

LLM-ASSISTED MARKUP OF ENTITIES IN TEI: A CASE STUDY

JONAH CAUSIN^a, MORGAN FUKSA^a, ARNE KÄFER^b, JOSEPH (SANG WUK) LEE^a,
AND CLIFFORD ANDERSON^{a*}

Abstract *This paper presents a case study of marking up TEI (Text Encoding Initiative) documents using large language models. We detail our experiments using LLMs for named entity recognition and entity linkage in a corpus of TEI documents. The corpus consists of four journals co-edited by the Swiss-German theologian, Karl Barth (1886–1968). We aimed to enrich the TEI markup by adding XML elements to mark up entities and attributes to link to QIDs on Wikidata. We experimented with using both cloud-based frontier models and local open-source models to enrich a subset of 34 articles from the corpus. After analyzing the outcome using both human reviewers and automated analysis, we indicate where our methodology proved successful and where we encountered problems. We conclude that LLMs can perform named entity recognition and linking in TEI documents, but that the combined financial and labor cost of scaling this procedure to the full corpus would be relatively high without optimizing our current pipeline.*

Keywords: TEI, markup, artificial intelligence, human-machine collaboration

1. INTRODUCTION

This paper offers a case study of LLM-assisted markup of TEI-encoded journal articles. Our primary goal was to develop a methodology for using LLMs to detect entities in journal documents. Secondly, we hoped that LLMs could connect detected entities to authoritative sources, in this case, to the corresponding Wikidata entities.

This case study is motivated by the Mapping Theology (MapTheo) project. MapTheo is an international research collaboration led by Clifford Anderson, Frederike van Oorschot, and Johannes Fröh (“Mapping Theology [MapTheo]” 2025). The MapTheo project explores questions such as how individual researchers gain and lose influence within the broader theological research network, how institutions such as universities and publishing houses shape the contours of theological scholarship, and how research programs in theology rise and fall, among other topics.

Broadly speaking, the Mapping Theology project applies techniques from the nascent field of the ‘science of science’ to the nonscientific domain of theology. The ‘science of science’ is

^a Yale University

^b University of Tübingen

* Corresponding Author

an interdisciplinary field that uses big data to study how scientific research happens in practice. As Santo Fortunato et al. (2018, eaao0185) remark, ‘The contribution of SciSci is a detailed understanding of the relational structure between scientists, institutions, and ideas, a crucial starting point that facilitates the identification of fundamental generating processes.’ We hope to uncover similar structures in religious studies and theology while also investigating related questions, such as how interdisciplinary linkages from theological literature to the humanities, social sciences, and natural sciences shape contemporary theology.

To do this, we first need data. The science of science was born, in part, from the growing availability of bibliographic datasets (Fortunato et al. 2018). In the field of religious studies and theology, datasets have been more difficult to obtain. The primary bibliographic dataset in the United States is the Atla Religion Database (Atla 2026) and in Europe the IxTheo Database (Universitätsbibliothek Tübingen 2026). Beyond these sources, various open-source and digital humanities projects in the field make their data available for analysis.

We also need relevant markup. If we want to examine how scholarly actors relate to institutions, for example, we need to identify those entities within the text. Named entity recognition (NER) is a subfield of information extraction within natural language processing (Cunningham 2006). To expose citation networks and other latent information in and across datasets, it is crucial to identify named entities and also to connect them with authority systems.

In this trial, we explore how readily an AI-assisted workflow can mark up named entities in 34 articles from the journal *Zwischen den Zeiten*, published between 1923 and 1933. We started with a Text Encoding Initiative (TEI-encoded) (TEI Consortium n.d.) version of the journal provided by the Center for Barth Studies at Princeton Theological Seminary. We aimed to develop an efficient pipeline for marking up entities with relevant TEI tags and connecting those entities to corresponding items on Wikidata.

2. RELATED WORK

Recent studies on automated TEI encoding examine structural annotation, digital publication, and semantic enrichment, drawing on rule-based workflows, machine learning, and large language models. Recurring concerns include scalability, reproducibility, the role of context in model performance, and the integration of encoded texts with linked knowledge resources. These studies underscore the distinction between producing structured markup and interpreting textual meaning, highlighting the importance of explicit annotation guidelines and human review. These studies inform our Claude-assisted workflow, which combines corpus-wide TEI annotation with scholarly oversight to analyze literary and personal references and their relationships.

One established part of this research focuses on automatically identifying textual structures and representing them in TEI markup. Khemakhem, Foppiano, and Romary (2017, 598) detail an experimental approach that allows them ‘to automatically structure digitized dictionaries, independently of the language or the lexicographic school or style.’ Incorporating GROBID (GeneRation Of Bibliographic Data), a machine learning system that utilizes Conditional Random Fields (CRF) to identify and extract bibliographic metadata, the team developed their own TEI “cascading model.” This consisted of independent sequential models, each with its

own annotative objective. After each experiment, the generalized model would then be reapplied to other dictionary entries, with varying results. Our study, however, differs from this approach in that, rather than relying on “top-down,” linearly structured, independent models, our approach was more “bottom-up” in scope, with multiple Claude prompts annotating a broad range of TEI elements.

