods for incorporating diacritics into the model. A link to the code and more details about can be found in Appendix A.

- We validate the usefulness of a monolingual BERT model for Ancient Greek, and demonstrate that domain-specific pretraining is of considerable benefit to downstream tasks, particularly when it comes to datasets that contain more syntactically complex genres.
- We show that diacritics are significant to tagging and parsing accuracy for Ancient Greek, and propose a novel way of incorporating them into existing BERT models.
- We perform extensive hyperparameter optimization for our models and report our findings.

5.2 FUTURE WORK

The various topics that this thesis touches on hit a broad range. As such, we have identified a number of points with which future research can be preoccupied. The first subsection considers model-specific improvements that we believe could provide even better results than the ones we were able to achieve for now. The second and third subsections include some final remarks regarding manual annotations and treebanks, as well as the task of dependency parsing seen in a broader perspective.

5.2.1 *Models*

Architecture

Taggers. At an early stage, we decided to follow Kondratyuk and Straka (2019) in treating morphological features as singular labels as opposed to having one label per feature, since they reported higher accuracies with the former. This goes against what has been suggested by for instance Keersmaekers (2020, p. 31), and may be somewhat counterintuitive. One would perhaps expect that liberating the individual features from each other introduces more nuance, and allows the model to learn individual features instead of “feature collections.” Furthermore, it prevents any model from predicting combinations of features that it has not seen before. While this behavior may be unreasonable to expect from a model, it would nonetheless be interesting to see to what degree it could be able to make generalizations and suggest previously unseen combinations for tokens for which it considers it more likely than other options. The simplest approach of having separate transformations per feature has the downside that it ignores a very important aspect of feature tagging, namely that the various categories are contingent on each other. It is, for instance, not possible (in Ancient Greek) to have a value for Gender if the VerbForm is Fin, and so on. Joint training would most likely minimize the worst risks in this case

either way, but it still seems intuitive that explicit interaction between the features is something to be desired. Therefore, pursuing morphological tagging as an instance of multi-label token classification with attention between features and UPOS/XPOS tags, similar to Qi, Dozat, et al. (2018), will surely be an interesting attempt for the future.

Parsers. In our experiments we did not employ any tree decoding algorithm, even though the two UD datasets only contain trees. Although we did not evaluate the models’ predictions for well-formedness, it is possible that incorporating such an algorithm could boost the performance further on the UD datasets. On the opposite end, incorporating more flexibility by empowering our parser with the ability to insert empty nodes—such as the Enchanced UD parsers (Oepen et al., 2021)—would enable it to operate directly on the AGDT and PROIEL, and do away with potentially troublesome solutions like orphan relations (in UD) or composite tags. Working with native treebanks that are not automatic conversions would likely give us more reliable goals to work towards, as well.

Joint modeling. We have so far only experimented with one way of doing joint tagging and parsing. There have, however, been studies in other languages showing that alternative strategies can be more effective. One of these is the so-called *stack* approach, where the tagger output is used as an extra input to the parser. Zhou et al. (2020) show that this generally outperforms the *share-tight* variant that we have employed. While this does deviate somewhat from the idea that BERT could learn to make use of the shared information by itself, it would nonetheless be interesting to see whether it would cause any immediate improvements, given that we already see good results in joint tagging and parsing.

Loss criteria. In figuring out learning rates and defining loss functions we have mostly followed the general practice in similar work. But given the relative complexity of the tasks (namely, generally large tagsets), particularly in the multi-task setting, it would make sense to look further into how the optimization is done. Most notably, we would have liked to implement some form of loss weighting instead of simply adding the losses of the five different classifiers together as it is done now.

Subword pooling. As we noted in Section 3.2, Ács, Kádár, and Kornai (2021) provide convincing indications that the choice of subword pooling strategy is significant. More specifically, their results imply that using attention over the subwords is most beneficial for morphological tagging, and that using an LSTM or averaging is best for ordinary POS tagging. Furthermore, they state that “tasks that rely more on

the orthographic representation such as morphology tend to benefit from [attention]” (Ács, Kádár, and Kornai, 2021, p. 2291). In the case of Ancient Greek with its complex morphology, and in-line with the fine-tuning idea of allowing the model to learn to detect patterns itself, the attention strategy seems very appealing, and we would like to see its effects on our setup in the future.

Positional embeddings. The positional embeddings in BERT are learned and absolute. This means that for each input position, the model keeps a specific embedding that is learned during pretraining and that can be fine-tuned. Recent work by Wang et al. (2021) offers a formal investigation into the nature of positional embeddings, and finds that there also are significant advantages to draw from using *relative* positional embeddings. Given the very free word order of Ancient Greek, trying out relative positional encodings seems like yet another tempting option.

Pretraining

Our experimentation has shown that our suspicions that polytonic Greek diacritics are of significance to downstream performance were warranted (Section 4.2.2). Even though we were able to devise a way to incorporate some of this information, the approach is admittedly far from the most elegant. Preferably, the appropriate diacritics should have been incorporated already from the beginning during pretraining. The reason why this is not already the case, of course, is that AGBERT utilizes the same tokenizer as GREEK-BERT. A large part of the the former’s success is naturally owed to it being initialized with the latter’s weights, which is only possible by having the same vocabulary. Of course, diacritics neither hold the same importance nor work in the same way in Modern Greek, and have as such been left out for good reason. The approach that would be most true to the target domain would therefore be to train a model with its own tokenizer vocabulary that incorporates diacritics. Whether to include tonal accents is a choice for later, but we claim to have shown that incorporating at least rough breathing marks and subscripted iotas is necessary to achieve optimal performance. Given the amount of available data, however, this can be a challenge, and we must leave the solving of this conundrum for the future.

