The Raymond and
Beverly Sackler Faculty
of Exact Sciences
Tel Aviv University

Transliteration of Judeo-Arabic Texts into Arabic Script Using Recurrent Neural Networks

A thesis
submitted in partial fulfilment
of the requirements for the Degree
of
Master of Science

by

Ori Terner

This research has been carried out under the supervision
of
Prof. Nachum Dershowitz

Tel Aviv University
School of Computer Science

October '19

Abstract

Many of the great Jewish works of the Middle Ages were written in Judeo-Arabic, a Jewish branch of the Arabic language family that incorporates the Hebrew script as its writing system. In this work we are trying to train a model that will automatically transliterate Judeo-Arabic into Arabic script; thus we aspire to enable Arabic readers to access those writings. We adopt a recurrent neural network (RNN) approach to the problem, applying connectionist temporal classification loss to deal with unequal input/output lengths. This choice obligates adjustments, termed *doubling*, in the training data to avoid input sequences that are shorter than their corresponding outputs. We also utilize a pretraining stage with a different loss function to help the network converge. Furthermore, since only a single source of parallel text was available for training, we examine the possibility of generating data synthetically from other Arabic original text from the time in question, leveraging the fact that, though the convention for mapping applied by the Judeo-Arabic author has a one-to-many relation from Judeo-Arabic to Arabic, its reverse (from Arabic to Judeo-Arabic) is a proper function. By this we attempt to train a model that has the capability to memorize words in the output language, and that also utilizes the context for distinguishing ambiguities in the transliteration. We examine this ability by testing on shuffled data that lacks context. We obtain an improvement over the baseline results (9.5% error), achieving 2% error with our system. On the shuffled test data, the error rises to 2.5%.

Table of Contents

Acknowledgments

בשם השם א־ל עולם
I give all my thanks to my family. I'm in gratitude to my parents and especially to my dear wife for her necessary and essential indefinite support.
I would like to thank my advisor for his patience and for making my visits so pleasant, and for the discussions on matters principled to the universe.
Also to Rabbi Shmuel Baum who has a most significant part in making this work possible.

Finally to the Creator of All. May we live to justify the name that he has called upon us.

List of Figures

List of Tables

1 Introduction

In this work we are trying to accomplish the automatic transliteration of Judeo-Arabic texts which were written using the Hebrew alphabet, to form readable Arabic texts. Judeo-Arabic here refers to the form of classical Arabic that was written in Hebrew script, particularly during the Middle Ages. Many prominent Jewish religious works during the Middle Ages were originally written in Judeo-Arabic. Great authors, such as Maimonides, Rabbi Judah Halevi, the Geonim and many more, have written much of their work in Judeo-Arabic. This has several reasons, as Judah ben Saul ibn Tibbon mentions in his preface to his Hebrew translation of the book *Al Hidayah ila Faraid al-Ḳulub [Direction to the Duties of the Heart]* by Rabbi Bahya ibn Paquda, which was also written in this manner.

> First, most of the *Geonim* in Babylon and in the region of Israel and the Persian Empire were living in Arab speaking countries, and were speaking the Arabic language. They did so [writing in Arabic] because all the people understood this language. Secondly, since it is broad by any means, and it is sufficient for every speaker and writer, and the rhetoric is straight and clear, reaching the objective in every subject more than is possible with the Hebrew language. This is because in the Hebrew language we don't have but what is found in the books of the Bible, and it is not sufficient for every need of a speaker. Also their aim was to benefit, in their writings, uneducated people who were not proficient in Hebrew [though they did know the alphabet].[1]

This phenomenon of transcribing the spoken language with Hebrew scripts among Jewish communities is not exclusive for Jews living in Arab speaking regions. Other notable examples are:

- *Loazei Rashi* – Rashi (Rabbi Shlomo Yitzchaki) in his commentary on the Talmud and the Bible occasionally explains a term by giving its translation in Old French, his vernacular language, transcribed in Hebrew script. There are thousands of such examples, giving latter-day scholars a window into the vocabulary and pronunciation of Old French [8].

[1] Translated from the source. See `http://www.daat.ac.il/daat/vl/hovatlevavot/hovatlevavot06.pdf`.

- A second example is Yiddish, the historical language of the Ashkenazi Jews. Yiddish is a Judeo-German language, and it is written in the Hebrew alphabet. Unlike the first example, Yiddish is transcribed as full sentences, not just single words, allowing context and syntax to be examined. Yiddish is still in use in the present day in certain communities as a vernacular.

- Also Ladino, the Judeo-Spanish Romance language, was formerly written with Hebrew script.

Those languages were written *by* Jewish authors *for* Jewish readers. It can be presumed that this use of Hebrew script instead of the local one is not solely a matter of convenience, but also in certain cases a matter of discretion, that is, placing a barrier to discourage non-Jewish readers.

To end this introduction we declare and define the problem at hand, of transliterating Judeo-Arabic, as the task of generating, from the original Judeo-Arabic text, a sequence of Arab letters that corresponds to the word sequence *intended* by the author of the text. This is what we set forth to accomplish.

2 Analyzing the Problem

There is a distinction between two terms, *transliteration* and *transcription*, both constitute the transformation of text written in the script of one language into the script of another. While transliteration is a process of changing each grapheme in the source language to a grapheme in the target language, usually in a one-to-one manner, transcription is a process that seeks to preserve the phonemes, that is the way the original words sound.

The border between the two terms is not always definite. In the matter under discussion, that is Judeo-Arabic, the classification tends more to the transliteration side. The writer almost always maintains the same number of letters in the transliteration of each word. He includes the *matres lectionis* (ا، و، ي) in the transliteration, and does not include for instance explicit *nunation* (as in بًا), which is pronounced (but not written) as an "n" sound at the end of a word. Yet, the correspondence between the letters in the transliteration is not one to one. Especially marked is the absence of a mapping for occurrences of the *hamza* sign (ء) when it is placed "on the line" (and not as a decoration for one of the *matres lectionis*). This might be of interest for researchers of the formation of the Arab script system, regarding the evolution of the *hamza*. It is known to be a relative latecomer into the Arabic writing system (see [11]).

Our work is actually trying to reverse the transliteration that was used by those authors. Certain features make the problem at hand more challenging. "Pure" transliteration such as the famous *Buckwalter* transliteration (a romanization scheme for Arabic) is mathematically an injection between symbols in the source language to symbols in the target language, so reversing it is simple. However, in our case:

1. There may be different mappings used by different writers.

2. Some extra signs, namely diacritics, were omitted, either in the original manuscript or by the digital extraction method, part of which is important for disambiguation. There is a diversity in the choice of grapheme for diacritics in the Hebrew script. In some manuscripts one can see dots above or beneath and other marks. This might not be consistent between writers. In the digital text all that is left is primarily an *apostrophe* sign, so the interpretation is less granulated.

3. There is a time barrier. The data we have at our disposal for training is a current-day transliteration of the Judeo-Arabic text into Arabic. The Arabic conventions for writing in the times when those texts were written might have some differences with respect to modern Arabic.

4. There are a few symbols that do not have a one-to-one relation but a one-to-many relation in the transliteration from Hebrew scripts to Arab scripts. Hence this mapping is not a proper injection! For example the letter י is the mapping for both ي and ى.

5. There might be errors by the copier/extractor of the digital text.

Our model will have to overcome all these issues.

The *The Friedberg Jewish Manuscript Society*[1] had endeavored to release to the internet a collection of more than a hundred Judeo-Arabic works. Those are available as e-text. A most important feature they possess is an annotation of Hebrew insertions. This phenomenon of incorporating words from one language or dialect in the fluent discourse of another is prevalent and is known as *code switching*. It poses a challenge for NLP systems. In our situation a complete solution would have to identify code switches, and translate them to the destination language. We did not cover this aspect in this work. Instead we replaced all non-Arabic words in the data with a reserved symbol.

