Transliteration Experiments on Chinese and Arabic

Grzegorz Kondrak, Xingkai Li and **Mohammad Salameh**
Department of Computing Science
University of Alberta
Edmonton, AB, Canada, T6G 2E8
{gkondrak,xingkai,msalameh}@ualberta.ca

Abstract

We report the results of our transliteration experiments with language-specific adaptations in the context of two language pairs: English to Chinese, and Arabic to English. In particular, we investigate a syllable-based Pinyin intermediate representation for Chinese, and a letter mapping for Arabic.

1 Introduction

Transliteration transforms an orthographic form of a word in one writing script into an orthographic form of the same word in another writing script. The problem is challenging because the relationship between the source and target representations is often ambiguous. The process is further complicated by restrictions in the target phonological system.

DIRECTL+ (Jiampojamarn et al., 2010a) is an online discriminative training system that incorporates joint n-gram features and many-to-many alignments, which are generated by M2M-ALIGNER (Jiampojamarn et al., 2007). Our team employed variants of DIRECTL+ in the previous editions of the Shared Task on Transliteration (Jiampojamarn et al., 2009; Jiampojamarn et al., 2010b; Bhargava et al., 2011). Recently, Bhargava and Kondrak (2012) show significant improvement in accuracy for the English-to-Japanese task by leveraging supplemental transliterations from other scripts.

In this edition of the Shared Task on Transliteration, we experiment with language-specific adaptations for the EnCh and ArEn data sets. The structure of the paper is as follows. In Section 2, we provide details about the system parameters used in M2M-ALIGNER and DIRECTL+. Section 3 provides details of our strategies adopted in the EnCh task, which incorporate Chinese-specific knowledge and system combination algorithm. In Section 4 we elaborate on the difficulty of Arabic name transliteration and propose a letter mapping scheme. In Section 5 we present the official test results.

2 Base System

We run DIRECTL+ with all of the features described in (Jiampojamarn et al., 2010a). System parameters were determined during development. For the EnCh experiments, we set the context feature size to 5, the transition feature size to 2, and the joint n-gram feature size to 6. For the ArEn experiments, we used the same settings, except that we set the joint n-gram feature size to 5.

The M2M-ALIGNER parameters were set as follows. For the English-Pinyin alignment, the maximum substring length was 1 on the English side, and 2 on the Pinyin side, with empty substrings (*nulls*) allowed only on the Pinyin side. For ArEn, the maximum substring length was 2 for both sides.

3 English to Chinese

In this section, we introduce the strategies for improving DIRECTL+ performance on the EnCh task, including the use of Chinese Pinyin for preprocessing, and the combination of different models.

3.1 Data preprocessing and cleaning

In general, the preprocessing is limited to removing letter case distinctions in English names, and replacing every non-initial letter *x* with *ks*. However, we observed that the provided development set contains a number of entries (about 3%) that contain multiple English words on the source side, but no corresponding separators on the target side, whereas no such entries occur in the training or testing set. Since this discrepancy between sets may cause problems for alignment and generation, we separated the multi-word entries into individual words (using whitespace and apostrophes as delimiters) and manually selected proper transliteration targets for them. We also removed individual words that have no corresponding transliterations on the target side. The cleaned development set contains 2483 entries.

3.2 Alignment via Pinyin

Following Jiampojamarn et al. (2009; 2010b), we utilize Pinyin as an intermediate representation of Chinese characters during M2M alignment with the objective of improving its quality. Pinyin is the formally-adopted Romanization system for Standard Mandarin for the mapping of Chinese characters to Roman alphabet. It uses the 26 letters of the English alphabet except for the letter v, with the addition of the letter *ü*. Every Chinese character can be represented by a sequence of Pinyin letters according to the way it is pronounced. Numerous freely available online tools exist for facilitating Chinese-Pinyin conversion[1].

In our experiments, the original Chinese characters from the target side of the training set are converted to Pinyin before M2M alignment. A small part of them (about 50 out of approximately 500 distinct Chinese characters in the Shared Task data) have multiple pronunciations, and can thus be represented by different Pinyin sequences. For those characters we manually select the pronunciations that are normally used for names.

After the alignment between English and Pinyin representation has been generated by M2M-ALIGNER, we use it to derive the alignment between English and Chinese characters, which is then used for training DIRECTL+. This preprocessing step results in a more accurate alignment as it substantially reduces the number of target symbols from around 500 distinct Chinese characters to 26 Pinyin letters.

