Akkadian Word Segmentation

Timo Homburg M.Sc., Dr. Christian Chiarcos
Institute for Computer Science
Goethe University, Robert-Mayer-Str. 10, 60325 Frankfurt am Main, Germany
timo.homburg@gmx.de, chiarcos@em.uni-frankfurt.de

Abstract

We present experiments on word segmentation for Akkadian cuneiform, an ancient writing system and a language used for about 3 millennia in the ancient Near East. To our best knowledge, this is the first study of this kind applied to either the Akkadian language or the cuneiform writing system. As a logosyllabic writing system, cuneiform structurally resembles Eastern Asian writing systems, so, we employ word segmentation algorithms originally developed for Chinese and Japanese. We describe results of rule-based algorithms, dictionary-based algorithms, statistical and machine learning approaches. Our results may indicate possible promising steps in cuneiform word segmentation that can create and improve natural language processing in this area.

Keywords: Assyriology, Cuneiform, Akkadian, Chinese, Word Segmentation, Machine Learning

1. Introduction

Word segmentation is the most elementary task in natural language processing of written language. In most alphabetical writing systems, this task is commonly referred to as tokenization and can be easily solved through the interpretation of orthographical markers for word and sentence boundaries, e.g., white spaces. Where these are lacking, however, word segmentation is a challenging task, a classical – and successfully addressed – problem in logographic writing systems like Chinese and logosyllabic writing systems like Japanese.

Here, we describe experiments on cuneiform, a writing system developed in the 4th m. BCE in Mesopotamia subsequently applied to various Semitic, Indo-European and isolate languages in the region. As a logosyllabic writing system, it shares important structural characteristics with Chinese and Japanese (Ikeda, 2007), so that we evaluate word segmentation methods successfully applied to these languages. However, these languages are unrelated to those of the Ancient Near East, so that future research will focus on developing aspects specific to languages with cuneiform writing.

As a writing system, cuneiform poses a number of unique challenges:

- The same character, e.g., 𒁕, can be read as a logograph or as a syllable, as the logograph *GURU* 'young man' or with its phonological reading as a syllabic sign.
- As a syllabic sign, a single character can have multiple different readings, e.g., grounded in the possible Sumerian pronounciation(s) of the logograph, or the pronounciation of their Akkadian translations, 𒁕 may be read as *dan/tan* (from Akk. *dannu* 'strong, powerful'), *kal* (from Sum. *kal* 'rare, valuable' and *kalag* 'strong'), *rib* (from Sum. *rib* 'outstanding, strong'), etc. (Tinney and others, 2006; Lauffenburger, nd; Borger, 2004).
- CVC syllables (e.g., *dan*) can be as a pair of CV-VC characters (𒁕𒀸 *da-an*) or with a single CVC character (𒁕 *dan*) (Gelb, 1957, p.8, *Da-an--ri* vs. *Dan-r-ri*).

We primarily consider the Akkadian language, the dominant language of the Ancient Near East from the 3rd to the 1st millennium BCE. Originally spoken in Mesopotamia, it became the lingua franca in the Near East during the 2nd m. BCE, with an extensive body of material comparable only to corpus languages such as Classical Latin or Ancient Greek. With a considerable amount of cuneiform clay tablets not yet deciphered, and new ones being continuously excavated, the automated processing of the Akkadian language is thus of tremendous importance. Previous research on automated digitization focused on producing 3D scans of tablets (Sect. 2.), with Optical Character Recognition (OCR) being a logical next step in the development. Successful cuneiform OCR, however, needs to be accomplished by knowledge-rich NLP methods for the contextual disambiguation of characters: One of the key characteristics of cuneiform is that a character can be read as an logograph, as a determinative, or as a syllabic sign (with different phonemic values). The contextual distribution of characters is thus heavily dependent on its context. Word segmentation approaches may thus be a key component to any approach on cuneiform OCR.

Akkadian is the oldest attested Semitic language, and has thus occasionally been considered in experiments on NLP for Semitic languages, but mostly focusing on (rule-based) morphological analysis. To our best knowledge, the present paper describes the first study of word segmentation in Akkadian cuneiform. It thus provides a primary point of orientation for any subsequent experiments on cuneiform word segmentation and will be of utmost importance to future experiments on cuneiform OCR and Akkadian NLP.

