Experiences with English-Hindi, English-Tamil and English-Kannada Transliteration Tasks at NEWS 2009

Manoj Kumar Chinnakotla and **Om P. Damani**
Department of Computer Science and Engineering,
IIT Bombay,
Mumbai, India
{manoj,damani}@cse.iitb.ac.in

Abstract

We use a Phrase-Based Statistical Machine Translation approach to Transliteration where the words are replaced by characters and sentences by words. We employ the standard SMT tools like GIZA++ for learning alignments and Moses for learning the phrase tables and decoding. Besides tuning the standard SMT parameters, we focus on tuning the Character Sequence Model (CSM) related parameters like order of the CSM, weight assigned to CSM during decoding and corpus used for CSM estimation. Our results show that paying sufficient attention to CSM pays off in terms of increased transliteration accuracies.

1 Introduction

Transliteration of Named-Entities (NEs) is an important problem that affects the accuracy of many NLP applications like Cross Lingual Search and Machine Translation. *Transliteration* is defined as the process of automatically mapping a given grapheme sequence in the source language to a grapheme sequence in the target language such that it preserves the pronunciation of the original source word. A *Grapheme* refers to the unit of written language which expresses a phoneme in the language. Multiple alphabets could be used to express a grapheme. For example, *sh* is considered a single grapheme expressing the phoneme */SH/*. For phonetic orthography like Devanagari, each grapheme corresponds to a unique phoneme. However, for English, a grapheme like *c* may map to multiple phonemes */S/,/K/*. An example of transliteration is mapping the Devanagari grapheme sequence प्रिन्स हैरी to its phonetically equivalent grapheme sequence *Prince Harry* in English.

This paper discusses our transliteration approach taken for the NEWS 2009 Machine Transliteration Shared Task [Li et al.2009b, Li et al.2009a]. We model the transliteration problem as a Phrased-Based Machine Translation problem. Later, using the development set, we tune the various parameters of the system like order of the Character Sequence Model (CSM), typically called language model, weight assigned to CSM during decoding and corpus used to estimate the CSM. Our results show that paying sufficient attention to the CSM pays off in terms of improved accuracies.

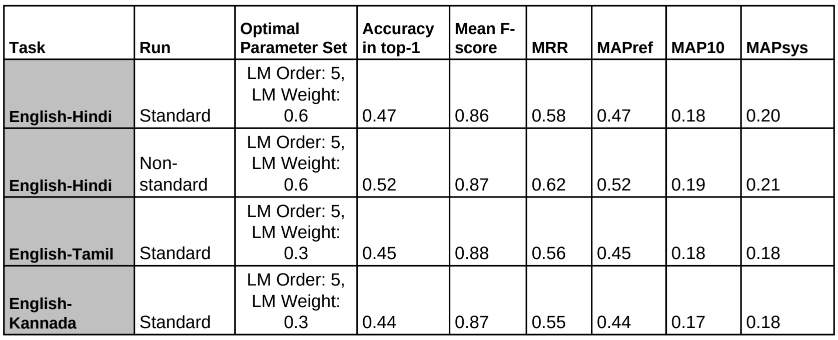
2 Phrase-Based SMT Approach to Transliteration

In the Phrase-Based SMT Approach to Transliteration [Sherif and Kondrak2007, Huang2005], the words are replaced by characters and sentences are replaced by words. The corresponding noisy channel model formulation where a given english word **e** is to be transliterated into a foreign word **h**, is given as:

$$\begin{aligned} h^* &= \operatorname*{argmax}_h Pr(h|e) \\ &= \operatorname*{argmax}_h Pr(e|h) \cdot Pr(h) \end{aligned} \tag{1}$$

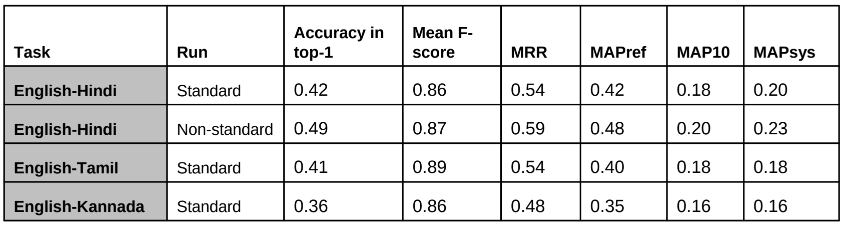
In Equation 1, $Pr(e|h)$ is known as the *translation model* which gives the probability that the character sequence h could be transliterated to e and $Pr(h)$ is known as the *character sequence model* typically called language model which gives the probability that the character sequence h forms a valid word in the target language.

