Automatic Romanization of Arabic Bibliographic Records

Fadhl Eryani and **Nizar Habash**
Computational Approaches to Modelling Language (CAMeL) Lab
New York University Abu Dhabi, UAE
`{fadhl.eryani,nizar.habash}@nyu.edu`

Abstract

International library standards require cataloguers to tediously input Romanization of their catalogue records for the benefit of library users without specific language expertise. In this paper, we present the first reported results on the task of automatic Romanization of *undiacritized* Arabic bibliographic entries. This complex task requires the modeling of Arabic phonology, morphology, and even semantics. We collected a 2.5M word corpus of parallel Arabic and Romanized bibliographic entries, and benchmarked a number of models that vary in terms of complexity and resource dependence. Our best system reaches 89.3% exact word Romanization on a blind test set. We make our data and code publicly available.

1 Introduction

Library catalogues comprise a large number of bibliographic records consisting of entries that provide specific descriptions of library holdings. Records for Arabic and other non-Roman-script language materials ideally include Romanized entries to help researchers without language expertise, e.g., Figure 1. There are many Romanization standards such as the ISO standards used by French and other European libraries, and the ALA-LC (American Library Association and Library of Congress) system (Library of Congress, 2017) widely adopted by North American and UK affiliated libraries. These Romanizations are applied manually by librarians across the world – a tedious error-prone task.

In this paper, we present, to our knowledge, the first reported results on automatic Romanization of *undiacritized* Arabic bibliographic entries. This is a non-trivial task as it requires modeling of Arabic phonology, morphology and even semantics. We collect and clean a 2.5M word corpus of parallel Arabic and Romanized bibliographic entries, and evaluate a number of models that vary in terms

Description	Tag	Romanized Entry	Arabic Entry
Author *details*	100a*	Taymūr, Aḥmad,	تيمور، احمد،
	100d	1871-1930,	،1871-1930
	100e	author.	مؤلف.
Title *details*	245a*	Qabr al-imām al-Suyūṭī wa-taḥqīq mawḍiʿihi /	قبر الإمام السيوطي وتحقيق موضعه /
	245c*	bi-qalam al-faqīr ilayhi taʿálá Aḥmad Taymūr.	بقلم الفقير إليه تعالى أحمد تيمور.
Place of *production*	264a*	al-Qāhirah :	القاهرة :
	264b*	al-Maṭbaʿah al-Salafīyah wa-Maktabatuhā,	المطبعة السلفية ومكتبتها،
	264c	1346 [H.], [1927? M.]	1346 [هـ]، [1927؟ م.]
Physical *description*	300a	24 pages, 1 unnumbered leaf of plates :	24 صفحة، 1 ورقة لوحات غير مرقمة :
	300b	illustrations, maps	مصورات، خرائط ؛
	300c		24 سم
Subject - *name*	600a*	Suyūṭī,	السيوطي،
	600d	1445-1505.	.1445-1505
Subject - *topic*	650a	Tombs	القبور
	650z	Egypt	مصر
	650z	Cairo.	القاهرة.

Figure 1: A bibliographic record for Taymūr (1927?) in Romanized and original Arabic forms.

of complexity and resource dependence. Our best system reaches 89.3% exact word Romanization on a blind test set. We make our data and code publicly available for researchers in Arabic NLP.[1]

2 Related Work

Arabic Language Challenges   Arabic poses a number of challenges for NLP in general, and the task of Romanization in particular. Arabic is morphologically rich, uses a number of clitics, and is written using an Abjad script with optional diacritics, all leading to a high degree of ambiguity. The Arabic script does not include features such as capitalization which is helpful for NLP in a range of Roman script languages. There are a number of enabling technologies for Arabic that can help, e.g., MADAMIRA (Pasha et al., 2014), Farasa (Abde-

---

[1] `https://www.github.com/CAMeL-Lab/Arabic_ALA-LC_Romanization`

Split	Bib Records	Entries	Words
Train	85,952 (80%)	479,726	∼2M
Dev	10,744 (10%)	59,964	∼250K
Test	10,743 (10%)	59,752	∼250K
Total	107,439	599,442	∼2.5M

Table 1: Corpus statistics and data splits.

lali et al., 2016), and CAMeL Tools (Obeid et al., 2020). In this paper we use MADAMIRA to provide diacritics, morpheme boundaries and English gloss capitalization information as part of a rule-based Romanization technique.

