Named Entity Transliteration Generation Leveraging Statistical Machine Translation Technology

Pradeep Dasigi
Computer Science Department
Columbia University
in the City of New York
pd2359@columbia.edu

Mona Diab
Center for Computational Learning Systems
Columbia University
in the City of New York
mdiab@ccls.columbia.edu

Abstract

Automatically identifying that different orthographic variants of names are referring to the same name is a significant challenge for processing natural language processing since they typically constitute the bulk of the out-of-vocabulary tokens. The problem is exacerbated when the name is foreign. In this paper we address the problem of generating valid orthographic variants for proper names, namely transliterating proper names in different scripts. We attempt to solve the problem for three different language pairs: English → Hindi, English → Persian, and Arabic → English. We adopt a unified approach to the problem. We frame the problem from a statistical Machine Translation perspective. We further post edit the output applying linguistically informed rules particular to the language pair and re-rank the output using machine learning methods.

1 Introduction

In a world of pervasive online media and globalization, we are flooded with streams of events where participants come from all over the world and they spell things in a myriad of ways especially where there are no orthographic standards. The problem is exacerbated for proper names especially when they are foreign. There are no standard spellings for such names. Accordingly orthographic variants are rampant. People typically rely on some form of phonetic transcription or what is referred to as transliteration. Humans have no issue identifying variants of names as the same, however for automatic algorithms in general and Natural Language Processing (NLP) in particular, proper name variants constitute a large portion of the out of vocabulary (OOV) phenomenon.

In this paper, we address the problem of generating valid transliterations for proper names in one language into some phonetic transcription (transliteration) in another language. The problem is not so bad if the two languages are phonetically close, share a script, and there exists an orthographic standard. However, if the two languages use different orthographic scripts and possess different phonetic inventories, we are faced with a much more complex situation.

We attempt to solve the problem for the latter case, namely for language pairs that are distant and that possess significantly different phonetic inventories. We target three language pairs: English → Hindi, English → Persian, and Arabic → English. English uses the Latin script, Arabic uses Arabic script, Persian uses an extended Arabic script to account for 6 extra sounds over Arabic, and Hindi uses Devanagari. We adopt a unified approach to the problem for the three language pairs. We leverage a statistical Machine Translation framework to address the problem. We apply linguistic expansion rules that are tailored for each language pair and transliteration direction. We view this as a generation problem, and we apply some post hoc filtering techniques to re-rank the output.

2 Linguistic Background

Hindi, Persian, Arabic, and English pertain to different language families but more importantly for the task at hand, they have different phonetic inventories. There are shared cognates between Hindi, Arabic and Persian due to historical reasons, however their sound repositories are significantly different from each other and in turn different from English. For instance, the /p/ and /v/ sounds in Persian do not exist in Arabic, the voiceless uvular plosive /q/ and the pharyngeal /h/ in Arabic have no real equivalents in English, the aspirated /b/ and /t/ in Hindi do not exist in English nor in Arabic or Persian for that matter. Such dis-

Proceedings of the 2011 Named Entities Workshop, IJCNLP 2011, pages 106–111,
Chiang Mai, Thailand, November 12, 2011.

tinctions in the sound inventories result in variable transcriptions, especially when a proper name in Hindi that has any of those aspirated letters such as the /b/, or the /q/ in Arabic. For example, the Arabic name *qAfy*[1] has a myriad of spelling variants such as **Kazafi, Qazafi, Kaddafi, Qadafy, Gaddafy, Gadaffy**, etc. This is partly a result of the lack of the phonetic sound in the inventory of English, but also due to the fact that different dialects of Arabic pronounce the /q/ sound differently affecting the foreign (in this case English) transliteration of it, for instance, in Egyptian Arabic, the /q/ sound is pronounced as a glottal stop, while in the Gulf it is pronounced as a /g/ sound.

The problem is further compounded for languages such as Arabic and Persian which have underspecified orthographies. In both languages, the short vowels and certain other phonetic markers such as consonantal gemination are underspecified in the surface orthography except when the genre of the text is liturgical such as in the Quran or the Bible, or in pedagogical materials for language learners, However the majority of text written for both languages lack short vowels which are typically expressed as diacritics. For instance the name *mHmd* in Arabic, as is evident in the transliteration, is expressed using only the consonants, and it corresponds to **Muhammad/Mohamed/Mohamad** etc., in English. We note the presence of the short vowels 'a, u, o' in the English transliteration, as well as the gemination of the medial letter 'm'.

