Joint Generation of Transliterations from Multiple Representations

Lei Yao and **Grzegorz Kondrak**
Department of Computing Science
University of Alberta
Edmonton, AB, Canada
{lyao1,gkondrak}@ualberta.ca

Abstract

Machine transliteration is often referred to as phonetic translation. We show that transliterations incorporate information from both spelling and pronunciation, and propose an effective model for joint transliteration generation from both representations. We further generalize this model to include transliterations from other languages, and enhance it with reranking and lexicon features. We demonstrate significant improvements in transliteration accuracy on several datasets.

1 Introduction

Transliteration is the conversion of a text from one script to another. When a new name like *Eyjafjallajökull* appears in the news, it needs to be promptly transliterated into dozens of languages. Computer-generated transliterations can be more accurate than those created by humans (Sherif and Kondrak, 2007). When the names in question originate from languages that use the same writing script as the target language, they are likely to be copied verbatim; however, their pronunciation may still be ambiguous. Existing transliterations and transcriptions can help in establishing the correct pronunciation (Bhargava and Kondrak, 2012).

Transliteration is often defined as *phonetic translation* (Zhang et al., 2012). In the idealized model of Knight and Graehl (1997), a bilingual expert pronounces a name in the source language, modifies the pronunciation to fit the target language phonology, and writes it down using the orthographic rules of the target script. In practice, however, it may be difficult to guess the correct pronunciation of an unfamiliar name from the spelling.

Phonetic-based models of transliteration tend to achieve suboptimal performance. Al-Onaizan and Knight (2002) report that a spelling-based model outperforms a phonetic-based model even when pronunciations are extracted from a pronunciation dictionary. This can be attributed to the importance of the source orthography in the transliteration process. For example, the initial letters of the Russian transliterations of the names *Chicano* ([tʃɪkɑno]) and *Chicago* ([ʃɪkɑgo]) are identical, but different from *Shilo* ([ʃɪlo]). The contrast is likely due to the idiosyncratic spelling of *Chicago*.

Typical transliteration systems learn direct orthographic mapping between the source and the target languages from parallel training sets of word pairs (Zhang et al., 2012). Their accuracy is limited by the fact that the training data is likely to contain names originating from different languages that have different romanization rules. For example, the Russian transliterations of *Jedi*, *Juan*, *Jenins*, *Jeltoqsan*, and *Jecheon* all differ in their initial letters. In addition, because of inconsistent correspondences between letters and phonemes in some languages, the pronunciation of a word may be difficult to derive from its orthographic form.

We believe that transliteration is not simply phonetic translation, but rather a process that combines both phonetic and orthographic information. This observation prompted the development of several hybrid approaches that take advantage of both types of information, and improvements were reported on some test corpora (Al-Onaizan and Knight, 2002; Bilac and Tanaka, 2004; Oh and Choi, 2005). These models, which we discuss in more detail in Section 2.1, are well behind the current state of the art in machine transliteration.

Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL, pages 943–952,
Denver, Colorado, May 31 – June 5, 2015.

In this paper, we conduct experiments that show the relative importance of spelling and pronunciation. We propose a new hybrid approach of joint transliteration generation from both orthography and pronunciation, which is based on a discriminative string transduction approach. We demonstrate that our approach results in significant improvements in transliteration accuracy. Because phonetic transcriptions are rarely available, we propose to capture the phonetic information from supplemental transliterations. We show that the most effective way of utilizing supplemental transliterations is to directly include their original orthographic representations. We show improvements of up to 30% in word accuracy when using supplemental transliterations from several languages.

The paper is organized as follows. We discuss related work in Section 2. Section 3 describes our hybrid model and a generalization of this model that leverages supplemental transliterations. Section 4 and 5 present our experiments of joint generation with supplemental transcriptions and transliterations, respectively. Section 6 presents our conclusions and future work.

2 Related work

In this section, we focus on hybrid transliteration models, and on methods of leveraging supplemental transliterations.

2.1 Hybrid models

Al-Onaizan and Knight (2002) present a hybrid model for Arabic-to-English transliteration, which is a linear combination of phoneme-based and grapheme-based models. The hybrid model is shown to be superior to the phoneme-based model, but inferior to the grapheme-based model.

