BabyLemmatizer 2.0 – A Neural Pipeline for POS-tagging and Lemmatizing Cuneiform Languages

A. J. Aleksi Sahala
University of Helsinki, Finland
aleksi.sahala@helsinki.fi

Krister Lindén
University of Helsinki, Finland
krister.linden@helsinki.fi

Abstract

We present BabyLemmatizer 2.0, a linguistic annotation pipeline for POS-tagging and lemmatizing cuneiform languages, as well as pretrained models for a variety of ancient Mesopotamian languages and dialects. We evaluate the system on two dialects of Akkadian: Assyrian and Babylonian, as well as on two genealogically unrelated cuneiform languages: Sumerian and Urartian. We also test our system on Ancient Greek and Latin to experiment with its performance on non-cuneiform languages. Our system achieves a POS-tagging accuracy between 95-98% and a lemmatization accuracy of 94-96% depending on the language or dialect. The system can predict correct POS-tags for 83-91%, and lemmata for 68-84% of out-of-vocabulary word forms depending on the language or dialect.

1 Introduction

Lemmatization is a linguistic annotation task that labels words with their dictionary forms. This is essential for morphologically complex highly inflectional and agglutinative languages, where the relationship between surface forms and their dictionary forms are opaque. In historical languages with less standardized spelling, lemmatization becomes even more crucial because also the relationship between the surface forms and their graphemic representations may be obscure, and make searching attestations of words belonging to highly inflectional part-of-speech classes difficult, or close to impossible, without cumbersome regular expression based search queries.

This issue can be demonstrated with the Akkadian verb *nadānu* "to give", which occurs in 367 different surface forms and in 477 different spellings in the Open Richly Annotated Cuneiform Corpus (Oracc) (Tinney et al., 2006). The simplest finite surface form, the third person singular G-present *inaddin* "he/she gives" is spelled in seven different ways in Oracc: logographically IN.SUM, SUM, SUM{+*in*}, logo-syllabically SUM-*in* and syllabically *i-na-din, i-na-di*$_3$*-in* and *ina-ad-din*. Similarly the third person singular G-preterite and G-perfect forms *iddin* and *ittadin* are spelled in eight and five different ways in Oracc, respectively.

Part-of-speech (POS) tagging is another important concept in the NLP of morphologically complex languages. Besides its obvious use, that is, searching for words that belong to a certain POS-class, POS-tags can be used to some extent to disambiguate lemmatization. For instance, in Akkadian the logogram IGI can denote various concepts depending on its context. If preceded by a preposition, it often denotes being in front of something (e.g. *ina* IGI = *ina pāni*), but in other contexts it can also mean *šību* "witness", *nāmuru* "be(come) visible" or *īnu* "eye", among many other readings and meanings.

Traditionally Akkadian and other cuneiform language lemmatization and POS-tagging has been done with rule-based systems, including a dictionary-based and morphology-based methods. The disadvantage of dictionary-based lemmatizers is that they are unable to provide POS-tagging or lemmatization for previously unseen word forms. Although morphology-based tools can produce annotations for unseen word forms as long as their lemmata and morphology have been defined, they struggle to deal with spelling variation, which is difficult to describe using rules without producing excessive over-generation.

In this paper we present an OpenNMT-based neural lemmatizer and POS-tagger for Akkadian and other cuneiform languages. The presented neural network based approach aims to solve both of the previously mentioned issues. It can learn many-

[0]Aleksi Sahala was responsible for developing the tool, training and evaluating the models and writing the paper. Krister Lindén was the PI of the project and provided feedback for the manuscript. BabyLemmatizer 2.0 is accessible at https://github.com/asahala/BabyLemmatizer

to-many relations between all spellings of word forms and their possible lemmata in context, and use these learned mappings to predict annotation for previously unseen word forms, also in previously unseen spellings.

Lemmatization of cuneiform texts opens them to a variety of computational methods, including, but not limited to semantic and network analysis, and enables harmonization of existing resources, which is essential for the digitalization of Assyriological research.

2 Cuneiform and the Cuneiform Languages

The cuneiform writing system was used in ancient Mesopotamia from the middle of the fourth millennium BCE until the first or the second century CE. According to modern understanding, it was first developed by the Sumerians and later adapted by speakers of several other languages such as Akkadian, Elamite, Hittite, Hurrian and Urartian.