Proceedings of the 2009 Named Entities Workshop, ACL-IJCNLP 2009, pages 44–47,
Suntec, Singapore, 7 August 2009. ©2009 ACL and AFNLP



Task	Run	Optimal Parameter Set	Accuracy in top-1	Mean F-score	MRR	MAPref	MAP10	MAPsys
English-Hindi	Standard	LM Order: 5, LM Weight: 0.6	0.47	0.86	0.58	0.47	0.18	0.20
English-Hindi	Non-standard	LM Order: 5, LM Weight: 0.6	0.52	0.87	0.62	0.52	0.19	0.21
English-Tamil	Standard	LM Order: 5, LM Weight: 0.3	0.45	0.88	0.56	0.45	0.18	0.18
English-Kannada	Standard	LM Order: 5, LM Weight: 0.3	0.44	0.87	0.55	0.44	0.17	0.18

Figure 1: NEWS 2009 Development Set Results



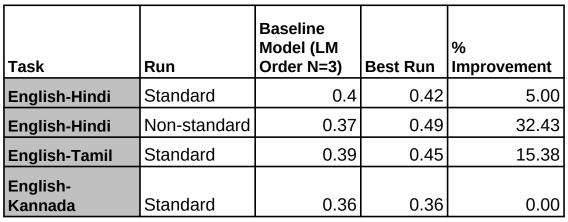
Task	Run	Accuracy in top-1	Mean F-score	MRR	MAPref	MAP10	MAPsys
English-Hindi	Standard	0.42	0.86	0.54	0.42	0.18	0.20
English-Hindi	Non-standard	0.49	0.87	0.59	0.48	0.20	0.23
English-Tamil	Standard	0.41	0.89	0.54	0.40	0.18	0.18
English-Kannada	Standard	0.36	0.86	0.48	0.35	0.16	0.16

Figure 2: NEWS 2009 Test Set Results

Given the parallel training data pairs, we preprocessed the source (English) and target (Hindi, Tamil and Kannada) strings into character sequences. We then ran the GIZA++ [Och and Ney2003] aligner with default options to obtain the character-level alignments. For alignment, except for Hindi, we used single character-level units without any segmentation. In case of Hindi, we did a simple segmentation where we added the halant character (U094D) to the previous Hindi character. Moses Toolkit [Hoang et al.2007] was then used to learn the phrase-tables for English-Hindi, English-Tamil and English-Kannada. We also learnt the character sequence models on the target language training words using the SRILM toolkit [Stolcke2002]. Given a new English word, we split the word into sequence of characters and run the Moses decoder with the phrase-table of target language obtained above to get the transliterated word. We ran Moses with the *DISTINCT* option to obtain the top *k* distinct transliterated options.

2.1 Moses Parameter Tuning

The Moses decoder computes the cost of each translation as a product of probability costs of four models: a) translation model b) language model c) distortion model and d) word penalty as shown in Equation 2. The distortion model controls the cost of re-ordering phrases (transliteration units) in a given sentence (word) and the word penalty model controls the length of the final translation. The parameters λ_T, λ_{CSM}, λ_D and λ_W control the relative importance given to each of the above models.



Task	Run	Baseline Model (LM Order N=3)	Best Run	% Improvement
English-Hindi	Standard	0.4	0.42	5.00
English-Hindi	Non-standard	0.37	0.49	32.43
English-Tamil	Standard	0.39	0.45	15.38
English-Kannada	Standard	0.36	0.36	0.00

Figure 3: Improvements Obtained over Baseline on Test Set due to Language Model Tuning

$$Pr(h|e) = Pr_T(e|h)^{\lambda_T} \cdot Pr_{CSM}(h)^{\lambda_{CSM}} \cdot Pr_D(h,e)^{\lambda_D} \cdot \omega^{length(h) \cdot \lambda_W} \qquad (2)$$

Since no re-ordering of phrases is required during translation task, we assign a zero weight to λ_D. Similarly, we varied the word penalty factor λ_W between $\{-1, 0, +1\}$ and found that it achieves maximum accuracy at 0. All the above tuning was done with a trigram CSM and default weight (0.5) in Moses for λ_T.

