Part-of-Speech Tagging of 16th-Century Latin with GPT

Elina Stüssi, Phillip Benjamin Ströbel
Department of Computational Linguistics
University of Zurich
{elina.stuessi,phillip.stroebel}@uzh.ch

Abstract

Part-of-speech tagging is foundational to natural language processing, transcending mere linguistic functions. However, taggers optimized for Classical Latin struggle when faced with diverse linguistic eras shaped by the language's evolution. Exploring 16th-century Latin from the correspondence and assessing five Latin treebanks, we focused on carefully evaluating tagger accuracy and refining Large Language Models for improved performance in this nuanced linguistic context. Our discoveries unveiled the competitive accuracies of different versions of *GPT*, particularly after fine-tuning. Notably, our best fine-tuned model soared to an average accuracy of 88.99% over the treebank data, underscoring the remarkable adaptability and learning capabilities when fine-tuned to the specific intricacies of Latin texts. Next to emphasising GPT's part-of-speech tagging capabilities, our second aim is to strengthen taggers' adaptability across different periods. We establish solid groundwork for using Large Language Models in specific natural language processing tasks where part-of-speech tagging is often employed as a pre-processing step. This work significantly advances the use of modern language models in interpreting historical language, bridging the gap between past linguistic epochs and modern computational linguistics.

1 Introduction

Understanding parts-of-speech (POS) is fundamental in linguistic analysis (Jurafsky and Martin, 2019). Automatic POS tagging offers vital clues for parsing and language analysis. Despite Latin's extensive dataset in the Universal Dependencies treebanks,[1] many historical texts lack syntactic analysis (Nehrdich and Hellwig, 2022).

Latin's enduring relevance in domains like the Catholic Church and classical studies persists despite its decline post-1800 (Leonhardt, 2013). However, this enduring relevance is intertwined with its extensive historical significance, contributing to the language's vast evolutionary timespan.

The evolution of the Latin language spans significant changes over time, notably evident in alterations to case endings and lexical transformations, particularly during the transition from Old Latin to Classical Latin (Allen, 1989). These alterations extend beyond mere word changes, reshaping meanings and structures and resulting in diverse linguistic variations. The historical evolution of the language continued until the Early Modern Latin period of the 16th century, thereby posing challenges for POS tagging systems that have been mostly trained on Classical Latin (Schmid, 2019).

Despite its historical significance, Latin remains classified as a low-resource language due to the scarcity of digitized texts and annotations (Hedderich et al., 2021).[2] The absence of speakers poses difficulties in creating a gold standard,[3] a process notably more labor-intensive and error-prone than that for modern languages. Nonetheless, Latin benefits from a wealth of linguistic expertise derived from its extensive historical legacy, offering substantial aid in overcoming these obstacles (McGillivray, 2013).

The nuances in 16th-century epistolary Latin pose challenges for POS taggers, especially. Tagging a sentence from the correspondence of Swiss reformer Heinrich Bullinger (1504–1575) by various systems[4] highlights discrepancies, as illustrated in Figure 1. *RDRPOSTagger* misclassified punctuation and the name "Erasmus" as verbs. *Lat-*

[1] There are five in total: Latin-ITTB, Latin-Perseus, Latin-PROIEL, Latin-LLCT, Latin-UDante, all of which we introduce in Section 3.1. Also, see https://universaldependencies.org/la.

[2] Although there are large text collections like the *Corpus Corporum*, see https://mlat.uzh.ch.

[3] i. e., a manually compiled and verified annotated version of a text (in our case, the annotation would concern POS tags only).

[4] We will introduce the different taggers in Section 3.2.

Proceedings of LaTeCH-CLfL 2024, pages 196–206

	Dominus	Erasmus	plurimam	salutem	tibi	adscribere	iussit	.
GS:	NOUN	PROPN	ADJ	NOUN	PRON	VERB	VERB	PUNCT
LC:	NOUN	PROPN	ADJ	NOUN	PRON	NOUN	VERB	PUNCT
RDR:	NOUN	VERB	ADJ	NOUN	PRON	VERB	VERB	VERB
GPT-4:	NOUN	PROPN	ADJ	NOUN	PRON	VERB	VERB	PUNCT

Figure 1: Demonstration of a sentence tagged with Gold Standard (GS), LatinCy (LC), RDRPOSTagger (RDR), and GPT-4.

inCy tagged "adscribere" as a noun while *RDRPOSTagger* and *GPT-4* identified it correctly as a verb. *GPT-4*'s similarity to the gold standard underscores the potential of Large Language Models (LLMs) to enhance accuracy in language processing tasks.

Our project's core revolves around tagging 16th-century Latin data with multiple taggers, showcasing the disparities and revealing the potential of LLMs to increase accuracy in the POS tagging process. Motivated by the need to enhance linguistic analysis for historians and linguists, our work addresses the challenges in POS tagging within this historical context. Moreover, our efforts in refining POS tagging algorithms preserve cultural heritage and drive advancements in natural language processing (NLP), extending their impact across machine learning and AI beyond linguistic analysis.

Our contributions encompass a detailed investigation into how fine-tuning influences POS tagging accuracy and the customization of models to distinct datasets for improved precision. We underscore the role of fine-tuning and prompting in notably enhancing performance, particularly when tailoring models to domain-specific data. By conducting extensive comparative analyses between fine-tuned and pre-trained models, we reveal each approach's distinct strengths and limitations, emphasizing the nature of domain-specific training for achieving superior accuracy in POS tagging. These evaluations offer insights important for future research, underscoring the need for tailored models and their potential applications in NLP tasks.