Machine Transliteration *Transliteration* refers to the mapping of text from one script to another. Romanization is specifically transliteration into the Roman script (Beesley, 1997). There are many ways to transliterate and Romanize, varying in terms of detail, consistency, and usefulness. Commonly used name transliterations (Al-Onaizan and Knight, 2002) and so-called Arabizi transliteration (Darwish, 2014) tend to be lossy and inconsistent while strict orthographic transliterations such as Buckwalter's (Buckwalter, 2004) tend to be exact but not easily readable. The ALA-LC transliteration is a relatively easy to read standard that requires a lot of details on phonology, morphology and semantics. There has been a sizable amount of work on mapping Arabizi to Arabic script using a range of techniques from rules to neural models Chalabi and Gerges (2012); Darwish (2014); Al-Badrashiny et al. (2014); Guellil et al. (2017); Younes et al. (2018); Shazal et al. (2020). In this paper we make use of a number of insights and techniques from work on Arabizi-to-Arabic script transliteration, but apply them in the opposite direction to map from Arabic script to a complex, detailed and strict Romanization. We compare rule-based and corpus-based techniques including a Seq2Seq model based on the publicly available code base of Shazal et al. (2020).

3 Data Collection

Sources We collected bibliographic records from three publicly available xml dumps stored in the machine-readable cataloguing (MARC) standard, an international standard for storing and describing bibliographic information. The three data sources are the Library of Congress (LC) (10.5M), the University of Michigan (UMICH) (680K), and New York University Abu Dhabi's Arabic Collections Online (ACO) (12K), amounting to 11.2 million records in total.

Extraction From these collections, we extracted 107,493 records that are specifically tagged with the Arabic language code (MARC 008 "ara").

Filtering Within the extracted records we filter out some of the entries using two strategies. First, we used a list of 33 safe tags (determined using their definitions and with empirical sampling check) to eliminate all entries that include a mix of translations, control information, and dates. The star-marked tags in Figure 1 are all included, while the rest are filtered out. Second, we eliminated all entries with mismatched numbers of tokens. This check was done after a cleaning step that corrected for common errors and inconsistencies in many entries such as punctuation misalignment and incorrect separation of the conjunction و+ $wa+$[2] 'and' clitic. As a result of this filtering, a small number of additional records are eliminated since all their entries were eliminated. The total number of retained records is 107,439. The full details on extraction and filtering are provided as part of the project's public github repo (see footnote 1).

Data Splits Finally, we split the remaining collection of records into Train, Dev, and Test sets. Details on the number of records, entries, and words they contain is presented in Table 1. We make our data and data splits available (see footnote 1).

4 Task Definition and Challenges

As discussed above, there are numerous ways to "transliterate" from one script to another. In this section we focus on the Romanization of undiacritized Arabic bibliographic entries into the ALA-LC standard. Our intention is to highlight the important challenges of this task in order to justify the design choices we make in our approaches. For a detailed reference of the ALA-LC Arabic Romanization standard, see (Library of Congress, 2012).

Phonological Challenges While Romanizing Arabic consonants is simple, the main challenge is in identifying unwritten phonological phenomena, e.g., short vowels, under-specified long vowels, consonantal gemination, and nunnation, all of which require modeling Arabic diacritization.

---

[2]Strict orthographic transliteration using the HSB scheme (Habash et al., 2007).

