Introducing an Open Source Library for Sumerian Text Analysis

Hansel Guzman-Soto and **Yudong Liu**
Computer Science Department
Western Washington University
Bellingham, Washington 98225
{guzmanh,liuy2}@wwu.edu

Abstract

The study of Sumerian texts often requires domain experts to examine a vast number of tables. However, the absence of user-friendly tools for this process poses challenges and consumes significant time. In addressing this issue, we introduce an open-source library that empowers domain experts with minimal technical expertise to automate manual and repetitive tasks using a no-code dashboard. Our library includes an information extraction module that enables the automatic extraction of names and relations based on the user-defined lists of name tags and relation types. By utilizing the tool to facilitate the creation of knowledge graphs, which is a data representation method offering insights into the relationships among entities in the data, we demonstrate its practical application in the analysis of Sumerian texts.

1 Introduction

The study of Sumerian texts offers a valuable opportunity to gain insights into the earliest written languages and its associated historical context. Assyriologists have conducted studies such as prosopography (Jacobs, 2007; Dahl, 2007; Liu, 2021) and social network analyses (Kulikov et al., 2021; Pottorf, 2022) on these texts, enabling a deeper understanding of administrative and economic history as well as the involved families and individuals during the Ur III period (ca. 2112-2004 BC). However, this type of studies often necessitates the identification of named entities and their relationships within a specific timeframe, demanding domain experts to meticulously examine a vast number of ancient Sumerian tablets. This process can be time-consuming and challenging.

Currently, non-technical users primarily depend on SQL and Excel to perform repetitive tasks such as manually locating and recording instances of individuals and their relationships across tablets. Not only does this result in a less intuitive interface, but it also is not scalable. Additionally, given that Sumerian is a low-resource language, the availability of dedicated software tools is scarce, limiting scholars' access to user-friendly NLP (natural language processing) toolkits.

To address these issues, we introduce an open-source library that facilitates the seamless integration of processing and NLP models, thereby enabling more comprehensive and expedited analysis of Sumerian texts. The library consists of two key components: a pipeline and a dashboard. Currently the pipeline provides functionalities for data processing and information extraction, equipping users with the necessary tools to build robust and efficient software solutions. The dashboard offers a user-friendly interface which requires minimal technical preparing for domain experts to automate their workflow in analyzing Sumerian tablets, ultimately accelerating their research progress.

2 Related Work

There are existing tools that process or perform NLP tasks tailored for specific tasks such as Machine Translation (Pagé-Perron et al., 2017; Punia et al., 2020) and Sumerian text annotation (Tablan et al., 2006; Smith, 2010; Liu et al., 2015; Luo et al., 2015; Chiarcos et al., 2018). Most notably, the Cuneiform Digital Library Initiative (CDLI) hosts several repositoriesthat process Sumerian in various data formats such as CoNLL-U and RDF (Resource Description Framework), and perform various NLP tasks. Although these tools may provide versatility for different tasks, they require adequate technical knowledge for modifying their utilization. Without such expertise, modifying these resources to accommodate the diverse requirements of Assyriology can be daunting. Therefore, the need for a more accessible platform becomes apparent, underscoring the importance of our work in this space. To the best of our knowledge, no existing dashboard currently allows scholars to easily uti-

Proceedings of the Ancient Language Processing Workshop associated with RANLP-2023, pages 133–137,
held in Varna Bulgaria, Sept 8, 2023.
https://doi.org/10.26615/978-954-452-087-8_016