Originally cuneiform was a logographic writing system, where all the signs denoted various concepts, such as numbers and commodities that were relevant to trade, taxing and ownership in the early Mesopotamian society. Around 2800 BCE the writing system took its first clear steps toward a more phonetic expression of human language by allowing certain logograms to be used for marking syllabic values (Michalowski, 2008). For example, the sign KA that originally denoted the Sumerian word for mouth /kag/ began also to mark a phonetic syllable /ka/, which allowed ancient scribes to express more abstract ideas such as combinations of grammatical affixes. After the cuneiform writing system had been adopted to the East-Semitic Eblaic and Akkadian languages around the 25th century BCE, the use of syllabic signs became widespread, as logograms alone were too ambiguous for expressing the Semitic stem-internal morphology (Michalowski, 2008).

Cuneiform signs can be used for four distinct purposes. The two basic uses are *logograms* that express ideas such as "king", "wife", "temple" or "to build", and *syllabograms* that express syllable-like sounds like /ma, mu, mi, me/. The remaining uses are *determinatives* and *phonetic complements*. The former were used to classify words into various categories, such as divine names, trees or wooden objects, and places among many others. Phonetic complements, on the other hand, were used sporadically to give hints on how a logogram next to them should be read by repeating some of its sounds syllabically (Jagersma, 2010).

2.1 Transliteration of Cuneiform

Transliteration of cuneiform aims to represent the original text in the Latin alphabet sign by sign. Conventionally, logograms are written in capital letters (except in Sumerian), and the syllabic signs are always written in lowercase. The marking of determinatives and phonetic complements vary. In paper publications they are written in superscript, but in the Oracc notation they are wrapped in curly brackets. Phonetic complements are distinguished from determinatives using a plus sign, as in APIN{+*ru*} for the Akkadian word *ikkaru* "farmer" (Tinney and Robson, 2019).

Another detail relevant to this paper in cuneiform transliteration is indexing that aims to separate cuneiform signs with similar readings from each other. The index of the sign (or its reading) is expressed in subscript numbers. For instance, there are two common cuneiform signs that indicate the syllable /šu/. To keep these two signs separate in transliteration, they are transliterated as *šu* and $\textit{šu}_2$, allowing the reader to know which sign was used in the original source. This makes the transliteration of cuneiform reversible and more transparent.

2.2 Languages

For this paper, relevant languages are Akkadian, Sumerian and Urartian.

Akkadian is best known as the language of the Babylonians and Assyrians. It belongs to the East-Semitic languages and is documented in writing from the Old Akkadian period ca. 2400 BCE to the first or the second century CE. The Assyrian dialect, once spoken in the northern Mesopotamia, is divided into three chronological variants: Old Assyrian (1950-1500 BCE), Middle Assyrian (1500-1000 BCE), and Neo-Assyrian (1000-612 BCE). The Babylonian dialect is divided into Old Babylonian (2000-1500 BCE), Middle Babylonian (1500-1000 BCE), Neo-Babylonian (1000-626 BCE) and Late Babylonian (626 BCE-100 CE). An artificial language known as Standard Babylonian was also used in literary contexts by Akkadian speaking scholars for over a millennium. Although this language was based on Old Babylonian, the texts written in Standard Babylonian often contain residue from the contemporary spoken Babylonian and Assyrian dialects.

Akkadian features a complex morphology that combines linear (prefixation and suffixation) and nonlinear (root-pattern morphology and infixation) processes (Huehnergard and Woods, 2008). Akkadian is written mostly using syllabic signs, but a selection of logograms is also used. The extent of logogram use varies depending on the time period and genre. Typically, everyday texts, such as letters, do not contain many logograms, but they are abundantly used until the later time periods in scholarly texts.

Sumerian was an isolate language first attested in writing in the middle of the fourth millennium BCE. Sumerian died as an everyday vernacular in the 18th century BCE and transformed into a literary language used by the Babylonian and Assyrian scholars in various contexts until the end of the cuneiform tradition circa the first or the second century CE (Jagersma, 2010). As an agglutinating language with 10 grammatical cases, possessive suffixes and heavy verbal prefixation its morphology is quite rich, not as opaque as that of Akkadian. Although Sumerian is generally written in a logo-syllabic manner, the earliest texts were purely logographic, and some texts written after the second millennium BCE used only syllabic signs. Counterintuitively, these syllabic, so-called unortographic, texts are often the most difficult ones to understand due to their high ambiguity (Michalowski, 2011).

Urartian was a language spoken in Asia Minor and the northern reaches of Mesopotamia. It belonged to the Hurro-Urartian language family and is attested between the 9th and the 7th century BCE on inscriptions written in the Neo-Assyrian cuneiform script (Wilhelm, 2008). Similarly to Sumerian, Urartian is a heavily agglutinating language with a complex morphology, including nine grammatical cases, Suffixaufnahme (stacking of nominal suffixes in genitive constructions) and rich verb affixation. Due to the relatively low number of surviving inscriptions and their repetitive nature, the Urartian language is far less understood than Akkadian or Sumerian.

