OdyCy – A general-purpose NLP pipeline for Ancient Greek

Jan Kostkan[1] and **Márton Kardos**[1] and **Jacob P.B. Mortensen**[2] and **Kristoffer L. Nielbo**[1]
[1]Center for Humanities Computing, Aarhus University, Denmark
[2]School of Culture and Society – Biblical Studies, Aarhus University, Denmark
{jan.kostkan, teojmo, kln}@cas.au.dk

Abstract

This paper presents a general-purpose NLP pipeline for Ancient or early forms of Greek (Classical, Koine, and Medieval) that achieves a slight state-of-art improvement by training on several Universal Dependencies treebanks jointly. We measure the performance of the model against other comparable tools. We show that the selected Greek language models tend not to generalize well to out-of-training set samples. More work is necessary to ensure interoperability between the existing datasets. We identify the main issues and list suggestions for improvements.

1 Introduction

The impact of digitization on literature research in contemporary English and other languages cannot be exaggerated. Computational linguistics and Natural Language Processing (NLP) have developed numerous tools that automate annotation and analysis that would otherwise have taken lifetimes of manual labor. Similar advances have not been made for historical and low-resource language areas, for instance, classical literature in Greek, Latin, and Hebrew. Computational studies of classical literature are limited not only by fewer tools but also paywalls and licensed access (ex. *Loeb Classical Library* and *Thesaurus Linguae Graecae*), altogether complicating the training of neural-based language technology. To remedy this and to contribute to a relatively small number of existing NLP resources in this domain, we present a general-purpose NLP pipeline for early forms of Greek that will enable a computationally assisted analysis of, among other things, early forms of Greek literature.

1.1 Related work

Most existing work on Ancient Greek NLP has focused on individual tasks such as lemmatization (Bary et al., 2017; de Graaf et al., 2022; Vatri and McGillivray, 2020) or morphological analysis and part of-speech-tagging (Celano et al., 2016; Singh et al., 2021). This work has primarily been conducted by subject-matter experts that incorporate their domain knowledge, making the model results more interpretable but less general.

There exists a few examples of full language pipelines for Ancient Greek. One notable example is The Classical Language Toolkit (CLTK) (Johnson et al., 2021), which has been the go-to option for classicists needing NLP tools. CLTK, however relies heavily on domain-specific knowledge. Other pipelines have been trained using well-known NLP frameworks, usually relying on neural components for individual tasks. These include Stanza's (Qi et al., 2020), UDPipe's (Straka, 2018), and Trankit's (Van Nguyen et al., 2021) pipelines. These neural models are language agnostic and hence general-purpose. One additional spaCy pipeline should be mentioned, greCy[1], that has been developed for the Diogenet project[2].

The pipelines mentioned have opted for training separate models for each UD Treebank in Ancient Greek. Raw accuracies generally tend to be higher for models trained and evaluated on *UD Proiel*, compared to *UD Perseus* models. However, due to Ancient Greek being a highly fragmented and low-resource language, high performance on one data set may not generalize for corpora of substantially different quality or nature.

The model presented here, *odyCy* relies on spaCy, which offers a fully modular framework in which individual components can be modified with relative ease. The goal is to allow researchers to integrate this model into their particular use case easily; for example, by fine-tuning the model for a downstream task such as document classification or using it to normalize raw texts for topic modeling. The model also easily integrates with other tools in the spaCy ecosystem, such as TextDescriptives

[1]https://github.com/jmyerston/greCy
[2]https://diogenet.ucsd.edu/

Proceedings of the 7th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature (LaTeCH-CLfL2023), pages 128–134

(Hansen and Enevoldsen, 2023) for calculating metrics from text.

2 Methods

2.1 Treebanks

For training the pipeline, both UD Treebanks, *UD Perseus* and *UD Proiel*, available for Ancient Greek, were used in order to increase the robustness of the model. The *UD Perseus* Treebank (Bamman and Crane, 2011) contains 13,919 sentences. This dataset contains texts in Ancient and Koine Greek distributed over various genres (e.g. tragedies by Aeschylus and Sophocles, biographies by Plutarch, or the Iliad) and various dialects. The *UD Proiel* Treebank (Haug and Jøhndal, 2008) contains 17,081 sentences. The content is mostly New Testament (in Koine Greek), with chapters from Herodotus' Histories. Notably, unlike the main branch of the Proiel Treebank, the UD version does not contain Sphrantzes' Chronicles, written in Medieval (Byzantine) Greek (Singh et al., 2021).