Bilac and Tanaka (2004) propose a hybrid model for Japanese-to-English back-transliteration, which is also based on linear interpolation, but the interpolation is performed during the transliteration generation process, rather than after candidate target words have been generated. They report improvement over the two component models on some, but not all, of their test corpora.

Oh and Choi (2005) replace the fixed linear interpolation approach with a more flexible model that takes into account the correspondence between the phonemes and graphemes during the transliteration generation process. They report superior performance of their hybrid model over both component models. However, their model does not consider the coherence of the target word during the generation process, nor other important features that have been shown to significantly improve machine transliteration (Li et al., 2004; Jiampojamarn et al., 2010).

Oh et al. (2009) report that their hybrid models improve the accuracy of English-to-Chinese transliteration. However, since their focus is on investigating the influence of Chinese phonemes, their hybrid model is again a simple linear combination of basic models.

2.2 Leveraging supplemental transliterations

Previous work that explore the idea of taking advantage of data from additional languages tend to employ supplemental transliterations indirectly, rather than to incorporate them directly into the generation process.

Khapra et al. (2010) propose a bridge approach of transliterating low-resource language pair (X, Y) by pivoting on an high-resource language Z, with the assumption that the pairwise data between (X, Z) and (Y, Z) is relatively large. Their experiments show that pivoting on Z results in lower accuracy than directly transliterating X into Y. Zhang et al. (2010) and Kumaran et al. (2010) combine the pivot model with a grapheme-based model, which works better than either of the two approaches alone. However, their model is not able to incorporate more than two languages.

Bhargava and Kondrak (2011) propose a reranking approach that uses supplemental transliterations to improve grapheme-to-phoneme conversion of names. Bhargava and Kondrak (2012) generalize this idea to improve transliteration accuracy by utilizing either transliterations from other languages, or phonetic transcriptions in the source language. Specifically, they apply an SVM reranker to the top-n outputs of a base spelling-based model. However, the post-hoc property of reranking is a limiting factor; it can identify the correct transliteration only if the base model includes it in its output candidate list.

3 Joint Generation

In this section, we describe our approach of the joint transduction of a transliteration T from a source orthographic string S and a source phonemic string P (Figure 1). We implement our approach by modifying the DIRECTL+ system of Jiampojamarn et al. (2010), which we describe in Section 3.1. In the following sections, we discuss other components of our approach, namely alignment (3.2), scoring (3.3), and search (3.4). In Section 3.5 we generalize the joint model to accept multiple input strings.

3.1 DirecTL+

DIRECTL+ (Jiampojamarn et al., 2010) is a discriminative string transducer which learns to convert source strings into target strings from a set of parallel training data. It requires pairs of strings to be aligned at the character level prior to training. M2M-ALIGNER (Jiampojamarn et al., 2007), an unsupervised EM-based aligner, is often used to generate such alignments. The output is a ranked list of candidate target strings with their confidence scores. Below, we briefly describe the scoring model, the training process, and the search algorithm.

The scoring model assigns a score to an aligned pair of source and target strings (S, T). Assuming there are m aligned substrings, such that the ith source substring generates the ith target substring, the score is computed with the following formula:

$$\sum_{i}^{m} \alpha \cdot \Phi(i, S, T) \quad (1)$$

where α is the weight vector, and Φ is the feature vector.

There are four sets of features. *Context* features are character n-grams within the source word. *Transition* features are character n-grams within the target word. *Linear-chain* features combine context features and transition features. *Joint n-gram* features further capture the joint information on both sides.