2 Recent Work

Exploration of Latin within the field of NLP has remained limited despite the existence of various methodologies designed to enhance its processing efficiency. The inaugural *Workshop on Language Technologies for Historical and Ancient Languages* (LT4HALA) held in 2020 represented a step forward in developing language technologies tailored for historically documented languages, including Latin (Sprungoli and Passarotti, 2020).

As a part of LT4HALA, the *EvaLatin* initiative focused specifically on Latin and investigated lemmatization and POS tagging, scrutinizing their performance across diverse temporal contexts (Sprungoli and Passarotti, 2020). *EvaLatin* encompassed works such as *LSTMVoter* (Stoeckel et al., 2020) and the *UDPipe2*-based system (Straka and Straková, 2020), showcasing advancements in techniques customized for historical Latin texts. Additionally, the *LiLa* project[5] significantly fortified the lexical foundation for Latin, fostering a symbiotic relationship between textual and lexical resources (Passarotti et al., 2023; Pellegrini et al., 2021).

However, despite efforts like Chu[6] highlighting the strengths of GPT models for POS tagging, there remains a conspicuous gap in research specifically exploring LLMs as POS taggers.

3 Data and Methodology

3.1 Datasets

This study focused on leveraging LLMs, particularly different flavours of GPT, for the POS tagging of historical texts from different periods. Utilizing UPOS tags[7] for consistency, our experiments encompassed samples from our own *Bullinger Digital* corpus (Bullinger Digital, 2023) and five treebanks: ITTB (Cecchini et al., 2018; Passarotti and

[5] See https://lila-erc.eu/.
[6] See https://bit.ly/3vUyqNu.
[7] See https://universaldependencies.org/u/pos.

Dataset	time	# of sentences	# of token-tag pairs
Bullinger	c. 16	200	3664
ITTB	c. 13	24,876	420,672
LLCT	c. 8 – c. 10	8,173	218,223
PROIEL	c. 1 BCE – c. 4	11,851	110,774
UDante	c. 13 – c. 14	1,157	38,086
Perseus	c. 1 BCE – c. 4	4,236	68,283
Total		**50,493**	**859,702**

Table 1: Overview of different datasets (c. = century).

Dell'Orletta, 2010), LLCT (Korkiakangas, 2021), UDante (Cecchini et al., 2020), PROIEL (Haug and Jøhndal, 2008), and Perseus (Bamman and Crane, 2006). Table 1 provides an overview of the data used.

The Bullinger corpus, derived from Heinrich Bullinger's 16th-century correspondence, offers insights into early modern societal aspects and the reformation process in Switzerland and Europe. This study uses a sample from a corpus comprising approximately 220k sentences, with the selected subset comprising 200 sentences. The sample intentionally includes only the Latin sections from digitized letters structured into XML format, excluding the Early New High German sentences.

The five treebanks served as training material and fine-tuning data for POS tagging models. The ITTB dataset offers morphosyntactic disambiguation and sentence-level syntactic annotation for Latin. The training and test sets provided in CoNLL-U format include 24,876 tagged sentences, totalling 420,627 token-tag pairs.

Similarly, the Universal Dependencies (UD) version of the Late Latin Charter treebank (LLCT) sheds light on 521 Early Medieval Latin records (charters) from 774 CE to 897 CE. These charters present a non-standard Latin variety, focusing on legal documentary genres, and pose linguistic challenges due to their formulaic nature.[8] The dataset utilized in this work encompasses 8,173 tagged sentences, totalling 218,223 token-tag pairs.

Additionally, UDante, a project annotating Dante Alighieri's Latin works, includes 1,157 tagged sentences amounting to 38,086 token-tag pairs, focusing on 14th-century literary Medieval Latin.[9]

Moreover, the Pragmatic Resources in Old Indo-European Languages (PROIEL) project comprises 11,851 tagged sentences totalling 110,774 token-tag pairs, involving annotated texts such as the New Testament and Latin works like Cicero's "Epistulae ad Atticum" (Eckhoff et al., 2009).

Lastly, the Perseus dataset (version 2.1) features semi-automatically annotated texts like Cicero's "In Catilinam", Ovid's "Metamorphoses," and Augustus' "Res Gestae." Perseus did not originally use UPOS tags, so we mapped the Perseus tags[10] to UPOS tags. Notably, not all UPOS tags had direct equivalents in the Perseus tags. For instance, Perseus employed only "c" to represent conjunctions, lacking the differentiation between subordinating conjunctions (SCONJ) and coordinating conjunctions (CCONJ) as observed in UPOS tags. We omitted the files that included Caesar's "Commentarii de Bello Gallico" and Jerome's "Vulgata," as these texts were already included in the PROIEL dataset. The shortened Perseus dataset utilized in this project comprises 4,236 tagged sentences, totalling 68,283 token-tag pairs.

3.2 POS Tagging Models

We evaluated the performance of various POS tagging models on 16th-century epistolary Latin sourced from the Bullinger letters (i. e., the Bullinger sample mentioned in Section 3.1 and Table 1). The comparative analysis involved *LatinCy*, *CLTK*, *UDPipe*, *RDRPOSTagger*, and *TreeTagger*, alongside the examination of GPT-3.5-Turbo and GPT-4. The taggers compared in this study encompassed a spectrum of approaches, such as Single Classification Ripple-Down Rule (SCRDR) trees, statistical methods, and other distinct methodologies.

