MDL-Based Models for Transliteration Generation

Javad Nouri[1], Lidia Pivovarova[1,2], and Roman Yangarber[1]

[1] University of Helsinki, Department of Computer Science, Finland
[2] St.Petersburg State University, Russia

Abstract. This paper presents models for automatic transliteration of proper names between languages that use different alphabets. The models are an extension of our work on automatic discovery of patterns of etymological sound change, based on the Minimum Description Length Principle. The models for pairwise alignment are extended with algorithms for prediction that produce transliterated names. We present results on 13 parallel corpora for 7 languages, including English, Russian, and Farsi, extracted from Wikipedia headlines. The transliteration corpora are released for public use. The models achieve up to 88% on word-level accuracy and up to 99% on symbol-level F-score. We discuss the results from several perspectives, and analyze how corpus size, the language pair, the type of names (persons, locations), and noise in the data affect the performance.

1 Introduction

The task of machine transliteration involves mapping the representation of a word to another language, typically using a different alphabet, based on its sound or spelling, rather than its meaning. Transliteration is commonly applied to proper names, as well as to terms in rapidly growing areas, such as medicine or technology, [12]. Two principal applications for machine transliteration are machine translation and information search—multilingual information retrieval (IR), information extraction (IE), and named-entity recognition (NER). While in machine translation the goal may be to produce only one correct transliteration for each word, in other tasks we may wish to produce several possible transliterations, and merge different variants of the same name.

There are two main approaches to machine transliteration: transliteration generation and transliteration mining (discovery). In transliteration generation one builds a transliteration model, which takes a source named entity as input and produces its representations in the target language as output.

Transliteration generation can be considered in a broad sense as a special case of alignment and transduction of words. The similarity of machine transliteration and alignment of etymologically-related words have been observed by [14], who applied the same model, a finite state transducer, for both tasks. In [1], transliteration is mentioned in the broader context of linguistic string-transduction tasks, such as paraphrasing, morphological transformation and co-reference resolution.

© Springer-Verlag Berlin Heidelberg 2013

The definition that [3] proposes for *cognate*—"words with a common form and meaning across the languages"—is applicable to both transliterated and etymologically related words. Machine transliteration is, of course, a different task from cognate alignment. The main principle for word changes in etymology is the regularity of sound change, whereas transliteration of names may not always follow regular rules. Transliteration does not always follow pronunciation in practice—it can be based on script, or tradition, or on translation. For example, the French name *Jean* [ʒɑ̃] used to be transliterated into Russian as *Жеан* [ʐɛan], i.e., based on its French *spelling* rather than its pronunciation. The same name may be transliterated into another language differently depending on tradition: e.g., the name of the famous Russian author *Лев Толстой* is commonly *translated* into English as *Leo Tolstoy*; the name of his son, which is identical in Russian, is commonly *transliterated* as *Lev Tolstoy*. Furthermore, transliteration rules may be different in different domains [6].

Thus, there may be more noise in transliteration data than in etymological data. On the other hand, words in remotely related languages, (e.g., Finnish–Hungarian), will have substantially more complex correspondences. Therefore, while the tasks of transliteration and etymological alignment may be similar from the algorithmic point of view, it is not *a priori* obvious that the same algorithms will work for both tasks.

The models that we use for transliteration were originally developed for automatic discovery of patterns in etymological sound change, [16]. We apply these models to 13 parallel corpora in 7 languages: Farsi, English, Russian, Greek, Hebrew, French, and Japanese (Katakana script for foreign names), which are extracted from Wikipedia headlines using *language links*. We report results on automatic transliteration of names of American actors from English into four languages that use different kinds of writing systems: alphabetic (Russian and Greek) and consonantal (Farsi and Hebrew). We also examine the performance on data of different semantic type—person names vs. location names. For example, for English-Russian transliteration we use 3 different datasets: person names of English origin (American actors), person names of Russian origin (Russian writers), and location names of Russian origin (Russian cities).

As far as we are aware, the comparison of transliteration results among different semantic types has not been addressed in the literature to date; [10] studied the influence of the origin of names and of noise in the data on the results of transliteration, but they did not apply their method to words of different types.

2 Related Work

A comprehensive survey in [9] classifies machine transliteration generation methods into several broad categories: rule-based, phonetics-based, spelling-based, hybrid—a mix of spelling and phonetics, and combined—applying several methods and then selecting the best transliteration via re-ranking. According to this classification scheme, the models we use would be classified as spelling-based, as they use only the spelling of the words with no information about their pronunciation. Some of our etymological models use phonetic features [18]; they will be applied to the transliteration task in future work.