Morphosyntactic Challenges Beyond basic diacritization modeling, the task requires some morphosyntactic modeling: examples include (a) proclitics such as the definite article, prepositions and conjunctions are marked with a hyphen, (b) case endings are dropped, except before pronominal enclitics, (c) the silent Alif, appearing in some masculine plural verbal endings, is ignored, and (d) the Ta-Marbuta ending can be written as *h* or *t* depending on the morphosyntactic state of the noun. For more information on Arabic morphology, see (Habash, 2010).

Semantic Challenges Proper nouns need to be marked with capitalization on their first non-clitic alphabetic letter. Since Arabic script does not have "capitalizations", this effectively requires named-entity recognition. The Romanization of the word القاهرة *AlqAhrħ* 'Cairo' as *al-Qāhirah* in Figure 1 illustrates elements from all challenge types.

Special Cases The Arabic ALA-LC guidelines include a number of special cases, e.g., the word بن *bn* 'son of' is Romanized as *ibn*, and proper noun عمرو *ςmrw* is Romanized as *'Amr*.

5 Romanization Models

We compare multiple Romanization models built using four basic techniques with different expectation about training data availability, contextual modeling, and system complexity. The models are listed in Table 2.

CharTrans Technique Our baseline technique is an extremely simple character transliteration approach utilizing regular expressions and exception lists. This technique is built based on the ALA-LC guidelines, and is inspired by the work of Biadsy et al. (2009); it comprises 104 regex, 13 exceptions, and one capitalization rule (for entry-initial words). This technique accepts diacritized, undiacritized or partially diacritized input. Model **Rules Simple** uses CharTrans only.

MorphTrans Technique This technique relies on the morphological disambiguation system MADAMIRA (Pasha et al., 2014) to provide diacritization, morpheme boundaries, POS tags and English glosses for the Arabic input. Morpheme boundaries are used to identify clitic hyphenation points. POS tags and capitalization in English glosses are used to decide on what to capitalize

in the transliteration. We strip diacritical morphological case endings, but keep other diacritics. We utilize the CharTrans technique to finalize the Romanization starting with the diacritized, hyphenated and capitalization marked words. For words unknown to the morphological analyzer, we simply back off to the CharTrans technique. Model **Rules Morph** uses MorphTrans with CharTrans backoff.

MLE Technique Unlike the previous two techniques, MLE (maximum likelihood estimate) relies on the parallel training data we presented in Section 3. This simple technique works on whitespace and punctuation tokenized entries and learns simple one-to-one mapping from Arabic script to Romanization. The most common Romanization for a particular Arabic script input in the training data is used. The outputs are detokenized to allow strict matching alignment with the input. Faced with OOV (out of vocabulary), we back off to the MorphTrans technique (Model **MLE Morph**) or CharTrans Technique (Model **MLE Simple**). In Table 2, we also study the performance of **MLE Simple** with different corpus sizes.

Seq2Seq Technique Our last technique also relies on existing training data. We use an encoder-decoder character-level sequence-to-sequence architecture closely following Shazal et al. (2020) (although in reverse direction).

The encoder consists of two gated recurrent unit (GRU) layers (Cho et al., 2014) with only the first layer being bidirectional, and the decoder has two GRUs with attention (Luong et al., 2015). For the input, we used character embeddings concatenated with embeddings of the words in which the characters appear. For all other setting details, see Shazal et al. (2020)'s Line2Line model. We also show how **Seq2Seq** performs with different corpus sizes in Table 2.

The Seq2Seq technique is known for occasionally dropping tokens, which in our case leads to misalignment with the Arabic input. To handle this issue in model **Seq2Seq**, we align its output and fill such gaps using the outputs produced by three other techniques, thus creating models **Seq2Seq+Rules Morph**, **Seq2Seq+MLE Simple**, and **Seq2Seq+MLE Morph**. The alignment technique we use relies on minimizing character-edit distance between present words to identify missing ones.