LatinCy, a *spaCy*-based Latin NLP toolkit introduced in 2023, employs *spaCy*'s (Montani et al., 2023) POS tagger,[11] backed by statistical models based on neural networks trained on the OntoNotes 5 corpus (Weischedel et al., 2013). With three core models, including "la_core_web_lg" utilizing sub-word vectors (Burns, 2023), *LatinCy* comprehends extensive vocabularies beyond its training data (Ács et al., 2021). Training incorporates diverse sources like Latin Universal Dependencies treebanks (Celano, 2019), Wikipedia or a preprocessed version of the cc100-latin corpus (Ströbel, 2023). Notably, the "la_core_web_lg" model we used for our research achieves an impressive

[8] See https://github.com/UniversalDependencies/UD_Latin-LLCT.

[9] See https://github.com/UniversalDependencies/UD_Latin-UDante.

[10] See https://github.com/PerseusDL/treebank_data/blob/master/v2.1/Latin/TAGSET.txt.

[11] See https://spacy.io/api/tagger.

97.41% accuracy for POS tagging on the respective test data (Burns, 2023).

The *Classical Language Toolkit* (CLTK),[12] established in 2014, caters to ancient languages like Latin and Greek, among others. CLTK's architecture supports various pre-modern languages, providing functionalities for POS tagging, tokenization, and lemmatization (Johnson et al., 2021). Utilizing *Stanza* (Qi et al., 2020) with bidirectional Long Sort-Term Memory networks (Hochreiter and Schmidhuber, 1997), it attains average accuracies of 68% for unigrams and 82% employing a 1, 2, 3-gram back-off tagger on the Perseus test data.[13]

UDPipe2 operates as a multifaceted pipeline, incorporating a neural network with a single joint model for tasks like POS tagging and dependency parsing. It leverages CoNLL-U format data and pre-trained word embeddings, where, e. g., the "Latin-ITTB" model achieved a high accuracy of 98.28% on the ITTB test data (Straka, 2018). Its flexibility spans over 50 languages, including non-Indo-European ones like Arabic and Irish (Wijffels, 2023).

RDRPOSTagger employs SCRDR trees for POS tagging across approximately 80 languages, with three available models showcasing varying accuracies for Latin (Nguyen et al., 2014). For our research, we employed the "UD_Latin-ITTB" model. Its conditional rule structure allows controlled interactions between rules, proving adaptable to languages like Latin. We used the "UD_Latin-ITTB" model that yielded an accuracy of 96.85% on the ITTB test set.

TreeTagger, developed through the University of Stuttgart's textual corpora project, adeptly annotates POS and lemma information in numerous languages like German, English, Chinese, Russian, Greek, and Latin. Its adaptability to new languages hinges on a lexicon and tagged training corpora, underscoring its versatility (Schmid, 1994). There are parameter files for numerous languages available; for Latin, we used the parameter file by Gabriele Brandolini. Functionally resembling traditional *n*-gram taggers, *TreeTagger* estimates transition probabilities using a binary decision tree and achieves accuracies ranging from around 95.8% to 96% on the Penn-Treebank for bigram and trigram versions (Schmid, 1995). However, its output does not directly use UPOS tags, necessitating a post-process tag mapping for compatibility.

Our study assessed two LLMs developed by OpenAI: GPT-3.5-Turbo and GPT-4. GPT-3.5, launched in 2022 and based on GPT-3 (Brown et al., 2020), boasts 175 billion parameters and excels in tasks like translation, text completion, and question-answering (OpenAI, 2023). Its architecture, excluding the encoder attention part, relies on an unmodified transformer decoder (Gupta, 2023). GPT-4, despite improvements, shares limitations like occasional unreliability and context window constraints. While GPT-4 shows superior task performance and visual data processing, only GPT-3.5-Turbo currently supports fine-tuning with custom data.[14]

The various models underwent a thorough assessment, revealing their capabilities and limitations in dealing with historical Latin texts. We applied each model mentioned above to the (test) datasets mentioned in Section 3.1 and compared the obtained accuracies. Incorporating conventional POS taggers and contemporary (fine-tuned) LLMs allowed for a direct comparison between traditional and LLM approaches.

4 Experiments and Results

4.1 Gold Standard

The initial phase involved the curation of a condensed Bullinger corpus comprising 200 sample sentences extracted from the Bullinger letters. Manual verification ensured a representative compilation spanning various editions, authors, and temporal contexts, subsequently stored in text files for further processing.

After the data curation process, the text was tokenized using the *spaCy* tokenizer with the "la_core_web_lg" model. We removed punctuation, including internal parentheses within words.[15] The subsequent application of the UDPipe tagger allowed for assigning reference tags to individual tokens, forming the foundation for creating an accurate gold standard dataset.

Multiple annotators, including a Latin expert, were involved in the verification and correction process of reference UPOS tags assigned by UDPipe. Any discrepancies between the assigned tags and the ideal classifications were addressed through

[12] See http://cltk.org/.

[13] For further details, see http://cltk.org/blog/2015/08/02/updated-accuracies-pos-taggers.html.

[14] As of 07.11.2023, when we conducted our experiments.