The most common approach for spelling-based transliteration uses source-channel models, [11,4]. Many methods used for machine transliteration are adopted from phrase-based statistical machine translation [15,5]. The importance of the representation of word-pairs is emphasized in [6], i.e., the features used in the transliteration task. The representation in [6] combines uni-grams and bi-grams, which are then used as features in an optimization task. It was shown in [12] that sequential n-grams outperform the bag-of-word n-gram models.

A Minimum Description Length (MDL) approach to transliteration is described in [21]. This work, as well as others [13,8], uses the Expectation-Maximization (EM) algorithm: in the Expectation stage the probabilities of substring correspondence are counted; in the Maximization stage word pairs are re-aligned using these probabilities.

Automatic transliteration and evaluation has been explored in a series of workshops on Named Entities, which organized shared tasks in transliteration generation and mining [23].

3 Data

We use datasets that we extracted from cross-language links in the Wikipedia. We did not try to extract as much data as possible; rather, we focused on language and on topical homogeneity of each dataset. In this paper, we present work on Wikipedia dumps dated up to 12 December 2012.[1] We used only the page titles, language links, and category links. The full text of the articles is not used in the transliteration task, with one exception, below. To extract the datasets we used Wikipedia Categories; we tried to focus on categories that can guarantee higher homogeneity in the data. For example, the majority of names in category *American actors* are of English origin, and most of the names in category *Russian Writers* are of Russian origin. However, there are many exceptions among person names; this is especially true for languages such as English or Russian, which are in use across wide geographic areas.

We also used location names, since toponyms may be more stable and more consistent than person names. For the dataset of Iranian Locations, we parsed the content of the corresponding Wikipedia pages. Since the number of titles for Iranian cities in the Russian Wikipedia was small, we collected names of towns and locations from Russian-Wikipedia pages about Iranian *Shahrestans* (counties). The data used our experiments is summarized in Table 1.

Each dataset was semi-automatically cleaned. Some amount of noise was intentionally left in data; e.g., the name of the Russian city Санкт-Петербург (*[Sankt-Peterburg]*) is usually "transcribed" into English as *Saint-Petersburg*, because this is a commonly accepted translation, although it is not phonetically accurate. We removed patronymics, which are very common in the Russian data,

[1] Wikipedia dumps are available under the GNU Free Documentation License and Creative Commons License at the Wikimedia Web-site.

Table 1. Transliteration corpora/datasets, extracted from Wikipedia headlines and language links. The language codes are: En–English, Fa–Farsi, Fr–French, Gr–Greek, He–Hebrew, Jp–Japanese (Katakana), Ru–Russian.

Dataset	Language pair	Size: # of pairs	Dataset	Language pair	Size: # of pairs
American Actors	En–Ru	1471	*Russian Cities*	Ru–En	1136
	En–He	1245		Ru–Fa	870
	En–Fa	840		Ru–Fr	828
	En–Gr	407		Ru–Jp	317
Russian Writes	Ru–En	1462	*French Cities*	Fr–Ru	828
Iranian Cities	Fa–En	439	*Iranian Locations*	Fa–Ru	1893
	Fa–Ru	469			

since in most cases they are omitted in other languages. We also removed all accent marks from the Greek dataset. Since these marks are obligatory in Greek script, transliteration models based on these data cannot be used in real-world application, though this data still interesting for a transliteration task.[2]

It was stated in [23] that "a reasonably large" dataset for the transliteration task should consist of ∼10 000 name pairs, which is orders of magnitude larger than our datasets. Some authors report satisfactory results using considerably smaller datasets. For example, a word-level accuracy of 33% for Arabic-to-English transliteration is reported using a training set of only 935 name pairs, and an accuracy of 46% for Russian-to-English, using a training set of 545 name pairs [22]. In this paper we present models that achieve good results (Section 6) on relatively small datasets; DirecTL+ [7], which we use as a baseline for comparison, also demonstrates reasonable results on our datasets.