Different considerations need to be paid attention to depending on the transliteration direction. Transliterating Arabic names into English is different from transliterating English names into Arabic. For instance, Arabic names when transliterated from English to Arabic, should lead to a smaller set of variants, than if an Arabic name is transliterated into English due to the underspecification of vowels inherent in the orthography of Arabic. For instance, the name **Bloomberg** can be spelled as **blwmbyrj/blmbrj/blwmbrj**, while a name such as **AbdAllTyf** would warrant at least the following variants in English **Abdel lateef, Abdallattif, Abdellatyff, Abd Allatif, Abd Allattyf**, etc. Accordingly in our algorithms we will be modeling for the language pair specifically bearing in mind the particularities of the transliteration direction.

[1]We use the Arabic Buckwalter transliteration scheme to express Arabic script throughout the paper. www.qamus.org.

3 Related Work

Automatic Transliteration has been well studied and various statistical approaches have been tried, starting from the seminal work by (Knight and Graehl, 1997). The noisy channel model has been extensively used by (Yuxiang et al, 2009) and the problem was dealt with in a manner similar to that of Statistical Machine Translation (SMT). Further, it has been modeled as a phrase based SMT problem in (Finch and Sumita, 2009), (Finch and Sumita, 2010), (Hong et al, 2009), (Noeman, 2009). (Finch and Sumita, 2009) reported accuracy of 0.788, F-score of 0.969 and Mean Reciprocal Rank of 0.788 on English $\rightarrow$ Hindi test data in NEWS 2009. (El-Kahky et al, 2011) modeled character sequence level alignments as bipartite graphs, and used graph reinforcement and link re-weighting to improve transliteration mining. They addressed two problems that arise from data sparsity - data coverage and erroneous translation probabilities due to ambiguous mappings. (Varadarajan and Rao, 2009) used Hidden Markov Models to derive substring alignments from training data and learn a weighted Finite State Transducer from these alignments. They reported an accuracy of 0.398, F-score of 0.855 and MRR of 0.515 on English $\rightarrow$ Hindi test data in NEWS 2009. (Noeman and Madkour, 2010) proposed a language independent technique for transliteration. They used Giza++ (2010) to model initial alignments. A Finite State Automaton (FSA) built from those alignments is used to generate transliterations at an edit distance of at most k from the source word. Their best performing system had an F-measure of 0.915 on English to Arabic transliteration task in NEWS 2010. In general, most of this work was to build an initial alignment and use statistical techniques in some form to generate better transliterations, and hence language independent. Our work differs in that it takes a more linguistically informed approach towards generating better transliterations by customizing the solutions per language pair and transliteration direction.

4 Approach and Experimental Design

In our basic approach, we model the problem as a noisy channel problem. We leverage Phrase Based Statistical Machine Translation (SMT) technology (Zens et al, 2002). Our statistical transliteration

system is implemented using Moses(Koehn et al, 2007). Each name is represented as a sentence for training, tuning and decoding. A name could be composite comprising multiple name units, such as **Michael Jackson** corresponding to **mAykyl jAkswn** in Arabic. Each character is treated as a separate token by the system, and name boundaries are marked using special characters. Accordingly, the sentence pair for the name **Michael Jackson** and it's Arabic counterpart will be represented as follows to the SMT system for training and tuning: **m i c h a e l # j a c k s o n** corresponding to **m A y k y l # j A k s w n**. Giza++ (Och and Ney, 2010) is used for building alignments between name pairs. For all the language pairs, the language scripts are represented in UTF-8 encoding. We further improve the output of the MT system by applying some language specific post-processing techniques. The following sub-sections describe those techniques for each language pair. All the techniques (except section 4.3.1) essentially expand the output given by our SMT system.

Since the methods of expansion yield large numbers of output candidates, a filtering technique is used to be able to distinguish the correct transliterations from the incorrect ones. We build a binary classifier that labels each candidate transliteration as correct or incorrect. We employ two features in training: a language model (LM) log probability for each name from the target side of the training data corpus to ensure that the generated candidate is a fluent target name; the second feature is the string edit distance of each candidate from its nearest name obtained from direct mapping. This second feature is a measure of how much the candidate has changed due to expansion. The filtering classifier is applied to the expanded data. The training data is synthetically generated from expanding the candidates according to the linguistic rules. We label the training data as correct and the expanded data as incorrect. To make sure that incorrect expansions do not overwhelm correct transliterations, we remove some incorrect candidates from the training data for the classifier.

