ITC 1/52

Information Technology and Control
Vol. 52 / No. 1 / 2023
pp. 128-139
DOI 10.5755/j01.itc.52.1.32353

Automatic Transliteration of Polish and English Proper Nouns into Lithuanian

Received 2022/09/23 | Accepted after revision 2022/12/20

https://doi.org/10.5755/j01.itc.52.1.32353

HOW TO CITE: Kasparaitis, P. (2023). Automatic Transliteration of Polish and English Proper Nouns into Lithuanian. *Information Technology and Control,* 52(1), 128-139. https://doi.org/10.5755/j01.itc.52.1.32353

Automatic Transliteration of Polish and English Proper Nouns into Lithuanian

Pijus Kasparaitis

Institute of Informatics, Faculty of Mathematics and Informatics; Vilnius University;
Didlaukio Str. 47, LT-08303, Vilnius, Lithuania; phone: +370 649 60094; e-mail: pijus.kasparaitis@mif.vu.lt

Corresponding author: pijus.kasparaitis@mif.vu.lt

As the world is becoming more globalized, proper nouns move from one language into other languages. In order to preserve the grammatical or phonetic structure of the target language, a desire arises to adapt them. The present work deals with adaptation (transliteration) of Polish and English words to the Lithuanian language. The set of context-sensitive and context-free rules was created manually for the Polish language. Manually creating such rules for the English language is too difficult, thus the algorithm to automatically generate transliteration rules from English-Lithuanian word pairs aligned at the letter level was developed in this work. For the Polish language, 100% accuracy was achieved. For English, word accuracy of about 50% and character accuracy of about 90% was achieved. The reasons for this accuracy are identified and directions for improving the set of rules are provided.

KEYWORDS: Natural Language Processing, machine translation, machine transliteration, context-sensitive rules, context-free rules, Lithuanian.

1. Introduction

The world is becoming more globalized, exchange of news and other information is becoming more and more intense, so the names, surnames, place names and other proper nouns travel from one language to another. Such words are not translated, however, when they enter another language, they can be adapted to conform to the spelling and phonetic rules of that language.

A few decades ago, in the "era of paper press", there were relatively few such words, and the latter usually had traditional firmly-established and well-known adapted (transliterated) forms, e.g., in Lithuanian Viljamas Šekspyras (William Shakespeare), Vinstonas Čerčilis (Winston Churchill). Attention should be drawn to how greatly the adapted word forms differed from the original spelling. Recently, in the "era of online press" the number of foreign words has been on the increase in every language.

Machine transliteration emerged as part of the machine translation (MT) process, which involves spelling out-of-dictionary words in the target language while preserving the pronunciation. Now it has evolved into an independent branch of Natural Language Processing (NLP).

Literature on transliteration falls into two major groups [8]: transliteration extraction and generative transliteration.

Transliteration extraction deals with the enrichment of the translation lexicon using existing transliteration instances from large multilingual corpora in order to reduce the need for on-the-fly transliteration. E.g., for the English-Latvian transliteration extraction see [11]. Generative transliteration focuses on algorithms that transliterate newly occurring terms that do not exist in any translation dictionary. There are four categories of generative transliteration methods [8]: phonetic-based, spelling-based, hybrid (both phonetic-based and spelling-based methods are used), and combined (several phonetic-based or several spelling-based methods are combined). Phonetic-based methods consist of several steps: grapheme to phoneme, phoneme to phoneme, phoneme to grapheme. The spelling-based method uses a single step. It has been reported by [8] that spelling based methods are more successful when compared with phoneme-based ones because errors accumulate during the steps. Our proposed method belongs to the category of generative spelling-based methods.

Further in this paper, related works are considered, the terms related to transliteration, legal regulation of transliteration, complexity of transliteration of English words are explained. The need for transliteration is justified. The algorithm to automatically generate transliteration rules is presented and the accuracy of these rules is evaluated.

2. Related Works

Various algorithms can be used for generative transliteration: handcrafted rules [4] (see below), statistical methods [14] (English-Nepali language pair, accuracy 87%), support vector machines (SVM) and hidden Markov model (HMM) [3] (Bengali-to-English forward and backward name transliteration), decision trees [7] (English/Korean transliteration with word accuracy 51.3%, character accuracy 83.4% and back-transliteration with word accuracy 37.2%, character accuracy 80%), n-gram models [6] (Punjabi person names from Gurmukhi script to English and back, word accuracy around 96%, char accuracy around 99% for both directions), weighted finite state transducers [10] (seven languages, 64-78% accuracy achieved), machine learning techniques such as neural networks of various architectures, e.g., gated recurrent unit (GRU) [1] (Devanagari to Roman Hindi transliteration and its back transliteration, word error rate 64.8% and character error rate 20.1% for forward transliteration), long short-term memory (LSTM) [16] (Romanized Arabic script called Arabizi to Arabic script, accuracy 79% and bilingual evaluation understudy score 88.49).

In this work context-sensitive rules are developed in the following form:

L [C] R → "T",
    F

where L – left context, C – current letter, R – right context, T – letters of the target language, F – features of the current letter.

