UNIVERSITY OF HELSINKI

https://helda.helsinki.fi

Transliteration Model for Egyptian Words

Jauhiainen, Heidi

2023-10-10

Jauhiainen , H & Jauhiainen , T 2023 , ' Transliteration Model for Egyptian Words ' , Digital Humanities in the Nordic and Baltic Countries Publications , vol. 5 , no. 1 , pp. 149-164 . https://doi.org/10.5617/dhnb

http://hdl.handle.net/10138/566540
https://doi.org/10.5617/dhnbpub.10659

cc_by
publishedVersion

Downloaded from Helda, University of Helsinki institutional repository.

This is an electronic reprint of the original article.

This reprint may differ from the original in pagination and typographic detail.

Please cite the original version.

Transliteration Model for Egyptian Words

Heidi Jauhiainen[1], Tommi Jauhiainen[1]

[1]*Department of Digital Humanities, University of Helsinki, Helsinki, Finland*

Abstract

In this paper, we describe token-based transliteration models for Egyptian words. We explain how we created them using an automatic alignment method we devised based on the Needleman-Wunsch sequence alignment algorithm. We use two sources where encoded Egyptian hieroglyphs and their transliteration pairs are available. Ancient Egyptian Sentences (AES) includes a collection of texts where c. 254,000 Egyptian words encoded using Manual de Codage (MdC) have been aligned with their transliteration counterparts. The second source is the Ramses Transliteration Corpus (RTC), with almost 500,000 MdC encoded words. The RTC consists of encoded hieroglyphic sentences, each on its line, and respective transliteration lines in another file. However, unlike the AES, there is no ready alignment of the MdC and its transliteration on the word level. In order to find word-transliteration pairs, we align the sentences of encoded words with the respective transliterations. The alignment task is made more difficult because many of the texts contain damaged parts and editorial additions.

Keywords

language modeling, word alignment, transliteration, hieroglyphs, Ancient Egyptian

1. Introduction

In this paper, we describe our current work on creating token-based transliteration models for Egyptian words, which are needed for an automatic transliteration method we aim at developing. In order to create such models for transliteration, a corpus of machine-readable Egyptian hieroglyphic texts with their transliterations is needed. We have identified two sources where encoded Egyptian hieroglyphic words and their transliteration pairs are available. *AES - Ancient Egyptian Sentences*, based on texts from the Thesaurus Linguae Aegyptiae (TLA) [1], includes a collection of sentences where Egyptian words encoded using Manual de Codage (MdC) have been aligned with their transliteration counterparts [2]. The second, even more extensive, source of MdC encoded and transliterated words is the *Ramses Transliteration Corpus* (RTC) [3]. MdC is the most often used encoding for hieroglyphic texts [4, 5, 6]. The RTC consists of encoded hieroglyphic sentences, each on its own line, and respective transliteration lines in another file. However, unlike the AES, there is no ready alignment of the MdC and its transliteration on the word level. In order to find word-transliteration pairs, we need to align the sentences of encoded words with the respective transliterations. The alignment task

DHNB2023 | Sustainability: Environment - Community - Data. The 7th Digital Humanities in the Nordic and Baltic Countries Conference. Oslo – Stavanger – Bergen, Norway. March 8–10, 2023.

{firstname.lastname}@helsinki.fi (H. Jauhiainen); {firstname.lastname}@helsinki.fi (T. Jauhiainen)
https://researchportal.helsinki.fi/en/persons/heidi-jauhiainen (H. Jauhiainen);
https://researchportal.helsinki.fi/en/persons/tommi-jauhiainen (T. Jauhiainen)
0000-0002-8227-5627 (H. Jauhiainen); 0000-0002-6474-3570 (T. Jauhiainen)

© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

DHNB Publications, DHNB2023 Conference Proceedings, https://journals.uio.no/dhnbpub/issue/view/875

is made more difficult by the fact that there are damaged parts for which transliteration has sometimes been guessed, as well as grammatical additions by editors.

We have developed a highly accurate alignment method that uses a sequence alignment algorithm together with a dictionary of MdC - transliteration pairs generated initially from the intact words within the AES and the sentences in the RTC corpus that have the same number of words in both MdC and transliteration. Once the rest of the lines in the RTC were aligned, we could extract MdC - transliteration pairs and create transliteration models. Manual inspection of the models proved to be useful in helping to notice actual errors and inconsistencies in the RTC transliterations. We created additional rules to handle the errors and re-created the models. In this paper, we describe the automatic alignment pipeline and the openly published models.

Section 2 introduces some of the challenges in Digital Egyptology and gives further context on the research presented in this paper. In Section 3, we present previous work for the automatic alignment of hieroglyphs and some research using the same sequence alignment algorithm as this paper. Section 4 describes the AES - Ancient Egyptian Sentences and the Ramses Transliteration Corpora. We describe the alignment method itself in Section 5 as well as our evaluation of it using the Levenshtein distance [7]. In Section 6, we present some further steps that were needed to produce the model and the resulting models.

2. Background

Contrary to, for example, Assyriology, which has several online corpora of cuneiform texts,[1] Egyptology has no tradition of publishing machine-readable hieroglyphic texts [8]. In a hieroglyphic text, individual signs can be next to, above, or over another, and two or more signs can be nested. When Egyptologists study the texts, they draw a facsimile of the object and its inscriptions [9]. Many Ancient Egyptian texts were written by hand, which made the hieroglyphs more cursive, and Egyptologists generally transcribe these cursive signs into clearer hieroglyphs [10, 6]. Traditionally this was done by hand or by typesetting physical fonts, but nowadays, pictures produced with computers are also possible [6]. Applying optical character recognition (OCR) on hieroglyphic texts would produce machine-readable corpora. However, currently, the techniques are not readily usable as annotated texts in the same handwriting are needed for training the methods [8, 11]. When Egyptologists interpret hieroglyphic texts, they transliterate them with Latin letters and diacritics. The transliteration method used in Egyptology does not represent the text sign by sign, but instead, it is always an interpretation of the text as words.