Transfer learning

Latin is a language that has close historical, temporal, and typological ties with Ancient Greek, in that they both retain certain morphological and syntactic qualities that are not readily found elsewhere. In the same way that AGBERT is able to leverage Modern Greek pretraining, it may be the case that Latin, for which there already exists a language-specific BERT model (Bamman and Burns, 2020), can offer a similar aid. This could come by either through fine-tuning on the same tasks in both

languages, or already at the pretraining step. In the latter case, there is once again the challenge of tokenization, and the question still remains on how exactly to construct such a multilingual “early Indo-European” model most effectively. Nonetheless, we believe that this is an interesting proposition for future research.

5.2.2 *Annotation*

Many of the special considerations and adaptations that we have had to make when it comes to Ancient Greek tagging and parsing stem from the lack of a common and consistent annotation format. Given the promising potential of using treebanks in language research, and particularly the considerable attention such projects have been given when it comes to classical languages, we would have liked to see more unified efforts pertaining to producing manual annotations. There is no particular reason why Ancient Greek should be lagging behind other languages in tagging and parsing accuracy, other than the fact that there are, admittedly, problems with the data. We have already referred to Keersmaekers’s (2020) observation that almost half of the parsing mistakes made by his system are not solely due to the parser. When, in fact, all of the data we have worked with in this thesis is the result of mostly automatic conversions, the situation we are in leaves enough to be desired. If the resources would allow it, and there is interest elsewhere, a native UD treebank for Ancient Greek would of course be warmly welcomed.

5.2.3 *Parsing*

In closing, we will make some remarks on the parsing task in general. These ideas may not be original, but we would nonetheless like to state that some reconsiderations might be made when it comes to the expectations that we set for our poor parsers. For instance, parsing is usually evaluated on a system’s ability to parse sentences one at a time, without any context. In many such cases, ambiguity can yield the task impossible due to the lack of context. One idea, therefore, is to allow the parsers to take document context into account. This would both make them more reliable, and probably more useful, in that they approach parsing with the same prerequisites as the annotators, with all of the information available, including both previous and later sentences. Furthermore, in those instances where ambiguity really is present even with context, we propose to allow the parser to offer multiple solutions and then evaluate those solutions against a *set* of gold annotations, instead of having to select a single one. The new accuracy measurement could then be a form of precision-at- k , where the parser is evaluated on its ability to make its top k scoring analyses overlap as much as possible with the k possible gold annotations.

6 BIBLIOGRAPHY

- Ács, J., Á. Kádár, and A. Kornai (2021). “Subword Pooling Makes a Difference.” In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Online: Association for Computational Linguistics, pp. 2284–2295. DOI: [10.18653/v1/2021.eacl-main.194](https://doi.org/10.18653/v1/2021.eacl-main.194).
- Bahdanau, D., K. Chu, and Y. Bengio (2014). *Neural Machine Translation by Jointly Learning to Align and Translate*. Version 7. DOI: [10.48550/arxiv.1409.0473](https://doi.org/10.48550/arxiv.1409.0473). arXiv: [1409.0473](https://arxiv.org/abs/1409.0473) [cs.CL].
- Bakker, E. J., ed. (2010). *A Companion to the Ancient Greek Language*. Blackwell. DOI: [10.1002/9781444317398](https://doi.org/10.1002/9781444317398).
- Bamman, D. and P. J. Burns (2020). *Latin BERT: A Contextual Language Model for Classical Philology*. DOI: [10.48550/arXiv.2009.10053](https://doi.org/10.48550/arXiv.2009.10053). arXiv: [2009.10053](https://arxiv.org/abs/2009.10053) [cs.CL].
- Bamman, D. and G. R. Crane (2011). “The Ancient Greek and Latin Dependency Treebanks.” In: *Language Technology for Cultural Heritage*. Ed. by C. Sporleder, A. van den Bosch, and K. Zervanou. Springer. DOI: [10.1007/978-3-642-20227-8_5](https://doi.org/10.1007/978-3-642-20227-8_5).
- Bohnet, B. et al. (2013). “Joint Morphological and Syntactic Analysis for Richly Inflected Languages.” In: *Transactions of the Association for Computational Linguistics* 1, pp. 415–428. DOI: [10.1162/tacl_a_00238](https://doi.org/10.1162/tacl_a_00238).
- Bojanowski, P. et al. (2017). “Enriching Word Vectors with Subword Information.” In: vol. 5. Cambridge, MA: MIT Press, pp. 135–146. DOI: [10.1162/tacl_a_00051](https://doi.org/10.1162/tacl_a_00051).
- Celano, G. G. A., G. R. Crane, and S. Majidi (2016). “Part of Speech Tagging for Ancient Greek.” In: *Open Linguistics* 2.1, pp. 393–399.
- Chu, Y.-J. and T.-H. Liu (1965). “On the Shortest Aborecence of a Directed Graph.” In: *Scientia Sinica* 14.10, pp. 1397–1400.
- Clemeth, D. (2022). *tagparse*. URL: <https://github.com/clemeth/tagparse> (visited on 05/29/2022).
- Conneau, A. et al. (2020). “Unsupervised Cross-lingual Representation Learning at Scale.” In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pp. 8440–8451. DOI: [10.18653/v1/2020.acl-main.747](https://doi.org/10.18653/v1/2020.acl-main.747).