3 Digital Resources

The most relevant digital resource to the work presented in this paper is the Open Richly Annotated Cuneiform Corpus, better known as Oracc (Tinney et al., 2006). It contains ca. 112,000 cuneiform texts in various languages, including but not limited to Sumerian, Akkadian and Urartian. Other important digital resources include the Cuneiform Digital Library Initiative (Englund et al., 1998) (ca. 350,000 entries, including texts and metadata), Database of Neo-Sumerian Texts (Molina, 2002) (ca. 105,000 Neo-Sumerian administrative documents), The Electronic Babylonian Library Fragmentarium (Jiménez et al., 2018) (16,000 fragments), Archibab (Charpin, 2009) (10,000 texts), Ebla Digital Archives (Milano and Maiocchi, 2016) (3,000 texts) and Achemenet (Briant and Henkelman, 2009) (4,000 texts). For a more detailed survey on cuneiform language resources, see Charpin (2014).

Oracc contains ca. 2.22 million words of Akkadian as of 2023. As the total number of words in known Akkadian tablets and inscriptions has been estimated to be around 10 million (Streck, 2010), a majority of Akkadian texts remain unannotated to date.[1]

For Sumerian, Oracc hosts texts comprising 4.45 million words and they include the vast majority of the important Sumerian texts and archives. Most of the Sumerian data has already been lemmatized, and therefore the need for Sumerian annotation tools is not as urgent as it is for Akkadian. Nonetheless, many witnesses of Sumerian composite texts still lack lemmatization.

For Urartian, Oracc contains texts comprising 26,000 words, 24,000 of which have already been lemmatized. The total number of non-digitized texts existing outside Oracc is not clear to us.

4 Previous Work

Akkadian lemmatization and POS-tagging have been approached with finite-state morphology on several occasions since the late 1980s. The first attempt to morphologically analyze, lemmatize and POS-tag Akkadian with finite-state transducers was taken by Kataja and Koskenniemi (1988). Barthélemy (1998) and Macks (2002) used Prolog Definite Clause Grammars for parsing Akkadian verbal morphology, and later a procedural approach to Akkadian verb morphology was taken by Sahala (2014). Bamman (2012) built a finite-state model for lemmatizing Old Assyrian letters, and Sahala et al. (2020) published the BabyFST, a finite-state model for Babylonian.

[1]There is no reliable estimate of the total number of Akkadian words in various digital resources, but alongside Oracc, at least 30,000 texts exist in other digital resources with varying accessibility (Charpin, 2014).

For Sumerian, morphological analysis, POS-tagging and lemmatization have been done with the GATE Java Suite (Tablan et al., 2006) and more recently with a dictionary-based approach by Chiarcos et al. (2018).

To date, the most comprehensive lemmatizer for cuneiform languages is L2 (Tinney, 2019), a dictionary and rule-based tool that has been used to annotate Oracc. L2 is also capable of providing morphological analysis for Sumerian.

For a more comprehensive survey on Computational Assyriology see Sahala (2021), and on the use of Machine Learning in ancient language processing Sommerschield et al. (2023).

5 Data

All our cuneiform language data comes from Oracc JSON dumps downloaded in January 2023. For the experiments done in this paper, we extracted all the texts written in Sumerian, Akkadian and Urartian.

We selected the data from Oracc as follows:

- The **Urartian** data set comprised all texts from Oracc labeled as Urartian, the majority of the data coming from the eCUT (Christiansen et al., 2016). In total, this set consisted of 24,000 words.

- The **Neo-Assyrian** data set comprised all texts from Oracc labeled as Neo-Assyrian dialect. In total, this set consisted of 331,000 words. This corpus consists mostly of royal inscriptions and letters that primarily come from SAAo (Radner et al., 2005) and ATAE (Novotny et al., 2017).

- The **First Millennium Babylonian** data set consisted of all Oracc texts labeled as any variant of Babylonian or Akkadian, excluding Neo-Assyrian, in the first millennium BCE, thus containing Standard Babylonian, Neo-Babylonian and Late Babylonian texts. In total, this consisted of 1.33 million words belonging to a wide range of genres. The largest portions of data came from RINAP (Frame et al., 2007), ADSD (Pirngruber et al., 2018), SAAo, RIBO (Frame et al., 2015) and HBTIN (Pearce et al., 2011).