Both treebanks are included in the Universal Dependencies framework (de Marneffe et al., 2021), which specifies annotation standards for multiple languages. Still, the two treebanks differ in some important aspects:

- Punctuation is absent from *UD Proiel* (except for elisions, e.g. ἀλλ'), making it a difficult resource to train a model for sentence segmentation.

- Proper nouns (PROPN) are only annotated in *UD Proiel*. For example, Ἑλλάς is labeled as a noun in *UD Perseus*, but as a proper noun in *UD Proiel*.

- Annotation standards for morphological features differ slightly between the two treebanks, even though the labels overlap for the most part. *UD Proiel* has richer annotations compared to *UD Perseus*, recognizing five additional morphological features (e.g. polarity, reflex or pronoun types) and 14 additional feature-value pairs[3].

- Ambiguous lemmas are handled differently between the two treebanks. *UD Perseus* contains lemmas of compound words in which the two stems are separated by a dash character (e.g., περί-κάθημαι). Furthermore, lemmas in *UD Proiel* may contain optional letters inside parentheses (e.g. Ἰωάν(ν)ης).

- The *UD Perseus* Treebank misrepresents some Ancient Greek characters, likely due to a problematic conversion of the annotations from beta code to Unicode. For example, the trailing apostrophe in the correct form ἀλλ' has been misinterpreted as a smooth breathing mark (᾽) above λ.

- *UD Perseus* has been 'semi-automatically annotated'[4]. This means texts were manually annotated and then corrected with the help of Morpheus (morphologizer of the Perseus project).

2.2 Model architecture

The pipeline uses Ancient-Greek-BERT (Singh et al., 2021) as the base model for acquiring the context-rich vector representation of tokens. Subsequent components in the pipeline use the representations as input features to generate predictions. The transformer component has been fine-tuned during training for all downstream tasks simultaneously. These vector representations can be directly accessed on every token for semantic analyses.

Single softmax-activated dense layer models in the pipeline are responsible for morphological analysis and part-of-speech tagging. Tags get assigned on a token level. The models' inputs are the contextual representations obtained from the transformer. We used the default transition-based dependency parser component of spaCy. The component learns both to parse dependency trees in the text as well as to segment sentences.

Lemmatization seems to be the most challenging task for Ancient Greek NLP software. Vatri and McGillivray (2020) provides an overview of different lemmatizers for Ancient Greek, where other approaches were evaluated manually by multiple annotators instead of being benchmarked automatically, which in the case of languages without a canonical orthography is particularly desirable. The paper also shows that, on average, multi-layer lemmatization strategies perform better than single-layer and that large lookup lexicons should have higher priority than machine learning-based layers.

[3] https://universaldependencies.org/treebanks/grc-comparison.html

[4] https://github.com/PerseusDL/treebank_data/tree/master/v2.1/Greek

In addition, the study found that lemmatizers sensitive to part of speech are better than lemmatizers that solely rely on lemma frequency as a heuristic.

In order to incorporate these findings in the odyCy pipeline, we employ a multi-layer strategy for lemmatization. Similar to the approach of GLEM (Bary et al., 2017), we produced a lexicon from the training set containing information about token-lemma pairs and morphological features part-of-speech tags. The lemmatization process searches for tokens in the lexicon and matches them with part-of-speech tags and morphological information. If at any point this process fails, the most frequent lemma will be returned from the last successful match. If the token cannot be found in the lexicon, the tokenizer returns to a default lookup table in spaCy that does not contain morphological or part-of-speech information. If a lemma cannot be identified for the token, a context-sensitive neural edit-tree lemmatizer (Müller et al., 2015) will try to produce a prediction for the given token. If all else fails, it will return the original form of the token. For a schematic overview see Figure 1.

Due to the modular nature of spaCy either the lookup or the neural component may be removed or disabled with a single line of code. Our experiments show that for unseen data, the entire pipeline and the neural component's performance are comparable (see Table 3).

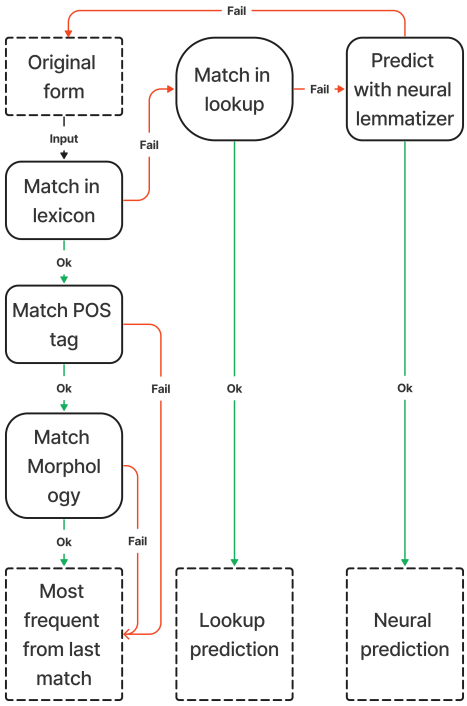

Figure 1: Schematic Overview of the Lemmatization Process.