4 Method

We use *Etymon*, a set of MDL-based models that we developed for analyzing etymological data, as the basis for our transliteration models.[3] The collection of models is described in [16,18,17]. The models take as input a phonetic representation of genetically related words, or *cognates*, and aim to discover regular phonetic changes between languages within a language family. This is done by searching for the best *pairwise alignment* of words, by optimizing the description length of the alignments. Transliteration is an analogous task, since in order to learn how to transliterate from one language to another, it seems that a natural prerequisite is to align the words in the training set. Thus, it seemed reasonable to suppose that we could use these models for aligning the data. After alignment, we introduce a set of *prediction* procedures for performing the actual transliteration—based on the alignment. We briefly describe the models; detailed explanations can be found in [16,18]—and the prediction procedure.

[2] We consider recovery of accents a separate task, beyond the scope of this work.
[3] The tools are publicly available from `http://etymon.cs.helsinki.fi/`

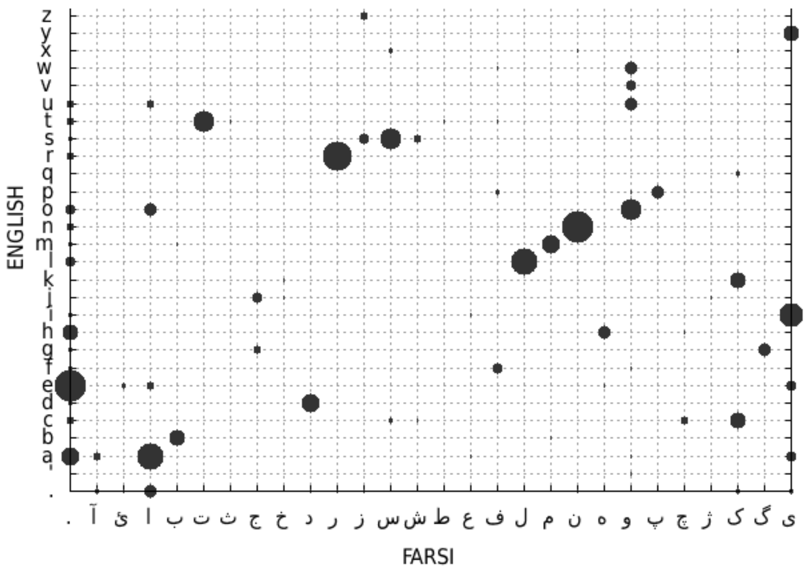

Fig. 1. English–Farsi alignment matrix; *American Actors* dataset. The size of each ball indicates the probability of the corresponding symbol-pair alignment.

4.1 1×1 Alignment

Our baseline 1×1 model finds an optimal alignment, where each symbol of the source word may align to at most one symbol of the target word, with possible insertions or deletions. Information about the context of the symbols is not used.

For example, the alignment matrix for English to Farsi transliteration on the *American Actors* is shown in Figure 1. The matrix shows that, e.g., English **e** is most frequently aligned to "." due to omission of short vowels in Farsi script. Mapping English **a** to Farsi آ is rare, as seen from the size of the corresponding bubble; this happens when **a** designates [a] that is situated at the beginning of the word. The 1×1 model is unable to capture this rule; the only information it uses is that **a** is transliterated to ا more frequently than to any other symbol.

4.2 2×2 Alignment

The 2×2 model extends the 1×1 model, by allowing for alignment of up to two consecutive symbols at a time on each level; this model also takes into account the start and end word boundaries. Unlike the 1×1 model, this model captures some information about the symbols' context when learning the correspondences. For example, it should discover cases where one symbol in language A corresponds to two symbols in language B; e.g., Russian 'ч' is often transliterated as "ch" in English—the 1×1 model is by definition unable to discover such correspondences. The 2×2 model can also discover that certain symbols are transliterated differently when they appear at the beginning or at the end of a word, as in the example of **a** and آ mentioned above, in Section 4.1. For example, the name *Alda* from the *American Actors* dataset is correctly transliterated into Farsi as آلدا by the the 2×2 model—with آ at the beginning. The 1×1 model incorrectly predicts the transliteration الدا—with ا as the most probable correspondence for **a** in both initial and final position, unable to exploit the context information.