However, this annotation of the Hebrew insertion by the *Friedberg* team is not noise-free. A recurring misclassification is the annotation of Hebrew words that are prefixed with the Arab definite article ال (and also other conjunction prefixes) as being Arabic (e.g. in the token אלשכינה, which decomposes to the Hebrew word שכינה prefixed by ال). We observe that these are the reverse conditions of where the need for transliteration usually appear, which is as part of a translation system. Such a system needs to identify words that are proper nouns, and to create a matching transcription to implant at the proper place in the final translation. A consequence of this difference is that the translate/transliterate relation is reversed. Also, while those systems only require the transliteration of a subset of words, namely proper nouns, our system needs to handle all parts of speech. On the other hand, with transliteration of Judeo-Arabic texts the transition is much more natural than with the problem of proper names since here all the words originate from Arabic. For example consider the word *Israel*. When transcribed to the much different Chinese language, it is pronounced: *Yĭ-sè-li-è*. Here the transformation was influenced by the great difference between the languages, especially the lack of an equivalent to the consonant "r" which only appear at the end of syllables in Chinese.

An additional point is that unlike its complement - translation, transliteration does not need the alignment to be reordered.

This current problem of Judeo-Arabic is of course easier than the problem of transliteration of names in general since there is more regularity in the material. Also we have at our service

[1] `https://fjms.genizah.org`.

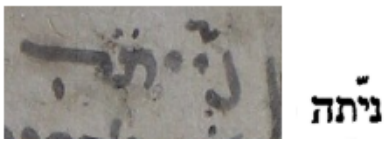
the context of each word which is in the same language as the word itself. Nevertheless the price of mistakes can be greater for the reader of the generated texts since the correct transliteration of names, places, etc., in a total body of text is not as crucial for the understanding of the whole as in our case when a single word can make the whole passage incoherent.

As was mentioned, It can be looked on as if the source text in Judeo-Arabic is itself a transliteration of Arabic to Hebrew scripts, and what we are trying to achieve is to reverse the transliteration. This "transliterator" to Hebrew had an exact knowledge of the Arabic writing system and of the Arabic language, leaving less room for ambiguities. In our case the two languages are very close to each other both being Semitic, and although a few of the Arabic consonants are not present in Hebrew, the transliterator obviously had knowledge of the Arabic writing system and tried to imitate it, and had the ability to be exact in the transliteration, a thing that can be hard for a person coming from outside (culturally), who tries to imitate the sound of the words in the source language in his own terms. It seems the way we interpret sound is also influenced by the language we speak. For instance, Japanese does not have a separate /r/ and /l/ sounds, and in transcribing names with the /r/ sound it sometimes obtains a sound that more resembles /l/. In a similar manner Arabs don't use the consonant /p/ and transliterate it /b/. Do they actually "hear" a /b/? Similarly, can we see something that we can't interpret? Think of the differences in animal sounds transcribed in children books across nations.

A closer problem is the transcription of *Arabizi*, which is a romanized transcription of Arabic that emerged naturally as a means for writing Arabic in chats. Especially in the early days of the internet the lack of support for the Arabic keyboard and problems with encoding created the need for a way to communicate written Arabic texts with Roman alphabets. This is a closer problem in the sense that whole sentences are transcribed; but it is different since this is a phonetic transcription, while we are dealing with transliteration as we distinguished the terms above. Furthermore, chat Arabizi language is colloquial and is dependent on the dialect that is used more than our texts that are more close to classical Arabic. (The Judeo-Arabic texts are also influenced by the local dialect but not at all in the same scale as Arabizi.)

As was mentioned before, some of the additional diacritics do appear in the original manuscript, and also in the printed publication. Furthermore in the published critical edition of the text there are also signs not present in the manuscript that were added by the editor such as *hamza* decoration for the letter ﻯ; though in the digital text they are not preserved. This is maybe due to extraction of the digital text by means of OCR [from the critical edition I presume, and not from the original manuscript] - OCR which was not trained to identify these extra signs. See Figure 2.1.

Hence the situation is that there is more information present in the manuscript that is not accessible digitally at this time to enable training the model for automatic transliteration. Our model will have to deal with this lack of information that the original copier may (or may not) have seen as necessary for the sake of understanding. This needs further investigation, whether or not in the original manuscript these diacritics appear constantly, or only where their absence cause ambiguity. *Shadda* especially is important for disambiguation. It is probably the most common



(a) Left: manuscript, right: printed edition, but digital-text reads: ניתא (missing *shadda* diacritic)

(b) Left: manuscript, right: printed edition, but digital-text reads: נאקצא (missing *tanwin* diacritic)

Figure 2.1: Information missing from the digital text and present in the manuscript and critical edition.

diacritic in use with written Arabic. For example: درس means "he studied", while درّس – "he taught".

There can also be an issue with generalization of the model to unseen text, especially by other copier. This is because the transliteration convention might have been different and there can be also differences in the dialect between regions. This is worsened by the current circumstances, where we only have a few data sources for supervised training at hand.

All of this means that a model that could solve the problem will have to have some knowledge and memory of Arabic words and the Arabic language to enable it to disambiguate words and to produce real words and coherent sentences.

3 Related Work

3.1 Transcription by RNN

In [10] transcription of proper names was handled with a recurrent neural network (RNN) model by two different approces. The first approach is similar to the procedure mentioned in the current work, which is a seq2seq model that deals with the unequal input and output lengths, with CTC alignment (see Section 4.4). A minor difference is their use of a method called *epsilon insertion* to deal with input sequences that are shorter than the matching output, which CTC can not handle. We instead used a similar solution of letter doubling that we will discuss later. The other approach they examined is a model inspired by recent work in the field of machine translation applying an encoder-decoder architecture with an attention mechanism. They report improved results using RNN compared to previous methods, as is usually the case when employing deep learning techniques with problem previously solved by other means.

3.2 Previous Attempts with Judeo-Arabic

In a previous work dealing with the same Judeo-Arabic texts [2], the authors tried a method that is inspired by machine translation (MT). This is the statistical MT (SMT) which was the state of the art before deep neural nets (DNN) took over. This consists of a log linear model where the main component is a phrase table that counts the number of occurrences in the train data. They also as in [10] regarded the transliteration as translation at the character level, i.e. regarding single letters as words. They improved their results by reranking their model top predictions by a word-level language model. This is expected to be beneficial since using a character-level mechanism is in danger of generating some non-words and nonsensical sequences of letters. A word-level language model which is rich enough to avoid high unknowns rates would screen those results out.

3.3 Arabizi Texts to Arab Transliteration

Transcription of *Arabizi* into Arabic graphemes was studied by [3]. Except for the different dialects and *code switches* from colloquial Arabic of different dialects to modern standard Arabic (MSA)

they also had to handle changes in the language induced by the platform it appears at, including *emoticons* and deliberate manipulation of words, such as repetition of a single letter for emphasis common with internet social medias.

The same problem was addressed in [1], which applied a mechanism that maps the Roman letters to Arabic scripts to produce a set of possibilities and chose from them by using a language model.