Model	Corpus Size	Morph Trans	Char Trans	Dev			Test		
				Exact	CI	CPI	Exact	CI	CPI
Rules Simple	0		✓	16.2	17.4	17.8	16.1	17.3	17.7
Rules Morph	0	✓	✓	67.4	83.5	84.8	67.4	83.6	84.9
MLE Simple 1/64	31K		✓	63.6	69.8	71.1			
MLE Simple 1/32	63K		✓	68.5	75.1	76.4			
MLE Simple 1/16	125K		✓	73.0	79.9	81.3			
MLE Simple 1/8	250K		✓	75.6	82.8	84.2			
MLE Simple 1/4	500K		✓	80.3	87.2	88.6			
MLE Simple 1/2	1M		✓	82.7	89.5	90.9			
MLE Simple	2M		✓	84.0	90.7	92.1	84.1	90.8	92.2
MLE Morph	2M	✓	✓	84.7	91.6	93.0	84.8	91.7	**93.2**
Seq2Seq 1/64	31K			6.3	7.5	10.2			
Seq2Seq 1/32	63K			28.3	31.0	38.1			
Seq2Seq 1/16	125K			64.9	69.1	70.5			
Seq2Seq 1/8	250K			75.5	79.6	80.9			
Seq2Seq 1/4	500K			82.5	85.8	87.1			
Seq2Seq 1/2	1M			85.9	88.6	90.1			
Seq2Seq	2M			87.2	89.7	90.9	87.3	89.8	91.0
Seq2Seq + Rules Morph	2M	✓	✓	88.8	91.6	92.9	88.9	91.7	93.0
Seq2Seq + MLE Simple	2M		✓	**89.2**	**91.8**	**93.1**	**89.3**	**91.9**	**93.2**
Seq2Seq + MLE Morph	2M	✓	✓	**89.2**	**91.8**	**93.1**	**89.3**	**91.9**	**93.2**

Table 2: Dev and Test Romanization word accuracy (%). (CI = case-insensitive, and CPI = case and punctuation-insensitive)

Comparing the Techniques The CharTrans and MorphTrans techniques do not need parallel data, while the MLE and Seq2Seq techniques do. Furthermore, the MorphTrans and Seq2Seq techniques make use of available context: in MorphTrans, we use context-aware monolingual morphological disambiguation; and in Seq2Seq we model parallel examples in context. In contrast, neither the MLE technique nor the CharTrans technique use the context of the words being mapped.

6   Experimental Results

Table 2 presents the Dev and Test results for the models discussed in the previous section. All results are in terms of three word accuracy metrics: exact match (Exact), case-insensitive match (CI), and case and punctuation-insensitive match (CPI).

The **Rules Simple** baseline manages to correctly produce an exact answer in close to 1/6th of all the cases. **Rules Morph**, which uses no training data, misses about 1/3rd of all exact transliteration matches; however, about half of the errors are from capitalization issues.

The **MLE Simple** with 2M words cuts the error

from Rules Morph by 51% (Exact) and 44% (CI). Notably **Rules Morph** outperforms **MLE Simple** with 31K words in Exact match, and **MLE Simple** with 250K words in CI match.

The **MLE Morph** model improves over **MLE Simple** by ~1% absolute in all metrics (5% and 10% error reduction in Exact and CI, respectively).

The **Seq2Seq** model outperforms the **MLE Morph** model by 2.5% absolute (16% error reduction) in Exact match, but under-performs in CI match. The **Seq2Seq** performance is comparatively much poorer with less data. With 31K words, **MLE Simple**'s performance is 10 times better than **Seq2Seq**; and their performance only becomes comparable with 250K words.

