TRANSLIT: A Large-scale Name Transliteration Resource

Fernando Benites, Gilbert François Duivesteijn, Pius von Däniken, Mark Cieliebak
Zurich University of Applied Sciences, Deep Impact
Switzerland
benf@zhaw.ch, gilbert@deep-impact.ch, vode@zhaw.ch, ciel@zhaw.ch

Abstract

Transliteration is the process of expressing a proper name from a source language in the characters of a target language (e.g. from Cyrillic to Latin characters). We present TRANSLIT, a large-scale corpus with approx. 1.6 million entries in more than 180 languages with about 3 million variations of person and geolocation names. The corpus is based on various public data sources, which have been transformed into a unified format to simplify their usage, plus a newly compiled dataset from Wikipedia. In addition, we apply several machine learning methods to establish baselines for automatically detecting transliterated names in various languages. Our best systems achieve an accuracy of 92% on identification of transliterated pairs.

Keywords: Transliteration of Names, Name Variant Discovery, Multi-lingual, Language Resource

1. Introduction

Identifying named entities (e.g. persons or locations) is a crucial task in many important applications, from anti-money laundering to reputation monitoring to online surveillance of terrorism activities. If the texts are written in languages with different characters (e.g. Cyrillic, Arabic, Latin etc.), this becomes particularly challenging: names and places that originate from one language have often numerous different spellings in another language. For instance, the Russian name Горбачёв can be written in Latin characters as Gorbachev, Gorbachov, and Gorbachyov, and the same applies to many other names or geoleocations. The task of transliteration focuses on the transformation of a word (normally a proper noun) into a language which has another phonology inventory and a different alphabet. Note that transliteration differs significantly from transcription, which focuses on the proper spelling of a foreign language sound into the target language.

In principle, there exists a set of ISC rules which can be applied when converting a proper noun from a source language into a different target language. Unfortunately, they neither cover all language pairs (which is especially true for low-resource languages), nor are they always consistently applied. Note that these rules are most suited to be applied manually, but not automatically. This creates a variety of (possible) transliterations which need to be considered when searching for entities. Thus, there is a strong need for large-scale corpora and high-quality methods to train automatic machine transliteration.

Automatic Machine Transliteration

There exist several approaches for automatic machine transliteration, and also some datasets on which automatic methods can be trained and evaluated (see Section 2 for more details). However, for most language-pairs there exist only little parallel data, or none at all. This is in particular true for low-resource languages. In addition, the few existing datasets have different formats and are usually limited to one language pair, which makes them unsuitable for developing cross- and multi-lingual solutions for transliteration.

Our Contribution

We present TRANSLIT, a new dataset for transliteration of person names and geolocations which merges transliterations from several data sources into a unified format. The resulting corpus contains about 1.6 million entries in more than 180 languages, and approx. 3 million name variations. TRANSLIT combines the existing public datasets JRC-Names (Ehrmann et al., 2017), Geonames (http://www.geonames.org), SubWikiLang (Merhav and Ash, 2018), and En-AR (Rosca and Breuel, 2016), and extends it with Wiki-lang-all, a newly created dataset where we automatically extracted potential transliterations from Wikipedia, following the methodology of (Liu et al., 2016) and (Merhav and Ash, 2018).

In this study, we present the data aggregation methods and the corpus details. Further, we use the corpus to train string-based and deep-learning methods (n-grams with SVM and random forests, siamese networks and convolutional networks (CNNs)) for automatic recognition of transliterated names.

Our main contributions are as follows:

• Merging existing datasets into a unified format
• Scavenging Wikipedia for name transliterations in arbitrary languages
• Building strong baselines for recognition of transliterated names and name variations

2. Related Work

Automatic transliteration is a field which is actively conducted since 20 years. Transliteration is especially a main concern for countries with a different script from Latin or that have multiple languages that use different scripts, like China or India.

A survey about transliteration can be found in (Karimi et al., 2011), which divided at that time the approaches into
phonetic, grapheme (spelling) and hybrid transliteration. A more recent survey is provided in (Prabhakar and Pal, 2018) which also handles NMT approaches.