Context-sensitive and context-free rules have already been used to manually compile a set of rules for transcribing English and French text into phonemes [4]. E.g., the following letter-to-sound rules were created for the letter C in English:

[C]A → "k"
[C]O → "k"
[C] → "s"

From now on we will use the following notation: the current letter of the source language will be written in square brackets (e.g., [A]), and the string of phonemes or letters of the target language – in double quotes, letters in all examples used for rule-making will be written in upper case, phonemes – in lower case. This notation differs from that in [4]. The first two rules are context-sensitive and the third one is context-free. These rules would successfully process words like CAKE, COOL, CEASE, CIGAR. If examples are found that contradict the rules created, such as CELLO [4], and when such examples are few, they should be included in the list of exceptions rather than changing existing rules and creating such a new rule:

[C]E → “tʃ”

Context-sensitive rules, supplemented by features of the current letter, were also successfully used to transcribe Lithuanian texts into phonemes [9].

The transliteration is similar to the phonetic transcription, only the representation of the target language is different. The transcription of Lithuanian has already been analyzed in [13, 15].

Interest in automatic transliteration into the Lithuanian language has arisen quite recently. Automatic transliteration from German has already been considered in [2]. Rules with the same form as in this work were used, 95% word accuracy achieved. Works on transliteration from other languages into Lithuanian have not been published. This work is the first dedicated to automatic transliteration from Polish and English into Lithuanian.

3. Background

3.1. Terminology

There are several other terms (besides transliteration) that refer to the adaptation of a foreign word to a specific language. Romanization is the conversion of a text from a different writing system to the Roman (Latin) script. The transliteration into a specific language can be named based on the target language, e.g., Anglicization or Lithuanization. The latter often include adaptation to a more drastic degree than that implied, for example, in Romanization.

As for the Lithuanian language, the following is stated in [19], approved by The State Commission of the Lithuanian Language:

“9.2. Definitions of terms used in Lithuanian when discussing the writing of personal and place names in other languages:

Transliteration – rewriting personal names (place names) written in non-Latin alphabets into letters (or as accurately as possible by rendering other written signs) in Lithuanian characters … .

Transcription – recording a personal name (place name) of another language according to the approximate pronunciation in Lithuanian characters.

Grammaticalization – adaptation of a personal name (place name) of another language to the morphological system of the Lithuanian language by adding inflectional endings of the Lithuanian language.”

The term transcription used in this definition means orthographic transcription where the pronunciation is conveyed with the help of letters of the target language. Usually (e.g., in speech synthesis or recognition) the term transcription refers to the phonetic transcription where the pronunciation is conveyed using special phoneme symbols (such as International Phonetic Alphabet). In order to avoid confusion with the term “transcription”, further in this article we will refer to the adaptation of Polish and English words into Lithuanian as transliteration.

3.2. Adaptation vs Original Spelling

It is obvious that transliteration is necessary if source and target languages use different writing systems (e.g., Cyrillic, Arabic, Greek, Latin, Chinese, Japanese). The answer is not so obvious when both languages use the same script.

On the one hand, there is a desire to make a foreign word easy to read, to adjust it to other words in a sentence. If foreign words are used unadapted or adapted insufficiently, problems can occur in the spoken language. For example, [5] reports that there are many English words in Italian, and their unphonetic spelling and pronunciation can create misunderstanding and confusion, because Italians often tend to pronounce these international imports the way they are written. Different reading of the letter “h” in the English and Italian languages would cause confusion among the Italians when reading such words as “hurt” – “art”, “hair” – “air”, “heat” – “eat”, “ham” – “am”, etc.

On the other hand, adaptation of foreign words in the written language can cause problems of recognizing how that word was spelled in the original language so that it could be used in the online search engines, etc. E.g., different orthographies of the name of the same person, namely, the former Lebanese Prime Minister Rafik Hariri, in nine languages are presented in [12].

A simple experiment was performed to test whether and how English word adaptation is used in different countries: the name of the city of Brighton in East Sussex, England was translated into other languages with the help of an online translation system https://translate.google.com/. In all languages tested that use the Cyrillic script, the word has been transliterated according to its pronunciation. Most European languages that use the Latin alphabet use original spelling, however, there are some countries that transliterate (see Table 1).

Table 1

How the world 'Brighton' is spelled in different languages

Language	Orthography
English	Brighton
Lithuanian	Braitonas
Azerbaijani, Uzbek	Brayton
Turkmen	Braýton
Serbian (Latin)	Brajton
Serbian (Cyrillic)	Брајтон
Russian, Ukrainian, Kazakh, Kyrgyz	Брайтон
Bulgarian	Брайтън

Thus, if the source and target languages use the same writing system, different levels of adaptation may be used depending on the similarity of the languages, the legislation of the target language country, etc.

