Transliterating From All Languages

Ann Irvine
anni@jhu.edu

Chris Callison-Burch
ccb@cs.jhu.edu

Alexandre Klementiev
aklement@jhu.edu

Computer Science Department
Johns Hopkins University
Baltimore, MD 21218

Abstract

Much of the previous work on transliteration has depended on resources and attributes specific to particular language pairs. In this work, rather than focus on a single language pair, we create robust models for transliterating from all languages in a large, diverse set to English. We create training data for 150 languages by mining name pairs from Wikipedia. We train 13 systems and analyze the effects of the amount of training data on transliteration performance. We also present an analysis of the types of errors that the systems make. Our analyses are particularly valuable for building machine translation systems for low resource languages, where creating and integrating a transliteration module for a language with few NLP resources may provide substantial gains in translation performance.

1 Introduction

Transliteration is a critical subtask of machine translation (MT). Many named entities (e.g. person names, organizations, locations) are transliterated rather than translated into other languages. That is, the sounds in the source language word are approximated with the target language phonology and orthography. Named entities constitute an open class of words. The names of people and organizations, for example, never seen before in training often show up in new documents. It is critical that MT systems properly handle these content-bearing words. Integrating a transliteration module into an MT system is one way of handing out of vocabulary NEs and cognates.

In this paper we use the machinery that is used to train statistical translation systems to build transliteration modules. In translation, words and phrases in the source language are translated and then reordered to form coherent sentences in the target language. In transliteration, characters and character sequences are transliterated to form words in the target orthography. In transliteration there is no reordering, making it a monotone translation task.

Although in many ways transliteration is a simpler task than translation, it has its own set of challenges. The phonetic inventories of languages are extremely varied. For example, Rotokas (a language spoken in Papa New Guinea) has only six consonant sounds, while Southern Khoisan (spoken in Botswana) has 122 (Maddieson, 2008). Additionally, in many languages, the spoken form of words is often not true to the written form. English has many such inconsistencies, including silent *e*'s, as in *Kate*, and unpredictable pronunciations patterns, exemplified by the *ai* in *Craig* and *Caitlin*. These types of differences make approximating source language sounds in the target language difficult and transliteration as a whole highly ambiguous.

In this paper, we build a large number of transliteration systems for a diverse set of languages, using name pairs mined from Wikipedia. We present the results in a variety of ways. We show how the volume of training data affects transliteration performance, and we discuss common transliteration errors.

The rest of the paper is organized as follows. In Section 2 we discuss prior work in transliteration, and in Section 3 we describe our model and dataset

in detail. In Section 3 we also compare our systems' performance with that of other systems and explain our evaluation metric. In Section 4, we describe experiments in transliterating from other languages to English. We conclude with some thoughts on future work in Section 5.

2 Previous Work

There has been a large amount of research focused on the task of transliteration with both discriminative and generative methods achieving good performance. Knight and Graehl (1997) reported the results of a generative model for back-transliterating from Japanese to English using a weighted FST. Recently, Ravi and Knight (2009) trained the same Japanese-English models on unsupervised data. Virga and Khudanpur (2003) and Haizhou et al. (2004) suggest using the traditional source-channel SMT model to 'translate' the sounds of one language into another and present results on Chinese-English transliteration.

Other recent work (Klementiev and Roth, 2006; Tao et al., 2006; Yoon et al., 2007) proposes to view transliteration as a classification task and suggests training a discriminative model to determine whether a pair of words are transliterations of one another. Subsequent work (Bergsma and Kondrak, 2007; Goldwasser and Roth, 2008) improves on this idea by focusing on selecting better pairwise features. Following this line of work, Sproat et al. (2008) developed a toolkit for computing the cost of mapping between two strings in any two scripts. Their toolkit also includes generative pronunciation modules for Chinese and English.