- Crane, G. R. (1991). “Generating and Parsing Classical Greek.” In: *Literary and Linguistic Computing* 6.4, pp. 243–245. DOI: [10.1093/llc/6.4.243](https://doi.org/10.1093/llc/6.4.243).
- ed. (1995). *Perseus Digital Library*. URL: <http://www.perseus.tufts.edu> (visited on 04/14/2022).
- de Marneffe, M.-C. and J. Nivre (2019). “Dependency Grammar.” In: *Annual Review of Linguistics* 5, pp. 197–218. DOI: [10.1146/annurev-linguistics-011718-011842](https://doi.org/10.1146/annurev-linguistics-011718-011842).
- Devlin, J. et al. (2019). “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.” In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4171–4186. DOI: [10.18653/v1/N19-1423](https://doi.org/10.18653/v1/N19-1423).
- Dozat, T. and C. D. Manning (2017). “Deep Biaffine Attention for Neural Dependency Parsing.” In: *International Conference on Learning Representations*. Toulon, France: OpenReview.net. URL: <https://openreview.net/forum?id=Hk95PK9le>.
- Dozat, T., P. Qi, and C. D. Manning (2017). “Stanford’s Graph-based Neural Dependency Parser at the CONLL 2017 Shared Task.” In: *Proceedings of the CONLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Vancouver, Canada: Association for Computational Linguistics, pp. 20–30. DOI: [10.18653/v1/K17-3002](https://doi.org/10.18653/v1/K17-3002).
- Eckhoff, H. et al. (2017). “The PROIEL treebank family: a standard for early attestations of Indo-European languages.” In: *Language Resources & Evaluation*. Vol. 52, pp. 29–65. DOI: [10.1007/s10579-017-9388-5](https://doi.org/10.1007/s10579-017-9388-5).
- Edmonds, J. (1967). “Optimum Branchings.” In: *Journal of Research of the National Bureau of Standards* 71B.4, pp. 233–240. DOI: [10.6028/jres.071b.032](https://doi.org/10.6028/jres.071b.032).
- Francis, W. N. and H. Kučera (1979). *Brown Corpus Manual*. URL: <http://icame.uib.no/brown/bcm.html> (visited on 04/14/2022).
- Glavaš, G. and I. Vulić (2021). “Is Supervised Syntactic Parsing Beneficial for Language Understanding Tasks? An Empirical Investigation.” In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Online: Association for Computational Linguistics, pp. 3090–3104. DOI: [10.18653/v1/2021.eacl-main.270](https://doi.org/10.18653/v1/2021.eacl-main.270).
- Gorman, V. B. (2020). “Dependency Treebanks of Ancient Greek Prose.” In: *Journal of Open Humanities Data* 6, pp. 1–3. DOI: [10.5334/johd.13](https://doi.org/10.5334/johd.13).
- Harrington, J. M., ed. (2020). *Tufts University Treebanked Commentaries*. URL: <https://perseids-publications.github.io/harrington-trees> (visited on 04/17/2022).
- Haug, D. T. T. (2015). “Treebanks in historical linguistic research.” In: *Perspectives on Historical Syntax*, pp. 185–202.

-
- Haug, D. T. T. and M. Jøhndal (2008). “Creating a parallel treebank of the old Indo-European Bible translations.” In: *Proceedings of the second workshop on language technology for cultural heritage data (LaTeCH 2008)*, pp. 27–34. URL: http://lrec-conf.org/proceedings/lrec2008/workshops/W22_Proceedings.pdf#page=31.
- Hugging Face (2022a). *bert-base-multilingual-cased*. URL: <https://huggingface.co/bert-base-multilingual-cased> (visited on 04/24/2022).
- (2022b). *nlpaueb/bert-base-greek-uncased-v1*. URL: <https://huggingface.co/bert-base-greek-uncased-v1> (visited on 04/24/2022).
- (2022c). *pranaydeeps/Ancient-Greek-BERT*. URL: <https://huggingface.co/pranaydeeps/Ancient-Greek-BERT> (visited on 04/24/2022).
- Keersmaekers, A. (2020). “A Computational Approach to the Greek Papyri: Developing a Corpus to Study Variation and Change in the Post-Classical Greek Complementmentation System.” PhD thesis. KU Leuven. URL: <https://limo.libis.be/primo-explore/fulldisplay?docid=LIRIAS3084305&vid=Lirias>.
- Keersmaekers, A. et al. (2019). “Creating, Enriching and Valorizing Treebanks of Ancient Greek.” In: *Proceedings of the 18th International Workshop on Treebanks and Linguistic Theories (TLT, SyntaxFest 2019)*. Paris, France: Association for Computational Linguistics, pp. 109–117. DOI: [10.18653/v1/W19-7812](https://doi.org/10.18653/v1/W19-7812).
- Kingma, D. and J. Ba (2014). *Adam: A Method for Stochastic Optimization*. Version 9. DOI: [10.48550/arxiv.1412.6980](https://doi.org/10.48550/arxiv.1412.6980). arXiv: [1412.6980](https://arxiv.org/abs/1412.6980) [cs.LG].
- Koehn, P. (2005). “Europarl: A Parallel Corpus for Statistical Machine Translation.” In: *Proceedings of Machine Translation Summit X: Papers*. Phuket, Thailand, pp. 79–86. URL: <https://aclanthology.org/2005.mtsummit-papers.11>.
- Kondratyuk, D. and M. Straka (2019). “75 Languages, 1 Model: Parsing Universal Dependencies Universally.” In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 2779–2795. DOI: [10.18653/v1/D19-1279](https://doi.org/10.18653/v1/D19-1279).
- Koutsikakis, J. et al. (2020). “GREEK-BERT: The Greeks Visiting Sesame Street.” In: *11th Hellenic Conference on Artificial Intelligence*. SETN 2020. Athens, Greece: Association for Computing Machinery, pp. 110–117. DOI: [10.1145/3411408.3411440](https://doi.org/10.1145/3411408.3411440).
- Kuncoro, A. et al. (2017). “What Do Recurrent Neural Network Grammars Learn About Syntax?” In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Valencia, Spain: Association for Computational Linguistics, pp. 1249–1258. URL: <https://aclanthology.org/E17-1117>.
- Lee, J., J. Naradowsky, and D. A. Smith (2011). “A Discriminative Model for Joint Morphological Disambiguation and Dependency Parsing.” In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human*

- Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, pp. 885–894. URL: <https://aclanthology.org/P11-1089>.
- Liu, Y. et al. (2019). *ROBERTa: A Robustly Optimized BERT Pretraining Approach*. Version 1. DOI: <http://arxiv.org/abs/1907.11692>. arXiv: 1907.11692 [cs.CL].
- Marneffe, M.-C. de et al. (2021). “Universal Dependencies.” In: *Computational Linguistics* 47.2, pp. 255–308. DOI: [10.1162/coli_a_00402](https://doi.org/10.1162/coli_a_00402).
- Nguyen, M. V. et al. (2021). “Trankit: A Light-Weight Transformer-based Toolkit for Multilingual Natural Language Processing.” In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*. Online: Association for Computational Linguistics, pp. 80–90. DOI: [10.18653/v1/2021.eacl-demos.10](https://doi.org/10.18653/v1/2021.eacl-demos.10).
- Nivre, J. et al. (2020). “Universal Dependencies v2: An Evergrowing Multilingual Treebank Collection.” In: *Proceedings of the 12th Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, pp. 4034–4043. URL: <https://aclanthology.org/2020.lrec-1.497>.
- Oepen, S. et al., eds. (2021). *Proceedings of the 17th International Conference on Parsing Technologies and the IWPT 2021 Shared Task on Parsing into Enhanced Universal Dependencies (IWPT 2021)*. Online: Association for Computational Linguistics. URL: <https://aclanthology.org/2021.iwpt-1.0>.
- Open Greek and Latin (2022). *First1KGreek Project*. URL: <https://opengreekandlatin.github.io/First1KGreek> (visited on 04/14/2022).
- Packard, D. W. (1973). “Computer-Assisted Morphological Analysis of Ancient Greek.” In: *COLING 1973 Volume 2: Computational And Mathematical Linguistics: Proceedings of the International Conference on Computational Linguistics*. URL: <https://aclanthology.org/C73-2026>.
- Pantelia, M. C. (2001). *Thesaurus Linguae Graecae*. URL: <http://www.tlg.uci.edu> (visited on 04/14/2022).
- Papyri.info (2010). Ed. by J. Sosin. URL: <https://papyri.info> (visited on 04/14/2022).
- (2022a). *Papyri.info IDP (Integrating Digital Papyrology) Data*. URL: <https://github.com/papyri/idp.data> (visited on 04/18/2022).
- (2022b). *The Duke Databank of Documentary Papyri*. URL: <https://papyri.info/docs/ddbdp> (visited on 04/14/2022).
- Paszke, A. et al. (2019). “PyTorch: An Imperative Style, High-Performance Deep Learning Library.” In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., pp. 1–12. URL: <https://proceedings.neurips.cc/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf>.
- Perseus Digital Library (2022). *Browse the Collections*. URL: <https://www.perseus.tufts.edu/hopper/collections> (visited on 04/07/2022).

-
- Qi, P., T. Dozat, et al. (2018). “Universal Dependency Parsing from Scratch.” In: *Proceedings of the CONLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Brussels, Belgium: Association for Computational Linguistics, pp. 160–170. DOI: [10.18653/v1/K18-2016](https://doi.org/10.18653/v1/K18-2016).
- Qi, P., Y. Zhang, et al. (2020). “Stanza: A Python Natural Language Processing Toolkit for Many Human Languages.” In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Online: Association for Computational Linguistics, pp. 101–108. DOI: [10.18653/v1/2020.acl-demos.14](https://doi.org/10.18653/v1/2020.acl-demos.14).
- Schmid, H. (2019). “Deep Learning-Based Morphological Taggers and Lemmatizers for Annotating Historical Texts.” In: *Proceedings of the 3rd International Conference on Digital Access to Textual Cultural Heritage*. DATECH 2019. Brussels, Belgium: Association for Computing Machinery, pp. 133–137. DOI: [10.1145/3322905.3322915](https://doi.org/10.1145/3322905.3322915).
- Schmid, H. and F. Laws (2008). “Estimation of Conditional Probabilities With Decision Trees and an Application to Fine-Grained POS Tagging.” In: *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*. Manchester, UK: COLING 2008 Organizing Committee, pp. 777–784. URL: <https://aclanthology.org/C08-1098>.
- Singh, P., G. Rutten, and E. Lefever (2021). “A Pilot Study for BERT Language Modelling and Morphological Analysis for Ancient and Medieval Greek.” In: *Proceedings of the 5th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*. Punta Cana, Dominican Republic (online): Association for Computational Linguistics, pp. 128–137. DOI: [10.18653/v1/2021.latechclfi-1.15](https://doi.org/10.18653/v1/2021.latechclfi-1.15).
- Stanza (2022). *Model Performance*. URL: <https://stanfordnlp.github.io/stanza/performance> (visited on 05/11/2022).
- Suárez, P. J. O., L. Romary, and B. Sagot (2020). “A Monolingual Approach to Contextualized Word Embeddings for Mid-Resource Languages.” In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 1703–1714. DOI: [10.18653/v1/2020.acl-main.156](https://doi.org/10.18653/v1/2020.acl-main.156).
- Taylor, A. (2020). “Treebanks in Historical Syntax.” In: *Annual Review of Linguistics* 6, pp. 195–212.
- Thesaurus Linguae Graecae® (2022). *The Thesaurus Linguae Graecae®: Our Mission and our Projects*. URL: <http://stephanus.tlg.uci.edu/tlg.php> (visited on 04/10/2022).
- Trankit’s Documentation (2022). *Building a customized pipeline*. URL: <https://trankit.readthedocs.io/en/latest/training.html> (visited on 05/22/2022).