It might seem obvious to use Unicode to produce machine-readable hieroglyphic texts. There are indeed over 1000 Unicode hieroglyphic characters, and since 2019, there are also so-called Format Control characters so that the signs can be presented properly in relation to each other [12, 13, 14].[2] However, the Unicode hieroglyphic characters are outside the Basic Multilingual

[1] E.g., Achemenet, http://www.achemenet.com; Open Richly Annotated Cuneiform Corpus (ORACC), http://oracc.museum.upenn.edu; ORACC at the Language Bank of Finland, http://urn.fi/urn:nbn:fi:lb-2019111601

[2] https://www.unicode.org/versions/Unicode14.0.0/ch11.pdf

Plane[3] and are not correctly handled by commonly used software applications. While the situation with the "digital divide" [16] between languages has improved as digital support for language diversity is expanding, many writing systems still have issues with common digital tools and technologies [17, 18]. Since the 1970s, egyptologists have been using special text editors to encode hieroglyphic texts so that the placement of the signs is maintained [13, 19]. Unfortunately, the machine-readable encodings have traditionally only been used for producing pictures of hieroglyphic texts and not published themselves [8].

Computer-assisted transliteration of hieroglyphic texts would aid Egyptologists in reading the encoded texts and publishing them for digital studies. Recently, Rosmorduc [20] proposed an automated transliteration method using neural networks. The method works on short sentences, but the sentence boundaries are not indicated in hieroglyphic texts, and sentences often span from one line to the next. A new method for transliterating whole texts at a time is, hence, needed. As Wiechetek et al. [21] have shown, using neural networks on less-resourced languages is often not feasible, and more traditional machine learning methods could be more effective. We have previously developed a back-off-based language identification method, HeLI [22], which still is superior to neural network-based identifiers in many tasks [23, 24]. Hence, the transliteration method we are currently developing is based on a similar back-off scheme, which at its core utilizes a transliteration model of hieroglyphic words and their transliterations together with the observed relative frequencies of the pairs.

3. Previous Work

3.1. Automatic Alignment of Hieroglyphs and Transliteration

The problem of automatic alignment of hieroglyphs and transliteration has been researched by Nederhof [25, 26]. His starting point for alignment differed from the current work since no word tokenization was present in the hieroglyphic texts he used. His method goes through the individual hieroglyphs and checks how well their possible transliterations match the existing transliteration. A customized scoring system gives varying penalties to different readings, and in the end, the reading with the lowest penalty is chosen.

In the corpus we use, the word tokenization is already indicated. Therefore it is possible to use much more straightforward methods for aligning the hieroglyphic words to their transliterated counterparts. By more straightforward, we mean that almost no expert knowledge in hieroglyphic writing is needed as our method learns the dictionaries automatically from the training corpus.

3.2. Sequence Alignment

In this work, we use the Needleman-Wunsch algorithm [27] to align encoded hieroglyphic text with their transliteration. This method was designed for and primarily used in bioinformatics. It works well, for example, when comparing proteins or genomic DNA sequences of up to tens of thousands of nucleotides [28]. The algorithm has also been used for aligning natural language text. Song et al. [29] evaluated several sequence alignment algorithms in identifying

[3] See the description by Tauber [15] for more information about the structure of the Unicode codespace.

sentence parallelism in student essays. They found the Needleman-Wunsch algorithm to perform best out of five individual algorithms but attained even better results when combining it with the others. In other areas of natural language processing, the Needleman-Wunsch algorithm has been used, for example, by Lai and Hockenmaier [30] and Itoh [31] to estimate semantic similarity in SemEval shared tasks. This method suits our needs exceptionally well as it supposes the sequences to be in the correct order but finds the places where items are missing from either sequence.

4. Corpora

In this section, we describe the two corpora from which we produced transliteration models AES - Ancient Egyptian Sentences [2], and The Ramses Transliteration Corpus [20].

4.1. AES - Ancient Egyptian Sentences

AES - Ancient Egyptian Sentences; Corpus of Ancient Egyptian sentences for corpus-linguistic research (AES) is a collection of more than 100,000 sentences with c. 254,000 annotated and transliterated words aligned with their MdC encoding [2]. The corpus was published in GitHub under a CC BY-SA license.[4] AES sentence corpus is formatted as JSON, which is more suitable for our pipelines than the TEI format of the Corpus of Egyptian Texts for the AED - Ancient Egyptian Dictionary (AED-TEI) [32],[5] from which AES has been extracted. AED-TEI contains more than 11,000 Egyptian texts and is itself based on a database snapshot from the "Strukturen und Transformationen des Wortschatzes der ägyptischen Sprache" project [1].[6] This database is also used to create the online version of the Thesaurus Linguae Aegyptiae (TLA) [33].[7]