In 2009, the Named Entities Workshop (NEWS) at the ACL-IJCNLP conference included a Machine Transliteration shared task (Li et al., 2009a). Over thirty teams participated in the task, which involved transliterating from English to the following languages: Hindi, Tamil, Kannada, Russian, Chinese, Korean, and Japanese. The workshop released a common dataset with training and development transliteration pairs for each language and used a common evaluation. We report results comparing our system to the workshop systems in Section 3.2.

With the exception of the shared task, most research papers present performance on just one or two language pairs. In this work, we evaluate a single transliteration framework for transliterating from many languages to English. We compare our systems to previous work where it is possible.

3 Transliteration Model

Following Virga and Khudanpur (2003), we treat transliteration as a monotone character translation task. Rather than using a noisy channel model, our transliteration models is based on the log-linear formulation of statistical machine translation (SMT) described in Och and Ney (2002). Whereas SMT systems are trained on parallel sentences and use word-based n-gram language models, we use pairs of transliterated words along with character-based n-gram language models. We use the Berkeley aligner (DeNero and Klein, 2007) to automatically align characters in pairs of transliterations. This is analogous to word-based alignment in SMT. Transliteration is simpler than translation, since phrases are often reordered in translation, but characters sequence are monotonic in transliteration. Our feature functions include a character sequence mapping probability (similar to the phrase translation probability), a character substitution probability (similar to the lexical probability), and a character-based language model probability.

For our experiments, we use the off-the-shelf Joshua open source statistical machine translation system (Li et al., 2009b). Joshua's translation model uses synchronous context free grammars, like the Hiero system (Chiang, 2005; Chiang, 2007). However, because transliteration is strictly a monotone task, we do not extract grammar rules that involve any hierarchical structure by restricting the number of nonterminals to zero. We have the grammar extractor identify rules for character-based phrases up to length ten. Our language models are also trained on up to 10-gram sequences of target language characters. Unlike in machine translation, our phrase tables and language models can support very large n-gram sizes because the number of characters in a given script is small compared to word vocabularies. As a preprocessing step, we append start-of-word and end-of-word symbols to all training pairs and test words. Table 1 shows examples of Russian to English and Greek to English transliteration rules

Russian→English			
Rule	**Feature Function Scores**		
ф о м → f a u t	0.301	1.456	3.118
ц ы → t s y	0.204	2.490	1.431
щ у к → s c h u k	0.845	2.185	2.034
а р д ж → a r j	0.398	1.432	0.506
Greek→English			
Rule	**Feature Function Scores**		
ο χ ΄α → o c h a	0.602	1.115	1.036
γ ε ρ → g e r	0.301	0.556	0.152
α λ μ → a l l m	0.699	0.214	0.175

Table 1: Examples of Russian to English and Greek to English transliteration rules learned by Joshua along with the following associated log probabilities: a character sequence mapping probability, a character substitution probability, and a character-based language model probability.

learned by Joshua along with their feature function scores. We use Joshua's MERT optimization to learn the feature weights. Although, as discussed below, we would actually like to minimize the edit distance between our systems' output and reference transliterations, we optimize using a character-based BLEU score objective function (BLEU-4), the MERT default in Joshua. Optimizing on a metric more suitable to transliteration is left to future work.

3.1 Training Data

All of the models that we describe are trained on name pairs mined from Wikipedia. Wikipedia maintains inter-language links between pages, making it possible to gather a set of pages that describe the same topic in multiple languages. Additionally, the site categorizes articles and maintains lists of all of the pages within each category. We have taken advantage of a particular set of categories that list people born in a given year. For example, the Wikipedia category page '1961 births' includes links to the 'Barack Obama' and 'Michael J. Fox' pages. By iterating through all categories that list people born in a given year and then all people listed, we follow all of the language links from each English page about a person and compile a large file of person names (the Wikipedia page titles) in many languages. The 100 languages with the most overlapping name pages with English are shown in Table 2. Our 14 languages