4.3 Prediction

Once all word pairs in the training set are aligned, the discovered symbol correspondences can be used to predict transliterations for new, unseen words. The result of the convergence of the alignment algorithm is a count matrix, such as one shown in Figure 1, which indicates how often each symbol (in the 1×1 model)—or pair of symbols (in the 2×2 model)—of the source language is aligned to symbols in the target language in an optimal way. We have implemented an algorithm that predicts the target representation of a given source word based on the alignment. Theoretically, this is done by searching among *all* possible strings in the target language to select the string that yields the lowest cost under the model, when aligned with the source word. In practice, this can be achieved efficiently, by using simple table lookup for the baseline 1×1 model, and by a Dynamic Programming algorithm for the more complex models. Prediction based on the 1×1 model is straightforward, since symbols are aligned independently of their context; we assign to each source symbol the single target symbol to which it is cheapest to align:

$$t_i = \arg\min_{t \in T} L(s_i, t) \tag{1}$$

where s_i is the ith symbol in source word, T is the alphabet of the target language augmented with the special symbol '.' to allow for deletions, and $L(x, y)$ is the cost (code-length) of aligning the source-language symbol x to the target-language symbol y under the learned model.

The 2×2 prediction is more complicated, since it is possible to align zero, one, or two source symbols to symbols of the target language, and we need to choose the lowest cost alignment for the entire source word. We solve this optimization problem using Dynamic Programming (DP). To predict a target word, the algorithm starts from the beginning of the source word, and for each symbol s_i, finds the best prediction up to s_i based on previously computed partial alignments. The algorithm computes the cost $L(i)$ of the best prediction up to s_i, for all i. Thus, for predicting the best target sequence corresponding to the source word *up to* the i-th symbol, s_i, the possible *final* candidate alignments are in the set C, where:

$$C = \big\{(s_i : .), (s_i : t), (s_i : tt'), (s_{i-1}s_i : .), (s_{i-1}s_i : t), (s_{i-1}s_i : tt')\big\},$$

and t and t' are symbols from the target alphabet. Each of these 6 alignments has a fixed cost under the learned 2×2 model. The optimal prediction up to the i-th source symbol is given by minimizing the sum of one of these alignments, plus the cost $L(i-1)$ of the optimal alignment up to symbol s_{i-1}, for the first three candidates in C, or $L(i-2)$, the optimal cost up to s_{i-2} for the last three candidates in C,—where $L(i-1)$ and $L(i-2)$ have been pre-computed by DP previously. Using this approach the best target word—under the model—can be predicted in linear time.

5 Evaluation

The quality of machine transliteration depends on the ultimate task for which transliteration is being developed. In multilingual IE and IR, the system may make use of multiple possible variants. For example, in many cases more than one transliteration may be acceptable for a particular name, but if the system populates a database with events or relationships, identifying and merging different references to the same real-world entity is needed across multiple sources, [20,19]. In a multi-lingual setting, this capability is indispensable, [2]

Here, we evaluate performance of the transliteration models at the word level and at the symbol level. The most common word-level measure is accuracy [9]:

$$A = \frac{\text{number of correct transliterations}}{\text{total number of test words}} \tag{2}$$

Symbol-based evaluation measures are more diverse than word-based ones; in general, they are based on an edit distance between the system response and the expected transliteration. In this paper we use Normalized Edit Distance and Mean F-score. The normalized edit distance is computed as:

$$NED = \frac{\sum_i ED(c_i, r_i)}{\sum_i |c_i|} \tag{3}$$

where c_i is the expected transliteration for word i, r_i is the system response, and $ED(c_i, r_i)$ is an edit distance (here, the Levenshtein edit distance).

The symbol-level Mean F-score [23] is based on the Longest Common Subsequence between an expected transliteration c and the system response r[4]:

$$LCS(c, r) = \frac{1}{2}(|c| + |r| - ED'(c, r)) \tag{4}$$

Recall, Precision and F-score for a particular word are calculated on the basis of LCS (distance ED' allows insertions and deletions and no substitutions):

$$P = \frac{LCS(c, r)}{|r|} \qquad R = \frac{LCS(c, r)}{|c|} \qquad F = 2\frac{R \times P}{R + P} \tag{5}$$

We average the F-score over all words to get the mean over the entire data set.

[4] We slightly simplify all formulae here, assuming only one expected transliteration and one system response for each word.

Alongside our models, we use two other models for comparison. One is a naive baseline, where each symbol of source alphabet is transliterated as fixed symbol (or a string of symbols) from the target alphabet. In many cases this is a one-to-one mapping, but there are many exceptions; e.g., the Russian щ corresponds to English *shch* while Russian ь is most frequently omitted in transliteration. We did not apply this baseline to Katakana, since it is difficult to make reasonable correspondence between Katakana and Russian symbols.