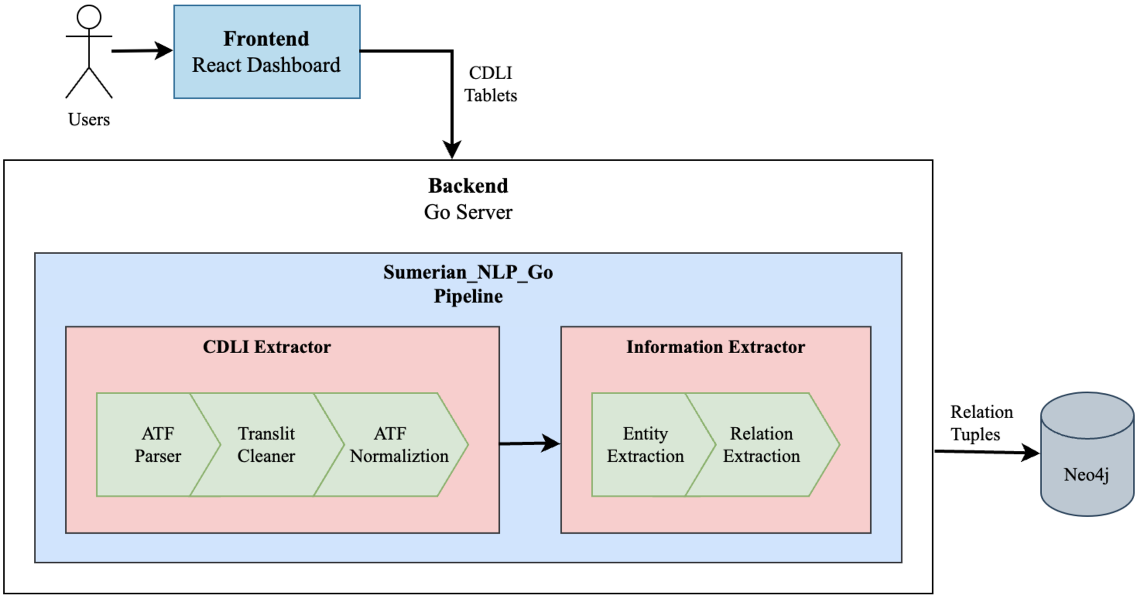


Figure 1: Architecture of the system.

lize tools or scripts specifically designed for the analysis of Sumerian tablets.

3 System Description

3.1 System Architecture

Fig. 1 illustrates our system's structure. It features a dashboard interface for users to upload their own data such as tablets or a customized list of named entity tags used by the backend pipeline. The backend pipeline handles user requests, such as data annotation, entity extraction or relation extraction, using specific library components which can be configured by the users on the fly. Additionally, users can create knowledge graphs, stored in a Neo4j database, leveraging the system's entity and relation extraction capabilities.

The library, accessible via this link[1] is designed to enable researchers to seamlessly integrate their workflow into our pipeline for their specific use cases. While the implementations are still relatively preliminary, the modular nature of the components involved ensures their adaptability for a wide range of applications. In the following sections, we will provide detailed descriptions of each component we have developed.

3.2 CDLI Extractor

The entry point to the pipeline is the CDLI Extractor, comprising three components: ATF (a text markup format used by CDLI to describe inscriptions on Cuneiform tablets and other artifacts) (Robson, 2014) Parser, Transliteration Cleaner, and ATF Normalizer. This component is built to load and process tablets from the CDLI repository, which are written in ATF.

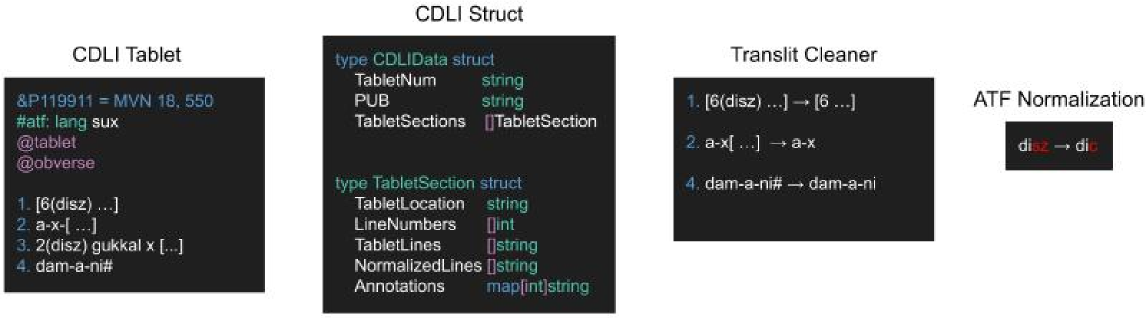
ATF Parser reads tablets in ATF format and stores it into an internal data structure that preserves all metadata, tablet content and positional information. For now, we support data from CDLI which has tablets in ATF format. Other formats exist such as Open Richly Annotated Cuneiform Corpus (ORACC) (Robson, 2014) and the Database of Neo-Sumerian Texts (BDTNS) (Molina, 2002) for which we plan to offer support.