Mani et al. compare in (Mani et al., 2013) a manual rule-based phonetic approach between source and target language against a monolingual machine-learning one. The results show that the latter approach produces much higher F-scores. In (Murat et al., 2017) Uyghur-Chinese transliteration was investigated using a semantic knowledge approach. Using gender detection and performance on language origin a probabilistic model could achieve a remarkable improvement in transliteration.

(Weichselbraun et al., 2019) examined the problem of name variation, i.e. if there are multiple transliterations for the same entity, which is an essential task for named entity recognition when linked to transliteration. The study applies an entropy-based approach to identify ambiguous generated name variations.

(Liu et al., 2016) examined the use of bidirectional LSTMs in a sequence to sequence (seq-2-seq) manner to the problem of transliterating jp-en, achieving very good results. Similarly, (Merhav and Ash, 2018) applied seq-2-seq to the problem, comparing LSTMs and Transformers, achieving better results with the later. In this context, the use of convolution neural networks in a siamese manner proved to be useful (Rama, 2016).

(Mahsuli and Safabakhsh, 2017) applied deep learning to transliterate between English and Persian. They used a sequence-to-sequence (seq-to-seq) architecture with attention, which usually is a good baseline for translation (Luong et al., 2015).

Also (Rosa and Breuel, 2016) uses a seq-to-seq with attention on an English to Arabic dataset achieving good results.

(Rama, 2015) used an SVM with string similarity features to solve the cognate identification of words which is a related problem to transliteration. In a subsequent study (Rama, 2016), convolutional siamese networks were used achieving mostly worse results.

The survey (O’Horan et al., 2016) discusses the problems of multi-lingual settings, specifically typological resources. This examines how the low-resource languages can profit from high-resource ones through a systematic use of typology. Although, this is relevant to us by enabling us to assess the transliteration quality of a model, it can only be used in a setting where target and source language are fixed, but we aim in this study at a more flexible setting.

There are some studies which focus on the problem of low resources for transliteration, since the names being incorporated in a low-resource language might not often occur in texts either. (Wu and Yarowsky, 2018) compares different system using bible names across 591 languages.

The NEWS 2018 Named Entity Transliteration Shared Task (Chen et al., 2018) used many news articles which are copyrighted and therefore not freely available. In contrast, we compiled here a corpus which is distributed under a Creative Commons license.

3 A prominent example was Muammar Mohammed Abu Minyar al-Gaddafi with about 140 different spellings.

3 We applied the library langdetect (https://pypi.org/project/langdetect/) on the words/names themselves 55 languages were detected.

3 Many are redundant or misspelled.
The other two datasets are based on name transliteration research studies. We now describe how we gathered Wiki-lang-all.

3.2. Construction of Wiki-lang-all

For construction of subset Wiki-lang-all, we used the same methodology for extracting transliterations from Wikipedia as in (Liu et al., 2016) and (Merhav and Ash, 2018), namely, we used inter-language tags and extracted the related words. We collected our data from the English Wikipedia data dump from 2017-08-07. We parsed every Wikipage from this dataset and searched in the first 6000 characters (approx. typical size of the abstract) for the "lang-" tag, which declared some passage in a foreign language, usually to describe the term in its original script/language. In contrast to (Merhav and Ash, 2018), we used any language in the lang-tag of Wikipedia next to the word. Note that this results in some noise in the dataset, since also translations and mislinked concepts may enter the data. This process resulted in 122k entities with 144k name variations. The mean number of characters per name of SubWikiLang was 10.3 which is lower than JRC (En-Ar is only about single names, i.e. either first name or surname). Wiki-lang-all has the longest, since many long titles are also included.

3.3. Data Merging

Each dataset was in a different format, favoring duplicates. However, finding duplicates in over 3 millions of transliterations is a huge effort, especially if a reliable machine learning method is not available. We unified the underlying datasets using the following procedure⁶. We gathered for every dataset the different variations/transliterations of a name spelling also through script, creating meta entities. This was performed by merging the names which were lexicographically equal. In each merge we saved the references to the original entities. Afterwards, the references were collected and the meta entities were merged in a final step, grouping the different name variations. As in some datasets, the English transcription was selected as key, we used UUIDs to differentiate different persons with the same name. We also added the language to the names in the beginning when available. This resulted in the format: UUID → {"en_name1", "zh_name2"...}. The produced file was stored in the JSON format.