- Universal Dependencies (2022a). *Ancient Greek data from the PROIEL project*. URL: https://github.com/UniversalDependencies/UD_Ancient_Greek-PROIEL (visited on 04/20/2022).
- (2022b). *Ancient Greek Dependency Treebank*. URL: https://github.com/UniversalDependencies/UD_Ancient_Greek-Perseus (visited on 04/20/2022).
- (2022c). *CONLL-U Format*. URL: <https://universaldependencies.org/format> (visited on 05/04/2022).
- (2022d). *Universal Dependency Relations*. URL: <https://universaldependencies.org/u/dep> (visited on 05/04/2022).
- (2022e). *Universal features*. URL: <https://universaldependencies.org/u/feat> (visited on 01/27/2022).
- Vannini, L. (2018). “Review of ‘Papyri.info.’” In: *RIDE* 9. DOI: [10.18716/ride.a.9.4](https://doi.org/10.18716/ride.a.9.4).
- Vierros, M. and E. Henriksson (2017). “Preprocessing Greek Papyri for Linguistic Annotation.” In: *Journal of Data Mining and Digital Humanities* (Special Issue on Computer-Aided Processing of Intertextuality in Ancient Languages). DOI: [10.46298/jdmdh.1385](https://doi.org/10.46298/jdmdh.1385).
- Virtanen, A. et al. (2019). *Multilingual is not enough: BERT for Finnish*. Version 1. DOI: [10.48550/arxiv.1912.07076](https://doi.org/10.48550/arxiv.1912.07076). arXiv: [1912.07076 \[cs.CL\]](https://arxiv.org/abs/1912.07076).
- Wang, B. et al. (2021). “On Position Embeddings in BERT.” In: *International Conference on Learning Representations*. OpenReview.net. URL: <https://openreview.net/forum?id=onxoVA9FxMw>.
- Wikipedia (2022). *elwiki dump*. URL: <https://dumps.wikimedia.org/elwiki/> (visited on 04/24/2022).
- Wolf, T. et al. (2020). “Transformers: State-of-the-Art Natural Language Processing.” In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, pp. 38–45. DOI: [10.18653/v1/2020.emnlp-demos.6](https://doi.org/10.18653/v1/2020.emnlp-demos.6).
- Wu, S. and M. Dredze (2020). “Are All Languages Created Equal in Multilingual BERT?” In: *Proceedings of the 5th Workshop on Representation Learning for NLP*. Association for Computational Linguistics, pp. 120–130. DOI: [10.18653/v1/2020.repl4nlp-1.16](https://doi.org/10.18653/v1/2020.repl4nlp-1.16).
- Yordanova, P. (2018). *Treebank of Aphthonius’ Progymnasmata*. URL: <https://github.com/polinayordanova/Treebank-of-Aphthonius-Progymnasmata> (visited on 04/14/2022).
- Zeman, D. et al. (2021). *Universal Dependencies 2.8.1*. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University. URL: <http://hdl.handle.net/11234/1-3687>.
- Zeman, D. et al. (2018). “CONLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies.” In: *Proceedings of the CONLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Brussels, Belgium:

-
- Association for Computational Linguistics, pp. 1–21. DOI: [10.18653/v1/K18-2001](https://doi.org/10.18653/v1/K18-2001).
- Zhang, X., J. Cheng, and M. Lapata (2017). “Dependency Parsing as Head Selection.” In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Valencia, Spain: Association for Computational Linguistics, pp. 665–676. URL: <https://aclanthology.org/E17-1063>.
- Zhou, H. et al. (2020). “Is POS Tagging Necessary or Even Helpful for Neural Dependency Parsing?” In: *Natural Language Processing and Chinese Computing*. Ed. by X. Zhu et al. Springer, pp. 179–191.

APPENDIX A:

CODE & REPRODUCIBILITY

We make our code available at <https://github.com/clemeth/tagparse>. For instructions and comments on the implementation, we refer to the repository documentation and commentary throughout the source code.

A.1 FINE-TUNING

We used the RNG seed 1 for all experiments. When we have reported multiple runs, we have used the seeds 1, 2, and 3. Table A.1 shows our complete configuration settings and hyperparameter search spaces.

Setting/hyperparameter	Value(s)
Seeds	(1,2,3)
<i>PyTorch</i> version	1.90
transformers version	4.12.3
Batch size	8
BERT learning rates	$[1 \times 10^{-6}, 1 \times 10^{-5}]$
Tagger learning rates	$[1 \times 10^{-5}, 1 \times 10^{-4}]$
Parser learning rates	$[6 \times 10^{-7}, 6 \times 10^{-6}]$
Last layer dropout	(0.1, 0.3)
Optimizer	<i>Adam</i>
Scheduler	Linear decay
Warmup steps	1000
Training epochs	50
Early stopping	5 epochs

TABLE A.1. Configuration settings and hyperparameter spaces for our experiments.

A.2 TRANKIT

In order to evaluate *Trankit*'s performance on our UD datasets, we followed the instructions in the official documentation, which can be found at <https://trankit.readthedocs.io/en/latest/training.html>. We initialized the pipelines with the category set to customized, and the task set to posdep. All other settings were left as is, and we did not set any RNG seeds. We supplied CONNL-U files for each dataset. The UD datasets were fetched from the official repositories as of version 2.8.