[15] E. g., the transcription could contain "vestr[um]" (EN *your*), where the editors added the "um" in parentheses. We removed the parentheses to obtain "vestrum".

manual verification. The collaboration with the Latin expert played a significant role in establishing a strong gold standard, incorporating mutually agreed-upon tagging principles. We employed Cohen's Kappa to assess inter-annotator agreement (IAA) on the corrected tag versions of the Bullinger corpus, yielding a noteworthy IAA of 0.97.

4.2 POS Tagging

Our study employed diverse POS tagging models, as detailed in Section 3.2, each utilizing distinct tagging methods. As test sets, we used the tokenized and manually tagged sample of the Bullinger corpus on the one hand and the tokenized test sets of the treebanks introduced in Section 3.1 on the other hand. We conducted manual post-processing on the taggers' output to ensure consistent token placement, aiming for uniformity in the models' outputs. This manual review became necessary because the taggers occasionally performed additional tokenization on certain words. Specifically, in the case of GPT models, they sometimes added extra text to the response, requiring careful verification to ensure uniform and comparable outputs.

Our experimentation also involved GPT-3.5-Turbo and GPT-4, which required specific prompts for accurate tokenization. We exclusively used system and user prompts, keeping all other parameters (like, e. g., temperature) unaltered. After encountering tokenization issues with the prompts used for the Bullinger test set, we refined the input by incorporating both the original sentence and a tokenized version aligned with our manually created gold standard. Figure 2 displays the used prompt. The variable "tokens" refers to the list of tokens and with the variable "sentence" we entered the sentence that should be tagged. Furthermore, we explored a token-only approach for the Bullinger corpus, submitting only tokens without a reference sentence in the prompt. The utilized prompt for this approach is displayed in Figure 3. Since we had used the already tokenized version of the corpora, we explicitly instructed the models in the user prompt not to perform additional tokenization. In contrast to the sentence-included approach, we focused solely on obtaining tags for individual tokens. Consequently, we requested only the tag for each token inserted into the prompt, replacing the "token" variable.

For the application of GPT models on treebank data lacking sentence boundaries, such as PROIEL, we developed an alternative strategy. Instead of providing complete sentences, we utilized sets of 65 tokens in the prompts, bypassing the requirement for punctuation to delineate sentences. This adaptation enabled the effective use of the GPT models without explicit sentence boundaries. The package size of 65 tokens was selected randomly, aiming to encompass the majority of sentences almost in their entirety. Even when complete inclusion was not possible, the chosen number ensured the presence of contextual information from the sentence, facilitating the disambiguation of words.

4.3 Fine-Tuning of GPT-3.5-Turbo

For the fine-tuning of GPT-3.5-Turbo, we obtained the training and test data from treebanks specified in Section 3.1 in the form of samples with an 80/20 split. When a pre-defined test set was absent, as was the case for PROIEL and Perseus, we randomly selected the test set. To explore how the model performance varies when provided with differently sized training sets, we crafted subsets for fine-tuning in sizes ranging from 50 to 10,000 sentences. We used stratified sampling to obtain the required examples from each training set to represent the different treebank sizes adequately. These subsets used for fine-tuning were formed by concatenating the chosen samples from the training sets from each resource listed in Section 3.1. E. g., "train5000" specifies a model fine-tuned on 5,000 sentences sampled from the different treebanks, taking the size of each treebank into account. We then prepared the data as required by the OpenAI API guidelines.[16]

4.4 POS Tagging Results

The pre-trained models' output required some manual post-processing, albeit not to the same extent as the GPT models' outputs. Significant discrepancies were noted within the GPT-generated content, encompassing repetitions, omissions of passages, instances of unexpected text occurrences (as illustrated in Figure 4), and irregularities such as unspecified tags and incorrect formats. These findings underscore limitations within the model's performance, notably occurring more frequently with models trained using larger datasets, such as train5000 and train10000.[17]

[16] See https://platform.openai.com/docs/guides/fine-tuning.

[17] The best-performing model on the Bullinger sample (train100) can be accessed with the model name `ft:gpt-3.5-turbo-0613:cl-uzh:train-100:8GWMiGKN`,

```
completion = openai.ChatCompletion.create(
    model = model,
    messages=[
        {"role":"system","content": """You are a Latin linguist and part-of-speech tagging expert.
        You are using UPOS (universal part of speech tags). UPOS tags are ADJ,ADP,ADV,AUX,CCONJ,DET,
        INTJ,NOUN,NUM,PART,PRON,PROPN,PUNCT,SCONJ,SYM,VERB and X. X stands for 'other'."""},
        {"role":"user",
        "content":f"""Return the UPOS tag for the tokens of the sentence: {sentence} The sentence should
        be tokenized like that: {tokens}. Return the tags in the format TOKEN \t Tag. Every Token-Tag
        pair should be on a new line in the output file, so add a newline character after the tags.
        Only output the token and the tag (no explanations, no translations, no additional text)."""}
        ]
    )
```

Figure 2: Prompt employed for POS tagging using the GPT API.

```
completion = openai.ChatCompletion.create(
    model = model,
    messages=[
        {"role":"system","content": """You are a Latin linguist and part-of-speech tagging expert.
        You are using UPOS (universal part of speech tags). UPOS tags are ADJ, ADP, ADV, AUX, CCONJ,
        DET, INTJ, NOUN, NUM, PART, PRON, PROPN, PUNCT, SCONJ, SYM, VERB and X. X stands for 'other'."""},
        {"role":"user",
        "content":f"""Do not tokenize the words further, they are already tokenized. Only output the tag
        (no explanations, no translations, no additional text). Return the UPOS tag for the token: {token}."""}
        ]
    )
```

Figure 3: Prompt employed for the token-only POS tagging approach using the GPT API.