3.4 Transliteration of English to IPA

In this task the aim is to translate written text to the International Phonetic Alphabet (IPA), a system for phonetic notation, capturing the pronunciation of the written text. This is also not similar to the task we have on hand because this is a transcription task according to the distinction made in the start of the previous chapter. The work in [9] tried different RNN models, handling unequal sized input-output pairs by epsilon post-padding. They experimented with time delays, that is postponing the output by a few timestamps (by pre-padding the output while post-padding the input accordingly). This allows the network to catch more of the input before deciding on the output. They also compared with a bidirectional LSTM that is able to see the entire context backward and forward. They combined the bLSTM with a CTC layer. They handled the longer output then input length issue with epsilon post-padding (we used doubling). They reported, as expected, that greater contextual information contributes to performance. The bidirectional LSTM performs better than the unidirectional one even when the full context (whole word) is fed to the network before it starts its prediction. The best performance was obtained using the bLSTM approach combined with an n-gram (non-RNN) model.

This transcription from a written language to IPA can potentially be used as a mediator to transcribe between any two languages if we had an encoder from IPA to the source language and a decoder from IPA for the target script. Another potential use for this ability of transliteration to IPA would be for improved spell checking and correcting (for those that only have basic knowledge of the sounds of the letters). A model could predict the utterances from the graphemes and recognize the word by similarity in the sound domain, to get the correct spelling.
For example:

neyber → *ˈneɪbər* → *neighbor*

Next we move to describe our own methods.

4 Methods and Results

4.1 Data

4.1.1 Book of Kuzari

In order to train and evaluate our model, we need considerable amounts of parallel text in Judeo-Arabic along with their Arabic transliteration. Thankfully, a large corpus exists, which was available to us digitally. This is the Book of the Kuzari, by *Rabbi Yehuda Halevi*, which was transliterated to Arabic by Dr. Nabih Bashir from Ben-Gurion University including translation for Hebrew insertions (the code switches). Note that there is no distinction in his edition between the *transliterated* and *translated* words. The Judeo-Arabic text of the Kuzari was taken from the Friedberg website. Their text is based on the critical edition edited by David H. Baneth, and published by Haggai Ben-Shammai. It was composed from several manuscripts and contains 47,384 words, about 11% of which are Hebrew insertion. As was mentioned we don't use the Hebrew insertions in our work, and we use the annotations found in the Judeo-Arabic text to filter out those insertions.

The first step in preparing the data for training our model is aligning the Judeo-Arabic source with Dr. Bashir's transliteration. This alignment also helped remove noise from the data source, such as comments, numbering of sections, etc. We used a dynamic algorithm building the alignment gradually, at each stage choosing between: (1) adding a word from the source text, (2) adding a word from the target text, or (3) adding both as an aligned source/target pair. The decision is made by the choices that minimize the total edit distance of the alignment. In order to calculate an edit distance between strings of different scripts, we applied a simple map to the Arabic source to get a basic transliteration to Hebrew letters. This is the more reliable transliteration direction since Arabic has more letters. Then the edit distance was calculated. Aligning is necessary because Bashir's transliteration does not 100 percent match the Judeo-Arabic text taken from the *Friedberg* website. There are some word insertions, deletions and changes. The pseudo-code for the algorithm described is presented in Appendix A.

After this alignment process we only keep pairs of words in the alignment that are close enough edit-distance wise. We set the threshold for edit-distance similarity heuristically, after normalizing the edit distance result by the maximum length of the two words that were compared, to the value

of 0.5. This left us with a total of 47,083 word pairs, of which 20% were chosen randomly as test data and the rest remains for training. We are aware of the generalization problem of training and testing on data from the same source. It will be hard to estimate how the model will perform on other texts different in style. Still we will try to suggest solutions to this shortcoming in the following sections.

The Hebrew insertions are identified by the annotations available in the Judeo-Arabic text and are replaced by a single symbol 'H' both in the source and target texts. We do this in order to preserve the sentence structure, since we suspect that this information is of value to the network for making better predictions.

Punctuation signs for the training data are inserted when in either one of the languages there is a punctuation sign. We see punctuation as an important feature of the language and we want to keep this information for the model to train with. Where there are disagreements (for instance: comma vs. period) between the parallel corpus we favor the information found in the source (Judeo-Arabic) text since the final objective is prediction from Hebrew letters to Arabic ones. We might also have wanted to consider other settings, for instance keeping punctuation only when it appears in the Judeo-Arabic. This will be better for generalization for other books and might be better for practical applications. Despite the manual annotation of code switches found in *Fridberg* website, there is another direction that we explore for cleaning the training data from this code switch, i.e. Hebrew insertion. This is by the described alignment of the source and target texts using the dynamic algorithm, identifying words that do not align well, that is, words having high edit distance as explained above, and in this way removing pairs of Judeo-Arabic/Arabic words that are *translations* of each other rather than *transliterations*, that usually result in high edit distance between them.

We removed from the Arabic transliteration certain diacritics, namely the three *harakat* (short vowel marks): *fathah*, *kasrah*, *dammah* and *sukun*. They appear rarely in written Arabic (except for text intended for children or in religious texts) and are usually considered as noise for NLP research due to their scarcity [7]. In the Judeo-Arabic text, the equivalent to these diacritics is the *niqqud* signs. They appear only for the Hebrew insertions, hence they are removed incidentally by the replacement of Hebrew insertions. It is also another way to identify Hebrew insertions. Note that the *tanwīn* symbols when combined with *alif* were standardised to the modern style so the diacritic symbol appears on top of the *alif* and not on the letter preceding it.

4.1.2 Additional data: Emunoth ve-Deoth

We identified an additional source for parallel data. This is the *The Book of Beliefs and Opinions* by Rabbeinu Sa'adiah Gaon. It contains roughly the same amount of data and was also transliterated by Dr. Bashir. We decided not to use it as training data. Instead we extracted 20% of it as additional test data, for evaluating how well the model perform on unseen text.

4.1.3 Synthetic Data

Although we have a considerable amount of parallel data, since we use only one text source for training, we're facing the problem of the model fitting to the specific style of the training data and generalizing poorly for other texts. In order to make the network capacity more robust and in order to expand the vocabulary and styles it memorize, we generated some synthetic data with a simple technique, again leveraging the fact that the opposite direction of transliteration, that is from Arabic to Hebrew scripts, has almost no ambiguities. We found some texts online that correspond to the time that the texts we are interested were written. We used the mapping to produce a pseudo-transliteration to Judeo-Arabic of those texts. This gives us a parallel dataset that is genuine in the target (Arabic) side and fictitious in the source (Judeo-Arabic) side. It might include words written in a different way than the original Judeo-Arabic author would have written them. We justify the use of such training data partly by the fact that we are likewise only interested in the accuracy of our model's prediction on the target (Arabic) side. Therefore we are less concerned with showing the network examples where the input side contains noise. This significantly enlarged the amount of data available for training.

4.2 Metric

The metric we use is simply the average edit distance over all examples in the test dataset. The edit distance (in our case Levenshtein distance) is calculated between the prediction and the ground truth. It is normalized by length of the ground truth.[1] The formula for *label error rate (LER)* is

$$LER(h,S) = \frac{1}{|S|} \sum_{(x,z)\in S} \frac{ED(h(x),z)}{|z|}$$

for model h on test data $S \subseteq X \times Z$, where X are the inputs, Z is ground truth, and $|z|$ is the length of z.

This is a natural measure for a model where the aim is to produce a correct label sequence [5].

4.3 Baseline Rule-Based Transliteration

In order to evaluate our results, we start by creating a baseline transliteration on the test data that translate each Hebrew letter to the most common letter according to a predefined map. The full mapping we used is presented in Table 4.1. Producing this mapping can be automated, for instance by taking the first letters of each source/target word pair, and taking the mode of the matchings for each letter in the source. We produced it manually according to the convention of transliterating Arabic to Hebrew.