ja	56786	mr	4847	bs	961	io	411
ru	47044	th	4610	br	894	cv	395
de	35365	ka	3624	ur	893	sq	377
fr	29317	sk	3536	cy	875	jv	326
zh	23345	da	3310	nn	857	wuu	322
pl	19731	tr	3281	zh-y	826	ku	287
it	17409	eo	2898	ms	708	kk	283
he	16436	ro	2857	sw	701	bat	256
es	16399	sl	2642	sh	692	nds	251
nl	14855	lv	2630	tg	667	an	244
ar	12253	id	2409	simp	664	gd	204
sv	11323	et	2407	yi	651	ast	204
ko	10782	hr	2275	tl	628	zh-m	186
pt	10734	mk	2124	oc	623	ceb	173
bg	10704	lt	2106	arz	621	gan	172
uk	8251	bn	2100	ga	584	qu	170
sr	8119	gl	2011	lb	584	als	160
fi	7981	hi	1811	is	573	vls	150
ca	7405	vi	1747	hy	540	vec	128
no	7364	ml	1543	af	501	uz	122
el	6506	ta	1463	scn	481	dv	117
hu	6484	be-x	1333	kn	456	am	116
la	6241	eu	1193	mn	456	sco	113
fa	5891	be	1146	ht	443	lmo	110
cs	5485	az	1087	fy	431	tt	106

Table 2: The 100 languages with the largest number of name pairs with English. The counts are for Wikipedia pages describing people that have a inter-language link with English, and whose title is not identical to the English page title.

of interest and the number of names that we gathered for each are listed in Table 3[1].

In addition to English, we have chosen to transliterate the Wikipedia languages that are written in a non-Roman script, have at least 1000 person names (see Table 3), and were relatively easy to word align. Word aligning multi-word names from Wikipedia page titles is not trivial. Table 4 shows a few problematic cases in the Russian and English pairs. Often one page title includes middle names while the corresponding page title in another language does not, or the pages may use abbreviations or titles inconsistently. In order to align multi-word names, we use simple romanization character mappings, also mined from Wikipedia. In comparing multi-word names, we compute the best word alignments and set an edit distance threshold to filter the noisy data.

[1]Our data is available for download at `http://www.clsp.jhu.edu/~anni/data/wikipedia_names`

Language	Number of names
English	826508
Russian	47044
Hebrew	16436
Arabic	12253
Korean	10782
Bulgarian	10704
Ukranian	8251
Serbian	8119
Greek	6506
Farsi	5891
Georgian	3624
Macedonian	2124
Old-Belarusian	1333
Belarusian	1146

Table 3: Languages of interest and the number of harvested person names. There are many more English names than there are for other languages and, correspondingly, its overlap with other languages is relatively large. Consequently, the amount of training data for transliterating between English and other languages is greater than between any other pair of languages.

We built our default English language model by tagging and counting named entities in the English Gigaword corpus[2]. We identified over 1.3 million unique NEs in the corpus. Using the name list and their corpus frequencies, we built a character-based language model that includes n-grams up to length ten. In this work, our non-English language models are built from monolingual Wikipedia name lists.

3.2 Comparison with other systems

Before presenting results from our novel set of experiments, we compare our transliteration system with those evaluated in a 2009 ACL shared task workshop (Li et al., 2009a). The workshop evaluated systems trained to transliterate from English to several other languages using a variety of metrics. Although the focus of our current work is transliterating into English, it is helpful to make sure that our framework can provide reasonable results that are comparable with the current state-of-the-art.

We used the workshop data to build English to Russian and English to Hindi transliteration systems and evaluated them using the workshop metrics. The results are presented in Table 5. In general, although our systems do not outperform the best participating systems (Jiampojamarn et al., 2009; Oh et al., 2009), they generate results that are comparable to the state of the art in English to Hindi and English to Russian transliteration. Thus, with a competitive system framework, we turn to our main focus, which is transliterating from a large, diverse set of languages into English.