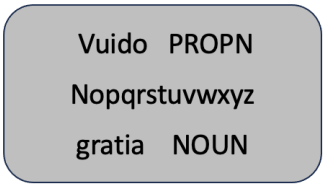

Figure 4: Example of GPT API output displaying unforeseen textual anomalies

After the manual clean-up of the output of all taggers on the Bullinger test set, involving the correction of tokens to adhere to the gold standard tokenization, especially in cases where taggers had further tokenized the input tokens, *LatinCy* demonstrated the highest accuracy (79.8%) among pre-trained models, slightly surpassing CLTK by 2.7%. RDRPOSTagger displayed the lowest accuracy (64.5%), indicating limitations in processing 16th-century data. Table 2 shows an overview of the results.

The token-only approach on the Bullinger corpus yielded the highest accuracy with the fine-tuned model train2000, reaching 78.2%. Remarkably, despite the absence of a reference sentence and the model being provided with only an individual token, the accuracy was nearly as high as for LatinCy. In contrast, the baseline model GPT-3.5-Turbo achieved only 62.2% accuracy in this approach, emphasizing the significant improvement brought about by fine-tuning. The non-fine-tuned GPT-4 achieved an accuracy of 74.3%, showcasing a clear performance difference between GPT-3.5-Turbo and GPT-4. The lowest accuracy, at 58.8%, was observed with the train500 model.

In the sentence-included approach, the best result was obtained using the train100 model, reaching an accuracy of 85.5% on the Bullinger corpus, surpassing the traditional tagging models by a significant margin. In this approach, the differences between the fine-tuned and the baseline models GPT-3.5-Turbo and GPT-4 were less pronounced. GPT-3.5-Turbo achieved an accuracy of 80.2%, and GPT-4 reached an accuracy of 83.5%. The model with the lowest accuracy in this scenario was train500, with an accuracy of 77.2%.

On the test data from the treebanks, *LatinCy* emerged with the highest average accuracy of the pre-trained models (83.22%), indicating effective performance as a POS tagger. The fine-tuned train1000 model exhibited the most effective performance (88.99%), whereas RDRPOSTagger had the lowest (72.36%). Notably, performance across various test sets varied, with ITTB showing the highest accuracy (84.95%) and PROIEL the lowest (74.73%).

the overall best model (train1000) is available under the name `ft:gpt-3.5-turbo-0613:cl-uzh:train-1000:8HUHOHgt`.

Tagger	Bullinger	ITTB	LLCT	UDante	Perseus	PROIEL	Avg TB
LatinCy	**79.8**	87.93	92.01	80.94	72.38	82.84	83.22
CLTK	77.1	88.45	79.32	77.36	72.44	80.63	79.64
UDPipe	72.8	71.59	71.45	70.51	69.2	84.49	73.45
RDRPOSTagger	64.5	82.67	67.41	71.72	64.79	75.22	72.36
TreeTagger	74.3	70.18	70.25	69.86	82.51	89.39	76.44
GPT-3.5-Turbo	62.2/80.2	74.82	78.82	74.33	68.81	79.22	75.2
GPT-4	74.3/83.5	79.73	84.89	77.62	73.9	84.38	80.1
train50	70.3/84.8	89.59	89.2	83.55	73.2	79.37	82.98
train100	69.4/**85.5**	89.73	91.03	85.26	74.19	82.19	84.48
train200	68.2/80.0	91.57	90.93	**85.8**	73.46	83.5	85.05
train500	58.8/77.2	93.2	93.85	82.71	72.09	86.72	85.71
train1000	65.8/82.5	**94.88**	**94.5**	84.94	81.39	89.25	**88.99**
train2000	**78.2**/78.3	87.95	87.43	81.31	**84.31**	87.83	85.77
train5000	71.9/76.6	88.11	84.85	74.62	81.99	**90.0**	83.91
train10000	74.4/76.4	83.84	85.39	75.36	76.34	86.01	81.39

Table 2: Tagger performance across different datasets. Bold numbers indicate the highest accuracy within each test set's column. The taggers starting with "train" are our fine-tuned GPT-3-5-Turbo models. For the GPT models, two numbers for the Bullinger data indicate the token-only and sentence-included approach (see Section 4.2). The average calculated over the test sets of the five treebanks is displayed in column Avg TB.

4.5 Tag Distribution

The taggers exhibit variations in their outputs. Some taggers allocate certain tags more frequently than others, owing to their dissimilar training data and learning algorithms. Moreover, not all taggers have been exposed to datasets encompassing all UPOS tags. In Figure 5, depicting tag distribution across four taggers and the gold standard on the Bullinger corpus data, common POS tags like NOUN and VERB are consistently assigned across all versions. However, notable differences emerge in the frequency of tags such as adverbs (ADV), determiners (DET) and pronouns (PRON). DET, for instance, appears significantly more in the gold standard compared to *LatinCy* or the basic GPT models. Only the fine-tuned model train100, boasting the highest accuracy on the Bullinger corpus, mirrors a similar frequency in assigning this tag. Conversely, *LatinCy* frequently uses ADV but assigns the tag for adpositions (ADP) less frequently than other taggers. The gold standard employs PRON less frequently, while GPT-4 allocates this tag almost twice as often as our gold standard did.