- The **Sumerian (literary)** data set consisted of all Sumerian texts in Oracc's ePSD2/Literary, eSD2/earlylit and ePSD/Praxis* (Tinney et al., 2017). In this data set, the subscript indices were not removed from the Sumerian data as homophones with different indices can belong to different POS-classes and denote completely different lemmata (see section on tokenization). We chose to separate literary texts from administrative texts due to their differing vocabulary and grammar. This data set comprised 268,000 words.

- The **Sumerian (administrative)** data set consisted of all Sumerian Early Dynastic, Old Babylonian, Old Akkadian, Ebla and Lagaš II administrative texts in Oracc's ePSD2 corpus. The Ur III corpus was excluded because it would have completely overwhelmed this data set with its 81,000 texts. As in the data set above, the subscript indices were preserved. This data set consisted of 570,000 words.

To test our system on historical non-cuneiform languages, we used the Latin and Ancient Greek PROIEL treebanks (Haug and Jøhndal, 2008), comprising 205,000 and 210,000 words respectively.

5.1 Training Data Cleanup

All our models are trained by using the Oracc data, but we run it through heuristic cleanup rules to provide more consistent learning results for the models and to minimize the amount of unwanted and meaningless errors in the evaluation, such as {*d*}x-x being tagged as a divine name in one place but as an unknown POS-class somewhere else.

For the Akkadian data, we merge some inconsistent lemmatizations with their most common representations in the data (e.g. *aganutillû* vs. *agannutillû*) and correct obvious lemmatization errors such as *bēlēšu*, which is de facto the phonological transcription of the word instead of lemma. We also apply the Helsinki normalizations (Jauhiainen et al., 2019) to all divine names in the corpus to make their lemmatization consistent. Therefore variation such as *Anunnaki, Anunnaku* and *Anunak* is consistently mapped into *Anunnaki*. Unfortunately the normalizations are available only for names that occur in the first millennium Akkadian texts.

For all cuneiform languages, we do the following normalizations:

1. Remove all lacuna indicators such as various brackets, exclamation marks and question marks.

2. Remove all entries that have been transliterated as asterisks. This convention is used in some composite texts where that only exist in phonological transcription, such as *iddinū* for varying spellings in the witnesses like *id-di-nu* and *id-di-nu-u*$_2$.

3. Remove all entries without lemmatization unless they are supposed to be unlemmatized, as is the case of lacunae (breakages in tablets) and numbers.

4. Remove all lemmatizations from numbers and force the POS-tag *n* to them.

5. Force POS-tags for broken personal names, place names and divine names if the determinative is visible but the words have not been POS-tagged in Oracc. This can be done in high confidence for divine names and personal names.

6. Force the POS-tag *u* for broken words in case they do not have a POS-tag.

We did not do any modifications to the Ancient Greek and Latin data. These data sets were used as they are distributed in the Universal Dependencies GitHub repository.

6 Description of the System

Our system first pre-annotates the input text using a encoder-decoder model, and then aims to correct possible errors by using simple post-correction rules. The system is based on the Open Neural Machine Translation Toolkit (OpenNMT) (Klein et al., 2017) and handles the POS-tagging and lemmatization almost completely as a machine translation task. Relying purely on OpenNMT makes the tool easy to setup and allows more flexibility and easier customization. The whole pipeline is written in Python and it comes with an easy-to-use command-line interface and extensive documentation.

As our tool is purely based on Oracc notation, it aims to harmonize various digital resources and to encourage various projects to publish their data openly in Oracc.

6.1 Network Architecture

Our neural network architecture for both, the POS-tagger and the lemmatizer, follows the architecture of the Universal Lemmatizer (Kanerva et al., 2021). We use a deep attentional encoder-decoder network, where the encoder is a two layer BiLSTM that reads the sequence of logo-syllabically tokenized input. The decoder for generating the output character sequences is a two layer unidirectional LSTM with input feeding attention. However, we train the model for a lesser amount of steps relative to the training data size, as it improves the training speed but does not seem to affect the model's performance. We use a batch size of 64 and start the learning rate decay halfway through the training process.

6.2 Tokenization

We tokenize the input sequences in a special way that is particularly suitable for the logo-syllabic cuneiform writing system. From here on, we refer to this as *logo-syllabic* tokenization. In logo-syllabic tokenization, syllabic signs and phonetic complements that represent phonetic sequences are encoded as space-separated character sequences, whereas logograms and determinatives are encoded as indivisible tokens. We retain indices for logograms, because homophonic logograms can refer to different parts-of-speech and lemmata (e.g. in Akkadian DUG$_3$ = *ṭābu* "good" and DUG$_4$ = *qabû* "speak"), but for syllabic signs indexation is removed to bring homophonic readings such as *šu* and *šu*$_2$ closer to each other. Based on our observations, splitting compound logograms such as MA.NA into MA and NA yields better results than handling them as monolithic units.

