Transliteration as Machine Translation*

Devanshu Jain
University of Pennsylvania
devjain@cis.upenn.edu

Chris Callison-Burch
University of Pennsylvania
ccb@cis.upenn.edu

Abstract

1 Introduction

Machine Transliteration is the process phonetic translation of a word across different scripts. When a word is translated from its native script to foreign script, then it is called *forward transliteration*. On the other hand, when it is translated from a foreign script back to its native script, then it is called *backward transliteration*.

With growing multilingualism, machine transliteration plays an important role in many downstream applications. One such example is Machine Translation Systems. Almost always, named entities (such as names, addresses, technical terms, festivals, etc.) are transliterated while generating annotated parallel corpora. Sometimes, there are no words in the target language corresponding to a word in the source language. Here, machine transliteration proves to be an important module to improve the performance of machine translation.

Another application is Cross Lingual Information Retrieval (CLIR) Systems. Many of the search engines do not consider transliterated content while responding to a query. It can be observed that the same query (for example, song lyrics) returns significantly different results when submitted to a search engine in a transliterated form.

An interesting human behaviour is also observed on websites such as Facebook, Twitter, etc. which are sources for a lot of user generated content. A lot of these posts/tweets are written in user's native language transliterated to Roman Script. Machine Transliteration would be a useful component for text-based applications such as Question Answering, Sentiment Analysis, etc.

Machine Transliteration presents its own set of challenges. One of them is of multiple target candidates. For example hamare can be written as *humare* or *hamare* or *hamaare* and so on. This leads to the system learning various rules, all of which are correct but may introduce test error. Thus, we need an appropriate performance measure that takes into account the partial correctness of transliteration process. Moreover, there is a lack of high quality training data which hurdles the creation of accurate models.

This study is based on Irvine et.al.'s paper titled ”Transliterating From All Languages” (2010). I attempt to recreate some of their experiments with focus on low resource languages to build an end to end machine transliteration system that takes a sequence, its source script and the target script and does the phonetic conversion from the source script to the target script.

The rest of the paper is organised as follows. Section 2 describes some of the related work done in this field. Section 3 describes the tools and data used for this study. Section 4 describes the Baseline System. Section 5 describes the experiments. Section 6 describes the results. Section 7 describes the Future Work.

2 Related Work

3 Tools and Data

3.1 Joshua

I will use Joshua decoder for the experiments.

3.2 Data

I will be using named entities extracted from wikipedia articles and their transliterated content as the training data for this study.

4 Baseline System

I will be using Theresa Breiner's system as a baseline for this study.

5 Experiments

For this study, I am considering the transliteration of languages from Non-Roman Script to Roman Script. I am trying the following experiments for this study. For these experiments, I plan to use Berkeley Aligner and 3-order Berkeley Language Model Generator, both of which are implemented in the Joshua decoder.

5.1 Experiment 1

The objective of this experiment is to confirm the hypothesis that with increase in training data, the system performance increases. I will divide the data into 3 parts: training, tuning and testing and will run the system using 60%, 70%, 80% and 90% of the data.

5.2 Experiment 2

The objective of this experiment is to observe how the system performance changes when we accumulate training data with languages belonging to same script family For example, Hindi, Sanskrit, Marathi, etc. belong to same family. The hypothesis here is that grouping languages with similar script will significantly increase the training data and if the above hypothesis (Experiment 1) is true, then this should increase the system performance.

5.3 Experiment 3

The objective of this experiment is to observe whether the presence of additional information such as the source of origin for a language can affect the system performance. I will cluster the training data based on the source of origin of the word. Then, I will train different models for each cluster.

5.4 Experiment 4

The objective of this experiment is to observe the changes in system performance when we try compositional transliteration. Given the source language script X and target script Z, I want to observe the system performance when an intermediate language is chosen in the process. That is, X $\rightarrow$ Y $\rightarrow$ Z. The hypothesis here is that the error will accumulate over the two conversions and the system performance should decrease drastically.

6 Results

6.1 Performance Measures

I will use the following evaluation metrics to calculate the system performance:

1. **Word Accuracy:** It measures the average correctness of the top transliterated candidate (in the list generated by the transliteration system) with respect to the reference transliteration.

2. **Mean F-Score:** The F-score for ith word is calculated by using Precision and Recall using Edit Distance between the top transliterated candidate (in the list generated by the transliteration system) and the reference.

3. **Mean Reciprocal Rank:** It calculates the average of the minimum reciprocal rank of a correct transliteration (in the list generated by the transliteration system).

4. **Mean Average Precision:** It calculates the mean of average precision over all the sources words.

6.2 Results

7 Future Work