The second model we used for comparison is the open-source system DirecTL+ [7]. It uses aligned data as input for the training; for alignment, we use the M2M-aligner, an open source program by the same authors, [8].[5]

We evaluate the models' performance via leave-one-out cross-validation.

6 Results

The results are shown in Tables 2 and 3, followed by the overall scores, average over all datasets that we tested. Although on some of the datasets, DirecTL+ beats the Etymon models, Etymon's performance appears higher overall.

One shortcoming of this evaluation scheme may be that only one correct answer is permitted for each word pair. For example, in the *American Actors* dataset, the English surname *Murray* is transliterated into Russian in two different ways: twice as Мюррей and twice as Мюррэй—therefore, for this name (*Murray*) any model can get at most 50% accuracy at the word level. We did not measure how this ambiguity ultimately affects the evaluation results, though it is common for person names.

By comparison, location names are more consistent; in most cases the toponyms are older and represent a more homogeneous transliteration scheme. Loan words and repetitions are more rare among location names. Thus, the results on location data are in general higher. For example, if we consider the results on three English-Russian datasets, namely *American Actors*, *Russian Writers* and *Russian Cities*, we can see from the tables that for both forward and backward transliteration the highest performance is achieved on the *Russian Cities* dataset. Comparing the datasets *Russian Writers* and *Russian Cities* is informative: both datasets use the same language pair, have the same language of names origin, and approximately the same size. However, we observe a difference of 20% in word-level accuracy on Ru-En transliteration and 14% on En-Ru, due to differences in the nature of the names.

It is also interesting to compare the *Iranian Cities* and *Iranian Locations* datasets for Russian-Farsi transliteration. As was described in Section 3, the latter contains a list of the Shahrestan's locations with population over 800. The dataset is four times larger, but it is also more noisy: Wikipedia editors seem to pay less attention to transliteration of smaller place names. In fact, we have

[5] We use default parameters for both programs. It may be possible to achieve better results through elaborate tuning of the parameters, though we did not explore parameter tuning. By comparison, our Etymon models have no parameters to tune.

Table 2. Transliteration results

Size: # of pairs	Model	Word level Accuracy	NED	Mean F-Score	Word level Accuracy	NED	Mean F-Score
American Actors							
		En → Fa			Fa → En		
	1x1	0.223	0.256	0.816	0.081	0.371	0.703
840	2x2	**0.393**	**0.180**	**0.867**	0.080	0.346	0.730
	Baseline	0.233	0.273	0.817	0.032	0.433	0.641
	DirecTL+	0.157	0.363	0.797	**0.118**	**0.324**	**0.756**
		En → Gr			Gr → En		
	1x1	0.157	0.312	0.776	0.079	0.385	0.692
407	2x2	**0.437**	**0.171**	**0.878**	**0.268**	**0.238**	**0.812**
	Baseline	0.179	0.343	0.750	0.101	0.456	0.650
	DirecTL+	0.342	0.232	0.849	0.140	0.417	0.693
		En → He			He → En		
	1x1	0.160	0.301	0.764	0.070	0.382	0.696
1245	2x2	**0.415**	**0.186**	**0.868**	0.104	0.337	0.738
	Baseline	0.074	0.426	0.725	0.043	0.430	0.640
	DirecTL+	0.160	0.331	0.817	**0.131**	**0.327**	**0.755**
		En → Ru			Ru → En		
	1x1	0.338	0.222	0.815	0.309	0.223	0.814
1471	2x2	**0.430**	**0.176**	**0.851**	**0.388**	**0.177**	0.853
	Baseline	0.298	0.250	0.799	0.282	0.250	0.795
	DirecTL+	0.387	0.214	0.834	0.373	0.189	**0.854**
Russian Cities							
		En → Ru			Ru → En		
	1x1	0.448	0.113	0.904	0.509	0.082	0.957
1136	2x2	**0.762**	**0.040**	**0.972**	**0.881**	**0.018**	**0.989**
	Baseline	0.379	0.176	0.868	0.823	0.028	0.983
	DirecTL+	0.501	0.163	0.886	0.813	0.028	0.985
		Fa → Ru			Ru → Fa		
	1x1	0.180	0.230	0.815	0.441	0.110	0.924
870	2x2	0.302	**0.170**	**0.866**	**0.684**	**0.060**	**0.964**
	Baseline	0.125	0.264	0.781	0.507	0.098	0.928
	DirecTL+	**0.325**	0.190	0.852	0.514	0.100	0.947
		Ru → Jp			Jp → Ru		
	1x1	0.013	0.552	0.470	0.016	0.377	0.733
317	2x2	**0.565**	**0.126**	**0.904**	**0.300**	**0.145**	**0.876**
	DirecTL+	0.022	0.742	0.541	0.287	0.159	0.870
		Ru → Fr			Fr → Ru		
	1x1	0.389	0.124	0.930	0.355	0.154	0.890
828	2x2	0.697	0.051	0.968	**0.668**	**0.065**	**0.953**
	Baseline	0.383	0.122	0.914	0.307	0.210	0.864
	DirecTL+	**0.736**	**0.042**	**0.973**	0.396	0.189	0.873