Transliteration Cleaner then handles broken tablets and normalizes transliterations to follow a specific format. For example, for the transliteration of "1(disz)", we may want this to map simply to "1" because the meaning is intact but it is easier for us to process.

ATF Normalizer aims to establish a standardized format enabling the uniform processing of data from diverse sources, including CDLI, ORAAC, and BTDNS. Currently, this component normalizes CDLI data to a unified format, with plans to extend its functionality to standardize data formats from other sources.

Fig. 2 shows a working example of this module.

[1] https://github.com/WWU-Sumerian-NLP



```
&P119911 = MVN 18, 550
#atf: lang sux
@tablet
@obverse

1. [6(disz) ...]
2. a-x-[ ...]
3. 2(disz) gukkal x [...]
4. dam-a-ni#
```

```
type CDLIData struct
    TabletNum        string
    PUB              string
    TabletSections   []TabletSection

type TabletSection struct
    TabletLocation   string
    LineNumbers      []int
    TabletLines      []string
    NormalizedLines  []string
    Annotations      map[int]string
```

```
1. [6(disz) ...] → [6 ...]

2. a-x[ ...]  → a-x

4. dam-a-ni# → dam-a-ni
```

```
disz → dic
```

Figure 2: A working illustration of the CDLI Extractor. The ATF parser takes a CDLI tablet and parses and stores it into an internal data structure. The Translit Cleaner then performs cleaning on numeric symbols, damaged annotations, and annotator's correction or guesses markers. Finally, the Normalizer standardizes transliterations from various tablet sources.

3.3 Information Extractor

As named entities and entity relationships are often the key information for Sumerian text analysis, our Information Extractor module currently contains two modules: Entity Extractor and Relation Extractor.

Entity Extraction We use a simple approach based on string matching. In this process, each word is examined, and target words are labelled with entity types drawn from a list of known entities (Bansal et al., 2021). As aforementioned, the pipeline is designed to be flexible, allowing users to input a customized list of named entity labels to be processed. While simple, this approach effectively automates the manual annotation work and establishes a centralized platform to leverage the annotated data for downstream tasks. Future iterations will port existing Named Entity Recognition models to our library and provide them as option to users.

Relation Extraction It involves finding connections between entities. The process involves the application of user pre-defined rules for relations using regular expressions. The pipeline allows users to define and pass a list of regular expressions for the system to search through.