An example of very simple adaptation could be German where words can be transliterated to English by simply replacing non-English letters with English ones. A trivial piece of software can be found in "How to anglicize German umlaut characters?" (https://knowledgebase.progress.com/articles/Article/000048933), where the whole process is described by the following elementary replacements:

[ä] → "ae",
[ö] → "oe",
[ü] → "ue",
[Ä] → "Ae",
[Ö] → "Oe",
[Ü] → "Ue",
[ß] → "ss".

3.3. Legal Regulation in Lithuania

An example of complicated adaptation of foreign words regulated by the legislation of the country could be Lithuanian. In Lithuania this issue is regulated by [19], approved by The State Commission of the Lithuanian Language. In principle, two questions arise here: in which cases and in what way to adapt. In answer to the first question, the following is stated:

"9.2.1. Personal names and place names of languages using the Latin alphabet are presented in Lithuanian texts taking into account the area of use of the text, the purpose and the age and education of the readers:

a they are transcribed in freer texts - works of fiction, popular and publications (texts) intended for children and young people;

b in subject texts - in scientific literature, advertising, information publications (texts), official documents and special texts, authentic forms of personal names of other languages are written, but more common place names are usually transcribed, ...;

c specific, well-established traditional forms of personal names and place names are used in all cases, for example: England → Anglija, Poland → Lenkija, Warsaw → Varšuva;

d authentic personal names and place names of other languages are grammaticalized in the coherent text - the Lithuanian endings of the required inflection are added to them, ... Transcribed personal names and place names are usually grammaticalized both in the coherent and incoherent text, ..."

Hence, it is evident that a need to adapt foreign words does exist. It should also be noted that the same word can be transliterated differently. Let us take two examples from popular scientific literature on the famous Gordon Moore's Law. The surname Moore is transliterated as "Mūras" (with the long vowel "ū") in "Kaku, M. Physics of the Future: How Science Will Shape Human Destiny and Our Daily Lives by the Year 2100. Eugrimas, Vilnius, 2013. Translated to Lithuanian by Dr. L. R. Tamošiūnas. https://www.mokslofestivalis.eu/wp-content/uploads/2015/06/ateities_fizika.pdf" and "Muras" (with the short vowel "u") in "Floridi, L. The Fourth Revolution: How the Infosphere is Reshaping Human Reality. Eugrimas, Vilnius. Translated to Lithuanian by D. Urbonė, 2018. http://www.lma.lt/uploads/files/KNYGOS/Ketvirtoji_revoliucija___.pdf". This is because the transliteration rules are not strictly defined, everything depends on the person transliterating the word. This is not what we would like. Automatic transliteration rules would facilitate the adaptation process and would help avoid differently transliterated words.

3.4. Transliteration Rules in Philological Sources

Rules for transliteration of proper nouns into Lithuanian from as many as 32 languages were published in [20]. The following languages are covered: Albanian, Ancient names transliterated into Latin and Greek, Bulgarian, Chinese names transliterated into Latin and Russian, Czech, Danish, Dutch, English, Estonian, Finnish, French, German, Greek, Hungarian, Icelandic, Italian, Japanese names transliterated into Latin and Russian, Latvian, Moldavian, Norwegian, Polish, Portuguese, Romanian, Russian, Serbo-Croatian, Slovak, Spanish, Swedish, Turkish.

For 31 languages, the rules are written in the following tabular form: Original letters - Position of letters - Lithuanian letters - Examples in original and Lithuanian languages. Many tabular entries contain an empty "Position of letters". This means that a single context free rule is enough to process a particular letter, e.g., in Polish [W] → "V". There are also many entries with certain conditions specified in the column "Position of letters". Usually, these conditions are easy to test. In this case, several context sensitive rules and one context free rule are needed, e.g., the rules for the Polish letter "Ą" are as follows:

[Ą]P → "OM" (e.g., SKĄPE → SKOMPĖ)

[Ą]B → "OM" (DĄBROWSKI → DOMBROVSKIS)

[Ą] → "ON" (MĄCZYŃSKI → MONČINSKIS, WĄSOWICZ → VONSOVIČIUS).

There are only few entries where conditions specified in the column "Position of letters" would be hard to check, e.g., in Norwegian:

[CH] → "Č" in names of English origin,

[CH] → "Š" in names of French origin,

[CH] → "K" in names of German origin.

The above assumes that a set of simple handwritten rules is sufficient to transliterate words into Lithuanian from 31 languages. That is why now, in the age of "deep learning algorithms", it is still worth talking about context-free and context-sensitive rules.

The only exception is English, where the transliteration rules are written in the following tabular form: English letters – English pronunciation - Lithuanian letters – Examples. I.e., merely all possible substitutions of a particular letter are listed, examples are given, however, the conditions for selecting the appropriate substitution are not specified, or such conditions (e.g., accented/unaccented syllable) are specified which are hard to be verified on the basis of the written word. Some examples are provided below:

[A] → "A" (e.g., FAST → FASTAS)

[A] → "E" (FAREHAM → FERAMAS)

[A] → "I" (VILLAGE → VILIDŽAS, in an unaccented syllable)

[A] → "O" (SCAFELL → SKOFELIS)

[A] → "EI" (AVON → EIVONAS, usually in an open syllable)

Consequently, it is impossible to apply directly the rules specified in this form. The idea occurred to take all the examples provided here and try to automatically generate the rules from them that would contain contextual information about when and what rule to apply. This would allow rules of the same form to be used for all 32 languages mentioned in [20].