3 Results

When evaluated on the UD Perseus Treebank, our model achieves state-of-the-art performance in POS Tagging, Morphological Analysis, and Dependency Parsing (see Table 1). We achieve close to state-of-art in Sentence Segmentation and Lemmatization. On the UD Proiel Treebank, we achieve the second-best performance across all measures except for Lemmatization (see Table 2). The odyCy joint model, which was trained on both UD treebanks, scores higher than odyCy versions trained on individual treebanks (see Table 1 and 2). Notably, models trained on a single treebank systematically underperform on the other.

3.1 Tokenization and Lemmatization Error Analysis

To investigate errors that occurred during lemmatization and tokenization, we conducted a qualitative error analysis of randomly selected batches from the treebanks. This investigation revealed the following causes for the mismatch between the gold standard and predicted lemmas. The causes may overlap.

- *Tokenization mistakes*. In some cases, a form that should correspond to a single lemma splits into two lemmas. This causes token misalignment and renders every following lemma in the sentence incorrect.
- *Incorrect or Ignored POS-tags or Morphological Features*. Incorrect predictions of a token's morphological features, especially of the POS-tag can cause lemmatization errors. This is because the lemmatizer relies on predictions from the preceding pipeline components. Proper nouns, for example, get frequently misinterpreted as regular nouns due to the disagreement between the two annotation schemes, which can result in incorrect inflection. When the component falls back to the lookup table, there is a possibility of ignoring morphological information, as the lookup table only contains form-lemma pairs without context or morphology.

	Token	POS	Morphology	Sentence Segmentation			Dependency Parsing		Lemma
Model	Accuracy	Accuracy	Accuracy	Precision	Recall	F-1 Score	UAS	LAS	Accuracy
CLTK	NA*	80.50	61.49	0.00	0.00	0.00	33.05	24.25	79.46
odyCy$_{perseus}$	99.98	<u>95.00</u>	<u>91.98</u>	<u>97.86</u>	98.16	<u>98.01</u>	76.71	70.31	82.56
odyCy$_{proiel}$	99.98	73.14	60.59	3.85	6.66	4.88	66.35	50.26	81.00
odyCy$_{joint}$	99.98	**95.39**	**92.56**	97.57	<u>98.32</u>	97.94	**78.80**	**73.09**	<u>83.20</u>
greCy$_{perseus}$	99.89	93.50	90.59	90.76	94.79	92.73	76.34	70.20	75.10
greCy$_{proiel}$	99.89	81.97	61.26	10.21	17.38	12.86	69.30	53.14	68.92
Stanza$_{perseus}$	100.00	91.05	91.03	**99.31**	**98.93**	**99.12**	<u>78.69</u>	<u>71.82</u>	**87.58**
Stanza$_{proiel}$	87.68	68.73	50.14	40.88	34.84	37.62	46.75	35.73	70.55
UDPipe$_{perseus}$	99.99	80.95	85.70	**99.31**	**98.93**	**99.12**	63.97	55.81	82.73
UDPipe$_{proiel}$	87.33	65.23	45.85	37.46	48.01	42.08	35.16	26.73	65.91

Table 1: Model performances on the test fold of *UD Perseus* Treebank. Highest performance in bold, second highest underlined.

*CLTK's tokenization had to be manually fixed as it routinely added punctuation to tokens, and spaCy's evaluation scripts could not align them against the gold standard.

	Token	POS	Morphology	Sentence Segmentation			Dependency Parsing		Lemma
Model	Accuracy	Accuracy	Accuracy	Precision	Recall	F-1 Score	UAS	LAS	Accuracy
CLTK	NA*	96.95	90.76	50.00	33.33	40.00	57.61	54.57	96.50
odyCy$_{perseus}$	100.00	84.88	57.44	2.08	0.29	0.50	64.55	48.72	91.36
odyCy$_{proiel}$	100.00	97.61	92.84	62.91	64.47	63.68	81.42	77.07	94.42
odyCy$_{joint}$	100.00	<u>97.81</u>	<u>93.46</u>	<u>64.03</u>	<u>65.81</u>	<u>64.91</u>	<u>83.17</u>	<u>79.03</u>	94.41
greCy$_{perseus}$	100.00	80.42	56.11	0.78	0.10	0.17	63.03	47.58	89.13
greCy$_{proiel}$	100.00	**98.23**	**94.05**	**71.76**	**71.82**	**71.79**	**85.74**	**82.28**	**98.06**
Stanza$_{perseus}$	99.99	80.93	56.00	0.93	0.10	0.17	59.00	43.79	87.14
Stanza$_{proiel}$	100.00	97.39	92.20	55.34	52.44	53.85	81.51	77.48	<u>97.21</u>
UDPipe$_{perseus}$	100.00	74.19	53.17	0.00	0.00	0.00	51.29	37.94	81.69
UDPipe$_{proiel}$	100.00	95.97	88.62	52.97	49.38	51.11	72.40	67.48	93.17

