A Machine Transliteration Tool Between Uzbek Alphabets

Ulugbek Salaev[1], Elmurod Kuriyozov[2] and Carlos Gómez-Rodríguez[2]

[1] *Urgench State University, Department of Information Technologies, 14, Kh.Alimdjan str, Urgench city, 220100, Uzbekistan*

[2] *Universidade da Coruña, CITIC, Grupo LYS, Depto. de Computación y Tecnologías de la Información, Facultade de Informática, Campus de Elviña, A Coruña 15071, Spain*

Abstract

Machine transliteration, as defined in this paper, is a process of automatically transforming written script of words from a source alphabet into words of another target alphabet within the same language, while preserving their meaning, as well as pronunciation. The main goal of this paper is to present a machine transliteration tool between three common scripts used in low-resource Uzbek language: the old Cyrillic, currently official Latin, and newly announced New Latin alphabets. The tool has been created using a combination of rule-based and fine-tuning approaches. The created tool is available as an open-source Python package, as well as a web-based application including a public API. To our knowledge, this is the first machine transliteration tool that supports the newly announced Latin alphabet of the Uzbek language.

Keywords

transliteration, uzbek language, natural language processing, low-resource language

1. Introduction

The term transliteration is ambiguous, as it refers to two similar tasks of Natural Language Processing (NLP), which differ according to their either inter-language or intra-language nature. More specifically, a transliteration can be described as a process of representing words from one language using the alphabet of another language [1], while the other use of the term stands for the act of transforming words from one alphabet into another alphabet within the same language [2]. We take the latter case as our goal in this work, and present a method for transforming words between three equally-important alphabets of the low-resource Uzbek language.

Uzbek language (native: *O'zbek tili*) is a low-resource, highly-agglutinative language with null-subject and null-gender characteristics from the Karluk branch of the Turkic language family. It is an official language of Uzbekistan, with more than 30 million speakers inside and

The International Conference and Workshop on Agglutinative Language Technologies as a challenge of Natural Language Processing (ALTNLP), June 7-8, 2022, Koper, Slovenia

ulugbek0302@gmail.com (U. Salaev); e.kuriyozov@udc.es (E. Kuriyozov); carlos.gomez@udc.es (C. Gómez-Rodríguez)

http://www.grupolys.org/~cgomezr (C. Gómez-Rodríguez)

0000-0003-3020-7099 (U. Salaev); 0000-0003-1702-1222 (E. Kuriyozov); 0000-0003-0752-8812 (C. Gómez-Rodríguez)

around the country, making it the second most widely spoken language among Turkic languages (right after Turkish language)[1].

The Cyrillic alphabet had been in use for a long time for Uzbek language, until it was replaced with Latin script in 1993[2] (with a reformation in 1995[3]), which is still an official alphabet. The use of both Cyrillic and Latin alphabets is equally popular in all areas of written language (law, books, web, media, etc.) even these days. Availability of texts in two writing systems make it harder and costlier for NLP researchers and practitioners to work on the language, such as by limiting the amount of collected data for a specific alphabet, or by creating a need to develop language resources and models for both alphabets. Furthermore, there is a new reformation[4] that has been introduced to change all the existing digraphs and replace them with diacritical signs[5], so every letter in the alphabet would be written with only a single character. Throughout this paper, we refer to this reformed Latin alphabet as "New Latin" alphabet.

Considering the existence of three distinctive alphabets currently in use in Uzbek language, we propose a methodology to perform the task of transliteration between those three alphabets, which is a combination of basic rule-based character-mapping, more sophisticated cross-alphabet specific rules, as well as fine-tuning approaches. Although there are some available web tools that offer transliteration between Cyrillic and Latin alphabets for Uzbek, none of them offer neither an open source code, nor an Application Programming Interface (API) for integration with other tools. Moreover, the only one tool with a good quality from Savodxon project[6] is commercial, and the free ones are not practical enough to be used, due to a bad implementation. In this paper, we also present a publicly available Python code[7] for research integration, together with a web-based tool[8] that also includes an API, which is, to our knowledge, the first ever transliteration tool between all three alphabets.