We observe that ~2% of the **Seq2Seq** output words are missing, contributing negatively to the system's results. Of the three models that address this issue through alignment and combination, **Seq2Seq+Rules Morph**, **Seq2Seq+MLE Simple**, and **Seq2Seq+MLE Morph**, the last two using the MLE technique are the best performers overall in Exact match. It's noteworthy that in CPI match, **MLE Morph**'s performance is almost equivalent to the best systems' performance.

Error Type		Counts	Source	Prediction	Target
Gold **52**	**Romanization**	34	إبراهيم	Ibrāhīm	Ibrāhim
			الأشقر	al-Ashqar	Ashqar
			ندوات	Nadawāt	Nadwāt
	Alignment	8	أحمد بو حسن.	Aḥmad Bū Ḥasan.	Bū Ḥasan, Aḥmad.
	Source	5	الطبع	al-Ṭab'	al-Ṭab'ah
	Translation	5	شعر.	shi'r.	Poems.
System **48**	**Romanization**	36	الريف	al-Rayf	al-rīf
			حدث	ḥadath	Ḥaddatha
			خسارة،	Khassārah,	Khasārah.
	Hallucination	10	الادارية.	Taṣal-Idārīyah.	al-Idārīyah.
	Valid variant	2	السوفياتي	al-Sūfyātī	al-Sūfiyātī

Table 3: Error types, counts, and examples on a sample of 100 **Seq2Seq+MLE Morph** predictions.

The CPI metric values are consistently higher than CI by ∼1.3% absolute for all models.

Blind test results presented in the right hand side of Table 2 are consistent with Dev results.

7 Error Analysis

We classified a sample of 100 word errors (ignoring capitalization and punctuation) from the Dev set of our best performing model (**Seq2Seq+MLE Morph**). Our classification results are presented in Table 3 along with representative examples.

Gold Errors   We found 52 gold errors, where the human-provided target reference is incorrect. Romanization errors such as typos, incorrect vowelization, and dropped definite articles, constitute roughly 65% of gold errors. The rest of the errors include issues such as first and last name flipping which we classify as an alignment issue, Arabic input source typos, and errors in which the target is a translation instead of a Romanization. Notably, we observe that our **Seq2Seq+MLE Morph** model generates correct predictions for 85% of all gold error cases.

System Errors   Romanization errors make up 75% of system errors. The vast majority of these mistakes are due to wrong prediction of vowels or gemination. An additional 21% of the errors is due to Seq2Seq model hallucinations of characters unsupported by the source input. We also encountered 2 predictions that did not match the target reference but are correct variants. In ∼44%

of system error cases, outputs generated by the **MLE Morph** or **Rules Morph** models are in fact correct, but were not chosen during alignment and combination because of existing Seq2Seq answers.

8 Conclusions and Future Work

We presented a new task for Arabic NLP, namely the Romanization of Arabic bibliographic records. Our extracted corpus and benchmark data splits, as well as our code base will be publicly available.

In the future, we plan to create an online Romanization interface to assist librarians. As more data is created efficiently, better models can be created.

We also plan to exploit the latent annotations in bibliographic records for improving Arabic NLP tools, e.g. using vowelization for automatic diacritization and possible morphological disambiguation (Habash et al., 2016), marked clitics for tokenization, and Roman-script capitalization for Arabic named entity recognition.

Acknowledgments

References

Ahmed Abdelali, Kareem Darwish, Nadir Durrani, and Hamdy Mubarak. 2016. Farasa: A fast and furious segmenter for Arabic. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 11–16, San Diego, California.

Mohamed Al-Badrashiny, Ramy Eskander, Nizar Habash, and Owen Rambow. 2014. Automatic transliteration of romanized Dialectal Arabic. In *Proceedings of the Conference on Computational Natural Language Learning (CoNLL)*, pages 30–38, Ann Arbor, Michigan.

Yaser Al-Onaizan and Kevin Knight. 2002. Machine Transliteration of Names in Arabic Text. In *Proceedings of the Workshop on Computational Approaches to Semitic Languages (CASL)*.