In 2023, Álvaro Andrés Rivera Parra described a method for automating TEI/XML encoding of José Asunción Silva poems as part of his master’s thesis in Digital Humanities (Rivera Parra 2023, 6). His research explored the advantages and challenges of using ChatGPT and OpenAI models, specifically monetary costs, labor requirements, and potential efficiency gains for humanities research (6, 32–33, 51–54). The study provides a baseline for larger-scale applications (51–54).

Floriane Chiffolleau’s “Keeping It Open: A TEI-Based Publication Pipeline for Historical Documents” (2024) develops the DAHN project’s publication workflow, which moves from digitization to encoding and publication. Developed using Paul d’Estournelles de Constant’s letters to Nicholas Murray Butler and the already-encoded *Letters and Texts. Intellectual Berlin around 1800*, the project utilizes eScriptorium, partially automated encoding scripts, and TEI Publisher to produce scholarly editions. The study focuses on transforming historical or ego documents from images into structured TEI-encoded digital editions.

Whereas Chiffolleau focuses on a TEI-based workflow for editing and publication, Santini (2024) analyzes how large language models (LLMs) and relation extraction (RE) can support extracting semantic relationships from TEI/XML-encoded Italian literary documents. Santini reports improvements in semantic accuracy and extraction consistency using REBEL over mREBEL, the multilingual relation extraction model developed by Huguet Cabot et al. (2023). The workflow uses ChatGPT to extract “natural language triples”—subject, predicate, object—statements. REBEL then processes these triples, converting natural-language inputs into RDF/XML and mapping them relationally via Wikidata QIDs. This integration of LLM-assisted extraction with structured knowledge resources offers a parallel to our case study, demonstrating the potential of such digital tools to support semantic enrichment in digital humanities research.

Working within computational drama analysis, Debaene and Hoste (2025, 137) apply ‘a semi-automated TEI encoding annotation methodology using transformer encoder language models’ to historical Dutch drama texts. Building on previous machine learning (ML) approaches, they aimed to reduce the time and labor required of human encoders in the Dutch Drama Corpora Project (DutchDraCor). By going beyond isolated text segments, they found that providing models with more context substantially improved task performance. Their study did not, however, examine how context quality affected model accuracy. Our study with Claude similarly highlighted the importance of context but also revealed its potential limitations: too much context can reduce accuracy and adherence to instructions. These observations informed our team’s decision to process files in batches.

Babenko and Athavale (2025) examine natural language processing (NLP) approaches and digital tools for literary analysis, situating TEI as a framework for encoding textual structure, contextual information, and intertextual connections. Rather than presenting an automated workflow for encoding documents as TEI, they advocate for combining computational analysis with qualitative philological interpretation, emphasizing the importance of cultural and

historical context in assessing results. Moving from literary analysis to historical parliamentary proceedings, Puren et al. (2025) also use NLP for their TEI model, using the French Chamber of Deputies (1889–1893) as a case study. Their initial Python-based approach encountered textual variability that limited reliable automated tagging, prompting them to adopt an annotation guide for manual processing. Their experience highlights the importance of explicit annotation criteria when automation cannot consistently accommodate heterogeneous sources, a concern shared by our human-supervised Claude workflow. Our project applies this principle through AI-assisted TEI encoding to identify and annotate names and references in literary texts. Like these scholars, we view human oversight as essential for validating the results of LLM-encoded markup.

Özcan, Erkoç, and Tokatlı (2026) describe a semi-automated workflow that combines TEI encoding with the Lexical Markup Framework (LMF) to create interoperable multilingual dictionaries, using Turkish as the pivot language. Automated encoding preserves the nested structure of dictionary entries, while expert review addresses contextual ambiguity, polysemy, and automation errors (Özcan, Erkoç, and Tokatlı 2026, 5, 15). Stanković et al. (2026) also use TEI, combining it with named entity recognition and linking (NER/NEL), textometry, and linked data to add semantic information to transcripts from the Serbian radio program *Digital Icons*. Their workflow links entities to Wikidata identifiers using QIDs and is intended to create a retrieval-augmented generation system. Although their corpus and retrieval goals differ from ours, their approach to entity normalization provides a useful methodological parallel to our study. Our project similarly uses Wikidata identifiers to normalize entities across the corpus, while combining LLM-assisted TEI annotation with human review to identify quotations and allusions and link them, where identifiable, to their sources.

Taken together, the studies reviewed here shift the question from whether TEI annotation can be automated to how reliably and efficiently it can serve scholarly research. The following sections explore this question through our annotation workflow, trials with a local model, and analysis of the corrections needed in practice.

3. METHODS

We analyzed a subset of the 1,005-article corpus: 34 articles from *Zwischen den Zeiten* (“Between the Times”), a theological journal co-founded by the Swiss-German theologian Karl Barth in 1923 and published by Chr. Kaiser Verlag in Munich during Germany’s Weimar Republic. The Center for Barth Studies at Princeton Theological Seminary originally digitized these articles in 2001–2002 as XML, following now-outdated TEI P3 or P4 guidelines. The Pacific Data Conversion Corporation carried out the original digitization using a double-keying process, which provides relatively accurate transcriptions of the underlying Fraktur text. The 34 articles are from volume 1, issues 1–7 of the journal (1923) and are labeled VBEB010101 through VBEB010706 in the original dataset and VBEB010101_P5 through VBEB010706_P5 in the working dataset, which we upgraded to the current TEI P5 standard. The TEI markup includes a `teiHeader` with a `sourceDesc` that provides information about the author, article title, editor, and imprint, among other information. In the TEI body, preexisting markup for headers, paragraphs, page breaks, and bibliographic references, along with other elements, provided initial structure for the analysis.