4. Properties of TRANSLIT

4.1. Dataset Statistics

A summary of characteristics of the dataset is displayed in Table 1. As one can see, the TRANSLIT dataset is much larger than any one of the constituting ones. In Figure 1 a histogram of name length in characters is displayed. Most names are 13 characters long (282’831 in total). In general, lengths of 13 and fewer characters cover already more than 50% (1’706’363) of all names. One can also see some names with more than 50 characters, which are most likely

5. Experiments

In certain named entity recognition tasks, it is important to detect names from a given list (e.g. gazetteers, politically exposed people (PEP) List). We produced a similar setup, in order to explore the potential of TRANSLIT, in which we performed experiments on how to detect name variation/transliteration across multiply languages automatically. We considered two settings, one where the confusion of names is rather unlikely, and one targeting the recall where names are similar and hence easily confused. Although it is possible to use the dataset to train a system for generating transliterations, we were also interested in low-resource language pairs for which there is not enough data to train such a system.

5.1. Setup

5.1.1. Bigram Difference for Name Variant Classification

We applied a new methodology: The use of character bigrams for identifying similar dialects was proven to be successful (Benites et al., 2018), especially if the notation is very similar to the phonetics. Therefore, we transform with unidecode⁷ source and target name to the same script (Latin), and count the character bigrams. We then subtract target bigram count from source bigram count to find common occurrences. Finally, we also calculate the Jaro similarity score (Jaro, 1989) between these names as a further feature. With these features we train a Random Forest (RF) (Breiman, 2001) classifier from the scikit-learn library (Pedregosa et al., 2011). We also compare the performance against a support machine classifier with a linear kernel (Fan et al., 2008). For the SVM, we used a TF-IDF bigram feature matrix.

5.1.2. Further Methods

The transliterations can be seen as a similar problem to finding cognates, i.e. words with a common root (in different languages). We employ the method from (Rama, 2016) to test our dataset, and compare the results. We also applied equivalent⁸ methods as (Rosca and Breuel, 2016) (LSTMs) by using a CNN with two layer convolution with 128, 64 filters respectively, maxpooling layer after each convolution layer and 3 dense layers (150, 100, 1) with relu and sigmoid activation.

⁶We assumed that there were multiple references to a name in the different datasets.

⁷https://github.com/avian2/unidecode

⁸LSTMs achieve for certain tasks, such as by short-text classification similar scores as CNNs, however need much longer training time.
in the last layer as activation function. We trained each network for 15 epochs.

5.2. Simple Name Variation Identification

We used 200’000 randomly chosen pairs (source/target) of name variations in a 50% split between positive and negative (i.e. 100’000 each). The results are depicted in Table 3. In this experiment, we used only the standard machine learning approaches, since this should be an easier task. We can see that the RF is much better, even against the TF-IDF method. The evaluation is based on accuracy, i.e., was the pair a transliteration of each other or not and was performed in a cross-validation setting (thus, the variance). This is comparable to the “Word Accuracy in Top-1” method used in the NEWS 2018 Named Entity Transliteration Shared Task (Chen et al., 2018), though we do not use generation, but choose an existing transliteration from the dataset.
Classification Method | Accuracy
--- | ---
baseline/Majority Class | 0.5
RF | 0.927 ± 0.002
SVM | 0.713 ± 0.003
SVM-TF-IDF on chars | 0.686 ± 0.002
Siamese | 0.871 ± 4.0e−5
CNN | 0.878 ± 8.6e−6

Table 4: CV classification results with variance for Extended Name Variant Identification Stage. RF: Random Forests