The feature weights α are learned with the Maximum Infused Relaxed Algorithm (MIRA) of Crammer and Singer (2003). MIRA aims to find the smallest change in current weights so that the new weights separate the correct target strings from incorrect ones by a margin defined by a loss function. Given the training instance (S, T) and the current feature weights α_{k-1}, the update of the feature weights can be described as the following optimization problem:

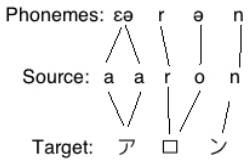

Figure 1: Triple alignment between the source phonemes, source graphemes, and the target graphemes ア ロ ン (A-RO-N).

$$\min_{\alpha_k} \|\alpha_k - \alpha_{k-1}\| \;\; s.t. \;\; \forall \hat{T} \in T_n :$$

$$\alpha_k \cdot (\Phi(S, T) - \Phi(S, \hat{T})) \geq loss(T, \hat{T})$$

where $\hat{T}$ is a candidate target in the n-best list T_n found under the current model parameterized by α_{k-1}. The loss function is the Levenshtein distance between T and $\hat{T}$.

Given an unsegmented source string, the search algorithm finds a target string that achieves the highest score according to the scoring model. It searches through all the possible segmentations of the source string and all possible target substrings using the following dynamic programming formulation:

$$Q(0, \$) = 0$$

$$Q(j, t) = \max_{t', t, j-N \leq j' < j} \alpha \cdot \phi(S^{j}_{j'+1}, t', t) + Q(j', t')$$

$$Q(J+1, \$) = \max_{t'} \alpha \cdot \phi(\$, t', \$) + Q(J, t')$$

$Q(j, t)$ is defined as the maximum score of the target sequence ending with target substring t, generated by the letter sequence $S_1 ... S_j$. ϕ describes the features extracted from the current generator substring $S^{j}_{j'+1}$ of target substring t, with t' to be the last generated target substring. N specifies the maximum length of the source substring. The \$ symbols are used to represent both the start and the end of a string. Assuming that the source string contains J characters, $Q(J+1, \$)$ gives the score of the highest scoring target string, which can be recovered through backtracking.

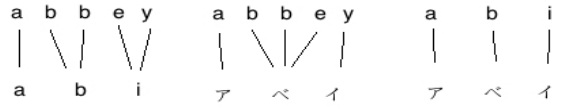

Figure 2: Three pairwise alignments between the English word *abbey*, its transcription [*abi*], and the Japanese transliteration アベイ (A-BE-I).

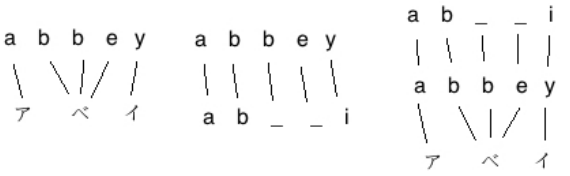

Figure 3: Obtaining a triple alignment by pivoting on the source word.

3.2 Multi-alignment

M2M-ALIGNER applies the EM algorithm to align sets of string pairs. For the purpose of joint generation, we need to align triples S, P and T prior to training. The alignment of multiple strings is a challenging problem (Bhargava and Kondrak, 2009). In general, there is no obvious way of merging three pairwise alignments. Figure 2 shows an example of three pairwise alignments that are mutually inconsistent: the English letter *e* is aligned to the phoneme [i] and to the grapheme ベ*(BE)*, which are not aligned to each other

Our solution is to select one of the input strings as the *pivot* for aligning the remaining two strings. Specifically, we align the pivot string to each of the other two strings through one-to-many alignments, where the maximum length of aligned substrings in the pivot string is set to one. Then we merge these two pairwise alignments according to the pivot string. Since the source phoneme string may or may not be available for a particular training instance, we use the source orthographic string as the pivot. The one-to-many pairwise alignments between the graphemes and phonemes, and between the graphemes and the transliterations are generated with M2M-ALIGNER. Figure 3 provides an example of this process.

An alternative approach is to pivot on the target string. However, because the target string is not available at test time, we need to search for the highest-scoring target string, given an unsegmented source string S and the corresponding unsegmented phoneme string P. We can generalize the original search algorithm by introducing another dimension into the dynamic-programming table for segmenting P, but it substantially increases the time complexity of the decoding process. Our development experiments indicated that pivoting on the target string not only requires more time, but also results in less accurate transliterations.