2. State Of the Art

We distinguish three types of word segmentation algorithms:

rule-based segmentation rules derived from grammar

dictionary-based segmentation by lookup in a (statically enhanced) dictionary

statistical/machine learning data-driven segmentation as learnt from segmented corpora

As shown in several SIGHAN BakeOffs in the last decade (Sproat and Emerson, 2003), in Chinese machine learning and dictionary-based approaches like MaxMatch (Chen and Liu, 1992) produce reasonable results while rule-based methods are commonly used as a Baseline (Palmer and Burger, 1997). In Japanese, however, rule-based algorithms like Tango (Ando and Lee, 2000) proved to be more successful. This is partially due to the morphological richness of Japanese as compared to Chinese.

As a point of orientation for subsequent studies on cuneiform, we evaluate selected approaches from these classes in their performance on Akkadian. Neither the Akkadian language nor cuneiform as a writing system have been addressed in this respect before.

Along with other cuneiform languages, Akkadian has a considerable research history in NLP. For the greatest part, existing approaches are concerned with rule-based morphological analyzers, e.g., Kataja and Koskenniemi (1988), Barthlemy (1998), Macks (2002), Barthlemy (2009), Khait (accepted) for Akkadian, or Valentin Tablan Wim Peters (2006) for Sumerian. As for data-driven morphological tools, the state of the art in the field is represented by the Lemmatizer of the Open Richly Annotated Cuneiform Corpus (ORACC),[1] which supports manual morphological annotation for Akkadian, Sumerian and (to a limited degree) Hittite with a lookup-functionality in the annotated corpus. Such example-based approaches can be extended to automatically transfer morphological rules through phonological equivalences, as demonstrated by Snyder et al. (2010) for the projection of Hebrew morphology and lexicon to Ugaritic, another Semitic cuneiform language. As for higher levels of linguistic analysis, we are not aware of any tools for syntactic or semantic annotation for Akkadian, however, the latter has been considered for administrative texts from the Sumerian period, whose highly conventionalized structure can be exploited for concept classification (Jaworski, 2008).

Aside from linguistic analysis, another aspect of cuneiform languages that recently aroused interest are approaches focusing on the material side of cuneiform writing, i.e., scanning and digitizing clay tablets (Subodh et al., 2003; Cohen et al., 2004), reconstructing tablets and tales by automatically combining their fragments (Collins et al., accepted; Tyndall, 2012), and recently, initial steps towards cuneiform OCR have been undertaken (Mara et al., 2010). As this line of research is flourishing mostly in the field of computer graphics, the obvious gap between both lines of research lies in the absence of any studies concerned with the transition from the (identified) sign and its linguistic interpretation, a challenging task, as mentioned before.

With our paper, we describe the first experiments in this direction, with a specific focus on segmenting character sequences into words as a core component for future approaches on transliteration.

[1] http://oracc.museum.upenn.edu/doc/builder/linganno/

3. Experimental Setup

3.1. Corpus Data

We use corpora from three different periods and dialects, namely Old Babylonian, Middle Babylonian and Neo-Assyrian, from the Cuneiform Digital Library Initiative (CDLI)[2], representing most of the available texts (clay tablets) of the given periods of time. The corpora were randomly split in a 80:20 ratio for training and testing purposes (on a per-tablet, not a per-line basis). For the experiments, we trained our segmentation algorithms on each of these language stages, and performed evaluations on each language stage respectively. For reasons of space, we only report results for the Middle Babylonian training corpus and evaluation against the Middle Babylonian test corpus in detail. Further experiments showing robust performance across different language stages will be represented in a graphical way. Additionally we will present results of classifications using corpora data of one epoch applied on other epochs of the same language to get an impression of the performance of the algorithms on related data.

The CDLI ATF format contains metadata, a (word-segmented) transliteration, and (optionally) a translation. CDLI data always represents cuneiform in lines as found on the clay tablets. To minimize ambiguity, Akkadian writers tried to avoid incomplete words at the end of a line, so the tablets themselves provide initial data on word segmentation. From the transliteration, we restored the original UTF-8 characters on the basis of a sign list that we compiled from various resources. Non-restorable characters were ignored and thus are not represented in the resulting texts. This data represents our gold standard. It should be noted that the mapping to UTF-8 can not be trivially reversed because of the highly ambiguous phonological and ideographic meaning of characters.

After conversion and the removal of whitespaces, segmentation algorithms of three categories have been applied. Figure 1 shows the segmentation process.