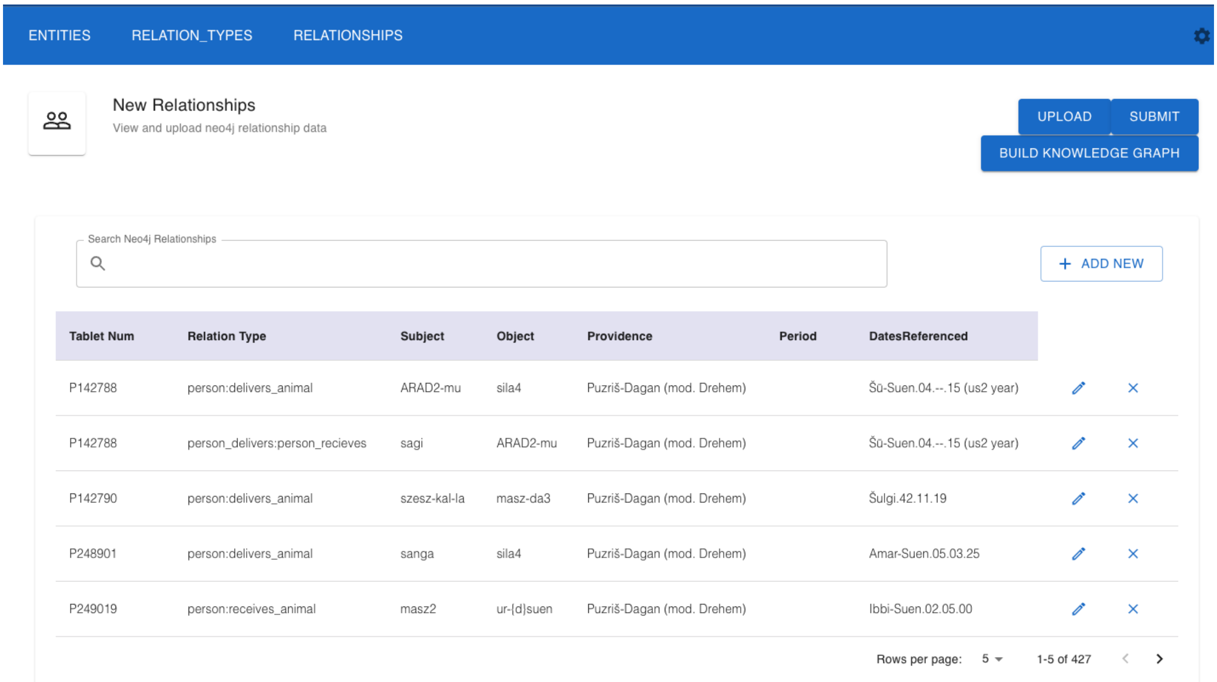
3.4 Dashboard for Non-technical Users

A No-Code Dashboard To facilitate the use of our library, we have developed a user-friendly dashboard that enables users to view, modify, and upload their data (see Fig. 3). It currently supports the following features: 1) Upload, add or delete entity names with their corresponding entity tag. 2) Define relation types with specific pattern rules. Our application takes these patterns, iterate through all data, and display the results to the user. 3) Configure different components within the pipeline. For example, users could configure ATF parser to filter by tablet metadata such providence or by broken tablets. 4) Search or filter for the results of each components output. 5) Download data or use the relationship tab to feed relations to a knowledge graph stored in Neo4j.

Server Architecture We have a server in place that acts as an intermediary between our dashboard and NLP libraries. Our server's backend imports our NLP libraries to use for each task and stores data in a relational database to maintain the state of data across multiple services. For server implementation, we use the Mux library in Go. The dashboard is designed for easy extension. To support a new tab for a service, users only need to create a new form in the frontend, add an entry in our server's database, and create a corresponding endpoint in our backend that uses the service. We are also developing features that will allow users to access their own Sumerian tablets for a variety of downstream tasks.

4 Evaluation and Implementation Considerations

We aim to create reproducible, replicable tools that can be easily customized and interchanged within the Assyriologist community. This is reflected in our pipeline architecture which will allow for the seamless integration of various components, enabling users to modify and adapt the tools according to their specific needs. This modularity not only promotes the reuse and repurposing of individual



Tablet Num	Relation Type	Subject	Object	Providence	Period	DatesReferenced		
P142788	person:delivers_animal	ARAD2-mu	sila4	Puzriš-Dagan (mod. Drehem)		Šū-Suen.04.--.15 (us2 year)	✎	✕
P142788	person_delivers:person_recieves	sagi	ARAD2-mu	Puzriš-Dagan (mod. Drehem)		Šū-Suen.04.--.15 (us2 year)	✎	✕
P142790	person:delivers_animal	szesz-kal-la	masz-da3	Puzriš-Dagan (mod. Drehem)		Šulgi.42.11.19	✎	✕
P248901	person:delivers_animal	sanga	sila4	Puzriš-Dagan (mod. Drehem)		Amar-Suen.05.03.25	✎	✕
P249019	person:receives_animal	masz2	ur-{d}suen	Puzriš-Dagan (mod. Drehem)		Ibbi-Suen.02.05.00	✎	✕

Figure 3: Dashboard interface designed for the streamlined data upload and knowledge graph generation through interactive widgets: “Entities”, “Relation_Types”, and “Relationships”. With the “Entities” widget, users can input entity lists with tags, triggering entity extraction across their dataset. Extracted entities are then cataloged in a database and displayed in a corresponding table.

components but also encourages collaboration and knowledge sharing within the Assyriologist community.

Our decision to utilize the Go programming language, is primarily motivated by its speed. It offers up to a 30-fold speed increase, resulting a highly responsive user dashboard. For example, tasks such as entity extraction, which could require 5-20 minutes in Python, now demand only 1-5 seconds in Go. As we continue to introduce more customization options, algorithms and features, maintaining this speed becomes essentials for a good user experience. Furthermore, this efficiency extends to server interactions, ensuring swift communication between the frontend and backend.