[1] The other option of normalizing by the length of the prediction would unrightfully profit models with a tendency to produce longer outputs.

Table 4.1: Simple mapping rules for baseline transliteration.

Judeo-Arabic	Arabic	Judeo-Arabic	Arabic
א	ا	כ	ك
ב	ب	כ'	خ
ג'	ج	ל	ل
ג	ع	מ	م
ד	د	נ	ن
ד	ذ	ס	س
ה	ه	ע	ع
ה'	ة	פ	ف
ו	و	צ	ص
ז	ز	צ'	ض
ח	ح	ק	ق
ט	ط	ר	ر
ט'	ظ	ש	ش
י	ي	ת	ت
		ת'	ث

This simple mapping achieves a relatively high accuracy (error rate 9.51%) and demonstrates the nature of this Judeo-Arabic transliteration problem. Though it is easy to achieve high accuracy, in order to produce readable text and to be able to confront ambiguities in the text, some language ability is desirable. The baseline results still do not guarantee fluent reading of the generated target text.

4.3.1 Common Baseline Mistakes

The baseline errors come mainly from ambiguous letters that have more than a single possibility for mapping. We will enumerate the prominent ambiguities.

4.3.1.1 Transliteration of the Hebrew Letter alef

The letter *alef* (א) most commonly should be transliterated as the Arabic letter *alif* (ا). This Arabic grapheme usually indicates an elongated /a/ vowel attached to the preceding consonant. However, it can also sometimes indicate a glottal stop, that is an *alif with hamza on top* (أ). As [7] mentions Arabic writers often ignore writing the *hamza* (especially with stem-initial *alifs*) and it is "de-facto optional". This will also lead to false negatives for the test data, deciding a transliteration is an error while in fact it would be accepted by a human reader, unjustifiably increasing the error (see [10, Section 2.3]).

A more complex model could hopefully predict the places were *hamza* is required for disambiguation (for instance words with stem-**non**-initial *alifs*). Alternatively, such a model would hopefully have a rich enough memory of the Arabic words it has seen, attaching the *hamza* sign for words it has seen in the training data. If indeed there are two legitimate forms, this model will also know to disambiguate according to the context, as we train on sequences, that is, words in context.

Rarer cases for transliteration of א are as follows: *hamza on the line* (ء), even though it is usually not transcribed in the Hebrew. Also it can mean an *alif maqsura* (ى) at the end of a word, but *alif maqsura* is usually mapped to the letter *yod* (י). See for instance the 3-letter word וגא that is transliterated as the 4-letter وجاء. In this example it is not clear whether the letter א corresponds to the *hamza* or to the long vowel *alif* that precedes it, in which case the *hamza* is missing from the transliteration. In the baseline we map א to the most common transliteration, i.e. a non-*hamza alif*. Thus we are missing all the other variations.

4.3.1.2 Transliteration of the Hebrew Letter yod

The two most common uses of the *yod* (י) transliteration are (1) the Arabic letter *ya* (ي) and (2) *alif maqsura* (ى), which is shaped as a dotless *ya* and appear always at the end of a word. Less frequently it can also be a transliteration of *ya hamza* (ـئ). But there are variations. For instance, in the first of the examples in Table 4.2, the *ya hamza* is transliterated as *yod* (י), while in the second when followed by a regular *ya*, it can be seen as either transliterated to a Hebrew *alef* (א) or dropped. In this example the variation can be due to the spelling in Hebrew of the translation that uses א: "ישראל". As mentioned in [7], in Egypt, but not in other Arabic countries necessarily, a final *ya* is often written dotless, that is as an *alif maqsura*. This seems to be the case in the transliteration of *Maimonides* book *The Guide for the Perplexed*, as transliterated by Hussein Attai[2]. Unluckily it is not available as digital text to enable using as training data. See Appendix B. In the baseline we map *yod* invariably to a regular *ya*.

[2] http://sepehr.mohamadi.name/download/DelalatolHaerin.pdf (text start page 45).

سُئِلْتُ	סילת
إسرائيل	אסראיל

Table 4.2: Different transliterations of ـئ. In the first row it matches Hebrew י, while in the second row it matches א. The unusual transliteration might stem from the spelling of the translation.

4.3.1.3 Shadda (Gemination)

Another critical issue is the *shadda* diacritic (e.g. بّ). This is not present in the Judeo-Arabic text, or was omitted in the digitized version of the text as described above. There is no simple rule of thumb regarding the presence of absence of shadda that could be adopted for the baseline.

4.3.2 Baseline Results

The baseline fully disregard *shadda* signs, *hamza*, *alif maqsura*, *tanwin (nunation)* and more. Nonetheless the error on the test data does not increase (LER 9.5%). Figure 4.1 depicts some results from the baseline system.

```
0.0500 | بالروحانيات وان يوصف | بالروحانيات وأن يوصف | באלרוחאניאת ואן יוצף
0.0000 | مخاطبة الله له فيقفون | مخاطبة الله له فيقفون | מכ'אטבה' אללה לה פיקפון
0.2000 | ير فيها اثر الاهي مع | ير فيها أثر إلهيّ مع | יר פיהא את'ר אלאהי מע
0.0741 | لها قابل مع الشرايط الشرعيه | لها قابل مع الشرائط الشرعية | קאבל מע אלשראיט אלשרעיה
0.2400 | الي النبوه كل من استعد | إلى النبوّة كلّ من استعدّ | אלי אלנבוה כל מן אסתעד
```

Figure 4.1: Baseline results. The column order is as follows (right to left): Judeo-Arabic text, ground truth, baseline prediction, error rate. The errors are marked in blue for convenience. Observe for instance the 4th word in the 3rd row, the lack of the *shadda* in the prediction and the extra א in the source sentence.

4.4 Training using CTC Loss with Letter Doubling

The connectionist temporal classification (CTC) loss is a method introduced in [5] that enables a neural model to learn to predict discrete labels from a continuous signal without requiring the train data to be aligned input to output. Instead the model produces a distribution over all alignments of all possible labels while facilitating an extra character (the *blank* symbol) added to its softmax layer in order to produce the alignment. Thus the probability of any label conditioned on the input can be calculated as the sum over all possible alignments of the given label. CTC is appropriate for

our mission of Judeo-Arabic transliteration because the Judeo-Arabic inputs and Arabic outputs are not always of equal length. Also there is no available alignment of the two texts at the character level.

Applying the CTC loss is a convenient solution for handling this problem. But CTC loss is only defined when the input signal is longer than the output, which is not always the case in our dataset. Actually, since there are diacritic signs in the Arabic transliteration included in the character count that do not appear in the Judeo-Arabic source, this is a prevalent situation. This requires an adaptation of the dataset in some way that we describe in the next subsection.

It should be noted that, as [4] mentions, the CTC mechanism implicitly assumes the independence of the characters over time. By using the multiplicative rule between probabilities over time it disregards the dependencies between timestamps. In spite of this strong assumption, CTC was able to achieve a substantial improvement in performance for various sequential tasks, such as speech recognition [5], OCR [12] and handwriting recognition [6].

4.4.1 Doubling

This technique of dealing with shorter input than output sequences that we used is similar to the one used by [10]. But instead of using a special character (epsilon) inserted between each timestamps a constant number of times, known as *epsilon insertions*, we filled the extra spaces by repetition of the previous timestamp label. For instance, the sequence:

עלי מא שהד וג'א פי

would become:

עעללי״ ממאא ששההדד ווגג״אא ppפפי״

Note the doubling of the *tag* sign which is performed separately from the letter ג which it decorates.