Kenneth R. Beesley. 1997. Romanization, Transcription and Transliteration. Http://www.xrce.xerox.com/Research-Development/Historical-projects/Linguistic-Demos/Arabic-Morphological-Analysis-and-Generation/Romanization-Transcription-and-Transliteration.

Fadi Biadsy, Nizar Habash, and Julia Hirschberg. 2009. Improving the Arabic Pronunciation Dictionary for Phone and Word Recognition with Linguistically-Based Pronunciation Rules. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 397–405, Boulder, Colorado.

Tim Buckwalter. 2004. Buckwalter Arabic Morphological Analyzer Version 2.0. LDC catalog number LDC2004L02, ISBN 1-58563-324-0.

Achraf Chalabi and Hany Gerges. 2012. Romanized Arabic transliteration. In *Proceedings of the Second Workshop on Advances in Text Input Methods*, pages 89–96, Mumbai, India. The COLING 2012 Organizing Committee.

Kyunghyun Cho, Bart van Merriënboer, Çağlar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar.

Kareem Darwish. 2014. Arabizi Detection and Conversion to Arabic. In *Proceedings of the Workshop for Arabic Natural Language Processing (WANLP)*, pages 217–224, Doha, Qatar.

Imane Guellil, cal Azouaou Fai Mourad Abbas, and Fatiha Sadat. 2017. Arabizi transliteration of Algerian Arabic dialect into Modern Standard Arabic. In *Social MT 2017/First workshop on social media and user generated content machine translation*.

Nizar Habash, Anas Shahrour, and Muhamed Al-Khalil. 2016. Exploiting Arabic diacritization for high quality automatic annotation. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4298–4304, Portorož, Slovenia. European Language Resources Association (ELRA).

Nizar Habash, Abdelhadi Soudi, and Tim Buckwalter. 2007. On Arabic Transliteration. In A. van den Bosch and A. Soudi, editors, *Arabic Computational Morphology: Knowledge-based and Empirical Methods*, pages 15–22. Springer, Netherlands.

Nizar Y Habash. 2010. *Introduction to Arabic natural language processing*. Morgan & Claypool Publishers.

Library of Congress. 2012. Arabic Romanization Table. https://www.loc.gov/catdir/cpso/romanization/arabic.pdf.

Library of Congress. 2017. ALA-LC Romanization Tables. https://www.loc.gov/catdir/cpso/roman.html.

Thang Luong, Hieu Pham, and Christopher Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1412–1421, Lisbon, Portugal.

Ossama Obeid, Nasser Zalmout, Salam Khalifa, Dima Taji, Mai Oudah, Bashar Alhafni, Go Inoue, Fadhl Eryani, Alexander Erdmann, and Nizar Habash. 2020. CAMeL tools: An open source python toolkit for Arabic natural language processing. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 7022–7032, Marseille, France. European Language Resources Association.

Arfath Pasha, Mohamed Al-Badrashiny, Mona Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. 2014. MADAMIRA: A fast, comprehensive tool for morphological analysis and disambiguation of Arabic. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, pages 1094–1101, Reykjavik, Iceland.

Ali Shazal, Aiza Usman, and Nizar Habash. 2020. A unified model for Arabizi detection and transliteration using sequence-to-sequence models. In *Proceedings of the Fifth Arabic Natural Language Processing Workshop*, pages 167–177, Barcelona, Spain (Online). Association for Computational Linguistics.

Ahmad Taymūr. 1927? Qabr al-imām al-Suyūṭī wa-taḥqīq mawḍi'ihi. Arabic Collections Online: http://hdl.handle.net/2333.1/m37pvs4b.

Jihene Younes, Emna Souissi, Hadhemi Achour, and Ahmed Ferchichi. 2018. A sequence-to-sequence based approach for the double transliteration of Tunisian dialect. *Procedia Computer Science*, 142:238 – 245. Arabic Computational Linguistics.