4.1 English-Hindi

4.1.1 Short vs long vowels

Hindi clearly distinguishes between short and long vowels, however English transliterations are not necessarily consistent in faithfully expressing that distinction. For example, the English transliteration of the names **amandip** and **parijat** both have the letter 'i', but in Hindi script it represents a long vowel in the first case and a short vowel in the second. Similarly, the 'a' sounds are short in the first word and long in the second. Accordingly, the SMT output is augmented by expanding short vowels with long vowels and vice-versa.

4.1.2 Initial vs Medial vowels

Like other Indian scripts, vowels in Devanagari are written as diacritic symbols if written after a consonant, and in independent form if not. So, when the SMT system is trained, vowels in English are aligned to both forms and some candidates have incorrect forms of vowels. As a post-processing step, those errors are automatically corrected. This is done by replacing diacritic symbols that occur at the beginning of names with vowel forms and vowels forms that occur after consonants with diacritic symbols.

4.2 English-Persian

4.2.1 Vowel interchange rule

It has been observed from the output of MT system that a common mistake is between long vowels 'A' and 'w', and 'A' and 'y'. To deal with this problem, the output is augmented by adding new candidates that have an 'A' sound replaced with 'w' or 'y' and vice-versa.

4.2.2 Words beginning with A

In many cases where the source word begins with letter 'A', that sound is not transliterated by the SMT system. The transliterated candidate begins with the sound of the consonant following the letter in these cases. This is probably because the sound corresponding to the letter is dropped in cases where it occurs in name medial positions. This is more common with words of Persian origin. Although a good language model takes into account the position of the letter in the name as well, some lower ranked candidates in the output have this error. To deal with this, 'A' is appended in those cases where the source word begins with 'A' and the output candidate does not begin with a vowel.

4.3 Arabic-English

4.3.1 Direct Mapping

A direct mapping of Arabic letters to their equivalent sounds in English is performed, for exam-

ple an 'm' is transliterated as an 'm'. However some of the letters are tricky since they have no equivalent simple orthographic forms in English such as the Arabic *'ain* or 'E' sound, the Arabic *ghain* or 'g' sound. In these cases we opted for multiple correspondents. In the former 'E' case, we expanded to a possible **' or A** sounds and for the 'g' sound we expanded to the following possibilities **gh, g, q**. We also noted in the development and training data the existence of some dialectal replacements indicating that the transliterations should also reflect dialectal variants, i.e. the transliteration is not only constrained to the modern standard Arabic (MSA) sound inventory, hence we allowed for dialectal expansions such as for the Arabic letter *thaal* or '$*$' was mapped to **th, z, d** and the letter *thaa* or 'v' was expanded to **th, s**. This mapping is devised by a native Arabic speaker. All possible sequences of sounds in English for a given Arabic name are treated as its transliteration candidates.[2] Accordingly, a name such as *mgrby* is translated directly as **maghrebi, magrebi, magreby, maghrabi,** etc.

4.3.2 Vowel Expansion

Arabic similar to Persian is underspecified for short vowels in its orthography hence two names such as *zamar* and *zumur* will be spelled the same way appearing as *zmr* in Arabic. Hence, we expand the names by placing short vowel between any two consecutive consonants. We maintain a vowelless version for every expansion spot. Also we do not epenthesize with a vowel at name boundaries where a name is composite and contains multiple names such as *Abw-MAzn*. We use rules such as: if two consonants are preceded by a long vowel *A, w, y*, then we should expect to expand with one of the 5 vowels of English.

4.3.3 Composite Names and their Internal Boundaries

In case of composite names that have subparts, we applied the following rules:

- If the candidate has a subpart that begins with *bn*, only vowels **i** or **e** is used between the two consonants. **bin** or **ben**, meaning 'son of', is frequent in Arabic names and hence other vowels are not likely to occur between these specific two consonants.
- One common problem in this language pair is to recognize the name may be segmented into parts when written in English such as *AbumAzn* may be transliterated in English to **Abu Mazen** or **Abu-Mazen**. To tackle this, if a candidate begins with patterns such as *Abw, AbA, Abn, ibn, bin*, a space or a hyphen is introduced after the first portion of the name.

5 Experimental Results

The official task training data was directly used for training. The official task development data was split into two equal parts, with half the data being used for tuning the system and the other half for initial testing (Dev). We report results of our systems on both the Dev and the official shared task Test data. Details of the data used, their sizes and sources can be found in the Task Organizer's Whitepaper (TOW) (Zhang et al, 2011).