Aesthetically speaking, this brings the usage of CTC closer to its original usage of transcribing a continuous signal to discrete labeling (e.g. in speech recognition) in the sense that the input text obtains at least the appearance of a continuous signal, while epsilon insertions would break the succession. It can be seen as an approximation of a continuous signal. More work is needed to examine the impact of these two options on performance.

4.5 Training the RNN

We used for our model GRU cells, stacked in four layers, followed by a softmax layer for the CTC loss. Each layer was bidirectional (meaning the cells observed the input both backwards and forwards). Each layer contained 1024 units. We used letter embedding for the input of dimension

8. The model was implemented using TensorFlow, and trained using the RMSprop optimizer. The text was divided into short 20 characters sequences (according to the lengths in the input side). The sequences contained only whole words. If the count of the 20th character happened to be mid-word the rest of the word was precluded from the sequence. Batch size was 128.

4.6 Pretraining on Single Letters

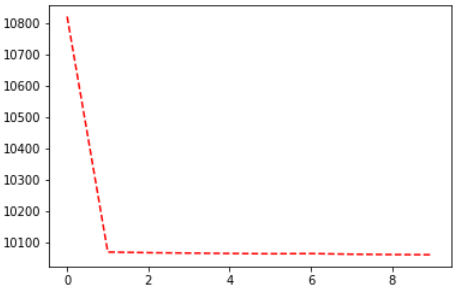
A method that was beneficial for speeding up convergence of the network was to pretrain the network with single letters according to the simple letter mapping between the Hebrew and Arabic alphabet. This is as if "to set the model in the right direction". This is intuitively reasonable if you think of the way children learn to read. First they learn to identify the individual letters and then to assemble them into words. We used for this training the *sparse cross entropy loss* trained on generated random parallel char sequences of length 10. As we will show, this makes it feasible to train deeper networks. It might also be helpful for instance in the task of speech recognition; to pretrain the network first on single time samples from a certain phoneme to teach the network to map to the correct grapheme, before training on continuous speech with CTC loss which is more complex, and with which it is less obvious for the network how to start to optimize than with the simpler cross entropy loss.[3]

4.6.1 Intermediate Results

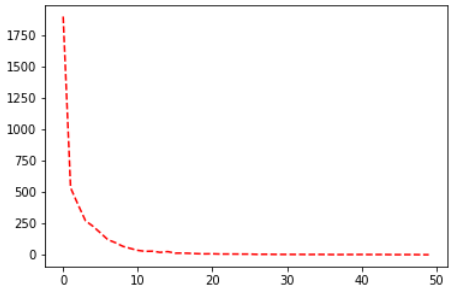
With pretraining with the cross entropy loss the network converges quite fast to reach error rate of 2% on the test data. On the other hand, without pretraining the network become stuck at a local minimum, transliterating each char in the input to the most prevalent char in the target data which is a space character, producing as output strings of repeating spaces. Figure 4.2 shows the losses and accuracy measures. In the rest of the experiments we include this pretrain stage.

Running this model against the additional text of *Emunoth ve-Deoth* yields an higher error rate of 3.24%. So we see a decrease in performance for unseen texts. We continue this section by exploring ways for improving prediction for unseen texts.

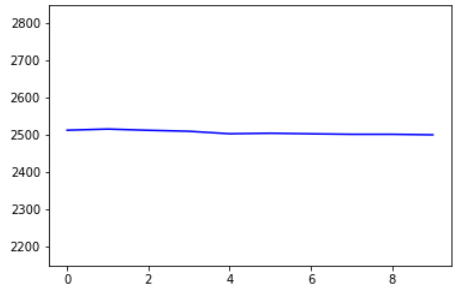
[3] Technical note: since we use the cross entropy loss we train only on variants of the Hebrew letters without *apostrophes* while mapping both to the transliteration of the Hebrew letter with apostrophe and without. For instance, the Hebrew letter ג maps in the pretrain stage to both غ – its transliteration, and ج, the transliteration of ג'. This is enough for our purpose.



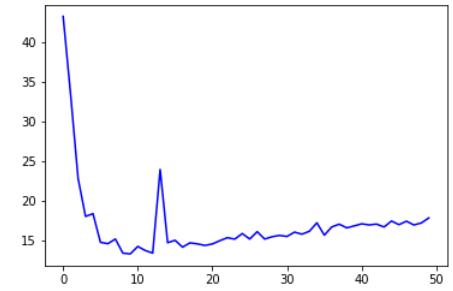
(a) Training loss. no pretrain



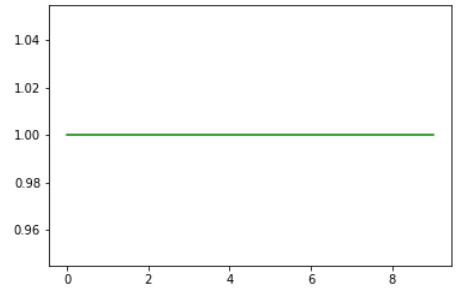
(b) Training loss. with pretrain



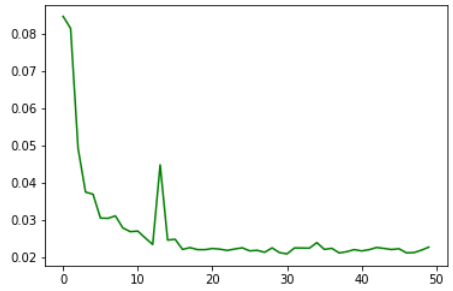
(c) Test loss. no pretrain



(d) Test loss. with pretrain



(e) Test accuracy. no pretrain



(f) Test accuracy. with pretrain

Figure 4.2: Comparing results with pretraining with single graphemes with CE loss (right) and without (left). Notice the differences in the graph scales.

4.7 Training with Synthetic Data

As described in Section 4.1.3 we propose a technique to augmented the amount of training data by generating pseudo parallel texts using Arabic writing of roughly the same era that our Judeo-Arabic text were written. We do this by transliterating them into Hebrew letters according to the simple mapping (see Table 4.1) to get the input for the network while using the original Arabic as output. For instance, for the first two words in the text of *Ilāhiyyāt*:

كتاب الشفاء

a pseudo-Judeo-Arabic transliteration will be generated according to Table 4.1 that was used for the baseline:

כתאב אלשפא

Now, the generated Judeo-Arabic together with its Arabic counterpart will be added to the parallel data for training.

The texts we used for this purpose in the current experiments are: (1) Avicenna's *Ilāhiyyāt*, (2) Al-Farabi's *Kitab Rilasa al-Huruf*, (3) Al-Farabi's *Kitab Tahsil al-Saida* and (4) Averroes's *Al-Darurī fī Isul al-Fiqh*. To this we add the original "real" dataset of the *Kuzari*. By this we enlarge significantly the amount of training data.

4.7.1 Results, Continued

With the synthetic data, the accuracy on the original text data *decreased* as expected to 2.48%. On the additional data the accuracy also *decreased* but to a lesser extent to 3.37%. This drop in performance might indicate that the Arabic texts that we chose for the production of the synthetic data are not exactly suitable for the Judeo-Arabic. Maybe there are other sources to consider that are more suitable. Note that the amount of fluctuation on the test is greater. This might be because the synthetic data is shuffled between epochs. The results are presented in Figure 4.3.