Since the aforementioned book [20] was published many years ago (in 1986), in recent years new editions of the transliteration rules approved by the State Commission of the Lithuanian Language have been published for some languages, such as English [17], Polish [18].

Next, this paper presents an automatic rule generation algorithm applied to the English language. As one of the languages with simple transliteration, Polish was chosen, and the transliteration rules for it were written manually (see section 6.1).

4. English Data

As for the data, only two data sets of minimal size are available. One set of English-Lithuanian word pairs (let us call it set X1) was taken from [17]. It provides from one to 12 examples to illustrate every possible transliteration; however, examples intended for other letters can also be used to transliterate a specific letter. For example, the rule [B] → “B” (BARBER → BARBERIS) is for the letter B, however, we can also see from this example that [A] → “A”. Set X1 contained a total of 751 English-Lithuanian word pairs.

Another analogous set Y1 was taken from [20] pp. 11-30. Two possible transliteration options were provided for five English words; one option was chosen for each word. Set Y1 contained a total of 476 pairs.

Sets X1 and Y1 partially overlap, however, the overlap is not large, only 132 words. All subsequent experiments were performed without taking account of a partial overlap of the sets.

Each of these two sets fully covers all possible transliteration options of all English letters. Therefore, it seemed logical for us to take one set for rulemaking and the other one for testing, and then swap them. If we had only one set, having taken even a few samples for testing, we would run the risk of creating an incomplete set of rules, i.e., it might happen that no rules are created to transliterate the cases illustrated by these samples.

Another problem that had to be solved was that some of the names were made up of several words. Spacing between the words made it difficult to make the rules. According to paragraph 9.3.15 of [19], the Lithuanian ending is usually added only to the last word, while other words in this group function as parts of a word. It was decided to combine such word groups by removing spaces. E.g., LONG ISLAND → LONG AILANDAS was brought together into LONGISLAND → LONGAILANDAS. In cases where Lithuanian endings were added to several words in a group, the group was divided into separate words. E.g., ROBERTS-AUSTEN → ROBERTSAS-AUSTENAS was split into ROBERTS → ROBERTSAS, AUSTEN → AUSTENAS. The word merging was applied to 23 cases in set X1, and to 17 cases in set Y1.

All examples were converted to uppercase letters to simplify processing. All pairs of examples were processed manually, assigning 0, 1 or more Lithuanian letters to each letter of the English word. An additional character (underscore) was added to the end of the English word, which means that a Lithuanian ending must be added at this point. E.g.,

D	Y	B	A	L	L	_
D	AI	B	O	-	L	AS

Let us refer to the datasets in the above-described format as X2 and Y2, respectively.

5. Algorithm

This section describes the automatic rule creation algorithm in detail. Making use of the above-described datasets X2 and Y2, new format data records were generated which contained the following information: the current letter of the English word, the left context of the current letter (6 letters to the left of the current letter; if the context was narrower than 6 letters, underscores were added), the right context of the current letter (6 letters to the right of the current letter, if the context was narrower than 6 letters, underscores were added), the string of Lithuanian letters corresponding to the current letter. The following 7 records were obtained from the previously mentioned English word DYBALL (see Table 2).

As many as 6173 records of the new format were created from set X2 (set X3) and 3895 records were cre-

Table 2
Sample records obtained from the English word DYBALL

No.	Left context	Current English letter	Right context	Lithuanian letters
1	______	[D]	YBALL_	“D”
2	_____D	[Y]	BALL_	“AI”
3	____DY	[B]	ALL_	“B”
4	___DYB	[A]	LL____	“O”
5	__DYBA	[L]	L_____	“”
6	_DYBAL	[L]	______	“L”
7	DYBALL	[_]	______	“AS”

ated from set Y2 (set Y3). To illustrate the creation of the rules, 12 records from set X3 with the current letter [A] were selected. Now let us sort the records further by the rightmost letter of the left context and place them into groups where the context letter is the same. The records are given in Table 3. For each group it is verified whether all its records contain the same string of Lithuanian letters. If so, a rule can be drawn up on the basis of the group. For example, from Table 3, we can already draw up the following two rules from rows 4-5 and 12, respectively:

$$\mathrm{D[A]} \rightarrow \text{“A”} \tag{1}$$

$$\mathrm{V[A]} \rightarrow \text{“EI”} \tag{2}$$

However, using these rules, only three out of 12 records could be processed (they are marked with the plus sign in the right-hand column in Table 3).