2. Related work

One of the very early mentions of machine transliteration was raised by Nida and Taber [3], stating that a problem of "untranslatability" arises when an exact equivalence of meanings is required in translation, rather than a comparative equivalence, so they referred to transliteration to tackle the issue. In early mentions, transliteration was described as a process of representing words from one language using the alphabet of another language, as part of machine translation [1]. Later on, it also has been used for similar purposes, but with intra-language perspective, describing it as a conversion of words from one written script to another one within the same

[1] More about Uzbek language: https://en.wikipedia.org/wiki/Uzbek_language

[2] Law of the Republic of Uzbekistan "On the introduction of the Uzbek alphabet based on the Latin script" (September 2, 1993 year, reg. number: 931-XII): https://lex.uz/docs/-112286.

[3] On Amendments to the Law of the Republic of Uzbekistan "On Introduction of the Uzbek Alphabet Based on the Latin Script" (May 6, 1995 year, reg. number: 71-I): https://lex.uz/docs/-116158.

[4] Resolution of the Cabinet of Ministers of the Republic of Uzbekistan "On measures to ensure a gradual transition to the Uzbek alphabet based on the Latin script." (February 10, 2021 year, reg. number: 61): https://lex.uz/uz/docs/-5281850.

[5] More about alphabets used in Uzbek language: https://www.omniglot.com/writing/uzbek.htm.

[6] https://savodxon.uz/

[7] https://github.com/UlugbekSalaev/UzTransliterator

[8] https://nlp.urdu.uz/?menu=translit

language [2, 4].

Instances of early works on transliteration can be Arabic-English names transliteration using a combination of a rule-based system with neural networks [1], and Japanese-English using finite state transducers [5]. Both approaches dealt with phonetic representations of words, which were replaced by a spelling-based approach to achieve higher results, as in the case of the Arabic-English model of [6]. Later modern approaches to transliteration include models with long short-term memories (LSTM) [4], and recurrent neural networks (RNN) [7], which perform equally well. Combination of old rule-based approaches with recent deep-learning methods improves the quality, according to a comparative study [8].

Transliteration between Cyrillic and Latin alphabets of Uzbek language has been done by Mansurovs [9], who used a data-driven approach, by aligning words and training a decision-tree classifier. Among some other NLP work that has been done on low-resource Uzbek language so far, there are a morphological analyzer [10], WordNet type synsets [11], Uzbek stopwords dataset [12], sentiment analysis and text classification [13, 14, 15], cross-lingual word-embeddings [16], as well as a pretrained Uzbek language model based on the BERT architecture [17].

3. Methodology

To check the accuracy of our tool, we collected text from the spelling dictionary of Uzbek language [18]. This dictionary is a printed resource that contains about 14K commonly-used words of Uzbek language in Latin and Cyrillic variants. We did not include multiword expressions, because we used word-level evaluation to check the performance analysis, and using them by splitting into single words would create duplications. After also removing words that could not be successfully digitalized using OCR, we ended up with around 9600 words to use in our experiments.

Although the dictionary size is limited, it includes words that are prone to spelling errors between Cyrillic and Latin. Since there is no publicly available data for the New Latin alphabet yet, we transliterated those words from Latin to New Latin, then manually checked resulting words, correcting them where necessary. Manual correction was only possible within our resources thanks to the fact that the majority of words stayed the same as in Latin, we focused only on words that changed their form.