In analyzing the LLCT treebank data, a distinct disparity emerges in tag distribution when juxtaposed with the Bullinger corpus. Figure 6 illustrates the tag distribution of the LLCT data. Our comparison involves the tagging outputs of *LatinCy*, GPT-3.5-Turbo, GPT-4, the fine-tuned

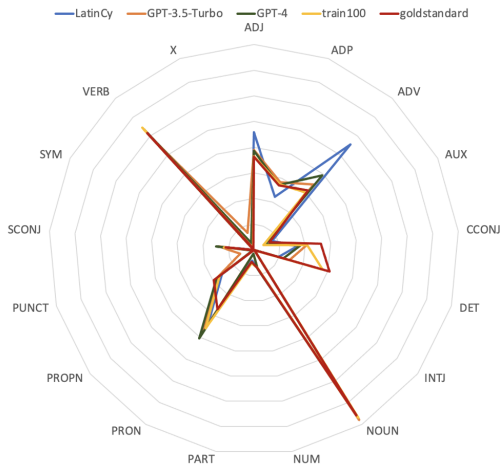

Figure 5: POS tag distribution in the Bullinger corpus.

model *train1000*, and the gold standard. Notably, a greater convergence among the models is evident, signifying more consistent tag assignments. Once again, NOUN and VERB exhibit striking similarities across these models, as do auxiliary verbs (AUX). However, a significant deviation surfaces with the other tag (X). *LatinCy* and the gold standard exclude its usage, while GPT-3.5-Turbo predominantly assigns this tag, potentially indicating problems encountered during tagging processes. A comparable pattern emerges when examining the distribution of coordinating conjunctions (CCONJ) and DET across Figures 5 and 6. While CCONJ dis-

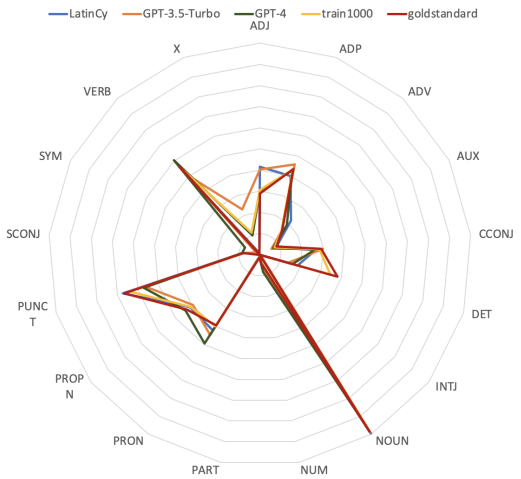

Figure 6: POS tag distribution in the LLCT test set.

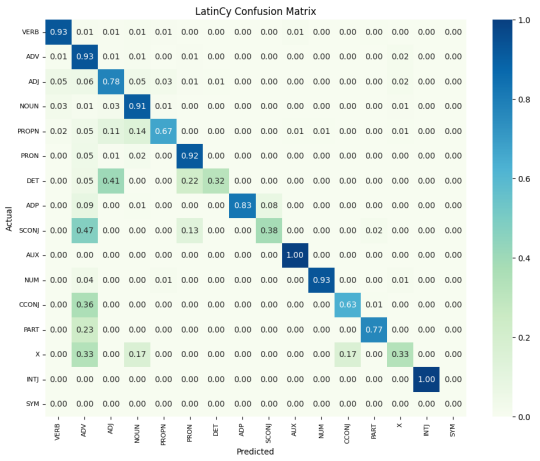

Figure 7: Confusion matrix for LatinCy on the Bullinger sample.

plays similarity across both test sets, the gold standard and fine-tuned models demonstrate a higher frequency in assigning this tag. Similarly, the DET tag exhibits a parallel pattern, with the fine-tuned models and the gold standard assigning this tag noticeably more frequently than the other three models.

5 Discussion

Evaluating part-of-speech tagging models within the context of 16th-century Latin texts provides valuable insights into language processing methodologies, particularly within historical frameworks. This comprehensive assessment reveals several noteworthy observations.

5.1 Fine-Tuning and Model Performance

One key finding is the significant impact of fine-tuning LLMs, such as GPT-3.5-Turbo, on POS tagging accuracy. Despite the superior performance of GPT-4 and GPT-3.5-Turbo, even without fine-tuning, the fine-tuned GPT-3.5-Turbo models outperformed conventional pre-trained taggers, especially on specific test sets. Especially the dominance over LatinCy, the most recent tagger that operates with transformer pipelines, is striking. This underscores the potential for domain-specific fine-tuning to enhance LLM efficacy in linguistically nuanced domains.

The evaluation aimed to assess the performance of various POS taggers on the Bullinger corpus and treebank test corpora. Evaluation of GPT models using token-only versus sentence-included approaches revealed notable differences. The sentence-inclusive method consistently achieved higher accuracy across all models, with a difference of over 11 percentage points compared to the token-only method. The model train500 showed the most significant difference between the two methods with 18.4 percentage points, while train2000 had only a difference of 0.1 percentage points.

The study also emphasizes the importance of prompting strategies when employing LLMs like GPT-4 and GPT-3.5-Turbo for POS tagging. Variations in accuracy between token-only and sentence-included approaches underscore the necessity of prompt engineering to improve contextual comprehension and enhance tagging performance.

