**ISM@FIRE-2014: Shared task on Transliterated Search**

Dinesh Kumar Prabhakar  
Indian School of Mines  
Dhanbad, Jharkhand  
India 826004  
dinesh.nitr@gmail.com

Sukomal Pal  
Indian School of Mines  
Dhanbad, Jharkhand  
India 826004  
sukomalpal@gmail.com

**ABSTRACT**

This paper describes approaches that we have used for official submission of FIRE-2014, for the Shared Task on Transliterated Search. Approaches solve the problem of word level labeling. The labeling classifies class of term with it belongs. MaxEnt a supervised classifier is used for classification and labeling of a word. This subtask completion is followed by back-transliteration of \( H \) (Hindi) labeled word. For the back-transliteration we have used generative and GEN-EXT(combination of generative and extraction) approaches. From the evaluation it is been seen that Runs has performed best in some metrics like \( LA, LP, LF, EF \) etc. In some metrics runs performed 2nd and 3rd best, in some other performance was poor as well.

1. INTRODUCTION

The user’s count on Social sites are increasingly becoming higher. They write messages (specially blog and post) on site (such as Twitter, Facebook et.), in their own language preferably using Roman scripts. These post might comprise terms of Non-English (or terms of user’s native) language, a simple English word, a Named Entity (NE) or an acronym. This transformation of a word of a language into a non-native or foreign script is called transliteration [5]. However more than one transformations are possible for a non-English word in Roman representation. These multiple transformations differ in spelling. Spelling variation is one of the serious issue in back-transliteration.

Transliterations, using Roman script, are used more frequently on the Web not only to create documents as well as for generating user’s queries to search required document. Last year transliterated search in Indian languages was included as a pilot task. The task, Shared Task on Transliterated Search had two subtasks:first, Query Word Labeling and second Multi-script Ad hoc retrieval for Hindi Song Lyrics. Like previous year this this year also task is divided into two same subtasks. Moreover, some more issues were considered to take into account such as Named entity tagging (P, L and O based on types).

As the task is divided into two part subtask-1 and subtask-2, where we participated in former one only. Again this subtask is divided into two phase: first phase classification of query words and second transliteration of Hindi language’s words.

Next in the Section 2 we discussed task description. Section 3 describes our approaches for labeling and transliteration. In Section 4 we have discussed results and analyze errors. Section 5 concludes the paper with directions of future work.

2. TASK DESCRIPTION

**Query Word Labeling**

Input:- Let \( Q \) be the query set containing \( n \) query word \( w_i (1 \leq i \leq n) \) written in Roman script. The word \( w_i \in Q \ (w_1, w_2, \ldots, w_n) \) could be a standard English word or transliterated form of a Hindi word, a Named Entity, an acronym or others(such as gr8, lol etc). Our approach the system should give labeled words followed by transliteration of Hindi words.

Output:- Result \( w_{ij} \) is produce in such a way that Hindi word \( w \) was be tagged with label \( H \) and aligned with its transliteration. Other than Hindi word all are only tagged with their corresponding label. The Named entity was tagged with \( P, L \) and \( O \) which refers the name of a person, location and organization respectively and an acronym was tagged with label \( A \). There are some word they do not come under these categories are considered as others and tagged with label \( O \).

3. APPROACHES

Our attempt to solve the problem Query word Labeling is phased in the order as Word Labeling followed by Transliteration of \( H \) (Hindi) labeled words.

3.1 Word Labeling

To accomplish this subtask, word level classification is needed. The word can be classified manually or using any classifier. But manual classification is not feasible for the large set of data. For the classification of terms at word level we have used Stanford classifier (MaxEnt - a supervised classifier) [4].

The classification is completed in two phase: first, train the classifier and then classify word based on extracted features.

3.1.1 Training

The FIRE-2013 & 2014 training dataset we have used to create two training files where second file is subset of first file. These file containing list of words \( w \) from training dataset aligned with their class tags (such as P, L, O etc). First file contains 23956 labeled terms from FIRE-2014 training data and second file contains 262665 words comprises additional 30823 [2] Hindi words and 207824 English words from FIRE-2013 dataset. These files stored data in column format (Stanford classifier’s required format). Using both the

---

Dinesh Kumar Prabhakar  
Indian School of Mines  
Dhanbad, Jharkhand  
India 826004  
dinesh.nitr@gmail.com

Sukomal Pal  
Indian School of Mines  
Dhanbad, Jharkhand  
India 826004  
sukomalpal@gmail.com
training files classifier was train and the trained classifiers were named as ISM-1 and ISM-2 respectively.