In this study, we used three software applications: git and GitHub for version control; Visual Studio Code, a code editor, for inspecting the XML documents; and Claude Code (versions 2.1.97–2.1.116) as a harness for calling a large language model. In particular, we used Claude Sonnet 4.6 as our LLM, though Claude Code directed some background requests to Claude Haiku 4.5, a cheaper and faster model. We also used Claude Opus 4.7 during the review phase. Claude Code sent 3,858 requests to Sonnet 4.6 and 438 to Haiku 4.5 during annotation (March 26–April 23, 2026), and 235 to Opus 4.7 during the audit of Wikidata QIDs (May 15, 2026) (see Table 7).¹

Our workflow proceeded as follows. First, we prompted the LLM to use a multi-pass approach, meaning it treats each group of taggable entities separately. The LLM therefore does not mix, for example, persons and quotations, which improved our results considerably. For each pass, the LLM is prompted to read the file in full and, if it exceeds 500 lines, divide it into chunks.

Next, the LLM compiles a checklist inventory of all taggable entities in the file. It then starts with the first entity group, such as persons, places, dates, quotations, or references, and draws up a second inventory specific to that group. Because German shows a great deal of variation in adjectival and genitive forms, the LLM is prompted to be aware of this characteristic and to cluster all related forms of a name such as “Luther”, including “Luthers” and “lutherisch.”

To identify these entities, we use the Model Context Protocol (MCP), an open standard created by Anthropic that allows AI assistants such as Claude to connect securely to external tools, databases, and data sources (Model Context Protocol 2026). Through MCP, we connected the LLM to the open knowledge base Wikidata.org. Wikidata assigns to every entry, such as a person, a place, a work, a specific QID, which allows us to enrich our tagging system with a stable identifier and thus to retrieve all mentions of an entity across the corpus, however it happens to be spelled. At this point, the LLM must be able to choose the correct QID among several entries bearing the same name: it needs to distinguish, for instance, between the person “Martin Luther” and the painting Martin Luther by Lucas Cranach. It is therefore prompted to draw on contextual information from the file, such as the date of publication and the content of the sentence in which the name occurs. A prompted double-check reduces errors, and a certainty level (low/medium/high/unknown) supplied by the LLM gives the human editor transparency.

The LLM also added the attribute `resp="#claude"`, which points to a `<respStmt xml:id="claude">` declared in the `<teiHeader>`. The key attribute holds the entity’s standard English label as given in Wikidata.² The certainty level ‘unknown’ does not mean the LLM found nothing. Instead, it signals that the LLM judged its result too uncertain to assert. In this case, it should omit both the reference attribute with QID and the key attribute.

Having gathered all this information, the LLM creates a template for every taggable entity it has found, here with Lazarus the Beggar as an example:

```
<persName ref="https://www.wikidata.org/wiki/Q21282948" key="Lazarus the
```

¹ We used OpenRouter to track requests to Claude, using two different accounts.

² In retrospect, we should have used the `n` attribute for our label since `key` should contain an identifier according to the TEI guidelines.

Beggar" cert="medium" resp="#claude">Lazarus</persName>

Finally, the LLM collects all templates in a Python script and applies them to the XML file by means of a simple replace function. An LLM self-check, which searches for missed occurrences and verifies the QIDs a second time, further reduces the number of errors. The LLM then continues with the next pass and thus with the next entity group.

One aspect of this process deserves closer explanation because of its complexity: the tagging of citations, unmarked quotations, and references. Three cases can be distinguished:

- First, whenever the LLM finds a full citation in the XML file, which is defined as a quotation marked as such by the author and accompanied by a source, it is prompted to a) mark the whole complex with a `<cit>` (citation) element, b) mark the quotation itself with `<quote>`, and c) mark the source given by the author with `<bibl>` (bibliographic citation). Within `<quote>`, the LLM decides whether the passage is a direct quotation, an indirect quotation, or a paraphrase, and supplies a certainty value and a responsibility attribute. If the quotation comes from the Bible, it adds a `<ptr>` (pointer) element carrying the exact biblical passage in OSIS format (a standardized format for consistency). The `<bibl>` element holds the source as printed by the author. Where that source is a biblical reference, the visible wording is wrapped in a `<ref>` element whose `cRef` (canonical reference) attribute holds the normalized OSIS form. However, where it is a non-biblical source, the LLM instead looks up the QID of the work and its English title for the `ref` and key attributes of the `<bibl>` on [wikidata.org](https://www.wikidata.org).
- Second, whenever the LLM finds a quotation, marked or unmarked, for which the author gives no source, it is prompted to tag it with `<quote>`, to decide whether it is a direct quotation, an indirect quotation, or a paraphrase, and to research the source on Wikidata. In the case of a biblical reference, it also identifies the exact passage, which is recorded in the `<ptr>` element described above. This feature is particularly delicate, since Claude is able to detect even indirect references. In Exod 3:14, for example, the biblical God says to Moses: ‘I am who I am’ (ESV). However, in one of our files the author alludes to this passage with ‘You are who you are’ (German original: *du bist, der du bist*; Gogarten, Friedrich: “Die Entscheidung,” ZDZ 1.1.3, München 1923), an allusion that is obvious to any theologically trained reader and therefore left unmarked by the author, but unclear to a general audience. Claude can identify such indirect references and link them to the biblical passage using a `<ptr>`.
- Third, whenever the LLM detects a quotation but cannot determine its source, it marks the passage with `<quote>` alone, without a Wikidata source or a pointer.