Figure 1 shows an example of a JSON entry for a single Egyptian word in AES. A JSON entry consists of keyword – value pairs: *"keyword": "value"*. The AES information defines the value of the "mdc" keyword as an Egyptological transcription in MdC. What it actually means is that in the value of "mdc", the characters used in the transcriptions conform to the Buurman et al. 1988 transliteration scheme [4].[8] The MdC allows the use of these transliteration characters for some of the hieroglyphs in addition to the letter–number combinations which are the codes coming from the so-called *Gardiner Sign List* [34], the standard reference list for Egyptian hieroglyphs.[9] The MdC equivalent of the original hieroglyphic characters we are seeking is actually found in the value of the "hiero" keyword: M20-X1-Z2. We were hoping to use a dictionary created from the AES to align the MdC and the transliteration in the RTC corpus. However, the "mdc" value in AES is not similar to the RTC transliteration, as it includes extra annotation such as the plural marker ".pl" in the example. The value for the "written_form" keyword is described to be the same as for "mdc", but in Unicode, which seems to use the character set

[4] https://github.com/simondschweitzer/aes

[5] https://github.com/simondschweitzer/aed-tei

[6] https://nbn-resolving.org/urn:nbn:de:kobv:b4-opus4-29190

[7] https://thesaurus-linguae-aegyptiae.de

[8] For the currently most complete list of transliteration schemes for Ancient Egyptian, see https://en.wiktionary.org/wiki/Appendix:Egyptian_transliteration_schemes.

[9] https://www.unicode.org/notes/tn32/Unicode-MdC-Mapping-v1.utf8

```
{
    "_id": "IBUBdx3kjk1StkQar5VP5tBFvOk",
    "lineCount": "[x+7]",
    "written_form": "sẖ,t.pl",
    "mdc": "sx,t.pl",
    "cotext_translation": "Feld; Weide; Marschland",
    "lemma_form": "sẖ.t",
    "lemmaID": "141480",
    "zaehler": "14",
    "pos": "substantive",
    "numerus": "plural",
    "status": "st_absolutus",
    "hiero_inventar": "M20;X1;Z2",
    "hiero_unicode": "&#x131cf;&#x133cf;&#x133e5;",
    "hiero": "M20-X1-Z2"
},
```


Figure 1: Example of an Egyptian word in AES corpus.

of the Werning 2013 (traditional) transliteration scheme [35]. The value for the "lemma_form" keyword conforms to the Werning scheme and does not include extra annotation, and it is also possible to automatically transform the Werning transliteration in the "lemma_form" into Buurman-compliant transliteration. Thus, this is the entry we use in the automatic alignment experiments, even though it sometimes omits information such as the plural "w" in the case of the example in figure 1. The RTC equivalent transliteration for "M20-X1-Z2" would be "sx.wt".

The AES corpus is very heterogeneous. The dates attributed to the texts span from the Ancient Egyptian Old Kingdom to the Roman period, that is, for a period of over 2,500 years (c. 2600 BCE - 300 CE). Furthermore, the corpus is composed of texts from many different kinds of genres, such as religious texts, administrative texts, letters, medical texts, rock inscriptions, and so forth. Since the Egyptian language changed over time and different genres use different vocabulary, the model produced from the corpus is bound to be miscellaneous.

4.2. The Ramses Transliteration Corpus

The Ramses Transliteration Corpus, V. 2019-09-01 (RTC) was created by Rosmorduc [20] for training and testing his automated transliteration method [36]. The corpus was published in Gitlab[10] and Zenodo[11] under CC-BY-NC-SA license, and it contains sentences of encoded hieroglyphic text and respective lines of their transliterations in separate files [3]. Original hieroglyphic texts do not include word boundaries. However, since the data was originally collected in the Ramses Project [37] and is available for word searches in Ramses Online,[12] the corpus contains, in addition to texts without word boundaries, also separate versions of the files where words have been separated with underscores. In order to find word-transliteration pairs, we use the files with word boundaries. Examples of lines from these files can be seen in

[10] https://gitlab.cnam.fr/gitlab/rosmorse/ramses-trl
[11] https://doi.org/10.5281/zenodo.4954597
[12] http://ramses.ulg.ac.be

```
M17 Z7 _ M17 Z7 _ A1 _ D21 _ S29 G36 D21 N35A A2 _ M17 G17 _ I9 _
U23 G17 D21 G37 _ LACUNA V30 _ LACUNA
O4 Ff1 N35 G1 Ff1 Ff1 O1 F35 I9 D21 A1 _
G41 G1 _ O4 D21 Z7 N5 Z1 _ N35 _ MISSING G41 G1 _ W25 N35 W24 Z7 Y1 Z2 _
```

```
i w _ i w _ = i _ r _ s w r _ m _ = f _
m H r _ L A C U N A _ n b _ L A C U N A _
h y n - n f r _
p A _ h r w _ n _ m s _ p A _ i n w _
```

Figure 2: Four corresponding lines from the "src-sep-train.txt" and the "tgt-train.txt" files which are part of the RTC training set.

Figure 2. These examples also demonstrate the fact that what we refer to as a word actually means a token, such as "A1" corresponding to "=i", which is a suffix pronoun. Both AES and RTC consider these as separate tokens from the words they are connected to.

Egyptian texts are often fragmentary and damaged in places. In RTC, there sometimes exists a possible transliteration for these damaged parts, whether individual signs or longer passages. Furthermore, grammatical forms not present in the hieroglyphic text have often been added in the transliteration. These guesses and additions have been marked in a variety of ways as transliterations have been produced by numerous scholars.

The RTC comprises over 71,000 sentences of Late Egyptian texts (c. 1550-1069 BCE). The texts are all from a single phase of the Ancient Egyptian Language, and the text types in Ramses Online are less varied than the ones in AES. The sentences are in random order, and there is no indication of which text they belong to. The training corpus contains 66,693 encoding and transliteration line pairs, whereas the validation set has 1,841 and the test set 2,729 line pairs.