The methodology used in this work is very similar to the work from Mansurovs [9], but we extend it by adding the New Latin alphabet. Additionally, instead of training a classifier, we rely on string replacement techniques for the sake of simplicity and speed. Following are the steps followed by the tool, and the steps that need more detail are explained separately afterwards:

1. `Tokenization:` Feeding text from the source alphabet as string buffer, and splitting into tokens;
2. `Replacement of exceptional words:` Checking each token to see if it is or contains a word from the exceptional words dataset (excluding punctuations, emojis, or unrecognized characters), if so, replacing it with its target version;
3. `Replacement using rules:`Going through a set of mapping rules specific to the pair of alphabets and conversion direction that were designed to use where one-to-one character mapping does not apply. Technically, each rule consists of a simple regular

exrpression that looks for a specific sub-string (usually one to three character long), and replaces it with desired sub-string(either empty, one or more characters long);

4. `Character-mapping` : Replacing the rest of characters from source alphabet to the target one using one-to-one mapping. This can also be made by very simple regular expression that replaces one character with another in a string;
5. `Re-uniting` : Merging resulting tokens that contain target alphabet characters back again, and returning them as a whole string.

3.1. Replacement of exceptional words

This is the step we came up with after applying a fine-tuning approach to the created tool. There are words that cannot be transliterated using a rule-based approach. Only one-directional transliteration (like from Cyrillic to Latin) may be possible, but it could fail in the opposite direction (like from Latin to Cyrillic). To solve this issue, we extracted words from the collected data that did not provide the same output when transliterated and back-transliterated between different combinations. So far, there are 233 words with their form in all three alphabets that are stored in the tool as an exceptional words database. Some examples of such words can be seen in Table 1. One interesting insight about those words is that they are mostly loan words from Russian language, and there is usually a change when converting Cyrillic letters *ц*, *ь* (phonetic glottal stop), and *я*. Although this process was done after the tool's creation, it is required that this step has to be applied before any further conversion steps are applied.

Table 1
Some examples from the exceptional words database where rule-based transliteration does not apply.

Latin	Cyrillic	New Latin	English
aksent	акцент	aksent	accent
budilnik	будильник	budilnik	alarm clock
batalyon	батальон	batalyon	batalion
feldsher	фельдшер	feldşer	paramedic
fransuz	француз	fransuz	french
intervyu	интервью	intervyu	interview
koeffitsient	коэффициент	koeffitsient	coefficient
korrupsiya	коррупция	korrupsiya	corruption
kuryer	курьер	kuryer	courier
medalyon	медальон	medalyon	medallion
oktabr	октябрь	oktabr	october
pavilyon	павильон	pavilyon	pavilion
porshen	поршень	porşen	piston
shpatel	шпатель	şpatel	scraper (putty knife)
cherepitsa	черепица	çerepitsa	roof tile (shingle)

3.2. Character-mapping

Steps 3 and 4 of the conversion deal with mapping characters from source alphabet to the target one. Although the majority of letters are replaced in a straightforward manner, the remaining

characters require set of pairwise rules based on the alphabets involved, and the direction of the conversion. A general idea of conversion between alphabets is given in Table 2.

Throughout the process, we found out that some conversion rules are not as straightforward as expected. There is a problem with handling a single character uppercase letter when converting to a digraph letter in other alphabet. For instance, if we convert Cyrillic uppercase letters *Ш* and *Ю* into *SH* and *YU* (respectively) in Latin, an error like these happen: "*Шўрва*>"*S**H**o'rva*"(soup), or "*Юлдуз*>"*Y**U**lduz*"(star); But if we convert it into *Sh* and *Yu*, then an error with acronyms occurs like these: "*АҚШ*>"*AQS**h***"(USA), or "*ЮНЕСКО*>"*Y**u**NESKO*"(UNESCO). A solution to this kind of problem is to consider surrounding letters when performing conversion.

Another complicated situation with mapping rules is the phonetic glottal stop (native: *Tutuq belgisi*), which is also part of an alphabet in Uzbek language. There are some words that a glottal stop appears in its Cyrillic form and is omitted in its Latin form. For instance: "*факул**ъ**тет*->"*fakultet*"(faculty), or "*кал**ъ**ций*>"*kalsiy*"(calcium). The problem with this omission is twofold: The algorithm has to be taught whether to omit it or not, also when these words are transliterated back to Cyrillic, the glottal stop has to appear out of nowhere. A solution to this kind of problem is to include this kind of words in the exceptional words list.