[2]see LDC corpus LDC2003T05

En-Wiki	Ru-Wiki	Ru-Gloss
Abbas I of Persia	Аббас I Великий	Abbas I the Great
Abbot Suger	Сугерий	Suger
Canute VI of Denmark	Кнуд VI	Canute VI
C. A. R. Hoare	Хоар, Чарльз Энтони Ричард	Hoare, Charles Antony Richard

Table 4: Examples of multi-word Russian-English name pairs that require word alignments and filtering.

Metric	Our System	Others
	English→Russian	
Top-1 Accuracy	.55	.35 - .61
Top-1 F-score	.91	.87 - .93
Mean Avg Prec. at 10	.20	.13 - .29
Training Pairs	5977	
	English→Hindi	
Top-1 Accuracy	.45	.00 - .50
Top-1 F-score	.87	.01 - .89
Mean Avg Prec. at 10	.18	.00 - .20
Training Pairs	4840	

Table 5: A comparison of our performance against the systems submitted to the Russian and Hindi transliteration shared tasks at the 2009 Named Entities Workshop.

3.3 Evaluation Metric

It is often the case that imperfect transliterations (i.e., inexact matches with the reference transliteration) are still readable in text. Since our goal is to integrate our model into an SMT system, it is important to know not only how frequently we produce perfect transliterations but how similar our output

Candidate	Reference	Edit D.	Norm. Edit D.
Burkin	Burkin	0	0.00
Andruck	Andruk	1	16.67
Shikai	Schikay	2	28.57
Gutsaev	Guzayev	3	42.86
Truxtun	Trakston	4	50.00

Table 6: Examples of candidate transliterations and their corresponding reference transliterations, and the edit distances and normalized edit distances between them. The normalized edit distance is the minimum number of insertions, deletions, and substitutions that must be made to transform one string into the other, normalized by the length of the reference string, and multiplied by 100.

is to the reference. So, we have used the standard Levenshtein edit distance metric for evaluation. To compute the similarity between a pair of strings, we count the minimum number of insertions, deletions, and substitutions that must be made to transform one string into the other and then normalize by the length of the reference string. The numbers that we report here are the normalized edit distances multiplied by 100, or the percent of characters in the reference that require a transformation for the string to match the system output. Prior work has also used edit distance as a metric for transliteration (Zhao et al., 2007; Noeman, 2009). Examples of transliteration candidates, references, edit distances, and the normalized edit distances between them are shown in Table 6.

4 Experimental Results

4.1 Transliteration from many languages

In our first experiment, we train systems to transliterate from 13 languages into English. Figure 1 plots the performance (average normalized edit distance between the output and reference) of the 13 systems versus the number of training pairs available in the Wikipedia data. To compute the averages, we performed a ten-fold cross validation (using 80% of the data for training, 10% for development, and 10% for testing at each stage) and report the overall averages. The Russian system, for which there is the most available training data, outperforms the other 12. However, it is not universally true that systems built for languages for which we have more training data outperform those built for

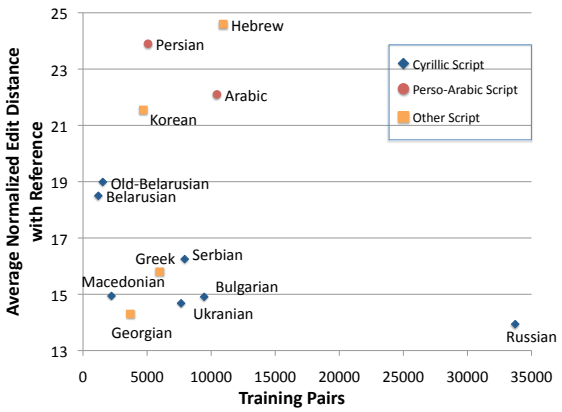

Figure 1: Number of training pairs vs. system performance as measured by average normalized edit distance from the reference. The normalized edit distance is the minimum number of insertions, deletions, and substitutions that must be made to transform one string into the other, normalized by the length of the reference string, and multiplied by 100.

languages for which we have little. For example, although there is a relatively large amount of data available for Arabic, it does not perform as well as we would predict on this factor alone. Linguistic differences make some languages inherently more difficult to transliterate to English. For example, the small Arabic vowel space makes transliterating, and especially back-transliterating (recovering names of English origin), into English relatively difficult.