All sets of the RTC were preprocessed to remove inconsistencies as far as possible. We inspected the sign values present in the training and the validation sets. As mentioned earlier, the Mdc allows using transliteration for certain signs, and, indeed, the corpus does include some annotations using transliteration rather than the Gardiner sign list keys for seven different signs.[13] In the first preprocessing step, we replaced them with the codes used elsewhere in the corpus for these hieroglyphs.[14]

Hieroglyphic texts in the RTC were originally written on papyri or so-called ostraca, that is, pieces of limestone or pottery. The texts that have survived until today have often suffered damage, particularly to their edges. When part of the text is missing or illegible, this is usually marked in the encoding. For missing parts of words, shading is often used. The Ramses corpus includes three different annotations, "SHADED1", "SHADED2", and "SHADED3". There was no apparent reason for the different numbers, so we unified these three annotations as "SHADED".

An entire illegible or missing word is in the corpus marked with "LACUNA" or "MISSING". A "LACUNA" in the hieroglyphic text usually corresponds to a "L A C U N A" in transliteration. When a hieroglyph or word is marked as missing, it may correspond to transliteration or not. The encoding files do not indicate a token break after "LACUNA" or "MISSING" as is done in

[13] "n", "A", "f", "t", "i", "nTrw", and "nn"

[14] "N35", "G1", "I9", "X1", "M17", "R8A", and "M22 M22"

the transliteration, so we added an underscore after these in the encoding files used.[15]

5. Alignment Method

The Needleman-Wunsch algorithm [27] was originally developed to align long protein sequences. It finds the tokens that are the same, and when there is a mismatch, it either aligns the two tokens or inserts a gap finding the optimal alignment. To use the algorithm for hieroglyphic sentences, we need to align the transliterated words with the encoded words, and, therefore, we need a dictionary to compare these two. Our alignment method is, hence, based on creating a dictionary of pairs of encoded words and their transliteration and using it together with the Needleman-Wunsch algorithm. We tested three slightly different ways of producing the dictionary from words in the RTC assumed to be aligned without further processing. The dictionaries were tested with the various configurations of the Needleman-Wunsch algorithm on a manually aligned part of the validation set.

5.1. Gold Standard Alignment

In order to assess the efficacy of the developed method, we created a gold standard alignment from the RTC validation set. We created an alignment test set by choosing the first 100 pairs that had an unequal number of tokens. Dividing the tokens into columns in a Microsoft Excel spreadsheet provided an easy and fast way to manually verify and correct the existing alignments. An example of an encoding-transliteration pair in its original and manually aligned forms can be seen in Table 1.

orig.	D37 Z7	O29 O1 O1 Z1 G7	S34 U28 S29	D21	T10 X1 Z1 A1 Z3A	N35	...
	r d i	<w i >	p r - a A	a . w . s	r	<H r y >	...
gold	D37 Z7	-	O29 O1 O1 Z1 G7	S34 U28 S29	D21	-	...
	r d i	<w i >	p r - a A	a . w . s	r	<H r y >	...

Table 1
Example of original and correctly aligned encoding transliteration pair from the validation set.

5.2. Dictionaries

We experimented with using dictionaries created from both the RTC and the AES corpora with our implementation of the Needleman-Wunsch algorithm. The dictionaries are files consisting of tab-separated values for tokens together with all their possible transliterations. Figure 3 shows an entry for the token "M17 Z7" from the third RTC dictionary. The token has five possible transliterations "in-iw", "ir", "iw", "iwf", and "r". The number in the third column indicates the frequency of the MdC – transliteration pair; however, this information is not used in the alignment algorithm.

[15]Examples of both "LACUNA" and "MISSING" can be seen in the second and fourth rows of the Figure 2.

```
M17 Z7  in-iw   1
M17 Z7  ir      3
M17 Z7  iw      3856
M17 Z7  iwf     1
M17 Z7  r       9
```

Figure 3: Example entry from the third RTC dictionary. The Egyptian token "M17 Z7" has five possible transliterations.

The dictionary from the AES corpus was built using each token that had a value for both the *hiero* and the *lemma_form* keywords. For our dictionary, the AES values written using the Werning scheme were automatically converted to the Buurman-compliant transliteration.

The three dictionaries from the RTC were built using the training set. After the preprocessing described in Section 4.2, quite a large portion of the encoding-transliteration pairs in the RTC training set had the same number of tokens in the encoding and the transliteration and were, therefore, assumed to be correctly aligned. This large number of correctly aligned pairs enabled us to create a token dictionary from the line pairs with an equal number of tokens with high confidence. When building the first and second versions of the RTC dictionary, we did not include any pairs of lines that had broken or partially broken transliteration, e.g., lines containing characters "[", "/", or "?", or inserted words that are not present in the encoding indicated by character "<" or parenthesis around the entire transliteration of a word.

For the first RTC dictionary, we used pairs of lines that did not contain annotations "SHADED", "LACUNA", or "MISSING" in the encoding line. There were 31,938 such intact line pairs in the training material and the dictionary contained 32,023 words. To create the second version of the RTC dictionary, we did use the lines with "LACUNA" if none of the other indications of broken text or insertion were present. The total number of lines from which the second RTC dictionary, with 36,891 words, was created was 42,541. For the third RTC dictionary, we used all 56,752 pairs of lines with an equal number of tokens and gathered 45,225 words. However, when adding words to the third dictionary, we still left out the word pairs that contained the characters mentioned above or "SHADED" or "MISSING" as annotations on the encoding line.