Table 1 contains the results of our system on English-Hindi. The metrics used Accuracy, Fscore, MRR and MAP are described in detail in TOW. The first set of results is of SMT output containing the top translation candidate for each source name (H-1best SMT[Dev]). H-Nbest SMT[Dev] corresponds to the output containing 10 top ranked transliterations per source language name. H-SMT+exp[Dev] and H-SMT+exp[Test] illustrate the results after application of the two expansion rules described in Section 4.1 on the Dev and Test data respectively. The results clearly indicate that yielding more candidates results in better performance, i.e. returning N-best results is better that the top result (N-best is better than 1-best), improving the overall accuracy, F score, MRR and MAP for the system as a whole. Moreover, applying expansion rules in the form of our devised linguistic rules significantly improves the quality of transliterations for the dev set on nearly all metrics except for MAP, (H-SMT+exp[Dev]) outperforms (n-best H-SMT[Dev]). We note a significant drop in accuracy between the Dev and Test data, however we see an improvement for the MAP metric.[3]

Table 1 shows three sets of results for English-Persian task. The first set is 10-best results from SMT system (P-SMT[Dev]), without any expansion. P-SMT+exp[Dev] and P-SMT+exp[Test] correspond to the output of Dev and Test, respectively, as expanded using rules described in sec-

[2] A full listing of the Transliteration mapping is available upon request.

[3] We do not have access to the Test data key answers for any of the language pairs to perform error analysis.

Condition	Acc.	F Score	MRR	MAP
H-1best SMT[Dev]	0.340	0.850	0.340	0.340
H-Nbest SMT[Dev]	0.631	0.937	0.631	0.393
H-SMT+exp[Dev]	0.718	0.951	0.718	0.316
H-SMT+exp[Test]	0.387	0.860	0.516	0.387
P-SMT[Dev]	0.575	0.920	0.587	0.481
P-SMT+exp[Dev]	0.710	0.953	0.725	0.339
P-SMT+exp[Test]	0.606	0.933	0.697	0.589

Table 1: English-Hindi and English-Persian results

Condition	Acc.	F Score	MRR	MAP
1. DirectMap[Dev]	0.018	0.763	0.045	0.022
2. DirectMap+vow-exp[Dev]	0.065	0.805	0.139	0.065
3. 10-best[Dev]	0.194	0.835	0.330	0.189
4. 10-best+vow-exp[Dev]	0.226	0.847	0.361	0.188
5. 40-best[Dev]	0.363	0.897	0.507	0.286
6. 40-best+vow-exp[Dev]	0.396	0.904	0.535	0.299
7. 40-best+vow-exp+filt[Dev]	0.375	0.898	0.512	0.288
8. 150-best[Dev]	0.559	0.941	0.677	0.426
9. 150-best+vow-exp[Dev]	0.590	0.946	0.702	0.442
10. 150-best+vow-exp+filt[Dev]	0.546	0.936	0.657	0.413
11. 150-best+vow-exp[Test]	0.526	0.928	0.628	0.386
12. 150-best+vow-exp+filt[Test]	0.519	0.927	0.612	0.383

Table 2: Arabic-English - Transliteration Results

tion 4.2. Clearly, these rules significantly improve the quality of the transliterations on the Dev set for all metrics. We note a similar trend to the English-Hindi results with a significant drop in accuracy, F-score, MRR between the Dev and Test data, however we see an improvement for the MAP metric.

For Arabic-English, Table 2 illustrates the results of the different conditions: 1. the direct mapping as described in section 4.3.1 for Dev; 2. DirectMap with vowel expansion of the Dev (DirectMap+vow-exp[Dev]); conditions 3, 5, and 8. are SMT N-best conditions for Dev data; conditions 4, 6, 9 and 11 are N-Best results for Dev and Test data; finally, conditions 7, 10 and 12 present the results after applying filtering to the output of the SMT expanded system for both Dev and Test data. We use three thresholds for N in the N Best conditions: 10, 40 and 150.

The Direct Map results in the worst performing conditions, however we do note relative improvement from DirectMap to DirectMap+vow-exp across the 4 metrics indicating that vowel expansion is a good move for this language pair. Using SMT for transliteration improves significantly over Direct Mapping as illustrated by the relative improvement of condition 3 (10-best[Dev]) over condition 2 DirectMap+vow-exp[Dev]. Increasing the number of returned N Best results from 10 to 40 and subsequently to 150 shows significant improvement comparing conditions 3, 5, and 8. Further applying vowel expansion shows consistent improvement in performance in conditions 4, 6, and 9. We further applied filtering to the resulting output however this did not yield improvements in the results as illustrated in conditions 7 40-best+vow-exp+filt[Dev] and 10 150-best+vow-exp+filt[Dev], however, filtering helped prune the 100s of outputs generated from the vowel expansion step in smart ways. In fact we note that on the Test data the difference between condition 11 (150-best+vow-exp[Test]) and 12 (150-best+vow-exp+filt[Test]) is not that significant, though 11 yields higher results.