APPENDIX B:

ADDITIONAL FIGURES

On the following pages, we print the additional figures that we did not make room for in the main text.



APPENDIX B. ADDITIONAL FIGURES

True label	GAP	596	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	PUNCT	3	8270	0	0	0	0	0	0	0	0	0	0	0	0	0	1		
	adjective	0	0	7588	18	7	127	0	2	4	0	0	1	6	0	36	3	66	1
	adverb	0	0	23	3198	4	4	5	3	5	0	1	0	0	17	0	1	12	5
	article	0	0	0	3	7968	0	0	3	0	0	0	0	0	0	1	0	0	1
	common	0	0	102	2	0	8689	0	1	9	2	0	0	13	1	1	2	158	4
	conjunction	0	0	0	0	0	0	1719	4	0	0	0	0	0	8	0	4	0	1
	coordinator	0	0	1	6	2	0	0	5955	1	0	0	0	0	15	0	0	0	1
	finite	0	0	4	3	0	15	0	3	7206	10	3	1	11	0	0	0	3	1
	infinitive	0	0	0	0	0	0	0	0	13	2216	0	0	2	0	0	1	0	0
	interjection	0	0	0	0	0	0	0	0	1	0	114	0	0	0	0	0	0	1
	numeral	0	2	2	1	0	5	0	0	1	0	1	632	0	0	0	4	2	0
	participle	0	0	10	4	0	13	0	0	7	3	0	1	3292	0	0	0	3	0
	particle	0	0	1	31	0	0	4	48	0	0	0	0	0	1738	0	1	0	0
	personal	0	1	39	0	1	1	0	0	0	0	0	0	0	0	2640	0	2	0
	preposition	0	0	2	4	0	4	0	2	0	0	0	0	2	0	0	4365	0	0
	proper	0	1	99	7	1	114	1	0	10	0	1	3	8	0	0	0	3149	1
	relative	0	0	1	9	8	0	3	1	0	0	0	0	1	0	0	0	0	701
		GAP	PUNCT	adjective	adverb	article	common	conjunction	coordinator	finite	infinitive	interjection	numeral	participle	particle	personal	preposition	proper	relative
		Predicted label																	

FIGURE B.1. Confusion matrix of the xpos predictions by the best performing AGBERT-based tagger with no diacritic expansion on the papyrus dataset.

True label	GAP	595	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	PUNCT	1	8272	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
	adjective	0	0	7632	19	7	118	0	2	3	0	0	1	11	0	14	3	48	
	adverb	0	0	32	3198	4	3	5	4	5	0	1	0	0	7	1	2	11	
	article	0	0	1	0	7969	0	0	3	0	0	0	0	0	0	1	0	0	
	common	0	0	115	1	0	8715	0	0	8	2	0	0	12	1	1	3	123	
	conjunction	0	0	0	0	0	0	1716	4	1	0	0	0	0	9	0	4	0	
	coordinator	0	0	1	4	2	0	0	5963	0	0	0	0	0	11	0	0	0	
	finite	0	0	4	2	0	8	0	2	7214	12	3	0	12	0	0	0	2	
	infinitive	0	0	0	0	0	0	0	0	10	2218	0	0	3	0	0	1	0	
	interjection	0	0	0	0	0	0	0	2	0	0	112	0	0	0	0	0	2	
	numeral	0	1	2	0	0	8	0	3	1	0	1	628	0	0	1	2	3	
	participle	0	0	4	3	0	13	0	0	5	2	0	1	3302	0	0	0	3	
	particle	0	0	1	37	0	0	3	46	0	0	0	0	0	1733	2	1	0	
	personal	0	1	53	0	1	0	0	0	0	0	0	0	0	0	2627	0	2	
	preposition	0	0	4	3	0	4	0	2	0	0	0	1	1	0	0	4363	1	
	proper	0	0	109	4	2	137	0	0	11	0	1	2	11	1	0	1	3115	
	relative	0	0	3	12	9	0	3	1	0	0	0	0	1	0	0	0	695	
		GAP	PUNCT	adjective	adverb	article	common	conjunction	coordinator	finite	infinitive	interjection	numeral	participle	particle	personal	preposition	proper	relative
		Predicted label																	

FIGURE B.2. Confusion matrix of the xpos predictions by the best performing AGBERT-based tagger with expanded iotas on the papyrus dataset.

APPENDIX B. ADDITIONAL FIGURES

True label	GAP	596	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	PUNCT	0	8273	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
	adjective	0	0	7634	21	6	129	0	2	3	0	0	1	9	1	7	3	42	1
	adverb	0	0	27	3198	5	7	4	3	5	0	1	0	0	17	1	2	3	5
	article	0	0	0	0	7968	0	0	1	0	0	0	0	0	0	0	0	0	7
	common	0	0	110	2	0	8734	0	0	8	3	0	0	13	1	1	1	108	3
	conjunction	0	0	0	0	0	0	1717	4	0	0	0	0	0	10	0	4	0	1
	coordinator	0	0	3	5	0	0	0	5954	1	0	0	0	0	18	0	0	0	0
	finite	0	0	4	2	0	13	0	4	7208	11	3	0	12	0	0	0	3	0
	infinitive	0	0	0	0	0	0	0	0	8	2221	0	0	2	0	0	1	0	0
	interjection	0	0	0	0	0	0	1	0	2	0	113	0	0	0	0	0	0	0
	numeral	0	1	3	0	0	5	0	4	0	0	0	635	0	0	0	0	2	0
	participle	0	0	8	3	0	10	0	0	5	4	0	1	3300	0	0	0	2	0
	particle	0	0	1	28	0	0	1	46	0	0	0	0	0	1745	1	1	0	0
	personal	0	1	22	0	1	0	0	0	0	0	0	0	0	0	2658	0	2	0
	preposition	0	0	4	2	0	7	0	2	0	0	0	0	1	0	0	4363	0	0
	proper	0	0	110	4	2	169	1	0	16	1	0	2	14	0	0	1	3074	1
	relative	0	0	2	3	7	0	2	0	0	0	0	0	0	0	0	0	0	710
		GAP	PUNCT	adjective	adverb	article	common	conjunction	coordinator	finite	infinitive	interjection	numeral	participle	particle	personal	preposition	proper	relative
	Predicted label																		