Table 2
Character-level mapping between alphabets for transliteration. **Cyr.** stands for Cyrillic alphabet, **Lat.** stands for Latin alphabet, and **NewLat.** stands for New Latin alphabet. ∅ denotes an empty string. Highlighted rows indicate a complex mapping, where one character from source alphabet is mapped to either two or zero characters from target alphabet. The character at the very end of the table is called a *phonetic glottal stop* (native: *Tutuq belgisi*), and although it is not a real letter, still it is considered a part of the Uzbek alphabet.

Cyr.	Lat.	NewLat.
А а	A a	A a
Б б	B b	B b
В в	V v	V v
Г г	G g	G g
Д д	D d	D d
Е е	**E/Ye e/ye**	**E/Ye e/ye**
Ё ё	**Yo yo**	**Yo yo**
Ж ж	J j	J j
З з	Z z	Z z
И и	I i	I i
Й й	Y y	Y y
К к	K k	K k

Cyr.	Lat.	NewLat.
Л л	L l	L l
М м	M m	M m
Н н	N n	N n
О о	O o	O o
П п	P p	P p
Қ қ	Q q	Q q
Р р	R r	R r
С с	S s	S s
Т т	T t	T t
У у	U u	U u
Ф ф	F f	F f
Х х	X x	X x

Cyr.	Lat.	NewLat.
Ҳ ҳ	H h	H h
Ц ц	**Ts/S ts/s**	**Ts/S ts/s**
Э э	E e	E e
Ю ю	**Yu/u yu/u**	**Yu/u yu/u**
Я я	**Ya/A ya/a**	**Ya/A ya/a**
Ў ў	**O‘ o‘**	**Ō ō**
Ғ ғ	**G‘ g‘**	**Ḡ ḡ**
Ш ш	**Sh sh**	**Ş ş**
Ч ч	**Ch ch**	**Ç ç**
Нг нг	**Ng ng**	**Ń ń**
ъ	’/∅	’/∅

4. Results

The created tool has been analysed using the collected parallel text data for all three alphabets, and comparing the tool's output for each word with the actual expected output. We have calculated micro-averaged F1 scores of each conversion using the *metrics* module of scikit-

Table 3
Micro-averaged F1 scores of word level transliteration process between alphabets. The direction of the transliteration is from the alphabet shown in the row to the alphabet shown in the column.

Alphabets	Latin	Cyrillic	New Latin
Latin	-	0.89	0.94
Cyrillic	0.90	-	0.92
New Latin	0.93	0.92	-

learn[9]. F1-scores are calculated at the word level (i.e., by considering words that the system transliterates correctly or incorrectly). Table 3 shows the results between each pair and each direction.

Although the analysis has been done using very limited amount of data, it gives us some insights about the tool's performance: The best performing pair is Latin->New Latin conversion (0.94 F1 score) due to the reason that there are only five letters that change during conversion with no exceptional cases (to our best knowledge), and those errors that still occur are only because of the problem with handling the abbreviations.The worst performing pair is Latin->Cyrillic (0.89 F1 score), likely due to many conversion rules to consider, plus many exceptional cases. Furthermore, It is also possible to see that transliteration to and from the New Latin alphabet performs better than any other alphabets do, which can be explained by the minimum number of conversion rules required compared to its counterparts. More specifically, Transliteration between New Latin and Latin would require only 5 specific conversion rules (and no exceptional cases), and 6 rules (plus exceptional cases) between New Latin and Cyrillic, while the same process would require 11 rules (and exceptional cases) from a transliteration between Latin and Cyrillic alphabets.

The Python tool created for this work is openly-accessible, and also can be easily installed, using the following command that is popular for the Python community:

```
pip install UzTransliterator
```

The user interface of the created web tool can be seen in Figure 1. There is also a public API based on this tool, and more detailed information about it can be found at the project's GitHub repository.

