Modeling Machine Transliteration as a Phrase Based Statistical Machine Translation Problem

Taraka Rama, Karthik Gali
Language Technologies Research Centre,
IIIT, Hyderabad, India.
{taraka,karthikg}@students.iiit.ac.in

Abstract

In this paper we use the popular phrase-based SMT techniques for the task of machine transliteration, for English-Hindi language pair. Minimum error rate training has been used to learn the model weights. We have achieved an accuracy of 46.3% on the test set. Our results show these techniques can be successfully used for the task of machine transliteration.

1 Introduction

Transliteration can be defined as the task of transcribing the words from a source script to a target script (Surana and Singh, 2008). Transliteration systems find wide applications in Cross Lingual Information Retrieval Systems (CLIR) and Machine Translation (MT) systems. The systems also find use in sentence aligners and word aligners (Aswani and Gaizauskas, 2005). Transcribing the words from one language to another language without the use of a bilingual lexicon is a challenging task as the output word produced in target language should be such that it is acceptable to the readers of the target language. The difficulty arises due to the huge number of Out Of Vocabulary (OOV) words which are continuously added into the language. These OOV words include named entities, technical words, borrowed words and loan words.

In this paper we present a technique for transliterating named entities from English to Hindi using a small set of training and development data. The paper is organised as follows. A survey of the previous work is presented in the next subsection. Section 2 describes the problem modeling which we have adopted from (Rama et al., 2009) which they use for L2P task. Section 3 describes how the parameters are tuned for optimal performance. A brief description of the data sets is provided in Section 4. Section 5 has the results which we have obtained for the test data. Finally we conclude with a summary of the methods and a analysis of the errors.

1.1 Previous Work

Surana and Singh (2008) propose a transliteration system in which they use two different ways of transliterating the named entities based on their origin. A word is classified into two classes either Indian or foreign using character based n-grams. They report their results on Telugu and Hindi data sets. Sherif and Kondrak (2007) propose a hybrid approach in which they use the Viterbi-based monotone search algorithm for searching the possible candidate transliterations. Using the approach given in (Ristad et al., 1998) the substring translations are learnt. They integrate the word-based unigram model based on (Knight and Graehl, 1998; Al-Onaizan and Knight, 2002) with the above model for improving the quality of transliterations.

Malik (2006) tries to solve a special case of transliteration for Punjabi in which they convert from Shahmukhi (Arabic script) to Gurumukhi using a set of transliteration rules. Abdul Jaleel (2003) show that, in the domain of information retrieval, the cross language retrieval performance was reduced by 50% when the name entities were not transliterated.

2 Problem Modeling

Assume that given a word, represented as a sequence of letters of the source language **s** = s_1^J = $s_1...s_j...s_J$, needs to be transcribed as a sequence of letters in the target language, represented as **t** = t_1^I = $t_1...t_i...t_I$. The problem of finding the best target language letter sequence among the transliterated candidates can be represented as:

Proceedings of the 2009 Named Entities Workshop, ACL-IJCNLP 2009, pages 124–127,
Suntec, Singapore, 7 August 2009. ©2009 ACL and AFNLP

$$\mathbf{t}_{best} = \arg\max_{\mathbf{t}} \{\Pr(\mathbf{t} \mid \mathbf{s})\} \quad (1)$$

We model the transliteration problem based on the noisy channel model. Reformulating the above equation using Bayes Rule:

$$\mathbf{t}_{best} = \arg\max_{\mathbf{t}} p(\mathbf{s} \mid \mathbf{t})\, p(\mathbf{s}) \quad (2)$$

This formulation allows for a target language letters' n-gram model $p(\mathbf{t})$ and a transcription model $p(\mathbf{s} \mid \mathbf{t})$. Given a sequence of letters **s**, the argmax function is a search function to output the best target letter sequence.

From the above equation, the best target sequence is obtained based on the product of the probabilities of transcription model and the probabilities of a language model and their respective weights. The method for obtaining the transcription probabilities is described briefly in the next section. Determining the best weights is necessary for obtaining the right target language sequence. The estimation of the models' weights can be done in the following manner.