5.3. Needleman-Wunsch

In the Needleman-Wunsch algorithm, a two-dimensional array represents all possible pairs of the units to be aligned. Only the pathways from the last items of the sequences to the first have to be considered. For each pair of encoded (x) and transliterated (y) sentences, we first construct an array of size *length of* $x + 1$ times *length of* $y + 1$ (see Fig. 4). The cells where $x = 0$ are filled with numbers descending from 0. The same is done to the cells where $y = 0$. The rest of the cells are filled according to Equation 1.

$$[x, y] = \max \begin{cases} [x-1, y-1] + dscore \\ ([x, y-1] - 1) \\ ([x-1, y] - 1) \end{cases} \tag{1}$$

That is, the cell is assigned the maximum of the scores in either the cell to the top left plus a dictionary score *dscore*, the cell to the left minus the penalty point (-1), or the cell above minus

the penalty point (-1). The value of *dscore* is based on whether the transliteration is found in the dictionary entry for the encoding of the words under consideration (5 points) or not (-1 points). In Figure 4, only two encoded words (I10 D46 and I9) have a matching transliteration (Dd and f, respectively) in our dictionary. For all the other pairs, the *dscore* is -1.

		L A C U N A	Dd	<n>	f	iA.tw	n-aDA
	0	-1	-2	-3	-4	-5	-6
LACUNA	-1	-1	-2	-3	-4	-5	-6
I10 D46	-2	-2	4	3	2	1	0
I9	-3	-3	3	3	8	7	6
M17 G1 Z7 X1 A30 A2	-4	-4	2	2	7	7	6
N35 D36 U28 G1 M17 Z7 G37	-5	-5	1	1	6	6	6

Figure 4: The Needleman-Wusch algorithm used on a encoding - transliteration sentence pair. The optimal alignment is found by backtracing the arrows that indicate from where the value assigned to the cell was received. Only the word pairs marked with red are found in our dictionary and get the *dscore* 5. See Table 2 for the final alignment.

Another array is constructed simultaneously and filled with the information from which direction – left, top, or top left – the maximum score was found. In Figure 4, the relevant directions are conveyed with arrows. After both arrays have been filled, the best alignment is found by starting from the cell [*length of x*][*length of y*], that is, from the bottom right cell. The encodings and transliterations are aligned, starting from the last word and moving toward the first. If the score came from the top left, the encoding and transliteration x and y are aligned. If the score was received from the left, the transliteration is aligned with an empty slot, marked by "-". When the score comes from the above cell, it is the transliteration that is left empty. One then moves on to the cell where the maximum score came from until reaching the top left corner of the array [$x = 0$][$y = 0$]. For the alignment of the sentences in Figure 4 see Table 2.

Encoding	LACUNA	I10 D46	-	I9	M17 G1 Z7 X1 A30 A2	N35 D36 U28 G1 M17 Z7 G37
Transliteration	L A C U N A	D d	<n>	f	i A . t w	n - a D A

Table 2
The final alignment of the sentence pair in Fig. 4. In the figure, the word pair I10 D46 - <n> gets its score from the cell to its left (indicated by the horizontal arrow) and, therefore, the transliteration <n> has been aligned with an empty slot.

When the sentence is long or has many words not present in the dictionary, the maximum score for a cell may sometimes be received from several directions; that is, there are multiple optimal alignments. In order to find out which passage is best, we sum up the *dscores* for each pair on every possible passage. The highest scored alignment is then returned. In case there are several alignments with the highest score, we favor the diagonal alignment and, since additional words are more common in transliteration, left over top.

5.4. Evaluation of the Segmentation Method

We evaluated the method using the Levenshtein distance, which calculates the minimum number of single token edits to change one string to the other.[16] Levenshtein distance was also used for assessing the correctness of transliteration of hieroglyphic texts by Rosmorduc [20]. We implemented the method according to "Algorithm X" described by Wagner and Fischer [38]. Like in the Needleman-Wunsch algorithm, when comparing strings x and y, one uses a two-dimensional array, but the cells where $x = 0$ are filled with numbers ascending from 0 instead of descending. The same is done to the cells where $y = 0$. Like in the Needleman-Wusch algorithm, when assigning a score to a cell, one always considers the score in the cells to the left, top left, and above. A penalty point is added to the scores to the left and above and, if the tokens to compare are not the same, also to the score in the top left cell. After filling the array, the Levenshtein distance is retrieved from the cell [*length of x*][*length of y*]. We used one as a penalty point. We compared both lines produced with our alignment method, that is the encoding line and the transliteration line to the respective ones in the gold standards. This means that, in fact we always compared 200 lines. The smaller the Levenshtein distance, the more similar the lines are.

The Levenshtein distance of our alignment test set to our gold standard was 124 before doing any aligning. Examining the lines showed there were sometimes long sections of insertions, even at the beginning and the end of the sentences.

Initially, in our Needleman-Wunsch algorithm, we simply checked whether the transliteration was in the entry of the encoded word we were trying to align it with. Since a hieroglyphic word could be written in many ways by using different sign combinations, we tested various slightly different ways of finding out whether the encoding and the transliteration refer to the same word.

The basic matching scheme in the alignment algorithm was to compare the transliteration to be aligned with all possible transliterations for the encoded hieroglyphic token. In case of a match, the *dscore* is 5. Already in the initial matching scheme, the "Simple", if there was no match for the first rule, the alignment was additionally considered a match if the transliteration or any of the possible transliterations started with and completely contained the other. This partial match gave a *dscore* of 4. Using the simple scheme, the performance of the AES dictionary was worse than expected, giving a distance of 147, which means there were more mistakes than in the original alignment. Using our first Ramses dictionary gave a distance of 40, which was topped by 30 by the second dictionary and 26 by the third one (see Table 3). Adding AES to each of the Ramses dictionaries improved the results slightly to 38, 28, and 24, respectively.