4.1. Discussion

Although the created tool is practical enough to be used for transliteration, there are some certain cases we still have to consider and improve on the go:

- Our database of exceptional cases (a result of fine-tuning approach) contains only lemmas of words, and due to the highly agglutinative nature of the Uzbek language, words mostly appear as inflections and derivations. For this reason we have to either store their root forms or add syntactic knowledge to handle all possible forms of lemmas;
- New loan words and proper nouns adopted from other languages might not produce expected output, thus we have to keep updating the database of exceptional cases;

[9] https://scikit-learn.org/0.15/modules/classes.html#module-sklearn.metrics

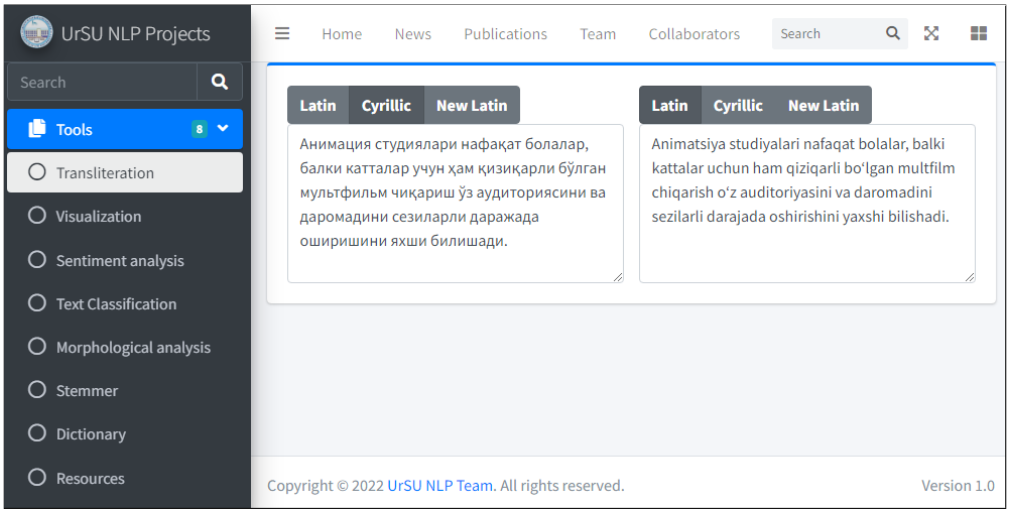

Figure 1: Web-interface of the created transliteration tool.

- We dealt with legal text properly written in Uzbek language, which is not always the case with user-generated text. Especially, there is a big deal of inconsistency in writing *o'* and *g'* letters in the currently official alphabet due to the use of apostrophe, which comes in many ways, such as *o‘*,*o‘*,*o'*, and *g‘*,*g‘*,*g'* forms respectively;
- Due to the lack of texts created in the New Latin alphabet, we worked only with manually created text, which is very limited and requires more analysis as the coverage starts to enlarge.

5. Conclusion

In this paper, we presented a Python code, a web tool, and an API created for the low-resource Uzbek language that performs machine transliteration between two popularly used Cyrillic and Latin alphabets, as well as a newly reformed version of the Latin alphabet which, according to the governmental decree, all legal texts will have been completely adapted to by year 2023. We have also shown the cases of alphabet-specific problems related to the transliteration between those three scripts that do not allow for a simple character mapping, including ongoing attempts to tackle user-input related issues.

Our future work will be to strengthen the output quality of the current tool by implementing more mapping rules, user input cleaning techniques, as well as integrating a pretrained neural language model that can handle unseen cases. Furthermore, we hope to be able to make a pipeline that can perform useful NLP tasks for Uzbek language, such as tokenization, POS tagging, morphological analysis, and parsing in a foreseen future.