We should note that our methodology was not fully consistent. While marking up entities in the documents, the LLM used two different methods. In the first, the LLM used a Python script to annotate the entities it identified in the XML documents. In the second, the LLM added the markup directly to the documents. The LLM used the first method on 15 articles and the second on 19 articles.

Table 1 Elements tagged by the annotation pass in the 34 study articles, and how many it linked to Wikidata

Element	Tagged	Linked	%
---------	--------	--------	---

persName	2,181	2,078	95
quote	450	342	76
ref	374	—	—
date	285	—	—
title	188	102	54
placeName	168	159	95
ptr	61	—	—
orgName	49	38	78
author	40	26	65
docAuthor	23	21	91
bibl	15	1	7
Total	3,834	2,767	89

After completing its passes over the documents, Claude had tagged 3,834 elements across the 34 articles (or an average of 112.8 elements per article [median 73, range 11–610]). The LLM marked most of them as “*high*” (91.4%) confidence; the rest were marked “*medium*” (5.3%), “*unknown*” (2.1%), or “*low*” (0.8%). The remaining fourteen elements (0.4%) have no @cert at all.

4. OTHER METHODS CONSIDERED

We previously noted that our XML-tagging workflow relied on a cloud-based LLM running on Claude. To address the associated costs of using a frontier model, we implemented an alternative method: using a cost-efficient local LLM (Rivera Parra 2023, 32–33, 51–54). Using Ollama as an inference engine capable of executing command-line prompts, we installed Claude Code to run on an open-source model, Gemma 4 26b, allowing us to run tasks locally. Ollama also provides a chat interface comparable to those of cloud-based frontier models but does not require paid cloud application programming interfaces (APIs). Our computing environment was a Linux-based NVIDIA DGX Spark designed for testing local LLMs. While our sample size of three local LLM-assisted XML edits is admittedly limited, we hope that our dual approach and findings aid other researchers running into similar issues. Below, we summarize our local LLM workflow and briefly reflect on opportunities to improve its optimization and reproducibility.

At first glance, our local LLM workflow appears similar to our Claude-annotated workflow. The exception is that we do not follow up with human review of AI-assisted annotations to gauge the local model’s accuracy of edits. First, we prompt the local model to read through a single unannotated XML file located in *Zwischen den Zeiten*. Our prompt looked like this:

Task: Perform TEI entity tagging of unnamed person names for a *Zwischen den Zeiten* XML file.

1. Extract all untagged person names only inside the <body> section. Ignore <teiHeader>.

2. Query Wikidata via MCP for all extracted names.
 3. Apply all XML edits to the file using strictly this schema:
- ```
<persName ref="https://www.wikidata.org/wiki/QID" key="First Last"
cert="high" resp="#claude">Original Name</persName>
```

The completion of this task depended on several factors, such as the XML file size and, most importantly, the number of names recalled from Wikidata to query specific QIDs of persons. Files with fewer names fared better than files with many person names, as the local model often struggled to efficiently recall entities. Querying QIDs led to an often sluggish response for at least two reasons: the loop of triggering the Wikidata API, followed by the model’s “re-reading” of this information; and an Ollama-integration issue where the MCP server briefly stutters and re-connects. In sum, the tasks took anywhere from twenty to forty minutes, substantially slower than a cloud-based model.

To counteract these slow responses, we experimented with time-saving procedures, including the pre-processing of XML files into chunks through a Python script executed by the terminal. We tweaked what constituted a “chunk” by the number of paragraphs—from 5, 2, and then to 1. After slicing the files, we then asked the AI agent to apply XML tags directly to each chunk, modifying how many chunks the agent processed from 2 to 1. We then reassembled the chunks into a revised whole using a different Python script. By prompting the model to only process smaller file chunk(s), we optimized our efficiency slightly. The average task times for all chunks ranged from fifteen to twenty-five minutes. While the method of pre-processing XML files reduced the LLM’s workload, this did not reduce the bottleneck of recalling Wikidata. Automating this workflow would require the local model to run two Python scripts, one for processing the XML files into paragraphs and another for recombining them.

At this point, we briefly considered other open-source LLMs, such as Alibaba Cloud’s Qwen2.5-Coder-32B. This exploration was short-lived due to a server crash caused by an improper configuration on our local system. There were many other LLMs we could have tested, including the latest Qwen3.8 lineup and models in the Gemma 4 family. We also experimented with an alternative inference engine, vLLM, and the AI pair-programming assistant Aider, but ultimately stuck with Ollama for its ease of use.

The issues with our local LLM workflow were likely not caused by insufficient computing power, as the NVIDIA DGX Spark could, in theory, handle XML processing. Rather, the workflow failed to make full use of the workstation’s processing capacity and required frequent QID lookups, which caused disruptions on the MCP server. If we were to repeat this experiment using a local model, we would manually compile a glossary of Wikidata entries for frequently referenced individuals to reduce repeated MCP searches. Since our work with frontier models has already tagged many person entities in *Zwischen den Zeiten*, we could construct an initial glossary from those XML files. We could then use our local model to continually expand the glossary, further streamlining the annotation process as it grows. Names such as “Immanuel Kant,” “Martin Luther,” and “Plato” appeared frequently, and adding them to the glossary would have substantially reduced task execution time.