Features.
Different features were set in property file for classifiers are listed in the box.

```
# Features
useClassFeature=true
1. useNGrams=true
1. usePrefixSuffixNGrams=true
1. maxNGramLeng=4
1. minNGramLeng=1
1. binnedLengths=10,20,30

# Optimization
intern=true
sigma=3
useQN=true
QNsize=15
tolerance=1e-4
```

Same property file is used for both the training file. Total six tags were identified during the training. Those tags are $H, E, P, L, O$ and $A$.

3.1.2 Classification

The given test data file was parsed on trained classifiers for classification. Words of test data classified in six classes such as Hindi word, English word, proper name (name of the person, location or organization), acronym and others and words were labeled with different class tags $H, E, P, L, O$ and $A$ for different classes respectively.

3.2 Transliteration

Transliteration can be obtained by extraction, by generation or combining both [3]. We have used later two techniques in our approaches for transliteration. The generative approach represented in algorithm 2 and combine (Extraction + Generation (GEN-EXT)) approach represented in algorithm 1.

General Terms.
The labeled query list $Q_i$ contain processed query words $w_i$ labeled with $lbl$. The $EH_P$ is a dictionary contains Hindi words written in Roman script along with its transliteration in Devanagari script. Procedure $Split(w_i)$ splits $w_i$. We have used $T_G(w)$ - a Hindi-English Indic character mapping system (RomaDeva) for automatic transliteration generation [7].

Algorithm 1

The procedure select a word $w_i$ form $Q_i$ and the label is checked using a simple string matching. If the word is not a Hindi term (i.e. $w$ is not labeled with $H$), then insert '/' between $w_i$ and $lbl$ which produce the result as $w_i$. Otherwise $w$ is a Hindi term, and it will be searched in the transliteration pair dictionary $EH_P$. During the search process, if $w \in EH_P$ then the corresponding transliteration $T$ will be extracted and $w_i$ produced as result. Otherwise, $w \notin EH_P$ i.e. out-of-EH_P dictionary, then the transliteration $T$ will be generated by using the procedure $T_G(w)$. After the generating $T$, the character '/' was inserted between $w$ and $lbl$, followed by '=' $T$. Finally, $w_i$ is produced as the result.

Algorithm 2

Extracting the transliteration of a word is not feasible for resource scar languages. So, the transliteration generation remains the only fair option[1]. The transliteration can be generated based on phonemes, grapheme, syllables, combined or hybrid techniques. We have used a rule (grapheme) based system for automatic transliteration generation. The system work based on Indic character mappings [7].

In this approach, procedure $Split(w_i)$ split word $w_i$ and label $lbl$ is checked using a simple string matching. If $w$ is non-Hindi term, then insert the character '/' between $w$ and $lbl$, produced result as $w_i$. Otherwise, $w$ is a Hindi term and the $T$ is generated using procedure $T_G(w)$ [7]. After the generating $T$, the character '/' was inserted between $w$ and $lbl$, followed by '=' $T$. Finally, $w_i$ is produced as the result.

The steps are given in Algorithm 2. Since this approach is rule based, so it may give inexact transliteration.

```
Algorithm 1 GEN-EXT Transliteration
1: procedure GET-LAB_TRANSLIT($w_i$) \( \triangleright (\forall w_i \in Q_i) \)
2: \( \{w, lbl\} \leftarrow Split(w_i) \)
3: if \( (lbl != H) \) then
4: \( w_i \leftarrow \{w, /, lbl, =, T\} \)
5: else \( w_i \leftarrow \{w, /, lbl\} \)
6: return \( (w_i) \)
7: \( T \leftarrow T_G(w) \) \( \triangleright \) Call Generative Function $T_G(w)$
8: \( w_i \leftarrow \{w, /, lbl, =, T\} \)
9: return \( (w_i) \)
```

3.3 Post-processing

Since ‘\’ is not allowed in programming so we used ‘/’. To bring the result in proper format we replaced later character by former character in our outputs.