3.3 Scoring Model

The scoring formula (1) is extended to compute a linear combination of features of three aligned strings (S, P, T):

$$\sum_{i}^{m} \alpha \cdot [\Phi(i, S, T), \Phi(i, P, T)] \qquad (2)$$

The transition features on T are only computed once, because they are independent of the input strings. We observed no improvement by including features between S and P in our development experiments.

3.4 Search

Our search algorithm finds the highest-scoring target string, given a source string and a phoneme string. Since we pivot on the source string to achieve multiple alignment, the input to the search algorithm is actually one-to-many aligned pair of the source string and the phoneme string. The search space is therefore the same as that of DirecTL+, i.e. the product of all possible segmentations of the source string and all possible target substrings. However, since we apply one-to-many alignment, there is only one possible segmentation of the source string, which is obtained by treating every letter as a substring. We apply the same dynamic programming search as DirecTL+, except that we extend the feature extraction function $\phi(S^{j}_{j'+1}, t', t)$ in the original formulation to $[\phi(S^{j}_{j'+1}, t', t), \phi(P^{k}_{k'+1}, t', t)]$ so that features between the current phoneme substring $P^{k}_{k'+1}$ and the target substrings are taken into consideration. The time complexity of this search is only double of the complexity of DIRECTL+, and is independent of the length of the phoneme string.

3.5 Generalization

Since we may need to leverage information from other sources, e.g., phonemes of supplemental transliterations, each training instance can be composed of a source word, a target word, and a list of supplemental strings. The size of the list is not fixed because we may not have access to some of the supplemental strings for certain source words.

We first align all strings in each training instance by merging one-to-many pairwise alignments between the source word and every other string in the instance, as described in Section 3.2. The generalization of training is straightforward. For the scoring model, we extract the same set of features as before by pairing each supplemental string with the target word. Since the alignment is performed beforehand, the time complexity of the generalized search only increases linearly in the number of input strings with respect to the original complexity.

4 Leveraging transcriptions

In this section, we describe experiments that involve generating transliterations jointly from the source orthography and pronunciation. We test our method on the English-to-Hindi and English-to-Japanese transliteration data from the NEWS 2010 Machine Transliteration Shared Task (Li et al., 2010). We extract the corresponding English pronunciations from the Combilex Lexicon (Richmond et al., 2009). We split each transliteration dataset into 80% for training, 10% for development, 10% for testing. We limit the datasets to contain only transliterations that have phonetic transcriptions in Combilex, so that each entry is composed of a source English word, a source transcription, and a target Japanese or Hindi word. The final results are obtained by joining the training and development sets as the final training set. The final training/test sets contain 8,264/916 entries for English-to-Japanese, and 3,503/353 entries for English-to-Hindi.

4.1 Gold transcriptions

We compare three approaches that use different sources of information: (a) graphemes only; (b) phonemes only; and (c) both graphemes and phonemes. The first two approaches use DIRECTL+, while the last approach uses our joint model described in Section 3. We evaluate each approach by computing the word accuracy.

Model	En→Ja	En→Hi
Graphemes only	58.0	42.6
Phonemes only	52.4	39.4
Joint	**63.6**	**46.1**

Table 1: Transliteration word accuracy depending on the source information.

Table 1 presents the transliteration results. Even with gold-standard transcriptions, the phoneme-based model is worse than the grapheme-based model. This demonstrates that it is incorrect to refer to the process of transliteration as phonetic translation. On the other hand, our joint generation approach outperforms both single-source models on both test sets, which confirms that transliteration requires a joint consideration of orthography and pronunciation.

It is instructive to look at a couple of examples where outputs of the models differ. Consider the name *Marlon*, pronounced [mɑrlən], which is transliterated into Japanese as マロン *(MA-RO-N)* (correct), and マレン *(MA-RE-N)* (incorrect), by the orthographic and phonetic approaches, respectively. The letter bigram *lo* is always transliterated into ロ in the orthographic training data, while the phoneme bigram */lə/* has multiple correspondences in the phonetic training data. In this case, the unstressed vowel reduction process in English causes a loss of the orthographic information, which needs to be preserved in the transliteration.