The posterior probability $\Pr(\mathbf{t} \mid \mathbf{s})$ can also be directly modeled using a log-linear model. In this model, we have a set of M feature functions $h_m(\mathbf{t}, \mathbf{s}), m = 1...M$. For each feature function there exists a weight or model parameter $\lambda_m, m = 1...M$. Thus the posterior probability becomes:

$$\Pr(\mathbf{t} \mid \mathbf{s}) = p_{\lambda_1^M}(\mathbf{t} \mid \mathbf{s}) \quad (3)$$

$$= \frac{\exp\left[\Sigma_{m=1}^{M} \lambda_m h_m(\mathbf{t}, \mathbf{s})\right]}{\sum_{t_1^{\prime I}} \exp\left[\Sigma_{m=1}^{M} \lambda_m h_m(t_1^{\prime I}, \mathbf{s})\right]} \quad (4)$$

with the denominator, a normalization factor that can be ignored in the maximization process.

The above modeling entails finding the suitable model parameters or weights which reflect the properties of our task. We adopt the criterion followed in (Och, 2003) for optimising the parameters of the model. The details of the solution and proof for the convergence are given in Och (2003). The models' weights, used for the transliteration task, are obtained from this training.

All the above tools are available as a part of publicly available MOSES (Koehn et al., 2007) tool kit. Hence we used the tool kit for our experiments.

3 Tuning the parameters

The source language to target language letters are aligned using GIZA++ (Och and Ney, 2003). Every letter is treated as a single word for the GIZA++ input. The alignments are then used to learn the phrase transliteration probabilities which are estimated using the scoring function given in (Koehn et al., 2003).

The parameters which have a major influence on the performance of a phrase-based SMT model are the alignment heuristics, the maximum phrase length (MPR) and the order of the language model (Koehn et al., 2003). In the context of transliteration, *phrase* means a sequence of letters(of source and target language) mapped to each other with some probability (i.e., the *hypothesis*) and stored in a phrase table. The *maximum phrase length* corresponds to the maximum number of letters that a hypothesis can contain. Higher phrase length corresponds a larger phrase table during decoding.

We have conducted experiments to see which combination gives the best output. We initially trained the model with various parameters on the training data and tested for various values of the above parameters. We varied the maximum phrase length from 2 to 7. The language model was trained using SRILM toolkit (Stolcke, 2002). We varied the order of language model from 2 to 8. We also traversed the alignment heuristics spectrum, from the parsimonious *intersect* at one end of the spectrum through *grow*, *grow-diag*, *grow-diag-final*, *grow-diag-final-and* and *srctotrg* to the most lenient *union* at the other end.

We observed that the best results were obtained when the language model was trained on 7-gram and the alignment heuristic was *grow-diag-final*. No significant improvement was observed in the results when the value of MPR was greater than 7. We have done post-processing and taken care such that the alignments are always monotonic and no letter was left unlinked.

4 Data Sets

We have used the data sets provided by organisers of the NEWS 2009 Machine Transliteration Shared Task (Kumaran and Kellner, 2007). Prior to the release of the test data only the training data and development data was available. The training data and development data consisted of a parallel corpus having entries in both English and Hindi.

The training data and development data had 9975 entries and 974 entries respectively. We used the training data given as a part of the shared task for generating the phrase table and the language model. For tuning the parameters mentioned in the previous section, we used the development data.

From the training and development data we have observed that the words can be roughly divided into following categories, Persian, European (primarily English), Indian, Arabic words, based on their origin. The test data consisted of 1000 entries. We proceeded to experiment with the test set once the set was released.