FIGURE B.3. Confusion matrix of the xpos predictions by the best performing AGBERT-based tagger with expanded rough breathing marks on the papyrus dataset.

True label	GAP	596	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	PUNCT	0	8273	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
	adjective	0	0	7634	21	6	129	0	2	3	0	0	1	9	1	7	3	42	1
	adverb	0	0	27	3198	5	7	4	3	5	0	1	0	0	17	1	2	3	5
	article	0	0	0	0	7968	0	0	1	0	0	0	0	0	0	0	0	0	7
	common	0	0	110	2	0	8734	0	0	8	3	0	0	13	1	1	1	108	3
	conjunction	0	0	0	0	0	0	1717	4	0	0	0	0	0	10	0	4	0	1
	coordinator	0	0	3	5	0	0	0	5954	1	0	0	0	0	18	0	0	0	0
	finite	0	0	4	2	0	13	0	4	7208	11	3	0	12	0	0	0	3	0
	infinitive	0	0	0	0	0	0	0	0	8	2221	0	0	2	0	0	1	0	0
	interjection	0	0	0	0	0	0	1	0	2	0	113	0	0	0	0	0	0	0
	numeral	0	1	3	0	0	5	0	4	0	0	0	635	0	0	0	0	2	0
	participle	0	0	8	3	0	10	0	0	5	4	0	1	3300	0	0	0	2	0
	particle	0	0	1	28	0	0	1	46	0	0	0	0	0	1745	1	1	0	0
	personal	0	1	22	0	1	0	0	0	0	0	0	0	0	0	2658	0	2	0
	preposition	0	0	4	2	0	7	0	2	0	0	0	0	1	0	0	4363	0	0
	proper	0	0	110	4	2	169	1	0	16	1	0	2	14	0	0	1	3074	1
	relative	0	0	2	3	7	0	2	0	0	0	0	0	0	0	0	0	0	710
		GAP	PUNCT	adjective	adverb	article	common	conjunction	coordinator	finite	infinitive	interjection	numeral	participle	particle	personal	preposition	proper	relative
		Predicted label																	

FIGURE B.4. Confusion matrix of the xpos predictions by the best performing AGBERT-based tagger with both diacritic expansion strategies on the papyrus dataset.

APPENDIX B. ADDITIONAL FIGURES

True label	acl	114	20	0	0	2	0	0	0	0	1	4	3	0	0	0	1	0	0	1	0	5	2	0	0	3	1	0	1	0	5	0	1		
	advcl	15	531	6	1	1	0	0	0	0	14	8	1	0	0	0	0	0	0	1	0	1	1	0	0	0	2	0	1	0	7	0	13		
	advmod	0	4	603	10	2	0	0	2	32	1	3	0	0	0	5	0	0	0	2	7	3	2	1	0	3	20	0	6	0	7	0	7		
	amod	1	0	5	178	1	0	0	1	0	0	2	0	0	4	0	0	0	0	2	1	11	1	1	1	1	1	0	2	0	1	0	2		
	appos	2	5	3	4	77	0	0	0	0	1	11	1	0	3	0	0	0	0	1	0	21	13	1	0	5	10	1	0	0	0	1	4		
	aux	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	aux:pass	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	case	0	0	3	0	0	0	0	874	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	
	cc	0	0	19	0	0	0	0	0	984	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	4	0	0
	ccomp	2	8	1	0	0	0	0	0	1	120	5	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1	0	19
	conj	3	8	2	2	9	0	0	2	1	5	686	1	0	7	0	0	0	0	2	0	3	8	0	1	6	5	0	10	1	5	0	3	3	
	cop	0	0	0	0	0	0	0	0	0	2	2	202	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	2	
	csubj:pass	0	2	0	0	0	0	0	0	0	2	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
	det	0	0	0	5	0	0	0	0	1	1	2	0	0	2217	0	0	0	0	10	1	3	17	1	0	12	4	1	0	0	3	0	1		
	discourse	0	0	3	0	0	0	0	0	0	0	0	0	0	0	722	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	
	dislocated	0	6	2	2	1	0	0	0	0	0	1	1	0	2	0	3	0	0	0	0	0	1	4	0	0	3	0	0	0	0	0	2	0	
	fixed	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	
	flat:name	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
	iobj	0	1	4	2	1	0	0	0	0	0	2	0	0	3	0	0	0	0	386	0	13	4	0	1	3	14	1	0	0	1	0	1		
	mark	0	0	1	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	211	0	0	0	0	0	0	0	0	0	0	0	0	0	
	nmod	0	2	2	9	12	0	0	0	0	0	7	0	0	9	0	0	0	1	11	0	418	8	0	0	4	29	0	5	0	1	0	0		
	nsubj	2	4	3	2	4	0	0	0	0	2	17	0	0	12	0	2	0	0	2	0	2	781	12	2	27	2	0	1	0	10	1	4		
	nsubj:pass	0	2	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	19	69	0	2	0	0	0	0	0	0	0	2		
	nummod	0	0	0	0	0	0	0	0	0	0	1	0	0	2	0	0	0	0	0	0	0	1	0	63	1	1	0	0	0	1	0	0		
	obj	1	1	5	3	0	0	0	0	0	2	5	0	1	12	0	1	0	0	1	0	2	14	1	1	843	12	0	0	0	0	0	2		
	obl	0	3	6	2	2	0	0	5	0	1	4	0	0	6	0	0	0	0	24	0	9	5	1	0	15	705	3	0	0	2	0	3		
	obl:agent	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	1	0	0	0	0	4	14	0	0	0	0	0		
	orphan	2	3	3	0	1	0	0	1	0	0	4	0	0	0	0	0	0	0	0	0	2	0	0	0	3	3	0	11	0	0	0	2		
	parataxis	1	3	0	0	0	0	0	0	0	2	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	5	2	0		
	root	3	6	3	1	6	0	0	1	0	3	5	13	0	1	1	1	0	0	0	0	3	9	2	1	3	2	0	1	3	978	0	1		
	vocative	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	4	52	0		
	xcomp	4	11	5	3	0	0	0	0	0	14	2	0	1	0	0	0	0	0	3	2	1	15	1	0	7	8	0	1	0	0	0	162		
		Predicted label																																	