4.2 Learning Curves

To gain a better picture of the effect that the number of training pairs has on system performance, we incrementally hold out data from each of the systems. The learning curves resulting from using 25%, 50%, 75%, and 100% of the available training data for each language are presented in Figure 2. For languages with a large amount of data, we perform additional experiments with very small training sets. All experiments are performed using 10-fold cross validation, and the results are shown in Figure 2.

Unsurprisingly, the average normalized edit distance between the hypothesis and the reference decreases with more training data nearly without exception. It is interesting to note that the average distance continues to decrease slowly as more data is added to the very large Russian training corpus.

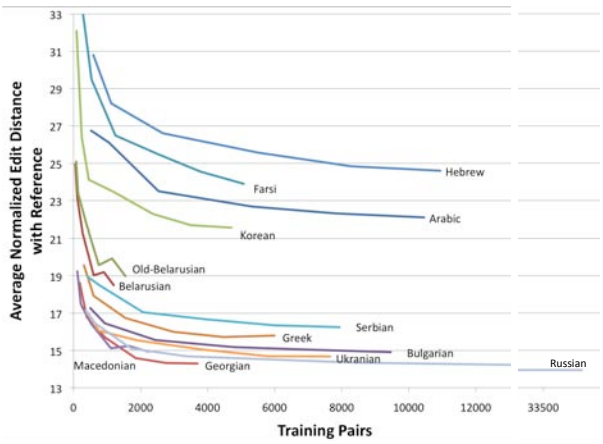

Figure 2: Learning curves resulting from holding out some training pairs from the models. The normalized edit distance is the minimum number of insertions, deletions, and substitutions that must be made to transform one string into the other, normalized by the length of the reference string, and multiplied by 100.

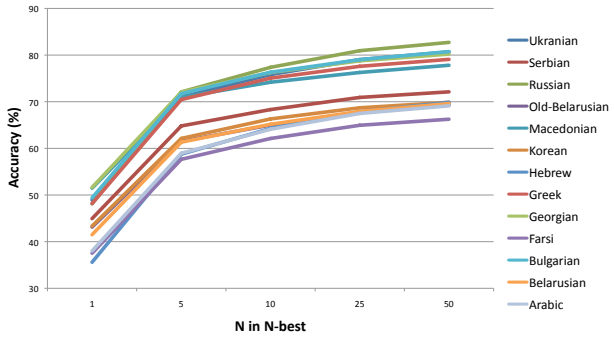

Figure 3: The percent of perfect transliterations found in the n-best output vs. n in n-best.

As in translation, more transliteration training data seems to only help performance.

4.3 N-best Transliterations

We also examine the transliterations in our systems' n-best lists. For this evaluation, we count the number of perfect transliterations (exact matches with the reference) that we find as we look at different size n-best lists for all items in the test set. The results are shown in Figure 3. On average, exact transliterations are found for an additional 20% of test set words in the 5-best list over those found in the 1-best transliteration. Nearly 10% more are in the 10-best list. These general trends hold for all of the languages that we tested. Re-ranking methods (e.g. Collins and Koo (2000)) may potentially improve the ranking of transliteration candidates, and this is part of our ongoing work. It is clear from the Figure that we should not expect much gain in re-ranking beyond the 10-best output.