Table 3

Sample records in the format used in sets X3 and Y3, sorted by the rightmost letter of the left context and grouped

No.	Left context		Current English letter	Right context	Lithuanian letters
1	__PIC	C	[A]	DILLY_	“A” -
2	____S	C	[A]	FELL__	“O” -
3	__CHI	C	[A]	GO____	“A” -
4	___RE	D	[A]	DDER__	“A” +
5	____I	D	[A]	HO____	“A” +
6	__PHI	L	[A]	DELPHI	“A” -
7	__VIL	L	[A]	GE____	“I” -
8	___ST	R	[A]	DFORD_	“A” -
9	_COYT	R	[A]	HEN___	“A” -
10	_____	R	[A]	LPH___	“A” -
11	_____	R	[A]	LSTON_	“O” -
12	FRUIT	V	[A]	LE____	“EI” +

It is obvious that the data records can be sorted according to the first letter of the right context, see Table 4.

Having sorted and grouped the records, it became evident that we can draw up the following 3 rules from rows 1-4, 5 and 8-9 in Table 4:

Table 4

Sample records in the format used in sets X3 and Y3, sorted by the leftmost letter of the right context and grouped

No.	Left context	Current English letter	Right context		Lithuanian letters
1	__PICC	[A]	D	ILLY_	“A” +
2	___RED	[A]	D	DER__	“A” +
3	__PHIL	[A]	D	ELPHI	“A” +
4	___STR	[A]	D	FORD_	“A” +
5	____SC	[A]	F	ELL__	“O” +
6	__CHIC	[A]	G	O____	“A” -
7	__VILL	[A]	G	E____	“I” -
8	____ID	[A]	H	O____	“A” +
9	_COYTR	[A]	H	EN___	“A” +
10	_____R	[A]	L	PH___	“A” -
11	_____R	[A]	L	STON_	“O” -
12	FRUITV	[A]	L	E____	“EI” -

$$\mathrm{[A]D} \rightarrow \text{“A”} \tag{3}$$

$$\mathrm{[A]F} \rightarrow \text{“O”} \tag{4}$$

$$\mathrm{[A]H} \rightarrow \text{“A”} \tag{5}$$

These three rules allow 7 out of 12 records to be processed. Thus, as it is not known in advance whether the transliteration rules are influenced more by the left or the right context, it is worthwhile to extend the context both to the left and right, then evaluate the result based on some criteria and choose a better context extension direction at this step (in this case to the right). Various criteria are possible: minimization of the number of rules (groups), maximization of the number of processed records, etc. The latter was used in our work. There might be a case where the same estimates are obtained for both the left and right context extensions. Then you can always take a fixed direction, choose the direction at random, try to ensure that the width of the context should be as even (balanced) as possible in both directions. We used the latter criterion. If the context was already balanced, we extended the context to the right.

If there are different strings of Lithuanian letters inside the group of records (see lines 6-7 and lines 10-12 in Table 4, the process is repeated within the group: we extend the context by one character to the left and right, sort, evaluate, choose a better direction of the context extension, where possible, draw up rules, and so on. For example, from rows 6-7 in Table 4, we could draw up rules equally successfully by extending the context both to the left and to the right. Extending the context to the left would give us the rules.

C[A]G → “A” (6)

L[A]G → “I” (7)

Extending the context to the right we would get the following:

[A]GO → “A” (8)

[A]GE → “I” (9)

The rules from lines 10-12 in Table 4 in the next step will be obtained only by extending the context to the right. In this case, the 3 rules will be as follows:

[A]LP → “A” (10)

[A]LS → “O” (11)

[A]LE → “EI” (12)

In addition, the rules can be combined and simplified. This does not only reduce the number of rules but also handles the cases that were not found in the data. Let us take three most recently created rules 10-12. Suppose that we want to process a word that contains a letter in a previously unseen context, such as [A]LB. None of the rules will fit. Let us choose one rule and reduce its context by one letter. Let us assume that this is rule 10. In general, it makes sense to choose the most often used rule here.

[A]L → “A” (13)

When transliterating an English word, rules 11 and 12 that have the broadest context must be examined first, and the reduced rule 13 will be applied in all other cases if rules 11 and 12 do not fit.

Similarly, rules 3-5 can be simplified to two rules:

[A]F → “O” (14)

[A] → “A” (15)

Suppose the current letter has index 0, the right and left context letters have indices 1, 2, ..., and -1, -2, ... respectively. The above-described algorithm in pseudocode is depicted in Algorithm 1:

Algorithm 1: automatic rule creation algorithm

Input: records

Output: rules

```
procedure P
for each group do
  if all records contain the same string of
    Lithuanian letters then
    create a rule from this group
  else
    sort records by the letter[L-1] and group them
    for each group do
      if all records contain the same string of
        Lithuanian letters then
        count them
      end if
    end for
    A = number of counted records
    sort records by the letter[R+1] and group them
    for each group do
      if all records contain the same string of
        Lithuanian letters then
        count them
      end if
    end for
    B = number of counted records
    if A > B then
      L=L-1
      sort records by the letter[L] and group them
      call procedure P for this group
    else
      R=R+1
      sort records by the letter[R] and group them
      call procedure P for this group
    end if
  end if
end for
end procedure

main program
sort all records by the letter[0] and group them
L = 0, R = 0
call procedure P for all records
return rules
```

Rule creation and testing algorithms were implemented as a computer program in C++. To make the set of rules available online, the rules have been ported to JavaScript. Software for transliteration of proper nouns from English and Polish to Lithuanian is available at: https://klevas.mif.vu.lt/~pijus/Lietuvinimas/.