4.8 Dropout

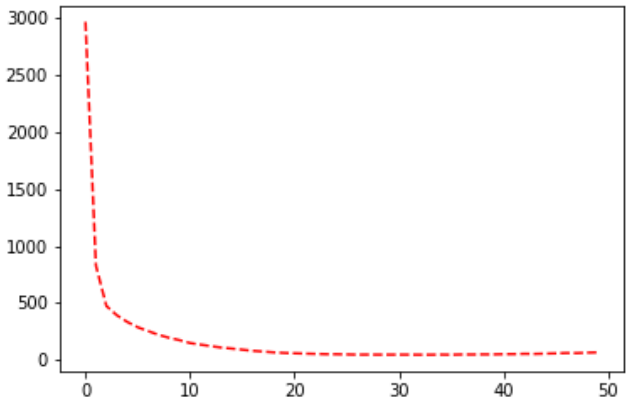
As mentioned, we are mainly interested in the network having memory of what would make sensible Arabic outputs, relying on the letter mapping, but allowing some flexibility. The network should be able to output Arabic words that were seen as output in the training data even though the input sequence is not the exact sequence that appeared in the training data. Experimenting with a small model showed the plausibility of this direction of thinking. An immediate implication was to use *dropout* on the input sequence to make the network more robust to noise in the Judeo-Arabic input. We achieved this by randomly setting a percentile of the non-space symbols in the input sequence (before doubling the letters) to the *blank* symbol.

4.8.1 Results

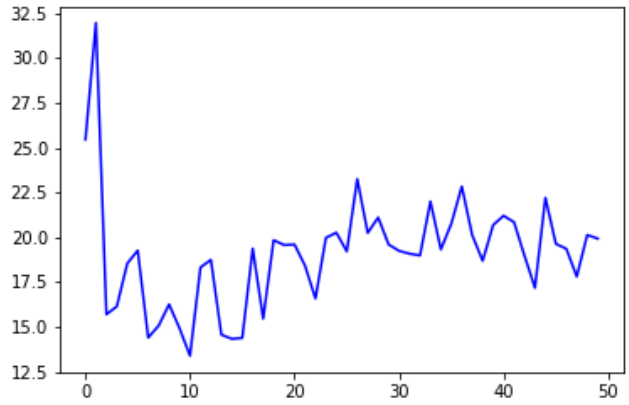
With dropout rate of 15% of the non-space symbols that was performed on the synthetic data, the test results for the original test of the *Kuzari* is set at 2.26%, which is worse than the accuracy without synthetic data and without dropout but better than the results when trained on the synthetic data without the dropout. On the other hand, for the additional text of *Emunot* an improvement is marked achieving an error rate of 3.14% (a 0.1 percent improvement).

4.9 Does the Network Learn Language Skills?

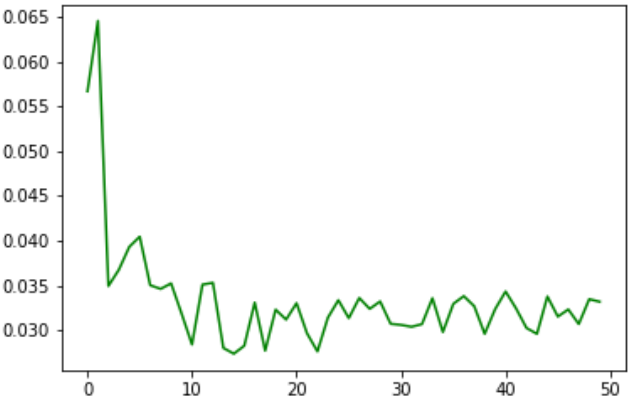
One question that arises is how much of the language the network catches. In order to examine this, we perform a forward pass on the shuffled test data. The shuffling is done at the word level. This generates a parallel dataset of words that lack context, sharing the same word distribution as the real test data. Indeed as expected, on the shuffled test data the error increases by $\sim$0.5% on average.



(a) Training loss



(b) Test loss



(c) Test accuracy

Figure 4.3: Results of training with added synthetic data.

5 Discussion

We designed a model for automatic transliteration of Judeo-Arabic texts into Arabic scripts. Endeavoring to overcome the problem of ambiguous mappings in the transliteration, we trained an RNN model using the CTC loss that has enabled us to cope with unequal length input/output sequences due to the addition of diacritics in the target (Arabic) side of the data. As mentioned, we wanted to create a network that will have memory, along with some language capability in the output side that will enable it to distinguish between different word senses and overcome small variations in the transliteration.

The results show a decrease in error compared to the baseline results from 9.5% to 2% LER, showing that the network is capable of attaching correct diacritics to the Arabic output. On unseen text the network incurs a 3.24% error rate. This is important since Judeo-Arabic texts vary according to the writer, the period, and the region.

Example results and some remaining problems are shown in Figure 5.1.

- For instance, in Example 5.1b with the last word the network misses the ground truth. Still, notice that it seems to struggle with it by producing a transliteration different from the simple rule based transliteration (الريا), suggesting that it does catch some language regularities and tries to imitate them on unseen words. Notice also the correct positioning of the *Alif Hamza*.

- In Example 5.1d the mistake is partly due to noise in the data since the ground truth form: موصياً is a conjugation of the noun موصي (meaning "recommender"), which the Judeo-Arabic מוצי form resembles more. The form موصّي (added shadda) that the network finally produces also exists.

- Example 5.1i shows a mistake in distinguishing between word senses, since both diacritizations of انا produce legitimate words.

We experimented with methods to enhance the training. First we exercised pretraining of the network with a different loss (cross entropy) to teach the network the simple mapping used in the baseline transliteration. This can enlarge and deepen the network and still guarantee convergence.

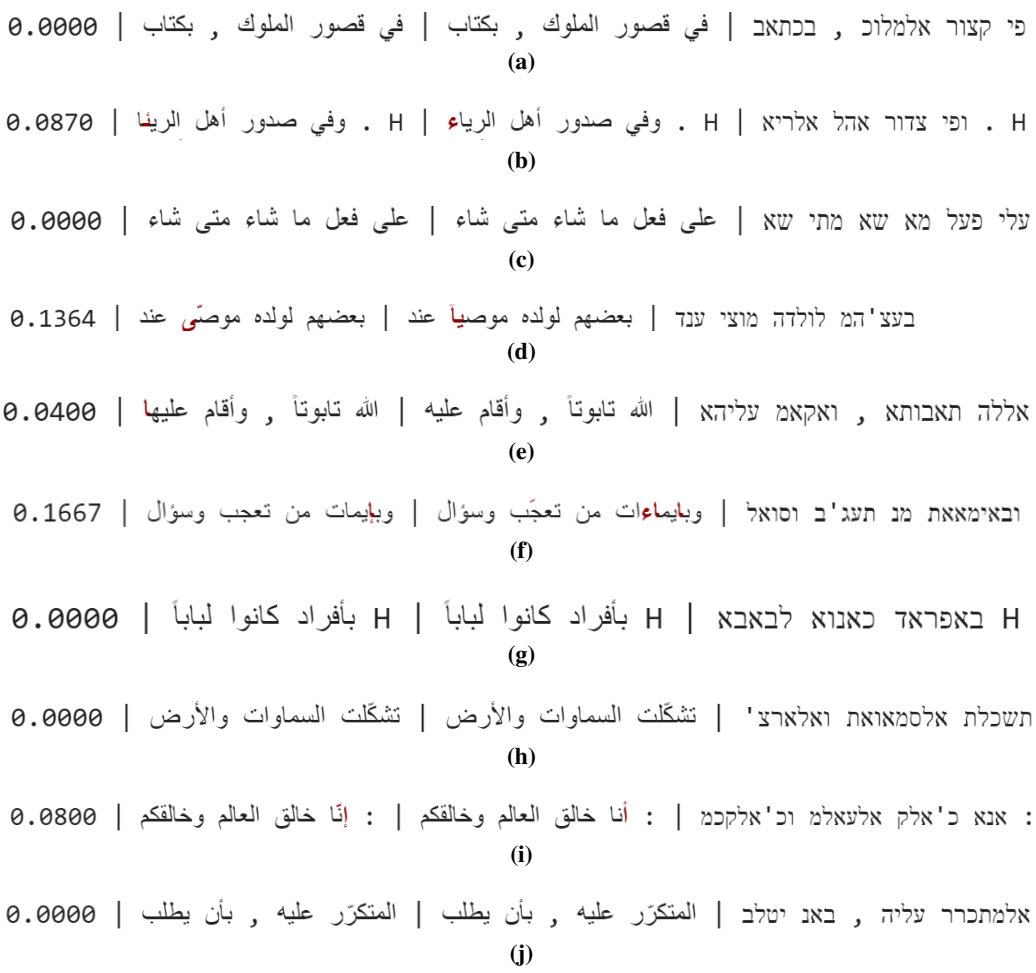


Figure 5.1: Example transliteration results for proposed model. The column order is as follow (right to left): Judeo-Arabic text, ground truth, baseline prediction, error rate.