3.2. Baseline

As our baseline we adopted the *Character-As-Word* algorithm (Palmer and Burger, 1997, p.1) common as a Baseline in Chinese, in the form of an *Average-Word-Length* (AVG) algorithm. The average word length L was determined empirically by analyzing the training corpora as follows:

$$L = \left\lfloor \frac{\sum_{i=1}^{\#(text)} len(w_i)}{\#(text)} \right\rfloor$$

with $text$ being the entire corpus, $len(w_i)$ being the length of a word and $\#(text)$ representing the number of words

[2] http://cdli.ucla.edu/

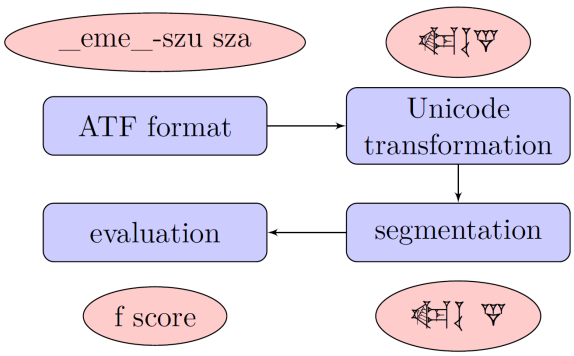


Figure 1: Classification process

in the entire corpus. Accordingly, a sequence of characters will be split after every Lth character.

3.3. Rule-based Algorithms

A first simple *bigram* method inserts a word break between two characters c_1 and c_2, if a word break between c_1 and c_2 is more frequent than a non-break in the training corpus.
Similarly, a *unigram* segmentation between two characters can be achieved by classifying every individual character according to the I(0)BES scheme – (preferred) intermediate (I), beginning (B), end (E) and single (S) characters – in the training corpus. The *prefix/suffix* algorithm collects usages for every characters in every word found in an already segmented training corpus. For every character found in a word classification in the I(0)BES scheme will be collected and frequencies will be counted for I,B,E and S respectively. For each pair of characters the algorithm will test if the probability of being E or S exceeds the probability of being I or B, therefore indicating a separation. Therefore the *prefix/suffix* algorithm includes information on a word basis to achieve the segmentation.
Furthermore, we apply the *Tango* algorithm (Ando and Lee, 2000) which uses a scoring system to determine possible segmentations in a sliding window mechanism. The threshold parameter and the window size have been empirically chosen to fit the needs of the Akkadian language.
Finally, we compare these methods with a random segmentation function which decides for every character whether a segmentation should occur after it. The randomized function is initialized with a random seed of the size of the corresponding charcount of each line respectively. Though substantially worse than the average-length baseline, it outperforms the bigram method.

3.4. Dictionary-based Algorithms

As dictionary-based approaches can despite their simplicity gain considerable segmentation efforts in languages like Chinese or Japanese, we apply the commonly used *Minimum WordCount Matching* Algorithm, a modified version of the *LCU Matching* algorithm (Pengyu et al., 2014), and the *MaxMatch* algorithm (Chen and Liu, 1992), and a modified version of the *MaxMatch* Algorithm (Islam et al., 2007).[3] Dictionaries used in those approaches have been generated from the provided training data used as a basis for the other approaches as well. Externa dictionary resources have not been considered. It is important to notice that we did not apply any new word detection which can be used to extend the dictionary using data from the test corpus. This may be a topic for further refinement and could be achieved by applying one of many statistical approaches.

3.5. Statistics & ML

The commonly used Maximum Probability Matching algorithm trivially maximizes the occurrence probability of a word sequence (here, a line on a tablet) by matching words against a frequency-tagged dictionary and returning the most probable word segmentation $\vec{s} \in \{(i|1 < i < n) \in \mathcal{N}^x | x < n\}$ for a character sequence (i.e., line) $c_1, .., c_n$:

$$Seg_{MaxProb}(c_1, ..., c_n) = \arg\max_{\vec{s}} \prod_{j=1}^{|\vec{s}|-1} P_{dict}(c_j, ..., c_{j+1})$$

We normalize the dictionary probability P_{dict} to values greater than 0 to account for out-of-vocabulary words.
More advanced approaches we studied here include clustering algorithms (*kNN*, *kMeans*), decision trees (*C4.5*), *NaiveBayes* and *MaxEnt*, sequence labelling models (Hidden Markov Models, *HMM* and Conditional Random Fields, *CRF*), as well as Support Vector Machines (*SVM*) and Multi-Layer Perceptrons (*MLP*). For most algorithms, we relied on the implementation provided by WEKA 3.7 [4], for HMMs the HMMWeka extension[5], for CRFs the MALLET[6] CRF SimpleTagger, for SVMs libsvm[7] with polynomial kernel.
Table 1 enumerates the feature sets we used in the experiments. For the purpose of our experiments, these were adopted without modification from the literature on Far Eastern languages and writing systems (see references for details) and *directly* applied to Akkadian cuneiform. The motivation is to establish a sound basis for the future development of cuneiform-specific algorithms, for which we expect substantial refinements if language-specific features for Akkadian are explicitly taken into account in statistical approaches.
For all data-driven classifiers, we simplified the target classification of the I(O)BES scheme to a binary distinction of Class 0 (i.e., IB: no segmentation after the current character) and Class 1 (i.e., ES: segmentation after the current character). All classifiers were trained on data sets of 10.000 instances in order to assess their performance under resource-poor circumstances. In addition, training against the full training set of 100.000 instances was performed successfully for most algorithms (and is reported below), with the exception of HMM and CRF whose training timed out.