Table 2: Model performances on the test fold of *UD Proiel* Treebank. Highest performance in bold, second highest underlined.

*CLTK's tokenization had to be manually fixed as it routinely added punctuation to tokens, and spaCy's evaluation scripts could not align them against the gold standard.

	UD Perseus		UD Proiel	
Lemmatizer	With Diacritics	Ignored Diacritics	With Diacritics	Ignored Diacritics
Lookup	75.27	76.52	89.13	90.69
Neural	84.16	85.54	93.5	94.58
Lexicon	76.79	78.4	94.4	94.53
Full	83.2	85.55	94.4	94.53

Table 3: Performances of individual lemmatizer components and the full lemmatization process of the odyCy$_{joint}$ model. Scores are accuracies.

- *Mismatch in diacritics*. Different Ancient Greek dialects and styles may employ diacritics in different ways. This leads to a mismatch in lemma annotations on multiple occasions; Consider ἐρῆμος versus ἔρημος. This problem can be alleviated by ignoring diacritics, which leads to higher lemmatization accuracies (see Table 3). However, diacritics are needed in the other components of the pipeline, for example, to distinguish between the nominative form πατήρ and the vocative πάτηρ.

- *Lemmas in neuter vs. masculine form*. Ancient Greek lacks a canonical lemmatization scheme, resulting in situations where both masculine and neuter forms of a word can function as a lemma in some instances, e.g. σίδηρος or σίδηρον.

- *Compound words*. Compound lemmas in Perseus are marked with dashes between the two words, sometimes leading to mismatches. This is either because the predicted lemma is missing a dash on Perseus data or it contains one when evaluating against the Proiel gold standard (see Section 2.1 for an example).

4 Conclusion

Comparing the performance of our model and several other comparable tools suggests there is a considerable amount of transferable information between *UD Perseus* and *UD Proiel* – the two most commonly used datasets for modeling linguistic features in Ancient Greek. We improved the state-of-art on some tasks, but more work is necessary to enhance the interoperability between the two datasets. We identify the main issues and list suggestions for improvement. Resolving these issues is a good way of addressing the low generalizability of Ancient Greek language models on out-of-training set samples (e.g., the bad performance of Proiel models on Perseus data).

Our best-performing model, odyCy$_{\text{joint}}$, comes with its own set of problems (see Limitations), but comparing its performance to similar tools suggests it generalizes better across the two datasets. This is advantageous when analyzing mixed corpora, where it is unclear whether the corpus to be analyzed is more Perseus-like or Proiel-like. However, it should be noted that a better solution exists for Proiel-like corpora, namely the greCy$_{\text{proiel}}$ model.

Finally, the model and its source code have been made open source[5], together with the source code for evaluating the performance of Ancient Greek NLP tools[6].

Limitations

Training on Both Treebanks

Since the two UD treebanks for Ancient Greek have different annotation schemes we are severely limiting the model's performance on certain corpora and tasks. Our model, for example is particularly bad at recognizing proper nouns as they are not included in Perseus at all. In our future work we intend to address these issues.

Low Variety in Training Data

Even though we are training odyCy on both available treebanks, the temporal, cultural, and literary variety of the data is relatively low. One perspective direction is to train and evaluate the model's performance on more datasets. This also poses some problems, because they have not been annotated following UD guidelines. An interesting data source that comes to mind are the Dependency Treebanks of Ancient Greek Authors (Gorman, 2020), which consists of Ancient Greek prose. As to texts picked for annotation, the treebank overlaps with *UD Perseus* to some extent. The partially annotated Collection of Greek Ritual Norms used by de Graaf et al. (2022) is also a good candidate for further annotation and usage.

Lemmatization Performance

On the *UD Perseus* testing data odyCy is outperformed by Stanza's Perseus model. We suspect that this difference might be attributed to the fact that Stanza uses full sequence-to-sequence lemmatizer models, which are much more flexible than the tree-based and lookup solutions we are using. We plan on addressing this issue either by implementing a sequence-to-sequence lemmatizer in our pipeline or by increasing the quality and quantity of the training data. Based on the results of the error analysis we have reasons to suspect that the latter might be sufficient.