[1]For instance, `http://www.chinesetopinyin.com`

Our approach is to utilize Pinyin only in the alignment phase, and converts it back to Chinese characters before the training phase. We do not incorporate Pinyin into the generation phase in order to avoid problems involved in converting the transliteration results from Pinyin back to Chinese characters. For example, a Pinyin subsequence may have multiple Chinese character mappings because of the fact that many Chinese characters have the same Pinyin representation. In addition, it is not always clear how to partition the Pinyin sequence into substrings corresponding to individual Chinese characters.

The choice of the appropriate Chinese character sequence is the problem further complicating the conversion from Pinyin. We experimented with a trigram language model trained on the target Chinese side of the training set for the purpose of identifying the correct transliteration result. However, this approach yielded low accuracy on the development set. In contrast, the strategy of using Pinyin only for the alignment introduces no ambiguity because we know the mapping between Pinyin sequences and the target Chinese side of the training set.

3.3 Syllabic Pinyin

The Pinyin sequences representing the pronunciations of Chinese characters should not be interpreted as combinations of individual letters. Rather, a Mandarin phonetic syllable (the pronunciation of one Chinese character) is composed of an optional onset (“initial”) followed by an obligatory rhyme (“final”). The rhyme itself is composed of an obligatory nucleus followed by an optional coda. Phonetically, the onset contains a single consonant, the nucleus contains a vowel or a diphthong, and the coda contains a single consonant ([r], [n] or [ŋ]). Both the onset and the rhyme can be represented by either a single letter or sequence of two or three letters. It is the initials and finals listed in Table 1 rather than Pinyin letters that are the phonemic units of Pinyin for Standard Mandarin. The pronunciation of a multi-letter initial/final is often different from the pronunciation of the sequence of its individual letters. Treating converted Pinyin as a sequence of separate letters may result in an incorrect phonetic transcription.

In this paper, we further experiment with encoding the converted sequences of Pinyin letters as the sequences of initials and finals for M2M alignment.

Initials							
b	p	m	f	d	t	n	l
g	k	h	j	q	x	zh	ch
sh	r	z	c	s	y	w	

Finals							
a	o	e	i	u	ü	ai	ei
ui	ao	ou	iu	ie	üe	er	an
en	in	un	ün	ang	eng	ing	ong

Table 1: The initials and finals in Chinese Pinyin.

Although the size of the alphabet increases from 26 letters to 47 initials and finals, the original Chinese pronunciation is represented more precisely. We refer to the new model which is trained on Pinyin initials and finals as PINYIN-SYL, and to the previously proposed model which is trained on Pinyin letters as PINYIN-LET.

3.4 System combination

The combination of models based on different principles may lead to improved prediction accuracy. We adopt the simple voting algorithm for system combination proposed by Jiampojamarn et al. (2009), with minor modifications. Since here we combine only two systems (PINYIN-LET and PINYIN-SYL), the algorithm becomes even simpler. We first rank the participating models according to their overall top-1 accuracy[2] on the development set. Note that the n-best list produced by DIRECTL+ may contain multiple copies of the same output which differ only in the implied input-output alignment. We allow such duplicates to contribute to the voting tally. The top-1 prediction is selected from the set of top-1 predictions produced by the participating models, with ties broken by voting and the preference for the highest-ranking system. For constructing n-best candidate lists, we order the candidate transliterations according to the highest rank assigned by either of the systems, with ties again broken by voting and the preference for the highest-ranking system. We refer to this combined model as COMBINED.

Table 2 shows the results of the three discussed approaches trained on the original training set, and tested on the cleaned development set. PINYIN-SYL performs slightly better than PINYIN-LET, which hints at the advantage of using Pinyin initials and finals over Pinyin letters as the intermediate representation during the alignment. The combination of the two models produces a marginally higher F-score[3]. The likely reason for the limited gain is the strong similarity of the two combined models. We experimented with adding a third model that is trained directly on the original Chinese characters without using Pinyin as the intermediate representation, but its accuracy was lower, and the accuracy of the resulting combined model was below PINYIN-SYL.

System	top-1	F-score
PINYIN-LET	0.296	0.679
PINYIN-SYL	0.302	0.681
COMBINED	0.304	0.682

Table 2: Development results on EnCh.

4 Arabic to English

Arabic script has 36 letters and 9 diacritics. Among these letters, the letters *Alif* and *Yaa* can be represented in different forms (أ آ إ ا and ي ى, respectively). The ArEn data set contains Arabic names without diacritics, which adds ambiguity to the transliteration task. When transliterated, such diacritics would appear as an English vowel. For example, it is difficult to tell whether the correct transliteration of the two-letter name بج is *Baj*, *Buj* or *Bij* because of the lacking vowel diacritic. Also, some Arabic consonants are transliterated into double English consonant because of the Shadda diacritic. Finally, some letters might have a different pronunciation (or none) when they occur at the end of the Arabic word. For example, the final letter ى is pronounced differently in أتمنى (*Atamana*) and بجانى (*Bagani*).