The tagger is trained with 5-grams of logo-syllabically tokenized word forms and it aims to predict the POS-label for the center word wrapped inside double angle brackets (see Table 1 for tokenization examples). The lemmatizer is trained with tokenized word forms followed by its POS-tag, as well as the word's previous and following POS-tags to provide shallow information about the word's context. This input string sequence is mapped to its lemma on the character level, enabling the system to infer unseen lemmata.

6.3 Lemmatization and POS-tagging Process

The input text is first tokenized logo-syllabically and fed into the tagger. The tagger output is then used as the context information for the lemmatizer, which produces the fully annotated output consisting of the lemma and the POS-tag.

The post-correction comprises two steps. First, we calculate the distribution of lemmata assigned for each word form + POS-tag pair in the training data and in case any single lemma constitutes more

Original	{*m*}KU$_6$-*li-i-di ina* **ŠA$_3$-*bi*** x MA.NA
Source	`{m} KU`$_6$` - l i - i - d i	i n a « ŠA`$_3$` - b i » x	MA . NA`
Target	`N`
Source	`ŠA`$_3$` - b i P0=PRP P1=N P2=u`
Target	`l i b b u`
Combined	`libbu + N`

Table 1: Example of logo-syllabic tokenization. The upper part shows the tokenization fed into the tagger, the center word wrapped in double angle brackets, and the wanted output **N** (noun). The lower table shows the tokenization fed into the lemmatizer, including the center word and its POS-tag along with the preceding and the following POS-tags, and the wanted output as a sequence of characters.

than 70% of the lemmatizations of the given pair, we replace the predictions made by the neural network with this lemmatization. Next, we repeat the same step but instead of using word forms and their POS-tags, we also use the POS-tags assigned to the preceding and the following words. These steps aim to ensure, that close to unambiguous lemmata are always lemmatized consistently. However, due to the fact that the context information is taken into account already in the neural lemmatization, the post-correction no longer improves the lemmatization results significantly as in the previous version of BabyLemmatizer (Sahala et al., 2022). Therefore the post-correction is now mostly used for assigning lemmatizations with confidence scoring.

Confidence scoring aims to assist humans to manually verify and correct the lemmatization results. This system is mainly designed for detecting out-of-vocabulary (OOV) words, that is, word forms that were not present in the training data, and categorizing these words into different classes based on their spellings. The lowest confidence score of 0 is given to OOV words written logographically as the logogram and its lemma has a suppletive relation. The score of 1 is given for logo-syllabic spellings, which may have partially suppletive relationship to their lemmata. Syllabic OOV spellings are given a confidence score of 2.

Confidence scores between 3 and 5 are assigned for in-vocabulary words. The score of 3 is given to highly ambiguous words, such as polyvalent logograms that exist in contexts that have not been observed in the training data. The score of 4 is given to words that show low or unlikely ambiguity, and the highest score of 5 is given to words that have low ambiguity and exist in a POS-context that has been witnessed in the training data.

The lemmatization process is designed to be iterative. For example, if a batch of 10,000 new texts are to be lemmatized, this data set should be broken into smaller subsets, for example in four batches of 2,500 texts each.

After each lemmatization batch, the tool generates OOV lexicons for all low confidence score classes. These lists are sorted by frequency, allowing maximal number of corrections per each corrected entry. The lemmatizations can be corrected simply entering the corrected lemma and POS-tag for any word form in the OOV list, or accepting the already given lemmatization by removing the symbol # from the beginning of the line. When the lemmatizer is run again, the system appends these changes to the model's lexicon, allowing it to lemmatize them correctly in the future. After the lemmatization results of the current batch are considered to be clean enough, the model can be retrained by using the the data from the current batch appended to the model's existing training data, yielding an updated model augmented with new manual corrections. This approach should significantly reduce the time needed for manual corrections after each batch.

6.4 CoNLL-U+ for Cuneiform

Our tool uses an extended CoNLL-U format for input and output.[2] The first ten columns follow the standard notation, reserving the XPOS field for the Oracc POS-label. In addition to the conventional fields (ID, FORM, LEMMA, UPOS, XPOS, FEATS, HEAD, DEPREL, DEPS, MISC), our CoNLL-U+ format has the following fields: ENG for the English translation, NORM for phonological transcription, LANG for word's language, FORMCTX and XPOSCTX for storing temporary context information for the system, SCORE for the confidence scoring and LOCK for write-protecting the field in

[2]See `https://universaldependencies.org/format.html`

case the file has manual corrections that the system should not overwrite.

The pipeline handles the conversion of CoNLL-U+ into OpenNMT-compatible target and source files and conversion of simple rawtext transliteration into CoNNL-U+.