5 Experiments and Results

The parameters described in Section 3 were the initial settings of the system. The system was tuned on the development set, as described in Section 2, for obtaining the appropriate model weights. The system tuned on the development data was used to test it against the test data set. We have obtained the following model weights. The other features available in the translation system such as *word penalty, phrase penalty* donot account in the transliteration task and hence were not included.

```
language model = 0.099
translation model = 0.122
```

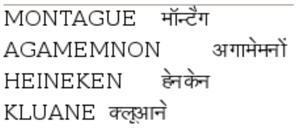
Prior to the release of the test data, we tested the system without tuning on development data. The default model weights were used to test our system on the development data. In the next step the model weights were obtained by tuning the system. Although the system allows for a distortion model, allowing for phrase movements, we did not use the distortion model as distortion is meaningless in the domain of transliteration. The following measures such as Word Accuracy (ACC), Mean F-Score, Mean Reciprocal Rank (MRR), MAP_{ref}, MAP_{10}, MAP_{sys} were used to evaluate our system performance. A detailed description of each measure is available in (Li et al., 2009).

Measure	Result
ACC	0.463
Mean F-Score	0.876
MRR	0.573
MAP_{ref}	0.454
MAP_{10}	0.201
MAP_{sys}	0.201

Table 1: Evaluation of Various Measures on Test Data

6 Conclusion

In this paper we show that we can use the popular phrase based SMT systems successfully for the task of transliteration. The publicly available tool GIZA++ was used to align the letters. Then the phrases were extracted and counted and stored in phrase tables. The weights were estimated using minimum error rate training as described earlier using development data. Then beam-search based decoder was used to transliterate the English words into Hindi. After the release of the reference corpora we examined the error results and observed that majority of the errors resulted in the case of the foreign origin words. We provide some examples of the foreign origin words which were transliterated erroneously.



MONTAGUE	मॉन्टैग
AGAMEMNON	अगामेम्नों
HEINEKEN	हेनकेन
KLUANE	क्लूआने

Figure 1: Error Transliterations of Some Foreign Origin Words

References

N. AbdulJaleel and L.S. Larkey. 2003. Statistical transliteration for english-arabic cross language information retrieval.

Y. Al-Onaizan and K. Knight. 2002. Machine transliteration of names in Arabic text. In *Proceedings of the ACL-02 workshop on Computational approaches to semitic languages*, pages 1–13. Association for Computational Linguistics Morristown, NJ, USA.

N. Aswani and R. Gaizauskas. 2005. A hybrid approach to align sentences and words in English-Hindi parallel corpora. *Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond*, page 57.

K. Knight and J. Graehl. 1998. Machine transliteration. *Computational Linguistics*, 24(4):599–612.

P. Koehn, F.J. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the NAACL:HLT-Volume 1*, pages 48–54. ACL Morristown, NJ, USA.

P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, et al. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *ACL*, volume 45, page 2.

A. Kumaran and T. Kellner. 2007. A generic framework for machine transliteration. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 721–722. ACM New York, NY, USA.

H. Li, A. Kumaran, M. Zhang, and V. Pervouchine. 2009. Whitepaper of NEWS 2009 Machine Transliteration Shared Task. In *Proceedings of ACL-IJCNLP 2009 Named Entities Workshop (NEWS 2009)*. ACL, Singapore, 2009.

M.G.A. Malik. 2006. Punjabi machine transliteration. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 1137–1144. Association for Computational Linguistics Morristown, NJ, USA.

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

F.J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on ACL-Volume 1*, pages 160–167. ACL, Morristown, NJ, USA.

T. Rama, A.K. Singh, and S. Kolachina. 2009. Modeling letter to phoneme conversion as a phrase based statistical machine translation problem with minimum error rate training. In *The NAACL Student Research Workshop*, Boulder, Colorado.

ES Ristad, PN Yianilos, M.T. Inc, and NJ Princeton. 1998. Learning string-edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5):522–532.

T. Sherif and G. Kondrak. 2007. Substring-based transliteration. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 45, page 944.

A. Stolcke. 2002. Srilm – an extensible language modeling toolkit.

H. Surana and A.K. Singh. 2008. A more discerning and adaptable multilingual transliteration mechanism for indian languages. In *Proceedings of the Third International Joint Conference on Natural Language Processing*.