Our overall impression of the local LLM’s annotation precision was that it handled the strict Wikidata QID schema well, though it occasionally missed person entities and introduced typographical errors, such as unnecessary spacing between lines and changes to German words. These issues were less apparent when we prompted the local agent to process the entire XML

file in a single pass, but they became more pronounced when we tasked it with processing multiple chunks. For instance, although we did not prompt the local agent to edit the language itself, it changed “*Grundtendenzen*” to “*Grundtenden denzen*.” Elsewhere, it removed a comma, changed “*entsprechenden*” to “*entsprechend*,” and added extra spaces between lines, even as it annotated person entities with some accuracy. We are uncertain whether these typographical changes and alterations to the German—especially those involving the separation of complex compound words into smaller units—resulted from how the agent processed and reassembled multiple chunks sequentially.

## 5. EVALUATION

We conducted both an automated review and a manual review of the LLM-generated markup. Among other things, the automated review involved verifying the QID assigned to the entities. We used Claude Opus 4.7 on May 15, 2026, to check their correctness. This process revealed that some entities had been matched to incorrect QIDs and that all these mismatches had originally been flagged as “high confidence.” The problem seems to have stemmed from using the Wikidata labels for identifying matches by string labels (as noted above), though some mismatches seem to have been hallucinated altogether. In these cases, we wonder whether the MCP tool failed to call Wikidata correctly.

*Table 2 Examples of QID mismatches*

| Intended entity            | Asserted QID | What that QID denotes                |
|----------------------------|--------------|--------------------------------------|
| Søren Kierkegaard          | Q37462       | Skinnskatteberg Municipality, Sweden |
| F.W.J. Schelling           | Q43180       | Basco, Philippines                   |
| Wilhelm Lütger             | Q1373026     | Wadō (Japanese era, 708–715)         |
| Glauben und Wissen (Hegel) | Q1373040     | Lehelitz, a German village           |
| Hamann und die Aufklärung  | Q113792649   | an Indian journalist                 |

We note that human review did not catch four of these five sampled mismatches; only Q113792649 was removed during the human review phase.

Between April and August 2026, we also manually reviewed all of Claude’s annotated XML files, carefully assessing and validating each annotation. These annotations were assessed for their accuracy (were the QIDs and identified references correct); completeness (were there missing pieces of the tag, such as the QID, location, person); consistency (were the annotation guidelines upheld throughout every XML file); and formatting (was the annotation free of syntax errors; as §6 reports, this check was not reliably applied). We also consulted the original file to see whether Claude’s annotations altered it in any capacity (e.g., removed quotation marks, commas, etc.). When Claude made these errors, each tag was manually edited to adhere to our annotation guidelines, including changing the responsibility from Claude to our own attribute, before being marked as complete.

While Claude’s precision accelerated the annotation process, its production was not entirely error-free. There were annotations that required manual editing, largely due to missed identification or incorrect tagging of an identity or textual reference. Textual ambiguities were

another limitation, as Claude would occasionally attribute an indeterminate quote to a specific author, thus resulting in the manual removal of the tag during our review process. We made this change when it became difficult to definitively accredit the reference or authorial information. Further, Claude’s ability to assess the XML file and directly identify individuals was compromised at times, as its higher levels of uncertainty correlated to the absence of an identification tag. Relatedly, Claude’s functionality decreased when assigned to a substantial number of XML files. To address this, Claude was given a reduced number of files to aid in preventing annotative mistakes. We assume that this was necessary because of the limitation of the model’s context window.

## 6. RESULTS

To analyze the combined LLM-human changes to the TEI, we ran the git commit history through an LLM-generated Windows PowerShell script that created Excel spreadsheets of the changes and a high-level report of the editorial corrections. **Table 3** below presents the general results from the corrections report, including the number of commits, files analyzed by the script, Wikidata QIDs, and LLM character changes and corrections. The LLM supplied 2,791 QIDs across the 34 articles, an average of 82 per article drawn from 425 distinct Wikidata items. The human editors then altered 160 of the 3,834 elements that the LLM had tagged (4.2%), reassigning responsibility for 76 and removing 84. We take this statistic as an indication of editorial burden rather than accuracy, since some LLM errors passed human review without notice; for instance, four of the five mismatched QIDs in Table 2 survived the human review phase. While writing this paper, we checked every identifier in the final dataset against the Wikidata API. Of 2,915 identifiers (more than the 2,791 the LLM supplied, since reviewers added and reassigned some), 157 (5.4%) pointed to the wrong item: 146 by the LLM, 10 by human reviewers, and 1 with no responsibility attribute. This tally does not include borderline cases, and may prove higher on closer manual inspection.

*Table 3 Results of the correction report for the 34 study articles*

| Measure                             | Value         |
|-------------------------------------|---------------|
| Articles                            | 34            |
| Commits analyzed                    | 313           |
| <i>The annotation pass</i>          |               |
| LLM enrichment commits              | 19            |
| Elements tagged                     | 3,834         |
| Wikidata identifiers applied        | 2,791         |
| Distinct Wikidata items             | 425           |
| Characters written                  | 658,718       |
| <i>Review</i>                       |               |
| Elements touched by review          | 160 (4.2%)    |
| Commits declaring a correction      | 11            |
| Characters changed in those commits | 12,028 (1.8%) |

Note: Of the 160 elements that the human reviewers altered, they reassigned the @resp to themselves in 76 cases and removed 84 altogether. The reviewers were inconsistent in signaling corrections in their commits, so the declared corrections undercount the true number.