7 Evaluation

For evaluation, we train ten models for each data set. We use a 80/10/10 train/dev/test split and estimate the model's accuracy, that is, the percentage of correct analyses over all analyses, using 10-fold cross-validation. We measure the accuracy in two categories: first, for all the word forms in the test set, and second for the OOV word forms only to examine the models' ability to predict labels for words that were not present in the training data. The results are summarized in Table 2 and Table 3 respectively. Confidence intervals of the cross-validation are shown in parentheses.

We ignored all fully broken words in the evaluation, as assigning empty labels for completely destroyed words is trivial.

7.1 Results for Cuneiform Languages

With the current data sets, the minimum accuracy scores for POS-tagging and lemmatization of known word forms are 95% and 92%, respectively. For OOV words, the system achieves minimum accuracy of 81% for POS-tagging and 68% for lemmatization. OOV lemmatization accuracy seems to vary greatly depending on the language and the diversity of the data set, the most striking difference being between the first millennium Babylonian (68%) and the Sumerian literary texts (84%). This difference can be explained partly by the diversity of the Babylonian data, and partly by Sumerian morphology, which is significantly more transparent than that of Akkadian. The low performance in the OOV lemmatization in administrative Sumerian can be explained by inconsistencies in the data especially in proper nouns. At times, Oracc renders their lemmata as sequences of signs separated by dots, whereas at times the dots are not used, for instance, **{d}nin-mar{ki}** is lemmatized as *Ninmar*, *Nin.mar*, *Nin.MAR*, which makes it difficult for the model to learn how to generalize. About 18% of OOV lemmatization errors and 13% of all lemmatization errors in this particular data set are caused by such inconsistencies.

7.2 Comparison Against Version 1.0

Compared with the earlier version of BabyLemmatizer, the current tool clearly outperforms its neural network performance, which significantly reduces the improvement gained from post-correction. For comparison, we used the same 500,000 word Babylonian evaluation data set as we used in our earlier report (Sahala et al., 2022). Better performance of the neural network translates directly into a better performance in OOV word lemmatization, improving the prediction accuracy of Lemma+POS labels

Category	Urartian	Neo-Assyrian	Babylonian	Sum. (lit.)	Sum. (adm.)
NN POS-tagger	96.97 (±0.39)	97.67 (±0.17)	96.80 (±0.18)	94.79 (±0.28)	96.32 (±0.08)
NN Lemmatizer	93.45 (±0.66)	95.35 (±0.20)	95.02 (±0.31)	94.67 (±0.25)	95.50 (±0.10)
NN Combined	92.49 (±0.72)	94.48 (±0.26)	93.82 (±0.36)	92.28 (±0.37)	94.57 (±0.09)
PC POS-tagger	96.97 (±0.39)	97.72 (±0.17)	96.80 (±0.18)	94.77 (±0.28)	96.32 (±0.08)
PC Lemmatizer	94.12 (±0.57)	95.47 (±0.21)	95.14 (±0.30)	94.66 (±0.27)	95.53 (±0.08)
PC Combined	93.18 (±0.63)	94.59 (±0.28)	93.94 (±0.34)	92.27 (±0.37)	94.60 (±0.07)
OOV-rate	8.26	9.20	6.06	16.80	5.21

Table 2: Results of the 10-fold cross-validation for the neural net (NN) and the post-corrected (PC) results. Combined represents word forms where both, lemma and POS-tag were predicted correctly. OOV-rate shows the average percentage of OOV words in the test set.

Category	Urartian	Neo-Assyrian	Babylonian	Sum. (lit.)	Sum. (adm.)
POS-tagger	83.00 (±1.96)	90.87 (±0.74)	85.78 (±0.81)	84.39 (±0.84)	81.17 (±0.77)
Lemmatizer	70.10 (±2.16)	71.16 (±1.14)	67.82 (±1.36)	83.93 (±1.00)	70.51 (±1.24)
Combined	65.73 (±2.06)	70.15 (±1.09)	65.52 (±1.31)	76.49 (±1.15)	67.20 (±1.28)

Table 3: Results of the 10-fold cross-validation for OOV words only. This table does not contain separate results for NN and PC, because post-correction does not affect OOV words.

for OOV words on average by 10 percentage points. The performance increase is summarized in Table 4.

Category	All	OOV
NN POS-tagger	+0.14	+4.06
NN Lemmatizer	+8.87	+8.84
NN Combined	+8.91	+10.48
PC POS-tagger	+0.14	+4.06
PC Lemmatizer	+0.35	+8.37
PC Combined	+0.45	+10.01

Table 4: Average improvement in accuracy-% from v1.0 to v2.0, overall and for OOV words (NN for neural net and PC for post-corrected).