The software interface has an Input box, an Output box, an Enter button, and two comboboxes: Language and Gender. It means that the gender of the word to be transliterated should be specified in advance. Gender defines the endings added to the Lithuanian word. If the word ends with a consonant, according to paragraphs 9.3.6, 9.3.7 and 9.3.9 of [19], endings "-as", "-is" or "-ius" should be added to the word of masculine gender and no ending should be added to the word of feminine gender. The rules for adding male endings were created from the data, while the rules for the feminine gender were created manually due to insufficient data.

In order for masculine and feminine words to be applied different rules, they have been given a feature of the current letter with the following possible values: masculine, feminine, both.

6. Experimental Results

Typical evaluation measures for machine transliteration are word accuracy and character accuracy [8]. Word accuracy (WA) is defined as the ratio of correctly transliterated words to all words tested. The character accuracy (CA) is usually defined based on the length of the word in the target language and the number of insertions, deletions, and substitutions required to change the system-transliterated word to the expected one. When transliterating a word using the algorithm we have developed, the number of rules applied corresponds to the length of the word in the source language, so we defined the character accuracy as a percentage of the correctly applied rules.

6.1. Transliteration from Polish

Based on [18], a set of 63 rules was compiled manually. The set contains a total of 37 context free rules (32 letters plus 'q', 'v', 'x', underscore and space) and 26 context sensitive rules. These rules cover all possible transliterations of Polish letters. Testing was carried out on all 129 examples found in [18]. Three words were added to the list of exceptions: ADELAJDA → ADELAIDA (an incorrect spelling according to the general Polish rules would be ADELIAIDA), JOLANTA → JOLANTA (JOLIANTA), JERZY → JEŽIS (JEŽAS). The remaining 126 words were transliterated without errors, i.e., with the 100% accuracy. In order to use these rules to transliterate any Polish proper nouns, the traditional forms of personal names and place names should be included in the dictionary of exceptions, e.g., POLSKA → LENKIJA, WARSZAWA → VARŠUVA, BIAŁYSTOK → BALSTOGĖ, KRAKÓW → KROKUVA, GRUNWALD → ŽALGIRIS. Compiling a complete list of traditional forms is not the aim of this work.

6.2. Transliteration from English

Using set X3 (751-word pairs, 6173 letters with contexts), a set of rules containing 2022 rules was automatically generated. Testing was performed using English words from set X1 and comparing the results with the Lithuanian ones from X1. The rules have been proved to be 100% correct.

Testing was performed with set Y1: 288 out of 476 words contained errors. No rules were found for as many as 382 letters. It was decided to supplement the set of rules with 23 context-free rules, i.e., rules of type [A] → "A", which are applied in case no context-specific rule fits. Again, testing with set Y1 was performed. Let us call this experiment X3Y1. More detailed results are presented in Table 5, column 2.

The rules in the set were manually combined and simplified as described in Section 5. An optimized

Table 5

Accuracy of word and letter transliteration

Item	X3Y1	X3oY1	Y3X1
Number of rules	2045	950	1582
Number of words in the test set	476	476	751
Number of misspelled words	233	227	443
Word accuracy	51.1%	52.3%	41.0%
Number of letters in the test set	3895	3895	6173
Number of incorrect letters	416	391	708
Character accuracy	89.3%	90.0%	88.5%

set of 950 rules was obtained. Testing was performed with set X1, and it was ascertained that no errors occurred. Testing with set Y1 (let us call it X3oY1) was performed, the results are shown in Table 5, column 3. Optimization corrected the transliteration of 11 words but damaged the transliteration of 5 words.

Using set Y3 (476-word pairs, 3895 letters with contexts), a set of rules containing 1582 rules was automatically generated. Testing was performed using English words from set Y1 and comparing them with the Lithuanian words from Y1. It was ascertained that the rules were 100% correct.

Testing was performed with set X1: 552 out of 751 words were with errors. No rules were found for as many as 862 letters. Similarly, the set of rules was supplemented with 23 context-free rules and testing was performed again with set X1. Let us call this experiment Y3X1, the results are presented in Table 5, column 4. Manual optimization was not performed because the optimization effect was not significant in experiment X3oY1.

7. Discussion and Future Work

Regarding the transliteration of foreign names into Lithuanian, manually created context-sensitive and context-free rules work well for the Polish language and are expected to work for many other languages except English.