4.4 Error Analysis

Finally, we explore the space of common mistakes made by our systems as well as some samples of output. The most common mistakes are shown in Table 7. The errors are not surprising for systems transliterating into English. The letters *e* and *h* are often not pronounced in English and, hence, not represented in some foreign language transliterations. Thus, our transliteration systems often do not produce them. For example, the final *e* in the English name *Pete* is not pronounced and, therefore, is unlikely to appear in a foreign transliteration. When our systems attempt to back-transliterate this name into English, they are likely to also leave out the final *e*. Although the phrase tables do include rules that are capable of generating silent English characters, the systems do not always choose to apply them. Additionally, it is not surprising that letters like *a* and *e* are often confused, as well as *c* and *k*.

There is not a large amount of variation among the types of mistakes made by the different language systems. However, we do see that in Russian, Bulgarian, Serbian, and Macedonian, *w* and *v* are frequently confused. This is not surprising due to the fact that these languages do not distinguish the two sounds, which makes it difficult to discriminate between the correct and incorrect potential English representations. Similarly, *p* and *b* confusion from Arabic is frequent for the same reason. Table 7 highlights some of the differences.

Table 8 shows some examples of the Russian to English transliteration system output. In the first three examples, the system transliterated correctly. The next group shows cases where the transliteration system outputs a correct alternative transliteration to the reference. That is, both the output and the reference are true alternative English spellings of the Russian named entities, and it would be impossible to be sure which alternative should be used with-

Overall		Russian		Serbian		Arabic		Hebrew	
E	_	E	_	E	_	E	_	E	_
H	_	H	_	**Ć**	**C**	E	I	E	A
Y	I	Y	I	H	_	A	_	A	_
A	_	C	_	Y	I	U	O	U	O
C	K	C	K	C	_	**I**	**E**	A	E
S	_	U	_	S	_	U	_	Y	I
C	_	**Y**	_	C	K	Y	I	H	_
_	I	T	_	T	_	**I**	_	_	**A**
U	_	**W**	**V**	A	_	S	_	E	I
E	I	S	_	U	_	_	**A**	**F**	**P**
T	_	**I**	**Y**	**L**	_	C	K	C	_
E	A	**I**	_	O	_	**P**	**B**	U	_
O	_	A	_	**E**	**A**	E	A	C	K
U	O	**E**	**A**	**W**	**V**	C	_	S	_
A	E	**S**	**Z**	A	E	H	_	**V**	**B**

Table 7: The 10 most common errors. The reference is on the left, and hypothesis is on the right. E _ indicates that the letter E is dropped from the hypothesis, and _ I indicates I is inserted.

Russian	**System**	**Reference**
Кастельнуово	Castelnuovo	Castelnuovo
Сигизмонди	Sigismondi	Sigismondi
Уильямсон	Williamson	Williamson
Раевский	Raevsky	Rayevskiy
Диттрих	Ditterich	Dittrich
Лукович	Lukovic	Lukowich
Шартье	Shartie	Chartier
Кавасуми	Kavasumi	Kawasumi
Макфарлейн	Macfarlein	Macfarlane

Table 8: Examples of Russian to English transliteration output. The system produced the reference transliteration in the first three examples, and it produced a correct alternative English transliteration in the second three examples. It incorrectly transliterated the final three.

out additional context. These examples highlight the difficulty in evaluating transliteration performance and suggest that evaluation metrics are likely to be pessimistic. In future work, we plan to use human annotations to measure the upper limit of transliteration performance. The system incorrectly transliterated the final set of names. These mistakes suggest that using additional evidence may assist transliteration. For example, as shown in Huang (2005), if a name's origin is known, we may be able to infer the proper spelling of the erroneous transliterations in the third part of Table 8.

5 Conclusion

In this work, we have demonstrated that freely available resources (both systems and data) are sufficient to build models capable of producing high quality transliterations from a large set of languages into English. We have shown that we can build high performing systems even for languages for which there are few available NLP resources. We have also shown that more data is better. We could supplement the training data pairs that we extracted from Wikipedia with those gathered by unsupervised or weakly supervised methods (Klementiev and Roth, 2006). Other directions for future work include building systems that transliterate into languages other than English and those that incorporate name origin information. Our results, which suggest that a language pair independent model trained with modest amounts of data can yield reasonable performance, are encouraging, especially for research on low-resource language NLP.