In our second matching scheme, we added the "First 3" rule, i.e., taking the first three MdC codes from the encoding under scrutiny and adding the transliterations of all the words starting with those three codes to the list of possible ones before checking. If neither of the simple scheme rules matched, the "First 3" rule gave a *dscore* of 4. The distance with the AES dictionary improved to 98 while the first Ramses dictionary gave 16, and the second and the third 10. This rule seemed to work especially while using the AES dictionary with the first and second Ramses

[16] See example analyses by Kruskal [7] (Section 4. Levenshtein And Other Distances) on how to calculate the Levenshtein distance.

Dict.	Distance Simple	+First 3	+ Logogram
AES	147	98	81
RTC 1st	40	16	8
RTC1 + AES	38	12	6
RTC 2nd	30	10	2
RTC2 + AES	28	6	2
RTC 3rd	26	10	2
RTC3 + AES	24	6	2

Table 3
Evaluation of the the first 100 lines of the validation set with unequal number of tokens against our gold standard using the first and second dictionaries with various matching methods in the Needleman-Wunsch algorithm.

dictionaries. The first RTC gave 12, and the second 6. The rule had no effect with the third dictionary compared to the second.

In the third matching scheme, a "Logogram" rule was added. With this rule, only the first sign of the encoding is considered. We used the sign similarly to the second rule of the "Simple" scheme. If none of the previous rules was usable, the "Logogram" rule gave a *dscore* of 3. This resulted in a small improvement of the Levenshtein distance with all the dictionaries: AES 81, the first Ramses dictionary 8, and the combination of these 6. All the other dictionaries gave the distance 2.

6. Transliteration Models

Since we wanted to align the sentences as well as possible and produce good-quality transliteration models, but we could not achieve Levenshtein distance 0 in our evaluation, we resorted to using a corpus-specific addition in the alignment method. The *dscore* was set to -5 if the transliteration contained any of the characters indicating an insertion by the annotator, i.e., parenthesis or angle brackets around the word. These could be possibly preceded by "=", indicating that a suffix pronoun was meant. This measure gave the distance 0 with all three Ramses dictionaries, even with the simple matching scheme.

In order to verify the reliability of the alignment of sentence pairs with an uneven number of words in the RTC, we used the third Ramses dictionary and all the matching rules, including the corpus-specific addition, for aligning the entire Ramses training set. We then made a wordlist out of the aligned lines and studied that manually. The broken words and additions are omitted from the word lists used for producing the transliteration model, but they were retained for this step of the process. When intact words were found to be paired with a '-', we studied the sentence where that happened. We found several sentences where the MdC and transliteration did not match. For example, some names of kings were always written as one word in the encoding but as two words in the transliteration. We made rules to align these names properly. For creating the transliteration model, we added these rules to the preprocessing step. Occasionally, some of the hieroglyphs had also been left untransliterated, and sometimes

there were transliterations that had been deduced from the context but with no counterpart in the MdC. Often this did not matter as our method could detect the missing words, but for some sentences, we just could not align properly, so we made a list of sentences to ignore.

Once we were content with the dictionary built from the aligned lines, we used the dictionary to build the transliteration model. The various dictionaries contain information on the frequency of each MdC-transliteration pair and can be used as transliteration models of the specific corpora they were built from. In order to publish the models in a more structured format, we made a script to write them using the JSON scheme. The script allows one to build a transliteration model from a desired number of word lists. We build JSON-format models from the AES words, the words in the Ramses training model, and a combined AES-Ramses model.

```
{
      "encoding": "M17 Z7",
      "interpretations": [
        {
          "transliteration": "iw",
          "freq": 8142,
          "relFreq": 99.75
        },
        {
          "transliteration": "r",
          "freq": 13,
          "relFreq": 0.16
        },
        {
          "transliteration": "ir",
          "freq": 4,
          "relFreq": 0.05
        },
        {
          "transliteration": "in-iw",
          "freq": 2,
          "relFreq": 0.02
        },
        {
          "transliteration": "iwf",
          "freq": 1,
          "relFreq": 0.01
        }
      ]
    },
```

Figure 5: Example of an Egyptian word in the transliteration model created from the RTC training corpus.

The models have been published under an open license on Zenodo and GitHub.[17] The models are JSON files with separate entries for each MdC token, as seen in Figure 5. The token in the example, "M17 Z7", is the same as in Figure 3. The "encoding" keyword gives the MdC for the

[17] http://doi.org/10.5281/zenodo.7991240, https://github.com/MaReTEgyptologists/TranslitModels

token, and all attested transliterations are collected under the "interpretations" keyword. For each transliteration, the frequency "freq" is indicated as is the observed probability, "relFreq", of the transliteration for the given MdC token. Due to being able to align the different length MdC – transliteration pairs, the number of "iw" transliterations for "M17 Z7" more than doubled even though the number of lines only grew from c. 57,000 to 67,000. This difference is due to the fact that misalignment occurs more often in longer sentences than in short ones.

The Table 4 shows the sizes of the different versions of the published models.

Model	# of unique tokens	Total frequency	# of unique MdC – transliteration pairs
AES	43,416	250,909	46,811
RTC	48,914	447,719	55,049
AES + RTC	84,558	698,628	97,777

Table 4
The published transliteration models and their sizes.