Our script also categorizes five types of corrections (or “axes”) to describe manual changes/markups at a broader level. These axes are as follows: coverage (supplying a markup that is absent altogether), linking (disputing *which* entity the LLM identified), transcription (removing <unclear/> markups and altering transcribed text), extent (disputing *where* a tag was drawn), and housekeeping (changing attributes, such as @cert and @resp, and other minor features). We subdivided these axes into more specific changes describing what human correction was done, such as adding, removing, or replacing a Wikidata QID. Additionally, our script included a categorization of the “severity” of manual changes (major or minor) for further subdivision, though nearly all manual edits were categorized as ‘major changes’ due to ‘minor changes’ being classified entirely under the housekeeping axis, which itself was relatively rare.

*Table 4 Most frequent categories of correction*

| Axis     | Category                               | Hunks |
|----------|----------------------------------------|-------|
| Coverage | Entity element added                   | 58    |
| Coverage | Entity element removed                 | 34    |
| Linking  | Wikidata QID replaced                  | 32    |
| Linking  | Reference target changed (not the QID) | 29    |
| Linking  | Entity label (@key) replaced           | 23    |

Note: A single hunk may satisfy more than one category, so the total does not sum to 134 hunks.

As expected, the coverage and linking categories comprised the vast majority of the collected data, as shown in **Table 4** in descending order of occurrence (for the five most common). Although not shown in the table, the coverage categories ‘Wikidata QID supplied’ and ‘Entity label (@key) supplied’ were also quite common—occurring 18 and 14 times, respectively.

Finally, **Table 5** provides an overview of our script-LLM-human workflow, including the number of commits, file edits, character changes, and edit categorizations (major, minor, and unclassified).

*Table 5 Overview of the script-LLM-human workflow for the 34 study articles*

| Edit class                       | Commits | File edits | Char. changes | Major  | Minor | Unclass. |
|----------------------------------|---------|------------|---------------|--------|-------|----------|
| Initial import of the 1 P5 files |         | 34         | 1,626,176     | —      | —     | —        |
| LLM enrichment 19 pass           |         | 102        | 658,718       | 4,153  | 721   | 24       |
| Script normalization             | / 37    | 399        | 542,689       | 20,887 | 397   | 0        |
| Human encoding                   | 245     | 320        | 159,708       | 1,061  | 412   | 69       |

|                                           |     |        |           |        |       |    |
|-------------------------------------------|-----|--------|-----------|--------|-------|----|
| Correction of LLM 11<br>output (declared) | 12  | 12,028 | 131       | 0      | 3     |    |
| Total                                     | 313 | 867    | 2,999,319 | 26,232 | 1,530 | 96 |

As a concluding note, we should indicate that several of the rules we set out in our method section were not followed or not applied consistently. In the final dataset, for instance, forty-seven elements carried an *@key* label even when the LLM marked the element with the *cert="unknown"* attribute. The LLM failed to wrap citations in *cit* elements. Biblical passages are referred to inconsistently. In a pre-project stage, we included a *<refsDecl>* indicating our intention to use Society of Biblical Literature (SBL) references for biblical texts; in the prompt, however, we indicated a preference for Open Scripture Information Standard (OSIS) references. The pre-project files used a third, different encoding. Claude’s conversion of these references to OSIS was incomplete, leaving 27 of 463 in their earlier form. In the case of non-biblical sources, the LLM generally failed to provide Wikidata QIDs, only linking 1 of 15 in the dataset. We hypothesize this may be because many bibliographical sources do not yet exist as items in Wikidata. In the *quote* elements, Claude interactively added a fourth item (“allusion”) to the three we enumerated in the original prompt. This value was applied to six elements; a human reviewer removed two of these instances, leaving the remaining four in the dataset. Both LLM and human reviewers broke the well-formedness of the XML documents: the LLM was responsible for one of these instances and the human reviewers for the other two. In retrospect, we should have added a parsing step to the verification stage to make sure the XML was well-formed and the TEI remained valid. Finally, we changed the *resp* attribute during human review, breaking the provenance of the LLM-supplied markup. We also did not add *<respStmt>* elements for the human reviewers in the TEI headers, leaving their *resp* attributes as dangling pointers.

## 7. DISCUSSION

The case study suggests that LLMs can mark up named entities in TEI and link them to Wikidata items. However, the accuracy of their markup remains in question since more than 5% of QIDs in the final dataset are incorrect.

In addition, the cost of using frontier models should also be considered. We tracked the cost of Claude Code’s LLM calls using OpenRouter (OpenRouter, Inc. n.d.). The average cost for annotating an article was thus \$9.35 (Table 7). However, this cost does not include student labor to check the added markup for accuracy or the LLM review pass. Were both additional costs factored in, the total estimated cost for extending this procedure to the entirety of the corpus would be significantly higher.