7.3 Experiment on Latin and Ancient Greek

To test our system on non-cuneiform languages, we tagged and lemmatized the PROIEL treebanks for Latin and Ancient Greek (Table 5). For these languages, we used character sequences as input format for both the tagger and the lemmatizer with the same context information as in the logo-syllabic tokenization (5-grams for tagger and adjacent POS-context for lemmatizer). We used the training, development and test data provided at the Universal Dependencies GitHub.

Category	Greek	Latin
POS-tagger	96.70	95.31
Lemmatizer	96.70	95.81
Combined	94.96	94.03
OOV POS-tagger	87.54	84.64
OOV Lemmatizer	74.47	75.92
OOV Combined	73.18	74.92
OOV-rate	11.02	10.58

Table 5: Results for the Ancient Greek and Latin data. The upper table shows the overall results and the lower table the results for OOV words only.

8 Conclusions and Future Work

We presented an updated version of BabyLemmatizer, a pipeline for POS-tagging and lemmatizing cuneiform languages and evaluated its performance on Sumerian, first millennium Babylonian, Neo-Assyrian and Urartian texts extracted from Oracc to observe its performance for the first time outside Babylonian texts. The system achieves a POS-tagging accuracy between 95-98% and a lemmatization accuracy of 94-96% depending on the language or dialect. For OOV words only, the current version can predict correct POS-tags for 83-91%, and lemmata for 68-84% of the input word forms from transliteration. Compared with the earlier version, the current one has about 10% higher accuracy in OOV lemmatization and POS-tagging due to better neural network performance. We also tested the system for lemmatizing and POS-tagging the PROIEL Ancient Greek and Latin treebanks, achieving results similar to those with the cuneiform languages.

In the future, we plan to add prediction for UD POS-tags, phonological transcription and morphological labels for Akkadian, Sumerian and Urartian. We also plan on adding full Oracc lemma prediction that includes the English translation of the word following Oracc's *lemma[translation]POS* format, but prior to this more data cleanup is required.

Acknowledgements

We wish to thank the Academy of Finland for funding the Origins of Emesal Project (PI Krister Lindén) and the Centre of Excellence in Ancient Near Eastern Empires (PI Saana Svärd). We also wish to thank Tero Alstola and Jonathan Valk (both at the University of Helsinki) for their valuable feedback on the Akkadian lemmatization results, Niek Veldhuis (UC Berkeley) for the Oracc data acquisition scripts,[3] and Steve Tinney (University of Pennsylvania) for correcting certain broken data sets in Oracc on our request.

References

David Bamman. 2012. *11-712 NLP Lab Report: Akkadian-morph-analyzer*.

François Barthélemy. 1998. A morphological Analyzer for Akkadian Verbal Forms with a Model of Phonetic Transformations. In *Computational Approaches to Semitic Languages*.

Pierre Briant and Wouter Henkelman. 2009. Achemenet.

Dominique Charpin. 2009. Archives Babyloniennes (Archibab).

Dominique Charpin. 2014. Ressources Assyriologiques sur Internet. In *Bibliotheca Orientalis LXXI no. 3-4*.

Christian Chiarcos, Ilya Khait, Émilie Pagé-Perron, Niko Schenk, Jayanth, Christian Fäth, Julius Steuer,

[3] `https://github.com/niekveldhuis/compass`

William Mcgrath, and Jinyan Wang. 2018. Annotating a low-resource language with LLOD technology: Sumerian morphology and syntax. *Information*, 9(11):290.

Birgit Christiansen, Mirjo Salvini, Dan Roberto, and Stephan Kroll. 2016. Oracc: Electronic Corpus of Urartian Texts (eCUT) Project.

Robert Englund, Jürgen Renn, Jacob L. Dahl, Bertrand Lafont, and Steve Tinney. 1998. Cuneiform Digital Library Initiative (CDLI).

Grant Frame, Jamie Novotny, and Karen Radner. 2015. Oracc: Royal Inscriptions of Babylonia Online.

Grant Frame, Karen Radner, Barry L. Eichler, Erle Leichty, and Steve Tinney. 2007. Oracc: The Royal Inscriptions of the Neo-Assyrian Period.

Dag T. T. Haug and Marius Jøhndal. 2008. Creating a parallel treebank of the old Indo-European Bible translations. In *Proceedings of the second workshop on language technology for cultural heritage data (LaTeCH 2008)*, pages 27–34.