7. Conclusions

Our alignment method manages to align even complex sentences with the difference of several words in the original encoding and transliteration lines. Increasing the size of the dictionary by including more pairs of words improved the results every time. Using a penalty for the obvious insertions by annotators was needed to align the original lines and our gold standard perfectly. In order to build good-quality transliteration models, some additional corpora-specific rules were needed. Although the alignment method depends on the corpus format, which is currently unique to this export done by Rosmorduc [20], we have published the software used for building the dictionaries and aligning the sentences. They cannot be used off-the-shelf for other corpora, but we believe that they can be modified for or at least give an example of how the Needleman-Wunsch can be used for aligning sentences that do not have the same format.

We have published a combined transliteration model by using the words from the AES and the RTC corpora. However, our attempts to use the AES for aligning MdC and transliteration sentence pairs in the RTC corpus were unsuccessful and showed that the corpora are very different. We assume that the separate models built from each of the corpora will be more useful.

The most obvious thing for future work would be to evaluate the alignment method using the test set in the Ramses Transliteration Corpus. However, as we wish to use the test set to evaluate the forthcoming automatic transliteration method, we are hesitant to look at it, which is needed for manual alignment.

We have used the RTC Transliteration Model in our first experiments on automatically segmenting hieroglyphic texts into words, a task that is needed before automated transliteration. We noticed that information on two or more words in a window might be useful for the task. Hence, we intend to produce additional token n-gram transliteration models from the texts we have aligned.

Acknowledgments

The Kone Foundation has funded this research. This work has been partially supported also by the Academy of Finland (Funding decision no. 341798). We thank Serge Rosmorduc for publishing the Ramses Transliteration Corpus and Simon Schweitzer for publishing the AES corpus with an open license. We are also grateful to the people behind the Ramses Online and Thesaurus Linguae Aegyptiae services for allowing the data produced in their projects to be used in research.

References

[1] T. S. Richter, I. Hafemann, H.-W. Fischer-Elfert, P. Dils, Teilauszug der Datenbank des Vorhabens "Strukturen und Transformationen des Wortschatzes der ägyptischen Sprache" vom Januar 2018, 2018. URL: https://nbn-resolving.org/urn:nbn:de:kobv:b4-opus4-29190.

[2] Schweitzer, Simon D., AES - Ancient Egyptian Sentences; Corpus of Ancient Egyptian sentences for corpus-linguistic research, 2021. URL: https://github.com/simondschweitzer/aes.

[3] University of Liege/Project Ramses, The Ramses transliteration corpus V. 2019-09-01, 2020. URL: https://gitlab.cnam.fr/gitlab/rosmorse/ramses-trl.

[4] J. Buurman, N. Grimal, M. Hainsworth, J. Hallof, D. van der Plas, Inventaire des signes hiéroglyphiques en vues de leur saisie informatique: Manuel de codage des textes hiéroglyphiques en vue de leur saisie sur ordinateur, Institut de France, Paris, 1988.

[5] M.-J. Nederhof, A Revised Encoding Scheme for Hieroglyphic, in: Proceedings of the XIV Computer-aided Egyptology Round Table, Pisa, Italy, July 2002, IE2002, 2002.

[6] S. Polis, S. Rosmorduc, Réviser le codage de l'égyptien ancien: vers un répertoire partagé des signes hiéroglyphiques, Document numérique 16 (2013) 45–67.

[7] J. B. Kruskal, An overview of sequence comparison: Time warps, string edits, and macromolecules, SIAM Review 25 (1983) 201–237. URL: https://epubs.siam.org/doi/pdf/10.1137/1025045.

[8] M.-J. Nederhof, OCR of Handwritten Transcriptions of Ancient Egyptian Hieroglyphic Text, in: Proceedings of Altertumswissenschaften in a Digital Age: Egyptology, Papyrology and Beyond, 2015.

[9] C. Thiers, The So-Called Karnak Method, in: The Oxford Handbook of Egyptian Epigraphy and Paleography, Oxford University Press, 2020. doi:10.1093/oxfordhb/9780190604653.013.21.

[10] A. H. Gardiner, The transcription of new kingdom hieratic, The Journal of Egyptian Archaeology 15 (1929) 48–55. URL: http://www.jstor.org/stable/3854012.

[11] B. Bermeitinger, S. A. Gulde, T. Konrad, How to compute a shape: Optical character recognition for hieratic, in: C. Gracia Zamacona, J. Ortiz-García (Eds.), Handbook of Digital Egyptology: Texts, Editorial Universidad de Alcalá, 2021, pp. 121–138.

[12] The Unicode Consortium, The Unicode Standard, Version 14.0.0, 2021. URL: https://www.unicode.org/versions/Unicode14.0.0/.

[13] R. B. Gozzoli, Hieroglyphic Text Processors, Manuel de Codage, Unicode, and Lexicogra-

phy, in: S. Polis, J. Winand (Eds.), Texts, Languages Information Technology in Egyptology, Presses Universitaires de Liège, Liège, 2013, pp. 89–101.

[14] M.-J. Nederhof, S. Polis, S. Rosmorduc, Unicode control characters for ancient egyptian, in: Proceedings of the International Congress of Egyptologists XII, F.R.S.-FNRS - Fonds de la Recherche Scientifique, IFAO, In press.

[15] J. K. Tauber, Character encoding of classical languages, in: M. Berti (Ed.), Digital Classical Philology, De Gruyter Saur, Berlin, Boston, 2019, pp. 137–158. doi:doi:10.1515/9783110599572-009.

[16] R. Cullen, Addressing the digital divide, Online information review 25 (2001) 311–320.

[17] F. Albarillo, Evaluating language functionality in library databases, International Information & Library Review 48 (2016) 1–10. doi:10.1080/10572317.2016.1146036.