*Table 6 Model usage costs for the 34 study articles*

| Model             | Requests | Spend    | % of total |
|-------------------|----------|----------|------------|
| Claude Sonnet 4.6 | 3,858    | \$312.57 | 85.1       |
| Claude Opus 4.7   | 235      | \$49.19  | 13.4       |
| Claude Haiku 4.5  | 438      | \$5.34   | 1.5        |

|       |       |          |       |
|-------|-------|----------|-------|
| Total | 4,531 | \$367.10 | 100.0 |
|-------|-------|----------|-------|

Note: Metered usage 26 March–15 May 2026

*Table 7 Cost by phase*

| Phase                    | Dates        | Requests | Spend    | Per article |
|--------------------------|--------------|----------|----------|-------------|
| Annotation, first passes | 26, 29 March | 600      | \$21.34  | \$0.63      |
| Annotation               | 7–23 April   | 3,696    | \$296.57 | \$8.72      |
| Verification of QIDs     | 15 May       | 235      | \$49.19  | \$1.45      |
| Total                    |              | 4,531    | \$367.10 | \$10.80     |

LLM costs ranged from \$1.16 to \$12.95 per article during April 16–23; this excludes the labor cost of human review.

We believe, however, that we can develop more efficient ways to combine Python code with LLM calls to reduce these costs. For example, we could build lookup tables of entities as we move through the corpus, effectively caching that information locally and only using LLM calls for new or indeterminate entities. We have constructed a new pipeline with that feature and other efficiencies, but we have not yet evaluated its effectiveness against our corpus.

## 8. LIMITATIONS

We note several limitations of this study, which stem from our trial-and-error methodology. As a lab, we were learning to use these tools while applying them to our data. For this reason, we limited the number of documents we processed to a subset of 34. As we wrote up our methodology and evaluation, we also candidly recognized that our experimentation with methods renders it difficult to reproduce our results. For example, we updated our prompt during the study, seeking for more accurate results. For this reason, we consider this paper a descriptive case study with suggestive metrics rather than a controlled analysis.

A further limitation of this study is that our process began before March 2026, evolving over several months in late 2025 and early 2026. We experimented with different LLM-assisted techniques and evaluated their performance using subjective measures until we found a pattern that produced results we considered satisfactory. The large number of commits in the git repository prior to the LLM enrichment pass indicates the amount of uncontrolled preparatory work we performed on the TEI before beginning our study.

Another limitation is that we did not perform a head-to-head comparison with existing NLP methods, such as the combination of spaCy (Honnibal et al. 2020) and entity-fishing (Lopez 2023). A head-to-head analysis of the differences in accuracy, compute time, and cost would help motivate the decision to switch (or refrain from switching) to LLM-assisted methods. We propose conducting this comparative analysis for future research.

## 9. CONCLUSION

We have shared our experience using a frontier LLM to detect and annotate entities in TEI documents. Using an MCP tool, the LLM can also link those entities to the corresponding items on Wikidata, when available. However, LLMs make considerable errors even when using MCP tools to look up information on Wikidata.

Of course, in the pre-LLM era, natural language processing tools could also perform named entity recognition in TEI texts with reasonably high accuracy, and tools such as entity-fishing disambiguated entities by connecting them to corresponding Wikidata items (Lopez 2023). So, it is not surprising that LLMs using an MCP tool can also carry out this twofold task.

However, it remains an open question whether this LLM-assisted workflow is the most efficient way to detect, annotate, and link entities in TEI documents. To answer that question, we need to establish compute time and cost compared with deterministic methods. LLM-assisted workflows provide a convenient way to perform named entity recognition and resolution in TEI documents, assuming you have the tokens to burn.

## ACKNOWLEDGMENTS

The authors thank Kaitlyn Dugan, Director of the Center for Barth Studies at Princeton Theological Seminary, for providing the initial TEI version of the journals. We also thank Olivia Mendlinger, former technical assistant at the Divinity Library, for her assistance with the literature review.

## DATA AND CODE AVAILABILITY

The TEI code for this project, along with the Claude Code artifacts, is available in a private GitHub repository. Please address requests for access to the TEI editions of Barth's journals to the Center for Barth Studies at Princeton Theological Seminary.

## COMPETING INTERESTS AND DISCLOSURES

The authors declare no competing interests. We disclose that we used Claude to develop our methodology and to analyze our results. We also used Grammarly and Claude to edit our prose and ChatGPT, Claude, and the AI LaTeX assistant in Overleaf to troubleshoot the LaTeX.

## ORCID

Jonah Causin  <https://orcid.org/0009-0008-9593-5672>  
Morgan Fuksa  <https://orcid.org/0009-0003-9745-2627>  
Joseph (Sang Wuk) Lee  <https://orcid.org/0009-0002-4946-0772>  
Clifford Anderson  <https://orcid.org/0000-0003-0328-0792>