5.2 Tag Assignment Challenges and Contextual Cues

Inconsistencies in tag assignments, especially for determiners and coordinating conjunctions, highlight the critical need for standardized definitions and categorizations. The study underscores the role of contextual cues in accurate POS tag assignments, particularly for ambiguous word classes like modal verbs and participles.

Confusion matrices were created to assess differences in tag assignment for the Bullinger sample. Figure 7 illustrates the comparison between the gold standard tags in the Bullinger sample and those assigned by LatinCy, while Figure 8 displays the tags assigned by GPT-4 using the sentence-included approach. These matrices present taggers' predictions along the x-axis and gold standard tags along the y-axis, providing insights into their performance.

LatinCy exhibits a nearly diagonal line, indicating generally accurate predictions. However, it

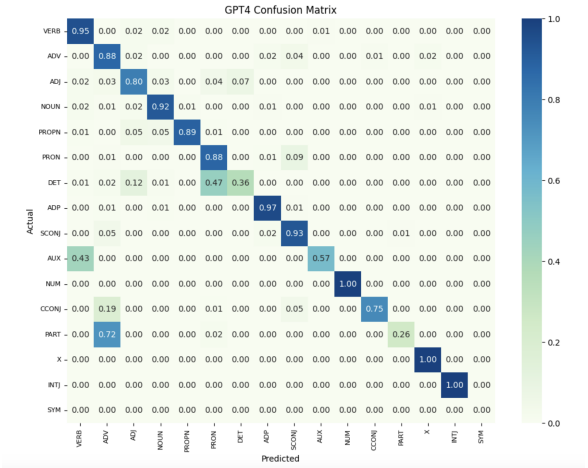

Figure 8: Confusion matrix for GPT-4 on the Bullinger sample.

correctly predicts the SCONJ tag only 38% of the time, incorrectly predicting ADV 47% of the time and PRON 13% of the time. In contrast, GPT-4 displays a more uniform and darker diagonal line, suggesting higher accuracy in tag assignments. GPT-4 demonstrates less dispersion in tag assignments than LatinCy, encountering difficulties in assigning the PART tag, mislabeling it as ADV (26%) and PRON (0.02%).

5.3 Comparative Analyses and Challenges in Applying Taggers

Comparative analyses, akin to prior studies such as Chu (2023), underscore the significance of prompting in part-of-speech tagging, emphasizing both similarities and disparities in the performance among various GPT models. Additionally, this investigation unveils challenges encountered when applying taggers to the Bullinger corpus, revealing notable differences in tagging standards and word usage.

The tagging process was time-consuming, particularly for GPT-4, hence each model was tested only once. While pre-trained models were faster, other GPT models occasionally necessitated several hours, contingent upon the overall global demand for GPT resources. Optimal efficiency was observed during off-peak hours for model testing, contrasting with markedly prolonged processing durations experienced during evening hours (UTC+1), reflecting heightened usage.

5.4 Implications and Future Directions

The study's findings underline the potential and constraints of traditional part-of-speech taggers and LLMs in historical Latin text analysis. The research serves as a pivotal impetus for future studies, prompting advancements in tagging precision and adaptability within historical and resource-limited language contexts. Further exploration of refined contextual models, standardized categorizations, and improved efficiency in deploying LLMs for extensive historical language analyses is encouraged.

5.5 Considerations for Resource Scalability

Despite the competitive performance, especially post-fine-tuning, concerns arise regarding the pragmatic utilization of LLMs in historical text analysis due to resource scalability challenges. The time-intensive nature of tagging procedures, particularly with models like GPT-4, raises considerations for their efficiency and scalability in large-scale historical language studies.

6 Conclusion

In conclusion, these multifaceted findings contribute to our understanding of part-of-speech tagging in historical Latin texts and pave the way for nuanced and targeted advancements in natural language processing within this domain. We could show that fine-tuning Large Language Models like OpenAI's GPT-3.5-Turbo can significantly heighten accuracy in part-of-speech tagging performance. We also provided insights into different prompting techniques for obtaining optimal results. However, the challenges related to stability and resource scalability, especially with time-intensive tagging procedures, raise considerations for the pragmatic utilization of Large Language Models in large-scale historical language studies.

Limitations

The study faced the following limitations: Tests, especially with GPT-4, were time-consuming, limiting the number of test runs due to extended processing times influenced by global demand and usage peaks. Performing only one test run per model with a single-epoch testing approach constrained a more thorough assessment of fine-tuning capabilities. Furthermore, formatting complexities in the output of models posed challenges, impeding the ability to adjust incorrectly formatted passages manually. This limitation hindered a more comprehensive analysis that could have been achieved through re-tagging sections if the process had been less time-intensive. Additionally, it is important to

note that our fine-tuning was exclusively conducted on the treebank data due to the absence of large-scaled gold standards for the 16th-century dataset. The 16th-century test set, comprising only 200 sentences, might not have fully represented that era's language complexities. Consequently, this limitation might have introduced an abundance of edge cases, potentially leading to decreased accuracy in the assessment.

References

Judit Ács, Ákos Kádár, and Andras 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*, pages 2284–2295, Online. Association for Computational Linguistics.

William Sidney Allen. 1989. *Vox latina*. Cambridge University Press.