In the joint model, the phonetic information sometimes helps disambiguate the pronunciation of the source word, thus benefiting the transliteration process. For example, the outputs of the three models for *haddock*, pronounced [hadək], are ハダク *(HA-DA-KU)* (phonetic), ハドドック *(HA-DO-DO-K-KU)* (orthographic), and ハドック *(HA-DO-K-KU)* (joint, correct). The phonetic model is again confused by the reduced vowel [ə], while the orthographic model mistakenly replicates the rendering of the consonant *d*, which is pronounced as a single phoneme.

Model	En→Ja	En→Hi
Graphemes only	63.1	43.5
Joint (gold phon.)	**67.4**	**48.0**
Joint (generated phon.)	65.8	46.1

Table 2: Transliteration accuracy improvement with gold and generated phonetic transcriptions.

Model	En→Ja	En→Hi
Graphemes only	53.3	46.4
Phonemes only	19.2	10.4
Joint (suppl. phonemes)	**54.8**	**50.0**

Table 3: Transliteration accuracy with transcriptions generated from third-language transliterations.

4.2 Generated Transcriptions

The training entries that have no corresponding transcriptions in our pronunciation lexicon were excluded from the experiment described above. When we add those entries back to the datasets, we can no longer apply the phonetic approach, but we can still compare the orthographic approach to our joint approach, which can handle the lack of a phonetic transcription in some of the training instances. The training sets are thus larger in the experiments described in this section: 30,190 entries for English-to-Japanese, and 12,070 for English-to-Hindi. The test sets are the same as in Section 4.1. The results in the first two rows in Table 2 show that the joint approach outperforms the orthographic approach even when most training entries lack the pronunciation information.[1]

Gold transcriptions are not always available, especially for names that originate from other languages. Next, we investigate whether we can replace the gold transcriptions with transcriptions that are automatically generated from the source orthography. We adopt DIRECTL+ as a grapheme-to-phoneme (G2P) converter, train it on the entire Combilex lexicon, and include the generated transcriptions instead of the gold transcriptions in the transliteration training and test sets for the joint model. The test sets are unchanged.

The third row in Table 2 shows the result of leveraging generated transcriptions. We still see improvement over the orthographic approach, albeit smaller than with the gold transcriptions. However, we need to be careful when interpreting these results. Since our G2P converter is trained on Combilex, the generated transcriptions of words in the test set are quite accurate. When we test the joint approach only on words that are *not* found in Combilex, the improvement over the orthographic approach largely disappears. We interpret this result as an indication that the generated transcriptions help mostly by capturing consistent grapheme-to-phoneme correspondences in the pronunciation lexicon.

[1]The improvement is statistically significant according to the McNemar test with $p < 0.05$. The differences in the baseline results between Table 1 and Table 2 are due to the differences in the training sets. The matching value of 46.1 across both tables is a coincidence. The comparison of results within any given table column is fair.

5 Leveraging transliterations

In the previous section, we have shown that phonetic transcriptions can improve the accuracy of the transliteration process by disambiguating the pronunciation of the source word. Unfortunately, phonetic transcriptions are rarely available, especially for words which originate from other languages, and generating them on the fly is less likely to help. However, transliterations from other languages constitute another potential source of information that could be used to approximate the pronunciation in the source language. In this section, we present experiments of leveraging such supplemental transliterations through our joint model.

5.1 Third-language transcriptions

An intuitive way of employing transliterations from another language is to convert them into phonetic transcriptions using a G2P model, which are then provided to our joint model together with the source orthography. We test this idea on the data from the NEWS 2010 shared task. We select Thai as the third language, because it has the largest number of the corresponding transliterations. We restrict the training and test sets to include only words for which Thai transliterations are available. The resulting English-to-Japanese and English-to-Hindi training/test sets contain 12,889/1,009, and 763/250 entries, respectively. We adopt DIRECTL+ as a G2P converter, and train it on 911 Thai spelling-pronunciation pairs extracted from Wiktionary. Be-

Language	Acc.	Data size
Thai	15.2	911
Hindi	25.9	819
Hebrew	21.3	475
Korean	40.9	3181

Table 4: Grapheme-to-phoneme word accuracy on the Wiktionary data.