[18] I. A. Zaugg, Digital inequality and language diversity: An ethiopic case study, in: M. Ragnedda, A. Gladkova (Eds.), Digital Inequalities in the Global South, Springer International Publishing, 2020, pp. 247–267. doi:10.1007/978-3-030-32706-4_12.

[19] S. Rosmorduc, Digital writing of hieroglyphic texts, in: C. Gracia Zamacona, J. Ortiz-García (Eds.), Handbook of Digital Egyptology: Texts, Editorial Universidad de Alcalá, 2021, pp. 37–53.

[20] S. Rosmorduc, Automated Transliteration of Late Egyptian Using Neural Networks: An Experiment in "Deep Learning", Lingua Aegyptia-Journal of Egyptian Language Studies 28 (2020) 233–257.

[21] L. Wiechetek, K. Hiovain-Asikainen, I. L. S. Mikkelsen, S. Moshagen, F. Pirinen, T. Trosterud, B. Gaup, Unmasking the myth of effortless big data - making an open source multi-lingual infrastructure and building language resources from scratch, in: Proceedings of the Language Resources and Evaluation Conference, European Language Resources Association, Marseille, France, 2022, pp. 1167–1177. URL: https://aclanthology.org/2022.lrec-1.125.

[22] T. Jauhiainen, K. Lindén, H. Jauhiainen, HeLI, a word-based backoff method for language identification, in: Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3), The COLING 2016 Organizing Committee, Osaka, Japan, 2016, pp. 153–162. URL: https://www.aclweb.org/anthology/W16-4820.

[23] T. Jauhiainen, H. Jauhiainen, K. Lindén, HeLI-OTS, off-the-shelf language identifier for text, in: Proceedings of the Thirteenth Language Resources and Evaluation Conference, European Language Resources Association, Marseille, France, 2022, pp. 3912–3922. URL: https://aclanthology.org/2022.lrec-1.416.

[24] B. R. Chakravarthi, M. Gaman, R. T. Ionescu, H. Jauhiainen, T. Jauhiainen, K. Lindén, N. Ljubešić, N. Partanen, R. Priyadharshini, C. Purschke, E. Rajagopal, Y. Scherrer, M. Zampieri, Findings of the VarDial evaluation campaign 2021, in: Proceedings of the Eighth Workshop on NLP for Similar Languages, Varieties and Dialects, Association for Computational Linguistics, Kyiv, Ukraine, 2021, pp. 1–11. URL: https://www.aclweb.org/anthology/2021.vardial-1.1.

[25] M.-J. Nederhof, Automatic Alignment Of Hieroglyphs And Transliteration, Gorgias Press, 2009, pp. 71–92. doi:doi:10.31826/9781463216269-007.

[26] M.-J. Nederhof, Automatic Creation of Interlinear Text for Philological Purposes, Traitement Automatique des Langues (2009).

[27] S. B. Needleman, C. D. Wunsch, A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins, Journal of Molecular Biology 48 (1970) 443–453. doi:https://doi.org/10.1016/0022-2836(70)90057-4.

[28] A. L. Delcher, S. Kasif, R. D. Fleischmann, J. Peterson, O. White, S. L. Salzberg, Alignment of Whole Genomes, Nucleic Acids Research 27 (1999) 2369–2376. doi:10.1093/nar/27.11.2369.

[29] W. Song, T. Liu, R. Fu, L. Liu, H. Wang, T. Liu, Learning to Identify Sentence Parallelism in Student Essays, in: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, The COLING 2016 Organizing Committee, Osaka, Japan, 2016, pp. 794–803. URL: https://aclanthology.org/C16-1076.

[30] A. Lai, J. Hockenmaier, Illinois-LH: A Denotational and Distributional Approach to Semantics, in: Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), Association for Computational Linguistics, Dublin, Ireland, 2014, pp. 329–334. URL: https://aclanthology.org/S14-2055. doi:10.3115/v1/S14-2055.

[31] H. Itoh, RICOH at SemEval-2016 Task 1: IR-based Semantic Textual Similarity Estimation, in: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), 2016, pp. 691–695.

[32] Schweitzer, Simon D., Corpus of Egyptian Texts for the AED - Ancient Egyptian Dictionary, 2019. URL: https://github.com/simondschweitzer/aed-tei.

[33] S. D. Schweitzer, Compiling the lexicon of ancient egyptian: State of the art, in: C. Gracia Zamacona, J. Ortiz-García (Eds.), Handbook of Digital Egyptology: Texts, Editorial Universidad de Alcalá, 2021, pp. 103–120.

[34] Sir A. Gardiner, Egyptian Grammar: Being an Introduction to the Study of Hieroglyphs, 3rd. ed., Griffith Institute, Oxford, 1957.

[35] D. A. Werning, Einführung in die hieroglyphisch-ägyptische Schrift und Sprache: Propädeutikum mit Zeichen-und Vokabellektionen, Übungen und Übungshinweisen, Humboldt-Universität zu Berlin, 2015.

[36] S. Rosmorduc, Ramses automated translitteration software, 2021. doi:10.5281/zenodo.4954597.

[37] J. Winand, S. Polis, S. Rosmorduc, Ramses. An Annotated Corpus of Late Egyptian, in: Proceedings of the 10th International Congress of Egyptologists. University of the aegean, rhodes 22-29 May 2008, Peeters, 2015, pp. 1513–1521.

[38] R. A. Wagner, M. J. Fischer, The String-to-String Correction Problem, J. ACM 21 (1974) 168–173. doi:10.1145/321796.321811.