As seen from Table 5, the percentage of correctly transliterated English words is not impressive at all, but the results are comparable to those presented in [8] Table II (35-74%). However, if you look at the percentage of correctly spelled letters, 10% of errors is not much, the vast majority of errors are one error per word. Main reasons for such a large number of errors are as follows:

1 Datasets are quite different, created by different authors. In addition, set X1 was created more than 30 years later than set Y1. During that time, a certain understanding of how some English words should be transliterated could have changed. Sets X1 and Y1 contain 19 English words, which are transliterated differently. Some of the differences grouped by rules are presented in Table 6 (the first two examples in each group). Amendments to the transliteration rules are also reflected in other words, which are different in sets X1 and Y1. See Table 6 for such examples (the last 2-3 examples in each group).

Table 6
Examples of different transliteration in sets X1 and Y1

English word	Set X1, Lithuanian word	Set Y1, Lithuanian word
ANGUS	ANGUSAS	ANGUS
DOUGLAS	DAGLASAS	DAGLAS
STROLLAMUS	STROLAMUSAS	-
CYRUS	-	SAIRUS
DURAND	DJURANDAS	DIURANDAS
UNION	JŪNJONAS	JŪNIONAS
BURE	BJURAS	-
BRIAN	BRAJANAS	-
DORIAN	-	DORIANAS
LIVERPOOL	LIVERPULIS	LIVERPŪLIS
MOORE	MURAS	MŪRAS
GOORNEY	GURNIS	-
POORE	-	PŪRAS
LONG ISLAND	LONG AILANDAS	LONG AILENDAS
WORDSWORTH	VORDSVORTAS	VERDSVERTAS
WORDINGHAM	VORDINGAMAS	-
LETCHWORTH	-	LEČVERTAS

2 Some English letters have lots of pronunciation variants, therefore transliteration is very complicated. E.g., a large number of rules were automatically generated for these letters: E – 298 rules, A – 259, O – 208, I – 168, H – 143, U – 122.

3 The transliteration of some similar words is very different. The difference can be accounted for at least by the fact that they were transliterated by different people, at different times, following a different sense of language. E.g., WORCHESTER → VUSTERIS, LEICESTER → LESTERIS, GLOUCESTER → GLOSTERIS vs. COLCHESTER → KOLČESTERIS, CIRENCESTER → SAIRENSESTERIS.

4 The rules we created are based merely on the context. It might be that in some cases the transliteration depends on some other features rather than on the context, for example, the letter E in similar contexts was sometimes omitted (COLERIDGE

→ KOLRIDŽAS, ISLEROYAL → AILROJALIS), sometimes it was retained (BILLERICAY → BILERIKIS).

The number of errors can be reduced and the rule set improved by the following:

1. The data sets used for rulemaking include only the minimum number of samples of each case. A larger set of English-Lithuanian word pairs validated by philologists and taken from some dictionaries of proper nouns, should be used to generate the rules. It should also be recalled that the words from which the rules were generated can be transliterated into Lithuanian with 100% accuracy.
2. The generated set of rules can be further adjusted manually, for example by adding new obvious contexts. Suppose that the rule A[W]A → "V" was created from OTTAWA → OTAVA. The rule with another vowel in the right context can be expected to be also correct, e.g., A[W]E → "V". The set of rules can be adjusted according to some regularities observed by philologists. In general, it is very difficult to draw up the rules from scratch, it is easier to adjust a generated set.
3. The set of rules can be supplemented with the attributes of the current letter, which are calculated in advance using some other means rather than the context. For example, a gender attribute was added to the last letter of an English word and then different endings can be added to a Lithuanian word based on it.
4. Some words are transliterated according to special rules, i.e., they are exceptions. Some of these words should be put on the list of exceptions and not be used to generate rules, as such words can make the set of rules very complicated. For example, we have word pairs: BRIGHTON → BRAITONAS, PIRBRIGHT → PERBRAITAS, KIRKCUDBRIGHT → KERKŪBRIS. We see that the third example does not correspond to the first two at all. To transliterate a word into Lithuanian, you should first look for it in the list of exceptions, and if it is not found there, apply the rules.
5. The automatic rulemaking algorithm can be applied to any language pair as long as there is a set of word pairs in the source and target languages.

8. Conclusions

Two sets of rules for the transliteration of Polish and English proper nouns into Lithuanian were created. The rules for Polish were created manually, they are 100% accurate, provided traditional forms are included in the exceptions list. The algorithm for automatic generation of transliteration rules is presented in the work. The rules are created from English-Lithuanian word pairs aligned at the letter level. The algorithm allows 100% accuracy in the training (rulemaking) set to be achieved. About 50% word accuracy and about 90% character accuracy were achieved in the test set. Low accuracy could be explained by differences in rulemaking and testing sets, the complexity of English pronunciation, and the fact that transliteration rules are not well-established as transliteration is different in similar situations. Accuracy would be enhanced by increasing the training set, removing inconsistencies between data records and compiling a list of exceptions. An automatically generated set of rules can serve as a basis for further manual tuning, and rules can be supplemented with the attributes pre-calculated by means of other methods.