Acknowledgments

This work has received funding from ERDF/MICINN-AEI (SCANNER-UDC, PID2020-113230RB-

C21), from Xunta de Galicia (ED431C 2020/11), and from Centro de Investigación de Galicia "CITIC", funded by Xunta de Galicia and the European Union (ERDF - Galicia 2014-2020 Program), by grant ED431G 2019/01. Elmurod Kuriyozov was funded for his PhD by El-Yurt-Umidi Foundation under the Cabinet of Ministers of the Republic of Uzbekistan.

References

[1] M. Arbabi, S. M. Fischthal, V. C. Cheng, E. Bart, Algorithms for arabic name transliteration, IBM Journal of research and Development 38 (1994) 183–194.

[2] E. Birnbaum, The transliteration of ottoman turkish for library and general purposes, Journal of the American Oriental Society 87 (1967) 122–156.

[3] E. A. Nida, C. R. Taber, The theory and practice of [Biblical] translation, Brill, 1969.

[4] M. Alam, S. ul Hussain, Sequence to sequence networks for roman-urdu to urdu transliteration, in: 2017 International Multi-topic Conference (INMIC), IEEE, 2017, pp. 1–7.

[5] K. Knight, J. Graehl, Machine transliteration, arXiv preprint cmp-lg/9704003 (1997).

[6] Y. Al-Onaizan, K. Knight, Machine transliteration of names in arabic texts, in: Proceedings of the ACL-02 workshop on Computational approaches to semitic languages, 2002.

[7] N. T. Le, F. Sadat, L. Menard, D. Dinh, Low-resource machine transliteration using recurrent neural networks, ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP) 18 (2019) 1–14.

[8] S. Najafi, B. Hauer, R. R. Riyadh, L. Yu, G. Kondrak, Comparison of assorted models for transliteration, in: Proceedings of the Seventh Named Entities Workshop, 2018, pp. 84–88.

[9] B. Mansurov, A. Mansurov, Uzbek cyrillic-latin-cyrillic machine transliteration, arXiv preprint arXiv:2101.05162 (2021).

[10] G. Matlatipov, Z. Vetulani, Representation of uzbek morphology in prolog, in: Aspects of Natural Language Processing, Springer, 2009, pp. 83–110.

[11] A. Agostini, T. Usmanov, U. Khamdamov, N. Abdurakhmonova, M. Mamasaidov, Uzwordnet: A lexical-semantic database for the uzbek language, in: Proceedings of the 11th Global Wordnet conference, 2021, pp. 8–19.

[12] K. Madatov, S. Bekchanov, J. Vičič, Automatic detection of stop words for texts in the uzbek language, 2022. URL: https://www.preprints.org/manuscript/202204.0234/v1. doi:10.20944/preprints202204.0234.v1.

[13] E. Kuriyozov, S. Matlatipov, Building a new sentiment analysis dataset for uzbek language and creating baseline models, in: Multidisciplinary Digital Publishing Institute Proceedings, volume 21, 2019, p. 37.

[14] I. Rabbimov, I. Mporas, V. Simaki, S. Kobilov, Investigating the effect of emoji in opinion classification of uzbek movie review comments, in: International Conference on Speech and Computer, Springer, 2020, pp. 435–445.

[15] I. Rabbimov, S. Kobilov, Multi-class text classification of uzbek news articles using machine learning, in: Journal of Physics: Conference Series, volume 1546, IOP Publishing, 2020, p. 012097.

[16] E. Kuriyozov, Y. Doval, C. Gomez-Rodriguez, Cross-lingual word embeddings for turkic

languages, in: Proceedings of The 12th Language Resources and Evaluation Conference, 2020, pp. 4054–4062.
[17] B. Mansurov, A. Mansurov, Uzbert: pretraining a bert model for uzbek, arXiv preprint arXiv:2108.09814 (2021).
[18] T. Togʻayev, G. Tavaldiyeva, M. Akromova, Oʻzbek tilining kirill va lotin alifbolaridagi imlo lugʻati, "Sharq" nashriyot-matbaa aksiyadorlik kompaniyasi bosh tahririyati, Taskent, Uzbekistan, 1999.