Table 3. Transliteration results, continued, including overall averaged scores.

Size: # of pairs	Model	Word level Accuracy	NED	Mean F-Score	Word level Accuracy	NED	Mean F-Score
Russian Writers							
		En → Ru			Ru → En		
1462	1x1	0.400	0.153	0.878	0.415	0.126	0.920
	2x2	**0.634**	**0.091**	**0.934**	**0.689**	**0.073**	0.943
	Baseline	0.347	0.201	0.856	0.651	0.075	**0.944**
	DirecTL+	0.462	0.176	0.875	0.588	0.090	0.933
French Cities							
		Ru → Fr			Fr → Ru		
828	1x1	0.088	0.338	0.745	0.113	0.357	0.715
	2x2	0.148	0.297	0.776	**0.381**	**0.182**	**0.863**
	Baseline	0.075	0.376	0.704	0.081	0.471	0.699
	DirecTL+	**0.199**	**0.259**	**0.808**	0.176	0.381	0.767
Iranian Cities							
		En → Fa			Fa → En		
439	1x1	0.196	0.334	0.787	0.109	0.280	0.817
	2x2	**0.435**	**0.155**	**0.896**	0.228	0.205	0.857
	Baseline	0.175	0.353	0.789	0.057	0.282	0.790
	DirecTL+	0.132	0.391	0.786	**0.289**	**0.185**	**0.863**
		Ru → Fa			Fa → Ru		
469	1x1	0.382	0.197	0.856	0.134	0.282	0.803
	2x2	**0.525**	**0.139**	**0.890**	**0.252**	0.237	0.827
	Baseline	0.267	0.277	0.803	0.092	0.296	0.775
	DirecTL+	0.151	0.332	0.800	0.222	**0.210**	**0.846**
Iranian locations							
		Ru → Fa			Fa → Ru		
1893	1x1	0.380	0.201	0.863	0.135	0.274	0.812
	2x2	**0.553**	**0.134**	**0.902**	0.278	0.217	0.841
	Baseline	0.285	0.270	0.816	0.078	0.318	0.752
	DirecTL+	0.155	0.345	0.813	**0.317**	**0.189**	**0.854**
	Results averaged over all datasets						
	1x1	0.235	0.259	0.804			
	2x2	**0.442**	**0.162**	**0.878**			
	Baseline	0.245	0.278	0.795			
	DirecTL+	0.311	0.253	0.832			

found many inaccuracies among the *Iranian Locations*. For example, the Iranian place-name بوئین /buin/ appears in Russian as Бу /bu/. Due to such noise in the data, for these datasets we achieved approximately the same results according to all measures, although the number of word pairs in the *Iranian Cities* dataset (439) is four times smaller than in the *Iranian Locations* dataset (1893). This may mean that it is possible to use quite small training sets for transliteration, if the data are highly homogeneous and clean.

7 Discussion and Current Work

To summarize, the main contributions of the presented work are: we provide a new, simple, and manually verified *data set* for evaluation of transliteration models; we apply models built for etymological alignment to the task of cross-lingual transliteration; we introduce simple extensions for *prediction* to the alignment models, which yield procedures for transliteration based on the alignment. We attempt to ground this work clearly in the context of other related approaches.

The MDL-based Etymon models, applied to the transliteration task without significant modifications, have achieved results that are comparable with state-of-the-art methods reported in the literature. We have discussed how the nature of data, as well as its homogeneity, impacts performance quality.