References

1. Ansari, M. Z., Ahmad, T., Beg, M. M. S., Ahmad, F. Hindi to English Transliteration Using Multilayer Gated Recurrent Units. Indonesian Journal of Electrical Engineering and Computer Science, 2022, 27(2), 1083-1090. https://doi.org/10.11591/ijeecs.v27.i2
2. Bazinys, A. Algorithms for Rewriting Foreign Names into Lithuanian. 2020. Bachelor theses. Vilnius university.
3. Chatterjee, S., Sarkar, K. Machine Transliteration Using SVM and HMM. International Journal of Advanced Intelligence Paradigms, 2021, 19(1), 3. https://doi.org/10.1504/IJAIP.2021.114584
4. Divay, M., Vitale, A. Algorithms for Grapheme-Phoneme Translation for English and French: Applications for Database Searches and Speech Synthesis. Computational Linguistics, 1997, 23, 495-523. https://aclan-

thology.org/J97-4001.pdf. Accessed on December 9, 2021.

5. Gani, M. Anglicizing Italian. English Today, 2007, 23, 40-41. https://doi.org/10.1017/S0266078407001071

6. Goyal, K., Abbas, M., Goyal, V., Saleem, Y. Forward-backward Transliteration of Punjabi Gurmukhi Script Using n-gram Language Model. ACM Transactions on Asian and Low-Resource Language Information Processing, 2022. https://doi.org/10.1145/3542924

7. Kang, B.-J., Choi, K.-S. Automatic Transliteration and Back-transliteration by Decision Tree Learning, 2000. http://www.lrec-conf.org/proceedings/lrec2000/pdf/227.pdf. Accessed on December 9, 2021.

8. Karimi, S., Scholer, F., Turpin, A. Machine Transliteration Survey. ACM Computing Surveys, 2011, 43, 17. https://doi.org/10.1145/1922649.1922654

9. Kasparaitis, P. Transcribing of the Lithuanian Text Using Formal Rules. Informatica, 1999, 10(4), 367-376. https://doi.org/10.3233/INF-1999-10401.

10. Lindén, K. Multilingual Modeling of Cross-lingual Spelling Variants. Information Retrieval, 2006, 9, 295-310. https://doi.org/10.1007/s10791-006-1541-5

11. Pinnis, M. Bootstrapping of a Multilingual Transliteration Dictionary for European Languages. In: Human Language Technologies - The Baltic Perspective: Proceedings of the Sixth International Conference Baltic HLT, 2014. https://doi.org/10.3233/978-1-61499-442-8-132.

12. Pouliquen, B., Steinberger, R., Ignat, C., Temnikova, I., Widiger, A., Zaghouani, W., Žižka, J. Multilingual Person Name Recognition and Transliteration. Corela, 2006, 3(3). https://doi.org/10.4000/corela.1219

13. Raškinis, G., Paškauskaitė, G., Saudargienė, A., Kazlauskienė, A., Vaičiūnas, A. Comparison of Phonemic and Graphemic Word to Sub-Word Unit Mappings for Lithuanian Phone-Level Speech Transcription. Informatica, 2019, 30(3), 573-593. https://doi.org/10.15388/Informatica.2019.219

14. Roy, A. K., Paul, A., Purkayastha, B. S. Statistical and Syllabification Based Model for Nepali Machine Transliteration. In: Mukhopadhyay, S., Sarkar, S., Dutta, P., Mandal, J. K., Roy, S. (eds) Computational Intelligence in Communications and Business Analytics. CICBA 2022. Communications in Computer and Information Science, 2022, 1579. Springer, Cham. https://doi.org/10.1007/978-3-031-10766-5_2

15. Skripkauskas, M., Telksnys, L. Automatic Transcription of Lithuanian Text Using Dictionary. Informatica, 2006, 17(4), 587-600. https://doi.org/10.15388/Informatica.2006.157.

16. Talafha, B., Abuammar, A., Al-Ayyoub, M. Atar: Attention-based LSTM for Arabizi transliteration. International Journal of Electrical and Computer Engineering, 2021, 11(3), 2327-2334. https://doi.org/10.11591/ijece.v11i3

17. The State Commission of the Lithuanian Language. Rules for Transcription of Proper Names from English into Lithuanian, 2021. http://www.vlkk.lt/media/public/file/Svetimzodziai/anglu-patvirtinta-VLKK-2021-09-30.pdf. Accessed on December 9, 2021.

18. The State Commission of the Lithuanian Language. Rules for Writing Lithuanian Polish Surnames and Names in the Passport of a Citizen of the Republic of Lithuania, approved on 28 March 1991 by a resolution of the meeting. http://www.vlkk.lt/aktualiausios-temos/svetimvardziai/perrasa-is-lenku-kalbos. Accessed on December 9, 2021.

19. Vladarskienė, R., Zemlevičiūtė, P. Spelling of the Lithuanian Language. Rules, Comments, Tips. Lithuanian Language Institute publishing house, Vilnius, 2022. https://vlkk.lt/media/public/file/Nutarimai/Rašyba_2022.pdf. Accessed on September 5, 2022.

20. Writing of non-Lithuanian proper names in the "Lithuanian Soviet Encyclopaedia". Vyriausioji enciklopedijų redakcija, Vilnius, 1986.