Model	En→Ja	En→Hi
Graphemes only	54.5	46.1
Joint (suppl. phonemes)	**58.6**	**46.4**

Table 5: Transliteration accuracy with transcriptions generated from multiple transliterations.

cause of the small size of the training data, it can only achieve about 15% word accuracy in our G2P development experiment.

Table 3 shows the transliteration results. The accuracy of the model that uses only supplemental transcriptions (row 2) is very low, but the joint model obtains an improvement even with such inaccurate third-language transcriptions. Note that the Thai pronunciation is often quite different from English. For instance, the phoneme sequence [waj] obtained from the Thai transliteration of *Whyte*, helps the joint model correctly transliterate the English name into Japanese ホワイト*(HO-WA-I-TO)*, which is better than ホイト*(HO-I-TO)* produced by the orthographic model.

5.2 Multi-lingual transcriptions

Transcriptions obtained from a third language are not only noisy because of the imperfect G2P conversion, but often also lossy, in the sense of missing some phonetic information present in the source pronunciation. In addition, supplemental transliterations are not always available in a given third language. In this section, we investigate the idea of extracting phonetic information from multiple languages, with the goal of reducing the noise of generated transcriptions.

We first train G2P converters for several languages on the pronunciation data collected from Wiktionary. Table 4 shows the sizes of the G2P datasets, and the corresponding G2P word accuracy numbers, which are obtained by using 90% of the data for training, and the rest for testing.[2] For the highly-regular Japanese Katakana, we instead create a rule-based converter. Then we convert supplemental transliterations from those languages into noisy phonetic transcriptions. In order to obtain representative results, we also include transliteration pairs without supplemental transliterations, which results in different datasets than in the previous experiments. The sets for English-to-Japanese and English-to-Hindi now contain 30,190/17,557/1,886 and 12,070/3,777/380 entries, where the sizes refer to (1) the entire training set, (2) the subset of training entries that have at least one supplemental transcription, and (3) the test set (in which all entries have supplemental transcriptions).

An interlingual approach holds the promise of ultimately replacing n^2 pairwise grapheme-grapheme transliteration models involving n languages with $2n$ grapheme-phoneme and phoneme-grapheme models based on a unified phonetic representation. In our implementation, we merge different phonetic transcriptions of a given word into a single abstract vector representation. Specifically, we replace each phoneme with a phonetic feature vector according to a phonological feature chart, which includes features such as *labial, voiced*, and *tense*. After merging the vectors by averaging their weights, we incorporate them into the joint model described in Section 3.3 by modifying $\Phi(i, P, T)$. Unfortunately, the results are disappointing. It appears that the vector merging process compounds the information loss, which offsets the advantage of incorporating multiple transcriptions.

Another way of utilizing supplemental transcriptions is to provide them directly to our generalized joint model described in Section 3.5, which can handle multiple input strings. Table 5 presents the results on leveraging transcriptions generated from supplemental transliterations. We see that the joint generation from multiple transcriptions significantly boosts the accuracy on English-to-Japanese, but the improvement on English-to-Hindi is minimal.

[2]We use the entire datasets to train G2P converters for the transliteration experiments, but their accuracy is unlikely to improve much due to a small increase in the training data.

Model	En→Ja	Ja→En	En→Hi	Hi→En
DIRECTL+	51.5	19.7	43.4	42.6
Reranking	56.8	30.3	50.8	48.9
Joint	56.4	38.8	51.6	51.1
Joint + Reranking	**57.0**	44.6	**53.0**	57.2
+ Lexicon	-	**53.1**	-	**61.7**

Table 6: Transliteration accuracy with supplemental information.

5.3 Multi-lingual transliterations

The generated transcriptions of supplemental transliterations discussed in the previous section are quite inaccurate because of small and noisy G2P training data. In addition, we are prevented from taking advantage of supplemental transliterations from other languages by the lack of the G2P training data. In order to circumvent these limitations, we propose to directly incorporate supplemental transliterations into the generation process. Specifically, we train our generalized joint model on the graphemes of the source word, as well as on the graphemes of supplemental transliterations.