Sentencization Performance

odyCy is outperformed by other neural pipelines trained on the *UD Perseus* Treebank in sentenciza-

[5] odyCy (Github)
[6] greevaluation (Github)

tion. This might be due to the fact that both Stanza and UDPipe use recurrent neural networks for sentencization and tokenization as well as the fact the sentences end with punctuation and the pipelines don't have to rely as much on dependency parsing. Our models learn dependency parsing and sentence segmentation jointly. This approach might not work best with text containing clear sentence boundaries but clearly outperforms both UDPipe and Stanza on *UD Proiel*, where sentence boundaries are missing. greCy performs exceptionally on *UD Proiel*, as it ships with its own sentence recognizer component. However, since it is only trained on one treebank, it performs worse on *UD Perseus*. This issue might be addressed by adding a separate sentence recognizer to odyCy. Still, odyCy seems to already provide a robust solution for sentencization.

Variants of Greek

Furthermore, the model can benefit from additional error analysis comparing the regional and temporal variants of Greek. We have not investigated how well the pipeline handles e.g. Doric morphology. In order to evaluate and possibly enhance the performance of our pipeline on other dialects of Ancient Greek or other literary genres we will need newly annotated texts. The already existing pipeline might be of substantial help here, as annotation would only consist of fixing the model's errors.

Acknowledgements

The study was supported by The Carlsberg Foundation's Semper Ardens instrument ('Computing Antiquity: Computational Research in Ancient Text Corpora', grant holder Jacob P.B. Mortensen). We would also like to thank Ross Deans Kristensen-McLachlan, Yuri Bizzoni and Jacobo Myerston for their helpful comments.

References

David Bamman and Gregory Crane. 2011. The ancient greek and latin dependency treebanks. In *Language Technology for Cultural Heritage*, pages 79–98. Springer Berlin Heidelberg.

Corien Bary, Peter Berck, and Iris Hendrickx. 2017. A memory-based lemmatizer for ancient greek. In *Proceedings of the 2nd International Conference on Digital Access to Textual Cultural Heritage*, DATeCH2017, page 91–95, New York, NY, USA. Association for Computing Machinery.

Giuseppe G. A. Celano, Gregory Crane, and Saeed Majidi. 2016. Part of speech tagging for ancient greek. *Open Linguistics*, 2(1). Publisher: De Gruyter Open Access.

Evelien de Graaf, Silvia Stopponi, Jasper Bos, Saskia Peels-Matthey, and Malvina Nissim. 2022. Agile: The first lemmatizer for ancient greek inscriptions. In *Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022)*, pages 5334–5344. European Language Resources Association (ELRA). The 13th Conference on Language Resources and Evaluation, LREC 2022 ; Conference date: 20-06-2022 Through 25-06-2022.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal dependencies. *Computational Linguistics*, pages 1–54.

Vanessa B. Gorman. 2020. Dependency treebanks of ancient greek prose. *Journal of Open Humanities Data*, 6(1). Publisher: Ubiquity Press.

Lasse Hansen and Kenneth Enevoldsen. 2023. Textdescriptives: A python package for calculating a large variety of metrics from text. *arXiv preprint arXiv:1503.06733*.

Dag TT Haug and Marius 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)*, pages 27–34.

Kyle P. Johnson, Patrick J. Burns, John Stewart, Todd Cook, Clément Besnier, and William J. B. Mattingly. 2021. The classical language toolkit: An NLP framework for pre-modern languages. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 20–29. Association for Computational Linguistics.

Thomas Müller, Ryan Cotterell, Alexander Fraser, and Hinrich Schütze. 2015. Joint lemmatization and morphological tagging with lemming. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2268–2274, Lisbon, Portugal. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 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*, pages 101–108. Association for Computational Linguistics.

Pranaydeep Singh, Gorik Rutten, and Els 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, pages 128–137, Punta Cana, Dominican Republic (online). Association for Computational Linguistics.

Milan Straka. 2018. UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 197–207, Brussels, Belgium. Association for Computational Linguistics.

Minh Van Nguyen, Viet Dac Lai, Amir Pouran Ben Veyseh, and Thien Huu Nguyen. 2021. Trankit: A light-weight transformer-based toolkit for multilingual natural language processing. *arXiv preprint arXiv:2101.03289*.

Alessandro Vatri and Barbara McGillivray. 2020. Lemmatization for ancient greek: An experimental assessment of the state of the art. *Journal of Greek Linguistics*, 20(2):179–196. Publisher: Brill.