In the transliterations provided in the ArEn dataset, the different forms of *Alif*, the *Hamza* letter (ء), and the *Ain* letter (ع) are sometimes rendered as an apostrophe. In order to reduce the ambiguity, we devised a mapping shown in Table 3. The

[2]Word accuracy in top-1 evaluates only the top transliteration candidate produced by a transliteration system.

[3]The mean F-score measures how different, on average, the top transliteration candidate is from its closest reference.

Arabic	English
أ آ ا Alif forms, ة Taa Marbouta	a
ص Sahd, س Seen	s
ض Dahd, د Dal	d
ط Tah, ت Taa	t

Table 3: The mapping of Arabic letters to their English equivalents.

mapping reduces sets of Arabic letters that have the same corresponding English letter to a single higher-frequency symbol. For example, both ض and د characters tend to correspond to the letter d in English, so we replace all occurrences of the former with the latter. We refer to this variant as Letter-Map, as opposed to No-Map, which is the baseline system with no additional mapping.

Arabic compound names may be separated by space in their Arabic form or when transliterated. We treated the space similar to any alphabetic character. Also, any punctuation characters such as the apostrophe and hyphen on the English side are also treated as an alphabetic character.

System	top-1	F-score
No-Map	0.529	0.926
Letter-Map	0.519	0.925

Table 4: Development results on ArEn.

Table 4 shows our results on the original development set (2588 names). For these experiments, we split the original training set into a new training (25114 names) and development (2064 names) sets. The results indicate that the additional mapping actually decreases the overall accuracy with respect to the baseline. It seems that the mapping decreases the amount of information available to DirecTL+, without sufficiently reducing the ambiguity. This confirms the previous findings that manually crafted rules for transliteration are generally ineffective (Karimi et al., 2011).

5 Final results

Table 5 shows our results as provided by the Shared Task organizers. For the EnCh task submission, we trained the Pinyin-Let and Pinyin-Syl models on the set that includes both the original training set and the cleaned development set. The output of the Combined system was designated as our Primary Run. The final results generally agree with our development results presented in Section 3, but the performance differences between models are smaller. For the ArEn task, we decided not to submit the output of the Letter-Map version because of the negative outcome of our development experiment.

Task	System	top-1	F-score
EnCh	Pinyin-Let	0.324	0.668
	Pinyin-Syl	0.325	0.673
	Combined	0.325	0.672
ArEn	No-Map	0.583	0.933

Table 5: Official test results.

According to the top-1 measure, our primary system was ranked second on the English-to-Chinese task, and third on the Arabic-to-English task. In both cases, we were within 0.5% of the best top-1 result. In addition, in both cases, we obtained the best results among the primary systems according to the F-score measure.

6 Conclusion

In this paper, we described our submission to the NEWS 2012 Shared Task on Machine Transliteration. In the EnCh task, our focus was on generating better alignment by employing Pinyin as the intermediate representation. A more coarse-grained representation that uses Pinyin initials and finals appears to be a step in the right direction. In the ArEn task, we found that reducing the number of distinct Arabic characters does not improve the accuracy of the base system.

Acknowledgments

We wish to thank Sittichai Jiampojamarn for help with DirecTL+. This research was supported by the Natural Sciences and Engineering Research Council of Canada.

References

Aditya Bhargava and Grzegorz Kondrak. 2012. Leveraging supplemental representations for sequential transduction. In *Proceedings of the NAACL-HLT*.

Aditya Bhargava, Bradley Hauer, and Grzegorz Kondrak. 2011. Leveraging transliterations from multiple languages. In *Proceedings of the 3rd Named Entities Workshop (NEWS 2011)*, pages 36–40, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.

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, 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)*, pages 28–31, Suntec, Singapore, August. Association for Computational Linguistics.

Sittichai Jiampojamarn, Colin Cherry, and Grzegorz Kondrak. 2010a. 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.

Sittichai Jiampojamarn, Kenneth Dwyer, Shane Bergsma, Aditya Bhargava, Qing Dou, Mi-Young Kim, and Grzegorz Kondrak. 2010b. Transliteration generation and mining with limited training resources. In *Proceedings of the 2010 Named Entities Workshop*, pages 39–47, Uppsala, Sweden, July. Association for Computational Linguistics.

Sarvnaz Karimi, Falk Scholer, and Andrew Turpin. 2011. Machine transliteration survey. *ACM Comput. Surv.*, 43(3):17:1–17:46, April.