The experiments that we have conducted so far suggest two additional methods of improving the transliteration accuracy. We have observed that n-best lists produced by our joint model contain the correct transliteration more often than the baseline models. Therefore, we follow the joint generation with a reranking step, in order to boost the top-1 accuracy. We apply the reranking algorithm of Bhargava and Kondrak (2011), except that our joint model is the base system for reranking. In order to ensure fair comparison, the held-out sets for training the rerankers are subtracted from the original training sets.

Another observation that we aim to exploit is that a substantial number of the outputs generated by our joint model are very close to gold-standard transliterations. In fact, news writers often use slightly different transliterations of the same name, which makes the model's task more difficult. Therefore, we rerank the model outputs using a target-language lexicon, which is a list of words together with their frequencies collected from a raw corpus. We follow Cherry and Suzuki (2009) in extracting lexicon features for a given word according to coarse bins, i.e., $[< 2000]$, $[< 200]$, $[< 20]$, $[< 2]$, $[< 1]$. For example, a word with the frequency 194 will cause the features $[< 2000]$ and $[< 200]$ to fire.

We conduct our final experiment on forward and backward transliteration. We utilize supplemental transliterations from all eight languages in the NEWS 2010 dataset. The English-Japanese and English-Hindi datasets contain 33,540 and 13,483 entries, of which 23,613 and 12,131 have at least one supplemental transliteration, respectively. These sets are split into training/development/test sets. The entries that have no supplemental transliterations are removed from the test sets, which results in 2,321 and 1,226 test entries. In addition, we extract an English lexicon comprising 7.5M word types from the English gigaword monolingual corpus (LDC2012T21) for the back-transliteration experiments.

We evaluate the following models: (1) the baseline DIRECTL+ model trained on source graphemes; (2) the reranking model of Bhargava and Kondrak (2011)[3], with DIRECTL+ as the base system; (3) our joint model described in Section 3.5; (4) "combination", which is a reranking model with our joint model as the base system; and (5) a reranking model that uses the English target lexicon and model (4) as the base system.

Table 6 present the results. We see that our joint model performs much better by directly incorporating the supplemental transliterations than by using the corresponding phonetic transcriptions. This is consistent with our experiments in Section 4 that show the importance of the orthographic information. We also observe that our joint model achieves substantial improvements over the baseline on the back-transliteration tasks from Japanese and Hindi into English. This result suggests the orthographic information from the supplemental transliterations is

[3]Code from http://www.cs.toronto.edu/~aditya/g2p-tl-rr/

particularly effective in recovering the information about the pronunciation of the original word which is often obfuscated by the transliteration into a different language.

Our joint model is more effective in utilizing supplemental transliterations than the reranking approach of Bhargava and Kondrak (2011), except on English-to-Japanese. The combination of these two approaches works better than either of them, particularly on the back-transliteration tasks. Finally, the incorporation of a target-lexicon brings additional gains.

Back-transliteration from Japanese to English is more challenging than in the forward direction, which was already noted by Knight and Graehl (1997). Most of the names in the dataset originate from English, and Japanese phonotactics require introduction of extra vowels to separate consonant clusters. During back-transliteration, it is often unclear which vowels should be removed and which preserved. Our approach is able to dramatically improve the quality of the results by recovering the original information from multiple supplemental transliterations.

6 Conclusion

We have investigated the relative importance of the orthographic and phonetic information in the transliteration process. We have proposed a novel joint generation model that directly utilizes both sources of information. We have shown that a generalized joint model is able to achieve substantial improvements over the baseline represented by a state-of-the-art transliteration tool by directly incorporating multiple supplemental transliterations. In the future, we would like to further explore the idea of using interlingual representations for transliteration without parallel training data.

Acknowledgements

We thank Adam St Arnaud for help in improving the final version of this paper.

This research was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

Yaser Al-Onaizan and Kevin Knight. 2002. Machine transliteration of names in Arabic texts. In *Proceedings of the ACL-02 Workshop on Computational Approaches to Semitic Languages*, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Aditya Bhargava and Grzegorz Kondrak. 2009. Multiple word alignment with Profile Hidden Markov Models. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Student Research Workshop and Doctoral Consortium*, pages 43–48, Boulder, Colorado, June. Association for Computational Linguistics.