<table>
<thead>
<tr>
<th>Dataset</th>
<th>RF</th>
<th>SVM</th>
<th>SVM-TF-IDF</th>
</tr>
</thead>
<tbody>
<tr>
<td>JRC</td>
<td>0.952 ± 0.002</td>
<td>0.883 ± 0.005</td>
<td>0.803 ± 0.016</td>
</tr>
<tr>
<td>Geonames</td>
<td>0.726 ± 0.049</td>
<td>0.560 ± 0.099</td>
<td>0.530 ± 0.081</td>
</tr>
<tr>
<td>SubWikiLang</td>
<td>0.757 ± 0.019</td>
<td>0.596 ± 0.059</td>
<td>0.537 ± 0.049</td>
</tr>
<tr>
<td>TRANSLIT</td>
<td>0.893 ± 0.000</td>
<td>0.713 ± 0.000</td>
<td>0.671 ± 0.000</td>
</tr>
</tbody>
</table>

Table 5: CV results with variance for each dataset with 21k generated pairs

5.3. Extended Name Variation Identification

In a second experiment, we wanted to assess how a harder selection of negative samples would influence the results. We generated further 100’000 samples by creating pairs from the names of randomly selected entities but the names needed to have a Jaro similarity score above 0.8. In Table 4 we see again, that RF was much better, but now the deep learning approaches were close, but still 0.05 points below the RF, with a negligible variance. The SVM TF-IDF based on characters approach produced the worst results in this setup.

6. Relevance between Datasets

In this experiment, we investigate if the created dataset TRANSLIT can improve the performance of the individual ones (JRC, Geonames, SubWikiLang). We selected, as in the previous experiment, 21k (7k positive, 7k negative and 7k similar) sample pairs for each dataset. We performed again cross-validation classification and measured the accuracy, as depicted in Table 5. One can see, that JRC is relatively easy to detect, where as Geonames is much harder.

We then added the 21k pairs from the TRANSLIT dataset to the training data of each dataset in each fold of the cross validation. The results are shown in Table 5. As one can see, there is not much difference, only that TRANSLIT is better recognized, as expected. We assume that the amount of samples is more important. Therefore, we investigate this hypothesis by selecting subsets of a 160k generated sample pairs from TRANSLIT. For that purpose, we checked the accuracy by increasing number of the training (1k,10k,20k,40k,80k,160k) again in a cross-validation setting. The results are depicted in Figure 3. One can see that the RF approach increases almost linearly with the logarithmic scale (between 10k and 160k). Especially, because of the multi-lingual nature of the problem, it is difficult to find prototype samples, that can represent many samples and so reducing the redundancy of the training data. This points to the fact, that a large dataset is indeed need.

7. Conclusions

We presented TRANSLIT, a new dataset for name transliteration and name variation detection, which merges and unifies several existing resources as well as a new one. The final dataset has 1.6 millions entities with 3 millions name variations/transliterations, which makes it - as far as we know - the largest resource of its kind. We also performed experiments on automatic transliteration detection across many languages, and achieved an accuracy of 92 % on a diffi-

9The variance of each run is not depicted, because they are too small to be depicted graphically.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>RF</th>
<th>SVM</th>
<th>SVM-TF-IDF</th>
</tr>
</thead>
<tbody>
<tr>
<td>JRC</td>
<td>0.953 ± 0.002</td>
<td>0.880 ± 0.004</td>
<td>0.762 ± 0.048</td>
</tr>
<tr>
<td>Geonames</td>
<td>0.728 ± 0.075</td>
<td>0.621 ± 0.102</td>
<td>0.546 ± 0.083</td>
</tr>
<tr>
<td>SubWikiLang</td>
<td>0.755 ± 0.021</td>
<td>0.623 ± 0.070</td>
<td>0.563 ± 0.075</td>
</tr>
<tr>
<td>TRANSLIT</td>
<td>1.000 ± 0.000</td>
<td>0.768 ± 0.000</td>
<td>0.918 ± 0.000</td>
</tr>
</tbody>
</table>

Table 6: CV results with variance for each Dataset with 21k generated pairs + 21k from TRANSLIT
We will publish the corpus and the scripts to build it on Github\footnote{\url{https://www.github.com/fbenites/TRANSLIT}}, as well as further experiments at. For future research, we will explore if the use of a multi-lingual transformer language model could improve the results.

8. Acknowledgements

We would like to thank Dandolo Flumini for the fruitful discussions and and his support with this project. The work was supported by the Swiss Commission for Technology and Innovation (CTI), Project No. 25256.1 PFES-ES “Libra”.

9. Bibliographical References