[3] We employ the basic version of (Chen and Liu, 1992). No additional neighbor checking performed.

[4] http://www.cs.waikato.ac.nz/ml/weka/

[5] http://doc.gold.ac.uk/~mas02mg/software/hmmweka/

[6] http://mallet.cs.umass.edu/

[7] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

feature set	algorithms
BASE base feature set (Low et al., 2005)	C4.5, MaxEntropy, NaiveBayes, kNN, kMeans, MLP, SVM
EXT extended feature set (Low et al., 2005)	C4.5, MaxEntropy, NaiveBayes, kNN, kMeans, MLP, SVM
MAXENT MaxEnt feature set (Raman, 2006)	C4.5, MaxEntropy, NaiveBayes, kNN, kMeans, MLP, SVM
PERC perceptron feature set (Song and Sarkar, 2008)	C4.5, MaxEntropy, NaiveBayes, kNN, kMeans, MLP, SVM
RED-BIG reduced bigram feature set (Papageorgiou, 1994)	CRF, HMM

Table 1: Feature sets

3.6. Ensemble Combination

Finally, we implemented simple ensemble combination architectures as a *meta* classifier, using

(a) a simple (unweighted) majority vote, tested for all possible combinations of individual segmentation algorithms described above,

(b) C4.5 meta classification, resp.

(c) SVM meta classification

Remarkably, the more elaborate C4.5 and SVM versions of the meta classifier did not outperform the best majority configuration (i.e., all classifiers except SVM), whose results are reported in Tab. 2.

3.7. Evaluation Metrics

We primarily evaluate against the following conventional metrics:

Boundary evaluation addresses character-based segmentation per boundary (Palmer and Burger, 1997, p.176), i.e., precision and recall of predicted and observed boundaries following a given character:

$$\text{rec}_b = \frac{\#\text{correctly predicted boundary}}{\#\text{gold boundaries}}$$

$$\text{prec}_b = \frac{\#\text{correctly predicted boundary}}{\#\text{predicted boundaries}}$$

Word boundary evaluation evaluates completely segmented words (Palmer and Burger, 1997, p.176), a metric especially relevant for any future practical application by Assyriologists or philologists:

$$\text{rec}_w = \frac{\#\text{correctly predicted words}}{\#\text{gold words}}$$

$$\text{prec}_w = \frac{\#\text{correctly predicted words}}{\#\text{predicted words}}$$

As these metrics are not differenciating between near and far misses of boundaries at all, we also employ **sliding-window-based metrics**:

WindowDiff aims to avoid penalizing near-matching boundaries too restrictively, with window size $k = \frac{N}{2*\#\text{segments}}$, reference segmentation R, a total number of N content units and C computed boundaries the correctness of a segmentation as follows:

$$WDiff = \frac{1}{N-k}\sum_{i=0}^{N-k}(|R_{i,i+k} - C_{i,i+k}| > 0)$$

WinPR is an established metric in the word segmentation community, it calculates precision and recall building on the basis of WindowDiff by defining $prec_{wp} = \frac{tp_{wp}}{tp_{wp}+fp_{wp}}$ and $rec_{wp} = \frac{tp_{wp}}{tp_{wp}+fn_{wp}}$ with the following definitions (Scaiano and Inkpen, 2012):

$$tp_{wp} = \sum_{i=1-k}^{k}\min(R_{i,i+k}, C_{i,i+k})$$

$$tn_{wp} = -k(k-1) + \sum_{i=1-k}^{N}(k - \max(R_{i,i+k}, C_{i,i+k}))$$

$$fp_{wp} = \sum_{i=1-k}^{N}\max(0, C_{i,i+k} - R_{i,i+k})$$

$$fn_{wp} = \sum_{i=1-k}^{N} max(0, R_{i,i+k} - C_{i,i+k})$$

PK metric is a standard metric in the field of text segmentation (Pevzner and Hearst, 2002).