FIGURE B.5. Confusion matrix of the *deprel* predictions by the AGBERT-based parser on PROIEL.

True label	acl	122	18	0	0	1	0	0	0	0	1	4	2	0	0	0	2	0	0	0	0	4	0	1	0	2	1	0	0	0	4	0	2	
	advcl	17	534	3	0	0	0	0	0	14	6	1	0	0	0	0	0	0	0	0	1	4	2	0	1	2	0	1	1	7	0	9		
	advmod	0	3	617	7	3	0	0	2	34	1	5	0	0	0	4	0	0	0	2	5	3	3	1	0	4	9	0	4	0	7	0	6	
	amod	1	1	5	187	1	0	0	1	0	0	2	0	0	3	0	0	0	0	0	0	10	0	1	0	1	0	0	1	0	1	0	1	
	appos	5	4	3	6	79	0	0	0	0	1	11	1	0	3	0	0	0	0	1	0	18	14	0	0	6	8	0	1	0	1	1	1	
	aux	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	aux:pass	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	case	0	0	0	0	0	0	0	878	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	cc	0	0	16	0	0	0	0	0	989	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0
	ccomp	6	6	1	0	0	0	0	0	1	113	5	1	1	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	25
	conj	3	10	2	4	12	0	0	1	1	4	676	2	0	6	0	0	1	0	1	0	6	9	0	0	5	4	0	9	3	7	0	4	
	cop	0	0	0	0	0	0	0	0	0	2	2	205	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2
	csubj:pass	0	2	0	0	0	0	0	0	0	3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
	det	0	0	1	3	0	0	0	0	1	1	2	0	0	2228	0	0	0	0	9	1	2	13	2	0	12	3	0	0	0	0	0	0	1
	discourse	0	0	3	0	0	0	0	0	0	0	0	0	0	0	724	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	dislocated	0	6	3	2	1	0	0	0	0	0	1	1	0	1	0	1	0	0	0	0	1	4	1	0	3	0	0	0	0	0	0	3	0
	fixed	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0
	flat:name	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
	iobj	0	0	3	2	0	0	0	0	0	0	2	0	0	5	0	0	0	0	382	0	15	4	1	0	3	17	1	0	0	1	0	1	
	mark	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	213	0	0	0	0	0	0	0	0	0	0	0	0	0
	nmod	0	1	4	8	11	0	0	0	0	0	4	0	0	12	0	0	0	0	10	0	420	12	0	0	3	28	0	4	0	1	0	0	0
	nsubj	1	4	3	2	4	0	0	0	0	5	13	0	0	15	0	1	0	0	1	0	3	780	16	2	23	3	0	1	1	10	1	3	
	nsubj:pass	0	1	0	1	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	12	78	0	1	0	0	0	0	0	0	0	0
	nummod	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	65	1	2	0	0	0	0	0	0	0
	obj	1	0	1	3	1	0	0	0	0	1	3	0	1	13	0	1	0	0	3	0	1	16	1	2	846	9	0	1	0	0	0	3	
	obl	0	6	6	0	2	0	0	5	0	1	2	0	0	5	0	0	0	1	20	0	8	4	1	1	14	714	2	1	0	1	0	2	
	obl:agent	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	5	13	0	0	0	0	0	0	
	orphan	2	2	5	1	1	0	0	1	0	0	5	0	0	0	0	0	0	0	0	1	0	0	0	3	3	0	9	0	0	0	0	2	
	parataxis	1	3	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	2	5	2	1	0	
	root	4	5	3	2	6	0	0	1	0	1	7	9	0	1	1	0	0	0	0	0	3	4	2	0	2	2	0	0	2	991	0	1	0
	vocative	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	0	0	0	0	0	0	0	3	54	0	0	0
	xcomp	4	10	4	4	2	0	0	0	0	7	2	0	0	0	0	0	0	0	3	2	2	12	1	0	3	9	0	1	0	0	1	173	0
		acl	advcl	advmod	amod	appos	aux	aux:pass	case	cc	ccomp	conj	cop	csubj:pass	det	discourse	dislocated	fixed	flat:name	iobj	mark	nmod	nsubj	nsubj:pass	nummod	obj	obl	obl:agent	orphan	parataxis	root	vocative	xcomp	
	Predicted label																																	

FIGURE B.6. Confusion matrix of the *deprel* predictions by the AGBERT-based joint model on PROIEL.