References

Shane Bergsma and Grzegorz Kondrak. 2007. Alignment-based discriminative string similarity. In *Proceedings of ACL*.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of ACL*.

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.

Michael Collins and Terry Koo. 2000. Discriminative reranking for natural language parsing. In *Proceedings of the International Conf. on Machine Learning*.

John DeNero and Dan Klein. 2007. Tailoring word alignments to syntactic machine translation. In *Proceedings of ACL*.

Dan Goldwasser and Dan Roth. 2008. Transliteration as constrained optimization. In *Proceedings of EMNLP*.

Li Haizhou, Zhang Min, and Su Jian. 2004. A joint source-channel model for machine transliteration. In *Proceedings of ACL*.

Fei Huang. 2005. Cluster-specific named entity transliteration. In *Proceedings of EMNLP*.

Sittichai Jiampojamarn, Aditya Bhargava, Qing Dou, Kenneth Dwyer, and Grzegorz Kondrak. 2009. Directl: a language-independent approach to transliteration. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009)*. ACL–IJCNLP.

Alexandre Klementiev and Dan Roth. 2006. Weakly supervised named entity transliteration and discovery from multilingual comparable corpora. In *Proceedings of ACL*.

Kevin Knight and Jonathan Graehl. 1997. Machine transliteration. In *Proceedings of ACL*.

Haizhou Li, A Kumaran, Vladimir Pervouchine, and Min Zhang. 2009a. Report of NEWS 2009 machine transliteration shared task. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009)*, pages 1–18. ACL–IJCNLP.

Zhifei Li, Chris Callison-Burch, Chris Dyer, Juri Ganitkevitch, Sanjeev Khudanpur, Lane Schwartz, Wren Thornton, Jonathan Weese, and Omar Zaidan. 2009b. Joshua: An open source toolkit for parsing-based machine translation. In *Proceedings of the Workshop on Statistical Machine Translation*.

Ian Maddieson. 2008. Consonant inventories. *The World Atlas of Language Structures Online*.

Sara Noeman. 2009. Transliteration using phrase based SMT approach on substrings. In *Proceedings of the International Conference on Arabic Language Resources and Tools*.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of ACL*.

Jong-Hoon Oh, Kiyotaka Uchimoto, and Kentaro Torisawa. 2009. Machine transliteration using target-language grapheme and phoneme: Multi-engine transliteration approach. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009)*. ACL–IJCNLP.

Sujith Ravi and Kevin Knight. 2009. Learning phoneme mappings for transliteration without parallel data. In *Proceedings of NAACL*.

Richard Sproat, Jim Baker, Martin Jansche, Bhuvana Ramabhadran, Michael Riley, Murat Saraclar, Abhinav Sethy, Patrick Wolfe, Sanjeev Khudanpur, Arnab Ghoshal, Kristy Hollingshead, Chris White, Ting Qian, Erica Cooper, and Morgan Ulinski. 2008. Multilingual spoken term detection: Finding and testing new pronunciations. In *Report on the Center for Language and Speech Processing Summer Workshop*.

Tao Tao, Su-Youn Yoon, Andrew Fister, Richard Sproat, and ChengXiang Zhai. 2006. Unsupervised named entity transliteration using temporal and phonetic correlation. In *Proceedings of EMNLP*.

Paola Virga and Sanjeev Khudanpur. 2003. Transliteration of proper names in cross-lingual information retrieval. In *Proceedings of ACL*.

Su-Youn Yoon, Kyoung-Young Kim, and Richard Sproat. 2007. Multilingual transliteration using feature based phonetic method. In *Proceedings of ACL*.

Bing Zhao, Nguyen Bach, Ian Lane, and Stephan Vogel. 2007. A log-linear block transliteration model based on bi-stream hmms. In *Proceedings of NAACL*.