4. Results

Table 2 and Fig. 3 provide results which are representative for the experiments conducted. For reasons of space, only the best-performing combinations of features and data-driven (statistical/ML) methods are reported. Also, while all combinations of cuneiform corpora have been tested, we report only results obtained by training the tools on the (training section of the) Middle Babylonian corpus and tested on the (test section of the) Middle Babylonian corpus. The general pattern, however, remains the same for all cuneiform corpora considered, both within a language stage (training and test corpus for the same language stage, Fig. 2), but also across language stages (e.g., Old Babylonian tools tested on Middle Babylonian corpus).

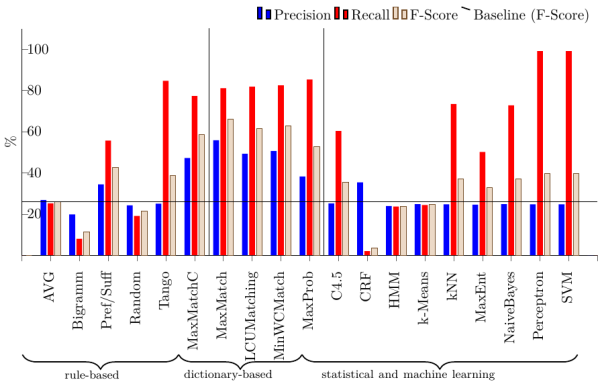
For the example of the Middle Babylonian tools, the scores of the best-performing configurations obtained on the Old Babylonian and Neo-Assyrian test set are reported in Tab. 3. Unsurprisingly, the scores are generally worse than for Middle Babylonian, yet, they still outperform the baseline. Remarkably, Neo-Assyrian boundary F-scores actually seem to improve over Middle Babylonian. This may be due to the fact that the Neo-Assyrian corpus is more homogenuous than the Middle Babylonian corpus, as the latter contains much material written by non-native speakers.

Method	Bound F-Score	Word F-Score	PK Score	WinPR F-Score
baseline (AVG length [2])	42.77	21.19	13.77	42.92
Bigram	14.90	7.22	47.72	20.84
Pref/Suff	34.59	10.17	23.08	34.86
Random	23.43	13.64	45.04	22.31
Tango	49.22	14.88	13.93	35.32
MaxMatch	65.02	**65.05**	9.03	57.47
MaxMatchCombined	**73.91**	58.48	8.69	**60.80**
LCUMatching	68.14	44.78	8.92	51.30
MinWCMatch	72.82	59.76	**8.17**	59.63
MaxProb	59.67	37.26	10.24	49.58
C4.5 (EXT 10K)	42.66	15.33	23.12	34.01
CRF (EXT 10K)	40.13	15.31	13.06	27.10
HMM (RED-BIG 10K)	23.10	12.98	21.64	35.55
kMeans (BASE 10K)	37.79	14.17	24.11	33.77
kNN (EXT 100K)	49.48	14.51	13.59	37.53
MaxEnt (EXT 10K)	46.91	14.97	15.82	36.59
NBayes (EXT 100K)	49.49	14.52	13.60	37.56
Percep (MAXENT 10K)	49.51	14.51	13.59	37.51
SVM (EXT 10K)	49.51	14.51	13.59	37.51
META (w/o SVM, majority vote)	49.42	14.31	13.59	37.47

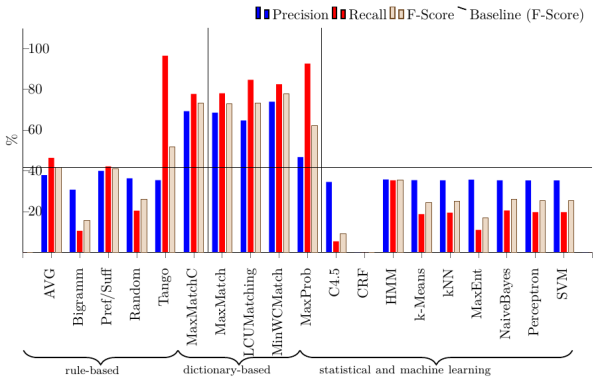
Table 2: Middle Babylonian tools on Middle Babylonian test set

As for rule-based methods, only Tango (Ando and Lee, 2000) outperformed the baseline, whereas dictionary-based algorithms performed clearly better. Dictionary-based approaches produced the best-performing classification, depending on the metric between 60% and nearly 80%.