## REFERENCES

- Atla. 2026. Atla Religion Database (Atla RDB). <https://www.atla.com/research-tool/atla-religion-database/>.
- Babenko, Irina, and Vijay Anant Athavale. 2025. “Methods of Literary Text Analysis with the Help of Artificial Intelligence.” In *DET-2024: International Conference on Distance Education Technologies*, edited by Maria Lapina, G.S. Prakasha, Deepanraj Balakrishnan, Natalia Gorbunova, Walaa H. Elashmawi, and Viktoriya Taran, 81–95. Cham: Springer Nature Switzerland. [https://doi.org/10.1007/978-3-032-04365-8\\_7](https://doi.org/10.1007/978-3-032-04365-8_7).
- Chiffolleau, Floriane. 2024. “Keeping It Open: A TEI-Based Publication Pipeline for Historical Documents.” *Journal of the Text Encoding Initiative*, no. 15. Accessed July 14, 2026. <https://doi.org/10.4000/12s01>.
- Cunningham, H. 2006. “Information Extraction, Automatic.” In *Encyclopedia of Language & Linguistics*, 2nd ed., edited by Keith Brown, 5:665–77. Oxford: Elsevier. <https://gate.ac.uk/sale/ell2/ie/preprint.pdf>.
- Debaene, Florian, and Veronique Hoste. 2025. “Classifying TEI Encoding for DutchDraCor with Transformer Models.” In *Proceedings of the 19th Linguistic Annotation Workshop (LAW-XIX-2025)*, edited by Siyao Peng and Ines Rehbein, 137–41. Vienna, Austria: Association for Computational Linguistics. Accessed July 14, 2026. <https://doi.org/10.18653/v1/2025.law-1.11>.
- Fortunato, Santo, Carl T. Bergstrom, Katy Börner, James A. Evans, Dirk Helbing, Staša Milojević, Alexander M. Petersen, et al. 2018. “Science of Science.” *Science* 359 (6379): eaao0185. <https://doi.org/10.1126/science.aaa0185>.
- Honnibal, Matthew, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-Strength Natural Language Processing in Python. <https://doi.org/10.5281/zenodo.1212303>.
- Huguet Cabot, Pere-Lluís, Simone Tedeschi, Axel-Cyrille Ngonga Ngomo, and Roberto Navigli. 2023. “REDFM: A Filtered and Multilingual Relation Extraction Dataset.” In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, edited by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, 4326–43. Toronto, Canada: Association for Computational Linguistics. <https://doi.org/10.18653/v1/2023.acl-long.237>.
- Khemakhem, Mohamed, Luca Foppiano, and Laurent Romary. 2017. “Automatic Extraction of TEI Structures in Digitized Lexical Resources Using Conditional Random Fields.” In *Electronic Lexicography in the 21st Century: Lexicography from Scratch. Proceedings of eLex 2017*, edited by Iztok Kosem, Jelena Kallas, Carole Tiberius, Simon Krek, Miloš Jakubiček, and Vít Baisa, 598–613. Leiden, Netherlands: Lexical Computing CZ s.r.o. Accessed July 14, 2026. <https://hal.science/hal-01508868v1/>.
- Lopez, Patrice. 2023. entity-fishing. <https://github.com/kermitt2/entity-fishing>. swh:1:dir:cb0ba3379413db12b0018b7c3af8d0d2d864139c.
- “Mapping Theology (MapTheo).” 2025. Forschungsstätte der Evangelischen Studiengemeinschaft (FEST). Accessed September 6, 2026. <https://www.fest-heidelberg.de/forschung/projektdatenbank-2/detail/nachricht/id/66641-mapping-theology-maptheo/?cb-id=214748>.
- Model Context Protocol. 2026. “What Is the Model Context Protocol (MCP)?” Documentation version 2026-07-28. Accessed September 23, 2026. <https://modelcontextprotocol.io/docs/2026-07-28/getting-started/intro>.
- OpenRouter, Inc. n.d. OpenRouter. Accessed September 26, 2026. <https://openrouter.ai/>.

- Özcan, Emrah, Mehmet Fatih Erkoç, and Hasan Tokatlı. 2026. “Machine-Readable Multilingual Dictionaries: A Semi-Automatic ISO–TEI Model.” *Mersin Üniversitesi Dil ve Edebiyat Dergisi* 22 (1): 1–20. Accessed July 14, 2026. <https://dergipark.org.tr/en/pub/mersinjll/article/1881023>.
- Puren, Marie, Fanny Lebreton, Aurélien Pellet, and Pierre Vernus. 2025. “From Parliamentary History to Digital and Computational History: A NLP-Friendly TEI Model for Historical Parliamentary Proceedings.” *Digital Scholarship in the Humanities* 40 (suppl. 1): i75–i86. Accessed July 14, 2026. <https://doi.org/10.1093/lc/fqae071>.
- Rivera Parra, Álvaro Andrés. 2023. “Automatización de marcado de texto en TEI.” Universidad de los Andes. Accessed July 14, 2026. <https://doi.org/10.57784/1992/70089>.
- Santini, Cristian. 2024. “Combining Language Models for Knowledge Extraction from Italian TEI Editions.” *Frontiers in Computer Science* 6: 1472512. Accessed July 15, 2026. <https://doi.org/10.3389/fcomp.2024.1472512>.
- Stanković, Ranka, Tamara Vučenović, Biljana Rujević, Milica Ikonić Nešić, and Mihailo Škorić. 2026. “Integrating TEI, NER/NEL, Textometry, and Linked Data for a Semantically Enriched Interview Corpus.” In *Proceedings of the Fifteenth Language Resources and Evaluation Conference*, edited by Stelios Piperidis, Núria Bel, Henk van den Heuvel, Nancy Ide, Simon Krek, and Antonio Toral, 1474–84. Palma de Mallorca, Spain: ELRA Language Resource Association. <https://doi.org/10.63317/4zjef9ycsupk>.
- TEI Consortium. n.d. Text Encoding Initiative. Accessed September 26, 2026. <https://tei-c.org/>.
- Universitätsbibliothek Tübingen. 2026. Index Theologicus (IxTheo). <https://ixtheo.de/>.