With the size of network that we used, without this pretraining stage, the network failed to discover this mapping by itself. Since we only train on parallel text from a single source, and we are interested in making the model generalize better to unseen texts, we augment the training dataset by generating parallel texts out of available Arabic texts from the Middle Ages, the time when the Jewish works we are dealing with were written. This had a slight reverse affect on accuracy and might depend on the choice of Arab texts that were used for the synthetic data. Dropout was thought of, and implemented on the synthetic data. This had a desired affect. While accuracy on the first test data, taken from the same source as the original train data decrease as expected, the accuracy for the additional test data improved. We suspect that the network assimilates more language skills due to the dropout on the training data, and suggest this as a direction for further experiments.

Next we will deal with some ideas for improvement of the existing model that were not imple-

mented.

5.1 Re-rank Top Transliterations Using a Model Language

Examining the top 5 results of the CTC-decode beam search, instead of only the top one, reveals that sometimes the correct transliteration, at least the one that was chosen by the human transliterator, is present among those top results. As was done in previous works, we believe that choosing between those top decodings by running an Arab language model could enhance the model accuracy.

5.2 Test Accuracy with a Context Window

Since we transliterate sequences of several words, we utilize the context of each word in predicting its correct transliteration. This is available only for words that are not the first and last in the sequence. We propose to change the setting of the testing to regard only words inside a fixed context window, and to calculate the loss and accuracy only on words that have context. We suspect this will reveal that errors occur more frequently at the start or end of a sequence. Another plan will be to test only on a single middle word inside a constant size context window. It will be interesting to check how results change with different window sizes, including a window size of *zero*, which will be another way to check how much of the language skills the model learns, a model that is not using explicitly a language model.

References

[1] Mohamed Al-Badrashiny, Ramy Eskander, Nizar Habash, and Owen Rambow. Automatic transliteration of romanized dialectal Arabic. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 30–38, 2014.

[2] Kfir Bar, Nachum Dershowitz, Lior Wolf, Yackov Lubarsky, and Yaacov Choueka. Processing Judeo-Arabic texts. In *ACLing*, pages 138–144, 04 2015. doi: 10.1109/ACLing.2015.27.

[3] Ann Bies, Zhiyi Song, Mohamed Maamouri, Stephen Grimes, Haejoong Lee, Jonathan Wright, Stephanie Strassel, Nizar Habash, Ramy Eskander, and Owen Rambow. Transliteration of Arabizi into Arabic orthography: Developing a parallel annotated Arabizi-Arabic script SMS/chat corpus. In *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*, pages 93–103, 2014.

[4] William Chan, Navdeep Jaitly, Quoc V. Le, and Oriol Vinyals. Listen, attend and spell. *CoRR*, abs/1508.01211, 2015. URL `http://arxiv.org/abs/1508.01211`.

[5] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 369–376. ACM, 2006.

[6] Alex Graves, Santiago Fernández, Marcus Liwicki, Horst Bunke, and Jürgen Schmidhuber. Unconstrained on-line handwriting recognition with recurrent neural networks. In *NIPS*, volume 20, 01 2007. doi: 10.1057/9780230226203.3287.

[7] N.Y. Habash. *Arabic Natural Language Processing*. Synthesis Digital Library of Engineering and Computer Science. Morgan & Claypool Publishers, 2010. ISBN 9781598297959. URL `https://books.google.co.il/books?id=kRIHCnC74BoC`.

[8] Raphael Levy. The background and the significance of Judeo-French. *Modern Philology*, 45(1):1–7, 1947. ISSN 00268232, 15456951. URL `http://www.jstor.org/stable/435416`.

[9] Kanishka Rao, Fuchun Peng, Haşim Sak, and Françoise Beaufays. Grapheme-to-phoneme conversion using long short-term memory recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4225–4229. IEEE, 2015.

[10] Mihaela Rosca and Thomas Breuel. Sequence-to-sequence neural network models for transliteration. *CoRR*, abs/1610.09565, 2016. URL `http://arxiv.org/abs/1610.09565`.

[11] Mehdy Shaddel. Traces of the hamza in the early Arabic script: The inscriptions of Zuhayr, Qays the Scribe, and 'Yazd the King'. *Arabian Epigraphic Notes*, 4:35–52, 2018.

[12] Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *CoRR*, abs/1507.05717, 2015. URL `http://arxiv.org/abs/1507.05717`.

A Alignment Algorithm

We used the following dynamic algorithm to align the Judeo-Arabic and Arabic texts.

For:
$T_i^{[0,\ldots,l_i-1]} = \left(t_i^0, t_i^1, \ldots, t_i^{l_i-1}\right)$ – word sequence with length l_i
ε – the empty string
ϕ – an empty sequence

Define:
$AlignScore\left(T_1^{[0,\ldots,l_1-1]}, T_2^{[0,\ldots,l_2-1]}\right)$

$$= \begin{cases} 0, & \text{if } l_1 = 0 \wedge l_2 = 0 \\ AlignScore\left(\phi, T_2^{[0,\ldots,l_2-2]}\right) + editDist\left(\varepsilon, t_2^{l_2-1}\right), & \text{if } l_1 = 0 \\ AlignScore\left(T_1^{[0,\ldots,l_1-2]}, \phi\right) + editDist\left(t_1^{l_1-1}, \varepsilon\right), & \text{if } l_2 = 0 \\ \min\left\{ \begin{array}{l} AlignScore\left(T_1^{[0,\ldots,l_1-2]}, T_2^{[0,\ldots,l_1-2]}\right) + editDist\left(t_1^{l_1-1}, t_2^{l_2-1}\right), \\ AlignScore\left(T_1^{[0,\ldots,l_1-2]}, T_2^{[0,\ldots,l_2-1]}\right) + editDist\left(t_1^{l_1-1}, \varepsilon\right), \\ AlignScore\left(T_1^{[0,\ldots,l_1-1]}, T_2^{[0,\ldots,l_1-2]}\right) + editDist\left(\varepsilon, t_2^{l_2-1}\right) \end{array} \right\}, & \text{otherwise} \end{cases}$$

Remarks:

1. Memoizing at each stage what choice was made by the *minimum* operator, enables to track back the desired alignment. In order to implement this algorithm in a space non-consuming manner only the alignment scores of two levels of recursion where kept. This is sufficient since the *minimum* operator operates only one level deep. Actual implementation was done by an iterative version.

2. The *editDist* maps both string to Judeo-Arabic according to the simple map from Table 4.1, before calculating the distance.