Machine Learning algorithms as applied here seem to face problems in Akkadian cuneiform: The feature sets used for segmenting Chinese seem to provide a significantly worse result in Akkadian, and even the best-performing algorithms hardly outperform the baseline. This indicates that a feature set specific to Akkadian needs to be developed in future research.



(2.a)



(2.b)

Figure 2: Within-language test error for (a) Old Babylonian tools on Old Babylonian test set, and (b) Neo-Assyrian tools on Neo-Assyrian test set (boundary evaluation)

It should be noted that this picture did not drastically change when different training and test corpora for Akkadian language stages were employed.

A note on transliteration and morphosyntax As mentioned before, retransforming cuneiform Unicode characters into the correct transliteration is far from easy, as every Unicode character may represent an ideograph, several different syllables, etc. In the process of word segmentation, we also conducted initial experiments on transliteration. By mapping every character to its according to the corpora data most frequent transliteration, we were able to establish a baseline that correctly transliterates up to 40% of the characters per corpus. In future research, this needs to be further improved by statistical, context-aware classifiers and tighter integration of the word segmentation and transliteration tasks. At the same time, word segmentation for Akkadian can certainly benefit from integrating higher levels of NLP, e.g., POS tagging, as this may be important for lexical disambiguation. Taken together, this calls for a uniform architecture capable to handle word segmentation, transliteration and morphosyntactic analysis in a single task. For such an integrated system, our experients with segmentation module may serve as a baseline.

Method	Bound F-Score	Word F-Score	PK Score	WinPR F-Score	language (alg.)
baseline (AVG length [2])	39.25	17.45	21.37	39.81	OBab
	41.63	17.06	11.12	36.13	NAss
rule-based	33.80	9.11	28.23	31.53	OBab (Prefix/Suffix)
	51.93	16.65	11.14	36.44	NAss (Tango)
dictionary-based	**62.73**	**53.12**	**2.43**	**56.10**	OBab (MinWCMatch)
	52.95	**24.51**	**10.97**	**38.46**	NAss (MinWCMatch)
statistical/ML	39.56	9.56	21.26	31.91	OBab (NaiveBayes, EXT 10k)
	52.31	16.64	10.95	36.47	NAss (NaiveBayes, PERC 10K)

Table 3: Across-language test error: Middle Babylonian tools on O(ld)Bab(ylonian) and N(eo-)Ass(yrian) test sets