2.2 Improving CSM Performance

In addition to the above mentioned parameters, we varied the order of the CSM and the monolingual corpus used to estimate the CSM. For each task, we started with a trigram CSM as mentioned above and tuned both the order of the CSM and λ_{CSM} on the development set. The optimal set of parameters and the development set results are shown in Figure 1. In addition, we use a monolingual Hindi corpus of around 0.4 million documents called Guruji corpus. We extracted the 2.6 million unique words from the above corpus and trained a CSM on that. This CSM which was learnt on the monolingual Hindi corpus was used for the non-standard Hindi run. We repeat the above procedure of tuning the order of CSM and λ_{CSM} and find the optimal set of parameters for the non-standard run on the development set.

3 Results and Discussion

The details of the NEWS 2009 dataset for Hindi, Kannada and Tamil are given in [Li et al.2009a, Kumaran and Kellner2007]. The final results of our system on the test set are shown in Figure 2. Figure 3 shows the improvements obtained on test set by tuning the CSM parameters. The trigram CSM model used along with the optimal Moses parameter set tuned on development set was taken as baseline for the above experiments. The results show that a major improvement (32.43%) was obtained in the non-standard run where the monolingual Hindi corpus was used to learn the CSM. Because of the use of monolingual Hindi corpus in the non-standard run, the transliteration accuracy improved by 22.5% when compared to the standard run. The improvements (15.38%) obtained in Tamil are also significant. However, the improvement in Hindi standard run was not significant. In Kannada, there was no improvement due to tuning of LM parameters. This needs further investigation.

The above results clearly highlight the importance of improving CSM accuracy since it helps in improving the transliteration accuracy. Moreover, improving the CSM accuracy only requires monolingual language resources which are easy to obtain when compared to parallel transliteration training data.

4 Conclusion

We presented the transliteration system which we used for our participation in the NEWS 2009 Machine Transliteration Shared Task on Transliteration. We took a Phrase-Based SMT approach to transliteration where words are replaced by characters and sentences by words. In addition to the standard SMT parameters, we tuned the CSM related parameters like order of the CSM, weight assigned to CSM and corpus used to estimate the CSM. Our results show that improving the accuracy of CSM pays off in terms of improved transliteration accuracies.

Acknowledgements

We would like to thank the Indian search-engine company Guruji (`http://www.guruji.com`) for providing us the Hindi web content which was used to train the language model for our non-standard Hindi runs.

References

Hieu Hoang, Alexandra Birch, Chris Callison-burch, Richard Zens, Rwth Aachen, Alexandra Constantin, Marcello Federico, Nicola Bertoldi, Chris Dyer, Brooke Cowan, Wade Shen, Christine Moran, and Ondej Bojar. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *In Proceedings of ACL, Demonstration Session*, pages 177–180.

Fei Huang. 2005. Cluster-specific Named Entity Transliteration. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 435–442, Morristown, NJ, USA. Association for Computational Linguistics.

A. Kumaran and Tobias Kellner. 2007. A Generic Framework for Machine Transliteration. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 721–722, New York, NY, USA. ACM.

Haizhou Li, A Kumaran, Vladimir Pervouchine, and Min Zhang. 2009a. Report on NEWS 2009 Machine Transliteration Shared Task. In *Proceedings of ACL-IJCNLP 2009 Named Entities Workshop (NEWS 2009).*

Haizhou Li, A Kumaran, Min Zhang, and Vladimir Pervouchine. 2009b. Whitepaper of NEWS 2009 Machine Transliteration Shared Task. In *Proceedings of ACL-IJCNLP 2009 Named Entities Workshop (NEWS 2009).*

Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51.

Tarek Sherif and Grzegorz Kondrak. 2007. Substring-Based Transliteration. In *In Proceedings of ACL 2007*. The Association for Computer Linguistics.

Andreas Stolcke. 2002. SRILM - An Extensible Language Modeling Toolkit. In *In Proceedings of Intl. Conf. on Spoken Language Processing*.