6 Discussion

The impact of each approach taken for English-Arabic transliteration can be seen from the example of >**bAbTyn**. When the direct mapping technique is used, one of the best transliterations is **Ababtyn**. When expansions are applied, it becomes **Aba Batyn**. The SMT system produces **Ababatin**, and after expansion, it becomes **Abaa Bateen**, which is in the reference list, although not in the first few ranks. Filtering this list reduced its size from 39 to 5 and removed incorrect names like **Ababwotyn** and **Ababoutyn**.

The English - Hindi system has specific limitations. Words like **Gertrude** and **Canada** are generally not transliterated correctly to Hindi. This can be because of the high number of names of Indian origin in the training data. Hindi names almost always have one to one letter to sound matching. The same holds when they are transliterated to English. So, a foreign origin word that has letters which do not have their most common pronunciation is a challenge for this approach. This may be resolved by trying to filter words that do not have Indian origin and treating them separately.

7 Conclusions and Future Directions

We showed that phrase based SMT systems can be useful for the problem of NE transliteration. But with the application of linguistic rules as a post-processing step, the performance can be significantly improved. For English Persian and English Hindi tasks, direct application of such rules improved the performance of the systems signifi-

cantly. However, Arabic-English task proved to be a different and a more complex problem, due to the transliteration direction from a highly underspecified orthography (Arabic) to a more phonetically specified one. We showed that this problem can be handled by a vowel expansion technique on the SMT output. Applying a filtering technique using a classifier proved to be an effective method of eliminating incorrect candidates in the expanded output without significantly affecting the performance of the system. In the future, we plan to apply these approaches to larger data sets and more language pairs in various transliteration directions.

References

Ali El-Kahky, Kareem Darwish, Ahmed Saad Aldein, Mohamed Abd El-Wahab, Ahmed Hefny, Waleed Ammar *Improved Transliteration Mining Using Graph Reinforcement.*. Emperical Methods in Natural Language Processing 2011

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, Evan Herbst. 2007. *Moses: Open Source Toolkit for Statistical Machine Translation*. Annual Meeting of the Association for Computational Linguistics (ACL), demonstration session, Prague, Czech Republic, June 2007.

Kevin Knight and Jonathan Graehl *Machine Transliteration*. Journal Computational Linguistics archive Volume 24 Issue 4, December 1998

Sara Noeman and Amgad Madkour *Language Independent Transliteration Mining System Using Finite State Automata Framework.*. Named Entity Workshop 2010

Franz Josef Och and Hermann Ney *Improved Statistical Alignment Models.*. Proc. of the 38th Annual Meeting of the Association for Computational Linguistics, pp. 440-447, Hongkong, China, October 2000

Jia Yuxiang, Zhu Danqing and Yu Shiwen *A Noisy Channel Model for Grapheme-based Machine Transliteration*. Named Entiy Workshop 2009, ACL-IJCNLP 2009

Sara Noeman *Language Independent Transliteration System Using Phrase Based SMT Approach on Substrings*. Named Entiy Workshop 2009, ACL-IJCNLP 2009

Balakrishnan Varadarajan and Delip Rao ϵ *extension Hidden Markov Models and Weighted Transducers for Machine Transliteration.*. Named Entity Workshop 2009

Richard Zens, Franz Josef Och, and Hermann Ney *Phrase-Based Statistical Machine Translation*. KI '02 Proceedings of the 25th Annual German Conference on AI: Advances in Artificial Intelligence

Andrew Finch and Eiichiro Sumita *Transliteration by Bidirectional Statistical Machine Translation*. Named Entiy Workshop 2009, ACL-IJCNLP 2009

Andrew Finch and Eiichiro Sumita *Transliteration using a Phrase-based Statistical Machine Translation System to Re-score the Output of a Joint Multigram Model*. Named Entiy Workshop 2010, ACL 2010

Gumwon Hong, Min-Jeong Kim, Do-Gil Lee and Hae-Chang Rim *A Hybrid Approach for English-Korean Name Transliteration*. Named Entiy Workshop 2009, ACL-IJCNLP 2009

Min Zhang, A Kumaran, Haizhou Li NEWS 2011 Shared Task on Machine Transliteration Whitepaper http://translit.i2r.a-star.edu.sg/news2011/news2011whitepaper.pdf