David Bamman and Gregory R. Crane. 2006. The design and use of a latin dependency treebank. In *Proceedings of The Third Workshop on Treebanks and Linguistic Theories*, pages 67–78, Tübingen.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Bullinger Digital. 2023. Bullinger Digital.

Patrick J. Burns. 2023. Latincy: Synthetic trained pipelines for latin NLP.

Flavio M. Cecchini, Marco Passarotti, Paola Marongiu, and Daniel Zeman. 2018. Challenges in converting the index Thomisticus treebank into Universal Dependencies. In *Proceedings of the Second Workshop on Universal Dependencies*, pages 27–36, Brussels, Belgium. Association for Computational Linguistics.

Flavio M. Cecchini, Rachele Sprugnoli, Giovanni Moretti, and Marco Passarotti. 2020. UDante: First steps towards the universal dependencies treebank of dante's latin works. In *Proceedings of the Seventh Italian Conference on Computational Linguistics*, pages 99–105, Torino. Accademia University Press.

Giuseppe G.A. Celano. 2019. *The Dependency Treebanks for Ancient Greek and Latin*, pages 279–298. De Gruyter Saur, Berlin, Boston.

Lai-Sik Fan Chu. 2023. GPT-4 is a very good hongkongese POS tagger.

Hanne Eckhoff, Marek Majer, Eirik Welo, and Dag Haug. 2009. Breaking down and putting back together: Analysis and synthesis of new testament greek. *Journal of Greek Linguistics*, 9(1):56–92.

Yashu Gupta. 2023. Chat GPT and GPT 3 detailed architecture study-deep NLP horse.

Dag T. T. Haug and Marius L. 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*, pages 27–34.

Michael A. Hedderich, Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. 2021. A survey on recent approaches for natural language processing in low-resource scenarios. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2545–2568, Online. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

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, Online. Association for Computational Linguistics.

Daniel Jurafsky and James H. Martin. 2019. Part-of-speech tagging. In *Speech and Language Processing*.

Timo Korkiakangas. 2021. Late latin charter treebank: Contents and annotation. *Corpora*, 16:191–203.

Jürgen Leonhardt. 2013. *Latin: Story of a world language*. Harvard University Press, Cambridge, Massachusetts.

Barbara McGillivray. 2013. *Methods in Latin computational linguistics*, volume 1. Brill, Boston.

Ines Montani, Matthew Honnibal, Adriane Boyd, Sofie Van Landeghem, and Henning Peters. 2023. spaCy.

Sebastian Nehrdich and Oliver Hellwig. 2022. Accurate dependency parsing and tagging of Latin. In *Proceedings of the Second Workshop on Language Technologies for Historical and Ancient Languages*, pages 20–25, Marseille, France. European Language Resources Association.

Dat Quoc Nguyen, Dai Quoc Nguyen, Dang Duc Pham, and Son Bao Pham. 2014. RDRPOSTagger: A ripple down rules-based part-of-speech tagger. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 17–20, Gothenburg, Sweden. Association for Computational Linguistics.

OpenAI. 2023. GPT-4 technical report.

Marco Passarotti and Felice Dell'Orletta. 2010. Improvements in parsing the index thomisticus treebank. revision, combination and a feature model for medieval latin. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation*, pages 1964–1971, Malta.

Marco Passarotti, Eleonora Litta, Flavio Massimiliano Cecchini, Matteo Pellegrini, Giovanni Moretti, Paolo Ruffolo, and Giulia Pedonese. 2023. The LiLa knowledge base of interoperable linguistic resources for latin. architecture and current state.

Matteo Pellegrini, Eleonora Litta, Marco Passarotti, Rachele Sprugnoli, Francesco Mambrini, and Giovanni Moretti. 2021. LiLa linking latin tutorial. In *Proceedings of the Workshops and Tutorials-Language Data and Knowledge*, pages 229–234, Spain.

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.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing*, Manchester.

Helmut Schmid. 1995. Improvements in part-of-speech tagging with an application to German. In *Proceedings of the ACL SIGDAT-Workshop*, pages 13–25, Dublin. Springer.

Helmut Schmid. 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*, DATeCH2019, page 133–137, Brussels, Belgium. Association for Computing Machinery.

Rachele Sprungoli and Marco Passarotti. 2020. *1st Workshop on Language Technologies for Historical and Ancient Languages, Proceedings*. European Language Resources Association, Paris, France.

Manuel Stoeckel, Alexander Henlein, Wahed Hemati, and Alexander Mehler. 2020. Voting for POS tagging of Latin texts: Using the flair of FLAIR to better ensemble classifiers by example of Latin. In *1st Workshop on Language Technologies for Historical and Ancient Languages, Proceedings*, pages 130–135, Marseille, France. European Language Resources Association.

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.

Milan Straka and Jana Straková. 2020. UDPipe at evalatin 2020: Contextualized embeddings and treebank embeddings.

Phillip Benjamin Ströbel. 2023. pstroe/cc100-latin · datasets at hugging face.

Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, Martha Palmer, Nianwen Xue, Mitchell Marcus, Ann Taylor, Ann Houston, Eduard Hovy, Robert Belvin, Mohammed El-Bachouti, and Michelle Franchini. 2013. Ontonotes release 5.0.

Jan Wijffels. 2023. udpipe: Tokenization, parts of speech tagging, lemmatization and dependency parsing with the 'UDPipe' 'NLP' toolkit.