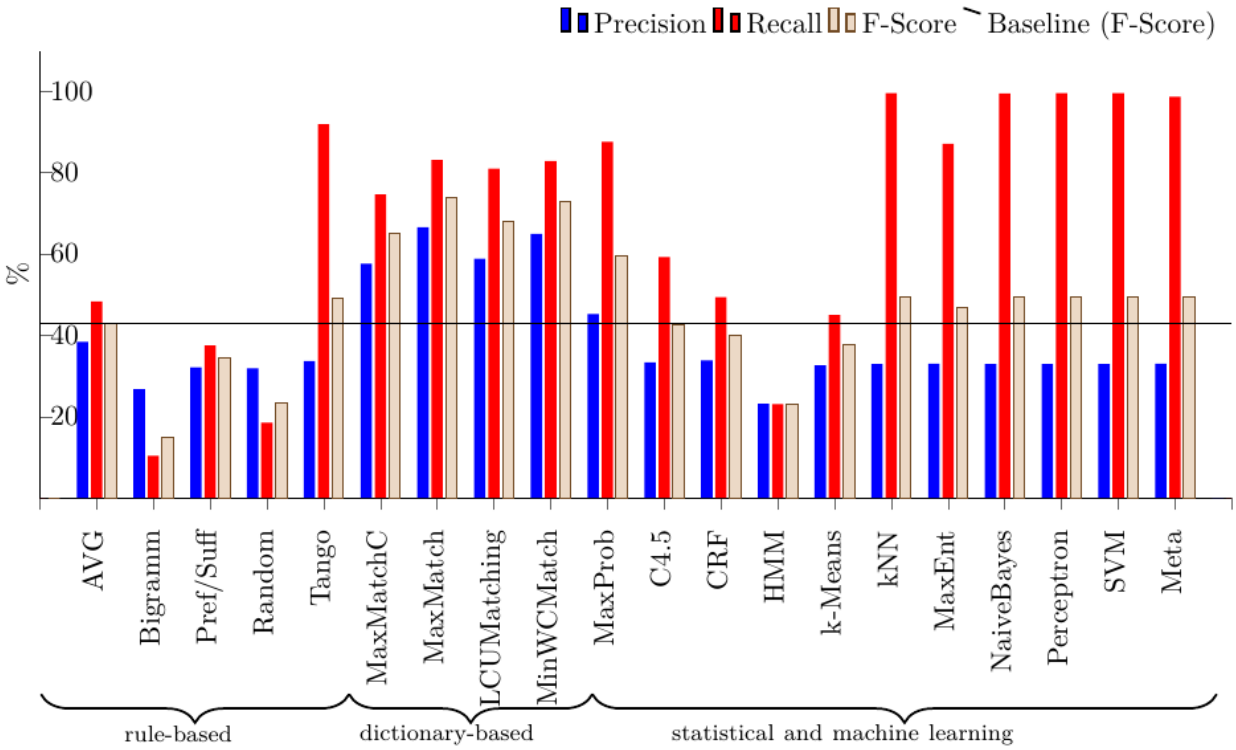


Figure 3: Middle Babylonian tools (80%) on Middle Babylonian test set(20%) (boundary evaluation)

5. Summary and Outlook

Our paper is the first experiment on word segmentation on Akkadian or cuneiform. It provides insights in what to expect by applying established word segmentation algorithms on the Akkadian language, and we showed that for Akkadian corpora of the dimensions available, dictionary-based approaches produce the best results in segmenting Akkadian texts.

In general, our results for Akkadian are substantially worse than those for Chinese and Japanese, and currently do not live up to the needs of philologists. However, given the high degree of ambiguity in the writing system, this result is unsurprising, and calls for intensified research efforts in this regard. Our experiments stipulate directions for future research and provide a point of orientation for any future approach in this direction.

A key result is that further research on linguistic characteristics of Akkadian and other cuneiform languages is required, and – likely – only this will improve results to production-ready quality.

Strategies to improve segmentation performance include

(1) extending dictionary-based algorithms with a new word detection component – yet, this is methodologically problematic if the test corpus is used to improve segmentation –,

(2) combining dictionary-based and rule-based approaches and extending the latter with a morphological component, and

(3) combining morphology-oriented rule-based systems with statistical and machine learning algorithms to benefit from both the context-awareness of data-driven methods and the high precision of rule-based morphological analysis.

The most promising (and the most challenging) extension

in this regard is the development of an integrated system that provides *uniform handling* for word segmentation, transliteration and morphosyntactic annotation. Akkadian is a morphologically complex language with a highly ambiguous writing system. Unlike Chinese, it shows heavy interference between morphology and word segmentation, and unlike Japanese, it does not have a 1:1 correspondence between syllabic signs and phonemic values. In this regard, any future word segmentation algorithm for cuneiform will have to be an *integrated approach*, and thus be very different from existing approaches for either Chinese or Japanese.

Remark to reviewers

Upon acceptance of this paper, code and data will be published under open licenses (Apache license and CC-BY). In addition to the algorithms described here, this includes an interactive GUI for visualizing and analyzing segmentation results, a cuneiform Input Method Engine, and a near-exhaustive list of UTF8 cuneiform characters and their readings. Providing a link would reveal author identity.

6. Bibliographical References

Ando, R. K. and Lee, L. (2000). Mostly-unsupervised statistical segmentation of japanese: Applications to kanji. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*, NAACL 2000, pages 241–248, Stroudsburg, PA, USA. Association for Computational Linguistics.

Barthlemy, F. (1998). A morphological analyzer for Akkadian verbal forms with a model of phonetic transformations. In *Proceedings of the ACL-1998 Workshop on Computational Approaches to Semitic Languages*, Montral.

Barthlemy, F. (2009). The Karamel System and Semitic languages: Structured multi-tiered morphology. In *Proceedings of the EACL 2009 Workshop on Computational Approaches to Semitic Languages*, page 1018, Athens, Greece.

Borger, R. (2004). *Mesopotamisches Zeichenlexikon*. Ugarit-Verlag.

Chen, K.-J. and Liu, S.-H. (1992). Word identification for mandarin chinese sentences. In *Proceedings of the 14th Conference on Computational Linguistics - Volume 1*, COLING '92, pages 101–107, Stroudsburg, PA, USA. Association for Computational Linguistics.

Cohen, J. D., Duncan, D., Snyder, D., Cooper, J., Kumar, S., Hahn, D., Chen, Y., Purnomo, B., and Graettinger, J. (2004). iClay: Digitizing Cuneiform. In *Proceedings of the 5th International Conference on Virtual Reality, Archaeology and Intelligent Cultural Heritage (VAST-2004)*, pages 135–143, Aire-la-Ville, Switzerland, Switzerland. Eurographics Association.