John Huehnergard and Christopher Woods. 2008. Akkadian and eblaite. In R. D. Woodard, editor, *The Ancient Languages of Mesopotamia, Egypt, and Aksum*, pages 83–152. Cambridge University Press.

Bram Jagersma. 2010. *A Descriptive Grammar of Sumerian*. University of Leiden.

Heidi Jauhiainen, Aleksi Sahala, and Tero Alstola. 2019. Oracc in Korp.

Enrique Jiménez, Jussi Laasonen, Aino Hätinen, Zsombor Földi, Adrian Heinrich, Tonio Mitto, Geraldina Rozzi, Ilya Khait, and Fabioan Simonjetz. 2018. The Electronic Babylonian Library (eBL).

Jenna Kanerva, Filip Ginter, and Tapio Salakoski. 2021. Universal lemmatizer: A sequence-to-sequence model for lemmatizing universal dependencies treebanks. *Natural Language Engineering*, 27(5):545–574.

Laura Kataja and Kimmo Koskenniemi. 1988. Finite-state Description of Semitic Morphology: A Case Study of Ancient Accadian. In *Coling Budapest 1988 Volume 1: International Conference on Computational Linguistics*.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72, Vancouver, Canada. Association for Computational Linguistics.

Aaron Macks. 2002. Parsing akkadian verbs with prolog. In *Proceedings of the ACL-02 workshop on Computational approaches to semitic languages*.

Piotr Michalowski. 2008. Sumerian. In R. D. Woodard, editor, *The Ancient Languages of Mesopotamia, Egypt, and Aksum*, pages 19–46. Cambridge University Press.

Piotr Michalowski. 2011. *The Correspondence of the Kings of Ur: An Epistolary History of an Ancient Mesopotamian Kingdom*. Eisenbrauns.

Lucio Milano and Massimo Maiocchi. 2016. Ebla Digital Archives.

Manuel Molina. 2002. The Database of Neo-Sumerian Texts (BDTNS).

Jamie Novotny, Karen Radner, and Poppy Tushingham. 2017. Oracc: Archival Texts of the Assyrian Empire (ATAE).

Laurie Pearce, Stephanie Langin-Hooper, Chris Bravo, Talia Prussin, and Jay Crisostomo. 2011. Oracc: Hellenistic Babylonia: Texts, Images and Names.

Reinhard Pirngruber, Maya Rinderer, and Craig A. Harris. 2018. Oracc: Astronomical Diaries Digital.

Karen Radner, Eleanor Robson, Steve Tinney, and Jamie Novotny. 2005. Oracc: The State Archives of Assyria online.

Aleksi Sahala. 2014. *Babylonian verbimorfologian automaattinen jäsentäminen*. University of Helsinki.

Aleksi Sahala. 2021. *Contributions to Computational Assyriology (PhD Thesis)*. University of Helsinki.

Aleksi Sahala, Tero Alstola, Jonathan Valk, and Krister Linden. 2022. BabyLemmatizer: A Lemmatizer and POS-tagger for Akkadian. In *CLARIN Annual Conference Proceedings, 2022*. CLARIN ERIC.

Aleksi Sahala, Miikka Silfverberg, Antti Arppe, and Krister Lindén. 2020. BabyFST-towards a finite-state based computational model of ancient babylonian. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 3886–3894.

Thea Sommerschield, Yannis Assael, John Pavlopoulos, Vanessa Stefanak, Andrew Senior, Chris Dyer, John Bodel, Jonathan Prag, Ion Androutsopoulos, and Nando de Freitas. 2023. Machine Learning for Ancient Languages: A Survey. *Computational Linguistics*, pages 1–44.

Michael Streck. 2010. Großes fach altorientalistik: Der umfang des keilschriftlichen textkorpus. In *Mitteilungen der Deutschen Orient-Gesellschaft zu Berlin, 142*.

Valentin Tablan, Wim Peters, Diana Maynard, Hamish Cunningham, and K Bontcheva. 2006. Creating Tools for Morphological Analysis of Sumerian. In *LREC*, pages 1762–1765.

Steve Tinney. 2019. *L2: How it Works*.

Steve Tinney, Phillip Jones, and Niek Veldhuis. 2017. Oracc: The electronic Pennsylvania Sumerian Dictionary 2.7.

Steve Tinney, Jamie Novotny, Eleanor Robson, and Niek Veldhuis. 2006. The Open Richly Annotated Cuneiform Corpus.

Steve Tinney and Eleanor Robson. 2019. Oracc ATF Primer. *Oracc: The Open Richly Annotated Cuneiform Corpus*.

Gernot Wilhelm. 2008. Urartian. In R. D. Woodard, editor, *The Ancient Languages of Asia Minor*, pages 105–123. Cambridge University Press.