3. The *editDist* on the empty string is equivalent to the length of the other string.

B Example Transliteration of Unseen Text

```
כנת איהא אלתלמיד' אלעזיז ר' יוסף ש"צ ב"ר
יהודה נ"ע למא מת'לת ענדי וקצדת
מן אקאצי אלבלאד ללקראה עלי , עט'ם שאנך ענדי
לשדה חרצך עלי
אלטלב ולמא ראיתה פי אשעארך מן שדה' אלאשתיאק
ללאמור אלנט'ריה וכאן ד'לך מנד' וצלתני רסאילך
ומקאמאתך מן
אלאסכנדריה קבל אן אמתחן
תצורך וקלת לעל שוקה אקוי מן אדראכה פלמא קראת
עלי מא קד
קראתה מן עלם אלהיאה ומא תקדם לך ממא לא בד מנה
תוטיה להא מן אלתעאלים
זדת בך גבטה לג'ודה' ד'הנך וסרעה' תצורך וראית
שוקך ללתעאלים
עט'ימא פתרכתך ללארתיאץ' פיהא לעלמי במאלך.
פלמא קראת עלי מא קד קראתה מן צנאעה' אלמנטק
תעלקת אמאלי בך
וראיתך אהלא לתכשף לך אסראר אלכתב אלנבויה חתי
תטלע מנהא עלי מא ינבגי
אן יטלע עליה אלכאמלון פאכ'ד'ת אן אלוח לך
תלויחאת ואשיר לך באשאראת
פראיתך תטלב מני אלאזדיאד וסמתני אן אבין לך
אשיא מן אלאמור
```

Figure B.1: First page of Maimonides' *The Guide for the Perplexed* in the original Judeo-Arabic orthography.

2 كنت أيها التلميذ العزيز ر يوسف شص بر
3 يهوده نع لما مثلت عندي وقصدت
4 من أقاصي البلاد للقراه على , عظم شأنك
عندي لشدة حرصك على
5 الطلب ولما رأيته في أشعارك من شدة
الاشتياق للأمور النظريه وكان ذلك منذ
وصلتني رسائلك ومقاماتك من
6 الاسكندريه قبل إن أمتحن
7 تصورك وقلت لعل شوقه أقوى من إدراكه فلما
قرات على ما قد
8 قراته من علم الهياه وما تقدم لك مما لا
بد منه توطئة لها من التعاليم
9 زدت بك جبطه لجودة ذهنك وسرعة تصورك
ورأيت شوقك للتعاليم
10 عظيما فتركتك للارتياض فيها لعلمي بمالك
11 فلما قرات على ما قد قراته من صناعة
المنطق تعلقت إمالي بك
12 ورايتك أهلا لتكشف لك أسرار الكتب النبوية
حتى تطلع منها على ما ينبغي
13 أن يطلع عليه الكاملون فاخذت أن الوح لك
تلويحات وأشير لك بإشارات
14 فرايتك تطلب مني الأزدياد وسمتني أن أبين
لك أشياء من الأمور

Figure B.2: Our model transliteration of the first page of Maimonides' *The Guide for the Perplexed* (final model trained with synthetic data).

(١٨٩-ب) ج

بســـم الله رب العالم [1]

كنت ايها التلميذ العزيز الرّبيّ يوسف، صانك الصخرة، ابن الرّبيّ يهودا، سكنت نفسه جنة عدن [2]، لما مثلت عندى [3] وقصدت الىّ [4] من اقاصى البلاد للقراءة علىّ، عظم شأنك عندى لشدة حرصك على الطلب، ولِما رأيت [5] فى اشعارك ومقاماتك التى وصلتنى — وانت مقيم بالاسكندرية — من شدة الاشتياق للامور النظرية. وقبل أن امتحن تصورك قُلت [6] : لعل شوقه اقوى من إدراكه، فلما قرأت علىّ ما قد [7] قرأته من علم الهيئة، وما تقدم لك مما لابدّ منه توطئة لها من التعاليم [8]، زدت بك غبطة لجودة ذهنك وسرعة تصورك ورأيت شوقك للتعاليم [9] عظيما، فتركتك للارتياض فيها لعلمى بمآلك [10].

فلما قرأت علىّ ما قد قرأته [11] من صناعة المنطق، تعلقت آمالى بك ورأيتك اهلا لتُكشف لك أسرار الكتب النبوية حتى تطّلع منها على ما ينبغى ان يطّلع عليه الكاملون؛ فاخذت الوّح [12] لك تلويحات واشيرلك باشارات فرأيتك تطلب منى الازدياد | وتسومنى [13] أن أبيّن لك اشياء من الامور (٢ - ب) م

Figure B.3: First page of Maimonides' *The Guide for the Perplexed* as transliterated by Hussein Attai.

תקציר

רבות מהיצירות היהודיות החשובות בימי הביניים נכתבו בשפה הערבית יהודית. זהו ניב ערבי שהיה בשימוש היהודים החיים בקרב דוברי ערבית, אשר עושה שימוש באלף בית העברי כמערכת כתיבה עבור ניב זה אשר דומה בעיקרו לשפה הערבית הקלאסית. בעבודה זו אני פועלים לייצר מכונה אוטומטית לתעתוק של הניב היהודי ערבי למער' הכתיבה הערבית, ובכך נשאף לאפשר לקוראים ערבים המתעניינים בפיסלוסופיה של ימי הביניים גישה לכתבים אלו. אנו מאמצים לצורך זאת גישה של למידה עמוקה, בפרט מודל רשת נוירונים סדרתית (RNN), תוך שילוב של פוני עלות המאפשרת ישור אוטומטי של טקסט מקבילי המכונה CTC. בחירה זו מכריחה התאמה של הטקסטים הזמינים לאימון. יש להאריך קלטים שאינם ארוכים כאורך הפלטים המתאימים להם. עשינו זאת ע"י הכפלה כוללת של האותיות בצד הקלט לקבלת סיגנל ארוך. כמו כן כיוון שברשותנו רק מקור אחד עבורו קיים טקסט מקבילי בשתי השפות, בחנו אפשרות להגדלת כמות המידע לאימון ע"י ייצור אוטומטי של טקסטים מלאכותיים מתוך כתבים ערבים שהם בערך מאותה התקופה של הטקסטים המדוברים. אנו עושים זאת תוך ניצול של העובדה כי התעתיק שבו השתמש המחבר המקורי מקיים יחס של אחד לרבים מיהודית-ערבית לערבית. כלומר ההתאמה ההפוכה מהווה פוני מוגדרת. ע"י כך אנו מנסים לבנות מודל שיאגור בתוכו זכרון לאוצר המילים הערבי ולצורת הכתיבה שלהן, ואשר יעזר בהקשר של המילים על מנת לבדל בין משמעויות שונות של מילים הנכתבות באופן זהה בערבית היהודית אשר לפנינו. אנו מגששים לראות עד כמה הרשת אכן משתמשת בידע זה ע"י בדיקה שניה של המודל על מקבץ ההערכה, אך הפעם באופן שהמילים מעורבלות, ולכן מחוסרות הקשר. באמצעות המודל שלנו הצלחנו לשפר את התוצאות של תעתוק הבסיסי באמצעות כללים קבועים משגיאה של 9.5% ל2%. כאשר בבדיקה מחוסרת ההקשר התוצאות יורדות ל2.5% שגיאה.

תעתוק אוטומטי של כתבים מערבית יהודית לערבית באמצעות מודל RNN

חיבור זה הוגש כעבודת מחקר לקראת התואר
"מוסמך אוניברסיטה" במדעי המחשב
על ידי
אורי טרנר

העבודה הוכנה בהנחייתו של
פרופ' נחום דרשוביץ

אוניברסיטת תל אביב
בית הספר למדעי המחשב
תשרי תש"פ