Collins, T., Woolley, S., Gehlken, E., Lewis, A., Munoz, L. H., and Ch'ng, E. (accepted). Automated reconstruction of virtual fragmented cuneiform tablets. *Electronics Letters (IET)*.

Gelb, I. (1957). *Glossary of Old Akkadian*. University of Chicago Press, Chicago, Illinois.

Ikeda, J. (2007). Early Japanese and early Akkadian writing systems. a contrastive survey of Kunogenesis. In *Proceedings of Origins of Early Writing Systems*, Peking University, Beijing.

Islam, M. A., Inkpen, D., and Kiringa, I. (2007). A generalized approach to word segmentation using maximum length descending frequency and entropy rate. In *Computational Linguistics and Intelligent Text Processing*, pages 175–185. Springer.

Jaworski, W. (2008). Contents modelling of neo-sumerian ur III economic text corpus. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 369–376, Manchester, UK, August. Coling 2008 Organizing Committee.

Kataja, L. and Koskenniemi, K. (1988). Finite-state description of Semitic morphology: A case study of Ancient Akkadian. In *Proceedings of COLING 1988*.

Khait, I. (accepted). Cuneiform Labs: Annotating Akkadian corpora. In *Rencontre Assyriologique Internationale (RAI-2015)*, Geneva and Bern, Switzerland, June 22-26, 2015.

Lauffenburger, O. (n.d.). Akkadian dictionary. `www.assyrianlanguages.org/akkadian`.

Low, J. K., Ng, H. T., and Guo, W. (2005). A maximum entropy approach to chinese word segmentation. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, volume 1612164.

Macks, A. (2002). Parsing Akkadian Verbs with Prolog. In *Proceedings of the ACL-02 Workshop on Computational Approaches to Semitic Languages*, Philadelphia, Pennsylvania.

Mara, H., Krmker, S., Jakob, S., and Breuckmann, B. (2010). GigaMesh and Gilgamesh 3D Multiscale Integral Invariant Cuneiform Character Extraction. In Alessandro Artusi, et al., editors, *VAST: International Symposium on Virtual Reality, Archaeology and Intelligent Cultural Heritage*. The Eurographics Association.

Palmer, D. and Burger, J. (1997). Chinese word segmentation and information retrieval. In *AAAI Spring Symposium on Cross-Language Text and Speech Retrieval*, pages 175–178.

Papageorgiou, C. P. (1994). Japanese word segmentation by hidden markov model. In *Proceedings of the workshop on Human Language Technology*, pages 283–288. Association for Computational Linguistics.

Pengyu, L., Jingchuan, P., Du Mingming, L. X., and Lijun, J. (2014). A lexicon-corpus-based unsupervised chinese word segmentation approach. *International Journal On Smart Sensing And Intelligent Systems*, 7(1).

Pevzner, L. and Hearst, M. A. (2002). A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics*, 28(1):19–36.

Raman, A. (2006). A dictionary-augmented maximum entropy tagging approach to chinese word segmentation.

Scaiano, M. and Inkpen, D. (2012). Getting more from segmentation evaluation. In *Proceedings of the 2012 Conference of the North American Chapter of the Associ-*

ation for Computational Linguistics: Human Language Technologies, pages 362–366. Association for Computational Linguistics.

Snyder, B., Barzilay, R., and Knight, K. (2010). A statistical model for lost language decipherment. In *Proceedings of ACL-2010*, Upsala, Sweden.

Song, D. and Sarkar, A. (2008). Training a perceptron with global and local features for chinese word segmentation. In *IJCNLP*, pages 143–146. Citeseer.

Sproat, R. and Emerson, T. (2003). The first international chinese word segmentation bakeoff. In *Proceedings of the second SIGHAN workshop on Chinese language processing-Volume 17*, pages 133–143. Association for Computational Linguistics.

Subodh, K., Snyder, D., Duncan, D., Cohen, J., and Cooper, J. (2003). Digital preservation of ancient cuneiform tablets using 3D-scanning. In *Proceedings of the 4th International Conference on 3-D Digital Imaging and Modeling (3DIM-2003)*, pages 326–333. IEEE.

Tinney, S. et al. (2006). The Pennsylvania Sumerian dictionary. `http://psd.museum.upenn.edu/epsd1/`.

Tyndall, S. (2012). Toward automatically assembling Hittite-language cuneiform tablet fragments into larger texts. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 243–247. Association for Computational Linguistics.

Valentin Tablan Wim Peters, Diana Maynard, H. C. (2006). Creating tools for morphological analysis of Sumerian. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-2006)*, pages 1762–1765, Genova, Italy.