Current work includes adapting Etymon's *context-sensitive* models for transliteration. These models were shown, [18], to achieve substantially lower compression cost and normalized edit distance than the 1×1 and 2×2 models. We are implementing the prediction algorithm for these models, which is more complex and requires a target language model. Another complication is that the context models require each symbol to be represented as a vector of phonetic features. Thus, the next step will be an implementation of phonetic representations of the data. We also plan to expand our datasets by including more language pairs, and more complex types of data, including company names.

References

1. Andrews, N., Eisner, J., Dredze, M.: Name phylogeny: A generative model of string variation. In: Proceeding of the 2012 Joint Conference of Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL (2012)
2. Atkinson, M., Piskorski, J., van der Goot, E., Yangarber, R.: Multilingual real-time event extraction for border security intelligence gathering. In: Wiil, U.K. (ed.) Counterterrorism and Open Source Intelligence. Springer Lecture Notes in Social Networks, vol. 2 (2011)
3. Bergsma, S., Kondrak, G.: Alignment-based discriminative string similarity. In: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (2007)
4. Ekbal, A., Naskar, S.K., Bandyopadhyay, S.: A modified joint source-channel model for transliteration. In: Proceedings of the COLING/ACL, Stroudsburg, PA (2006)
5. Finch, A., Sumita, E.: Phrase-based machine transliteration. In: Proceedings of the Workshop on Technologies and Corpora for Asia-Pacific Speech Translation, TCAST (2008)
6. Goldwasser, D., Roth, D.: Transliteration as constrained optimization. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (2008)
7. Jiampojamarn, S., Cherry, C., Kondrak, G.: Joint processing and discriminative training for letter-to-phoneme conversion. In: Proceedings of ACL 2008: HLT, Columbus, Ohio (2008)

8. Jiampojamarn, S., Kondrak, G., Sherif, T.: Applying many-to-many alignments and hidden markov models to letter-to-phoneme conversion. In: Human Language Technologies 2007: North American Chapter of the Association for Computational Linguistics, Rochester, New York (2007)
9. Karimi, S., Scholer, F., Turpin, A.: Machine transliteration survey. ACM Computing Surveys 43(3) (2011)
10. Karimi, S., Turpin, A., Scholer, F.: Corpus effects on the evaluation of automated transliteration systems. In: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (2007)
11. Li, H., Zhang, M., Su, J.: A joint source-channel model for machine transliteration. In: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics (2004)
12. Lindén, K.: Multilingual modeling of cross-lingual spelling variants. Information Retrieval 9(3) (2006)
13. Pervouchine, V., Li, H., Lin, B.: Transliteration alignment. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (2009)
14. Schafer, C.: Novel probabilistic finite-state transducers for cognate and transliteration modeling. In: 7th Biennial Conference of the Association for Machine Translation in the Americas (AMTA) (2006)
15. Sherif, T., Kondrak, G.: Substring-based transliteration. In: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (2007)
16. Wettig, H., Hiltunen, S., Yangarber, R.: MDL-based Models for Alignment of Etymological Data. In: Proceedings of RANLP: The 8th Conference on Recent Advances in Natural Language Processing, Hissar, Bulgaria (2011)
17. Wettig, H., Nouri, J., Reshetnikov, K., Yangarber, R.: Information-theoretic modeling of etymological sound change. In: Approaches to Measuring Linguistic Differences, Mouton de Gruyter (2013)
18. Wettig, H., Reshetnikov, K., Yangarber, R.: Using context and phonetic features in models of etymological sound change. In: Proceedings of EACL Workshop on Visualization of Linguistic Patterns and Uncovering Language History from Multilingual Resources, Avignon, France (2012)
19. Yangarber, R.: Verification of facts across document boundaries. In: Proc. IIIA 2006, Helsinki, Finland (2006)
20. Yangarber, R., Best, C., von Etter, P., Fuart, F., Horby, D., Steinberger, R.: Combining information about epidemic threats from multiple sources. In: Proc. RANLP 2007 MMIES Workshop, Borovets, Bulgaria (2007)
21. Zelenko, D.: Combining MDL transliteration training with discriminative modeling. In: Proceedings of the Named Entities Workshop: Shared Task on Transliteration (2009)
22. Zelenko, D., Aone, C.: Discriminative methods for transliteration. In: Proceedings of EMNLP: Conference on Empirical Methods in Natural Language Processing (2006)
23. Zhang, M., Li, H., Kumaran, A., Liu, M.: Report of news 2012 shared task on machine transliteration. In: Proceedings of NEWS 2012 Named Entities Workshop, vol. 12 (2012)