Aditya Bhargava and Grzegorz Kondrak. 2011. How do you pronounce your name? Improving G2P with transliterations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 399–408, Portland, Oregon, USA, June. Association for Computational Linguistics.

Aditya Bhargava and Grzegorz Kondrak. 2012. Leveraging supplemental representations for sequential transduction. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 396–406, Montréal, Canada, June. Association for Computational Linguistics.

Slaven Bilac and Hozumi Tanaka. 2004. A hybrid back-transliteration system for Japanese. In *Proceedings of Coling 2004*, pages 597–603, Geneva, Switzerland, Aug 23–Aug 27. COLING.

Colin Cherry and Hisami Suzuki. 2009. Discriminative substring decoding for transliteration. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1066–1075, Singapore, August. Association for Computational Linguistics.

Koby Crammer and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *J. Mach. Learn. Res.*, 3:951–991, March.

Sittichai Jiampojamarn, Grzegorz Kondrak, and Tarek Sherif. 2007. Applying many-to-many alignments and hidden Markov models to letter-to-phoneme conversion. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 372–379, Rochester, New York, April. Association for Computational Linguistics.

Sittichai Jiampojamarn, Colin Cherry, and Grzegorz Kondrak. 2010. Integrating joint n-gram features into a discriminative training framework. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 697–700, Los Angeles, California, June. Association for Computational Linguistics.

Mitesh M. Khapra, A Kumaran, and Pushpak Bhattacharyya. 2010. Everybody loves a rich cousin: An empirical study of transliteration through bridge languages. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 420–428, Los Angeles, California, June. Association for Computational Linguistics.

Kevin Knight and Jonathan Graehl. 1997. Machine transliteration. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pages 128–135, Madrid, Spain, July. Association for Computational Linguistics.

A. Kumaran, Mitesh M. Khapra, and Pushpak Bhattacharyya. 2010. Compositional machine transliteration. *ACM Transactions on Asian Language Information Processing (TALIP)*, 9(4):13:1–13:29, December.

Haizhou Li, Min Zhang, and Jian Su. 2004. A joint source-channel model for machine transliteration. In *Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Haizhou Li, A Kumaran, Min Zhang, and Vladimir Pervouchine. 2010. Report of NEWS 2010 transliteration generation shared task. In *Proceedings of the 2010 Named Entities Workshop*, pages 1–11, Uppsala, Sweden, July. Association for Computational Linguistics.

Jong-Hoon Oh and Key-Sun Choi. 2005. Machine learning based English-to-Korean transliteration using grapheme and phoneme information. *IEICE - Trans. Inf. Syst.*, E88-D(7):1737–1748, July.

Jong-Hoon Oh, Kiyotaka Uchimoto, and Kentaro Torisawa. 2009. Can Chinese phonemes improve machine transliteration?: A comparative study of English-to-Chinese transliteration models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, EMNLP '09, pages 658–667, Stroudsburg, PA, USA. Association for Computational Linguistics.

Korin Richmond, Robert Clark, and Sue Fitt. 2009. Robust LTS rules with the Combilex speech technology lexicon. In *Proceedings of Interspeech*, pages 1259–1298, Brighton, UK, September.

Tarek Sherif and Grzegorz Kondrak. 2007. Substring-based transliteration. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 944–951, Prague, Czech Republic, June. Association for Computational Linguistics.

Min Zhang, Xiangyu Duan, Vladimir Pervouchine, and Haizhou Li. 2010. Machine transliteration: Leveraging on third languages. In *Coling 2010: Posters*, pages 1444–1452, Beijing, China, August. Coling 2010 Organizing Committee.

Min Zhang, Haizhou Li, A Kumaran, and Ming Liu. 2012. Whitepaper of NEWS 2012 shared task on machine transliteration. In *Proceedings of the 4th Named Entity Workshop*, pages 1–9, Jeju, Korea, July. Association for Computational Linguistics.