5 Use Case: Creating Knowledge Graphs with Our Tools

Knowledge graphs serve as a powerful tool for representing data as a network of interrelated entities, enabling us to answer queries such as “who did what to whom”. For illustrative purposes, we draw upon the work (Liu, 2021) to demonstrate the use of knowledge graphs in studying prosopography of a family engaging in an animal delivery business during the Ur III period. In this knowledge graph, nodes represent entities such as people, animals, and locations. Connections between nodes depict relationships or actions, and each connection is enriched with tablet metadata, including tablet number, year, and region. For instance, the node ‘ARAD2-mu’ (a person) is connected to the node ‘sila4’ (lambs) with an edge labeled ‘delivers’. The graph not only illustrates the volume of deliveries, recipients, and geographic routes but also provides a comprehensive view of individual interactions over time and space. It gives insights into the networks of individuals and the broader prosopological landscape, shedding light on societal structures, relationships, and economic dynamics.

6 Conclusion and Future Work

This paper introduces an open-source library designed to empower domain experts in processing and analyzing Sumerian cuneiform tablets through an no-code dashboard. The application of knowledge graphs enhances the analysis via large-scale entities and relation visualization. The current implementation shows an initial but promising step in accommodating configurable components that are agnostic to various NLP tasks. As the pipeline’s capabilities expand, we invite collaborations to broaden its applications, potentially encompassing a wider range of ancient Mesopotamian languages.

References

Rachit Bansal, Himanshu Choudhary, Ravneet Punia, Niko Schenk, Jacob L Dahl, and Émilie Pagé-Perron. 2021. How low is too low? a computational perspective on extremely low-resource languages. arXiv:2105.14515.

Christian Chiarcos, Ilya Khait, Émilie Pagé-Perron, Niko Schenk, Jayanth, Christian Fäth, Julius Steuer, William Mcgrath, and Jinyan Wang. 2018. Annotating a low-resource language with llod technology: Sumerian morphology and syntax. *Information*, 9(11):290.

J.L. Dahl. 2007. *The Ruling Family of Ur III Umma: A Prosopographical Analysis of an Elite Family in Southern Iraq 4000 Years Ago*. Peeters Publishers Booksellers.

Dennis Jacobs. 2007. The secret life of judges. *75 Fordham L. Rev. 2855*.

Anya Kulikov, Adam Anderson, and Niek Veldhuis. 2021. Sumerian Networks: Classifying Text Groups in the Drehem Archives. *IDEAH*. Https://ideah.pubpub.org/pub/q22859lx.

Changyu Liu. 2021. Prosopography of individuals delivering animals to Puzriš-Dagan in Ur III Mesopotamia," Akkadica 142/2, 2021, pp. 113-142. *Akkadica*, 2021(24.0):112–142.

Yudong Liu, Clinton Burkhart, James Hearne, and Liang Luo. 2015. Enhancing sumerian lemmatization by unsupervised named-entity recognition. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1446–1451.

Liang Luo, Yudong Liu, James Hearne, and Clinton Burkhart. 2015. Unsupervised sumerian personal name recognition. In *The Twenty-Eighth International Flairs Conference*.

Manuel Molina. 2002. Bdtns: Database of neo-sumerian texts.

Émilie Pagé-Perron, Maria Sukhareva, Ilya Khait, and Christian Chiarcos. 2017. Machine translation and automated analysis of the Sumerian language. In *Proceedings of the Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 10–16, Vancouver, Canada. Association for Computational Linguistics.

Andrew Pottorf. 2022. *Social Stratification in Southern Mesopotamia during the Third Dynasty of Ur (ca. 2100–2000 BCE)*. Ph.D. thesis, Harvard University Graduate School of Arts and Sciences.

Ravneet Punia, Niko Schenk, Christian Chiarcos, and Émilie Pagé-Perron. 2020. Towards the first machine translation system for sumerian transliterations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3454–3460.

Steve Tinney Eleanor Robson. 2014. Oracc: The open richly annotated cuneiform corpus.

Eric JM Smith. 2010. *Query-Based Annotation and the Sumerian Verbal Prefixes*. University of Toronto.

Valentin Tablan, Wim Peters, Diana Maynard, Hamish Cunningham, and K Bontcheva. 2006. Creating tools for morphological analysis of sumerian. In *LREC*, pages 1762–1765.