4. RESULT AND DISCUSSION

We have submitted total three Runs namely Run1, Run2 and Run3. In this section first we discussed results and then analysed relate issues. The Runs were evaluated using following performance metrics [6]:

LP, LR and LF are the Token level precision, recall and F-measure for the Indian language in the language pair. EP, ER and EP are the Token level precision, recall and F-measure for English tokens. TP, TR and TF are the Token level transliteration precision, recall, and F-measure. EQMF (Exact query match fraction) as defined in [6], EQMF (without transliteration) as defined in [6], but only considering
language identification. ETPM (Exact transliterated pair match) as defined in [6].

\[ \text{LA} = \frac{\text{Correct label pair}}{\text{Correct label pair} + \text{Incorrect label pair}} \]

Relative scores of various metrics for our runs are included in Table 1.

4.1 Results

In total we submitted 3 runs using two different approaches discussed in previous section. We compare the evaluation results w.r.t. MAX and MEDIAN of all runs (including run of other teams) submitted at FIRE-2014.

4.1.1 Run1:

For the first run Label identification was done using ISM-1 followed by back-transliteration using combined approach of algorithm 1. These are the position in different metrics: LP-1st, LF-5th, EP-9th, ER-2nd, EF-6th, LA-4th

4.1.2 Run2:

For our second run we used ISM-2 for label identification and algorithm 2 for automatic transliteration. Run2 performed best in LF, EF, EQMF-All-NT (No transliteration), EQMF-wt-NE (No transliteration), EQMF-wt-Mix (No transliteration), EQMF-wt-Mix and NE-NT (No transliteration). Other metrics position are as LP-2nd, ER-2nd, EP-4.

4.1.3 Run3:

In this run ISM-2 is used to identify the label of Tokens and automatic transliteration approach in algorithm 1 is used. Run2 performed best in LF, EF, EQMF-All-NT (No transliteration), EQMF-wt-NE (No transliteration), EQMF-wt-Mix (No transliteration), EQMF-wt-Mix and NE-NT (No transliteration). Other metrics position are as LP-2nd, ER-2nd, EP-4. On a different note, there were some errors in the transliteration pairs of training data.

4.2 Comparison

We have compare our approaches score with Max. of all. We could not perform well in NEs related metrics, since our system fail to recognize NEs correctly. Possibly use of some NER tool can solve this problem. Since our system is heavily biased by the training data, we got inexact transliteration for some terms.

5. CONCLUSIONS

Our work comprises two subtasks labeling and transliteration. We used classifier for word labeling. Label accuracy relatively better than last year of our submission. We identified some terms were incorrectly labeled. Perhaps this happened due to two reasons, first less corpus size and second we did not use use any NER system. In the transliteration approaches we observed that results are inclined to extraction approaches which requires corpus of large size. By applying some learning based approaches for transliteration generation we are planning to improve result of our systems.

6. REFERENCES


Table 1: Performance of different Runs

<table>
<thead>
<tr>
<th>Run_ID</th>
<th>LP</th>
<th>LR</th>
<th>LF</th>
<th>EP</th>
<th>ER</th>
<th>EF</th>
<th>TP</th>
<th>TR</th>
<th>TF</th>
<th>LA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run1</td>
<td>0.942</td>
<td>0.852</td>
<td>0.895</td>
<td>0.823</td>
<td>0.942</td>
<td>0.878</td>
<td>0.131</td>
<td>0.636</td>
<td>0.217</td>
<td>0.872</td>
</tr>
<tr>
<td>Run2</td>
<td>0.93</td>
<td>0.892</td>
<td>0.911</td>
<td>0.871</td>
<td>0.932</td>
<td>0.901</td>
<td>0.07</td>
<td>0.363</td>
<td>0.118</td>
<td><strong>0.886</strong></td>
</tr>
<tr>
<td>Run3</td>
<td>0.93</td>
<td>0.892</td>
<td>0.911</td>
<td>0.871</td>
<td>0.932</td>
<td>0.901</td>
<td>0.122</td>
<td>0.628</td>
<td>0.204</td>
<td><strong>0.886</strong></td>
</tr>
<tr>
<td>Median</td>
<td>0.853</td>
<td>0.861</td>
<td>0.81</td>
<td>0.767</td>
<td>0.881</td>
<td>0.797</td>
<td>0.109</td>
<td>0.6345</td>
<td>0.1835</td>
<td>0.792</td>
</tr>
<tr>
<td>MAX</td>
<td>0.942</td>
<td>0.917</td>
<td>0.911</td>
<td>0.895</td>
<td>0.987</td>
<td>0.901</td>
<td>0.2</td>
<td>0.76</td>
<td>0.304</td>
<td><strong>0.886</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Run_ID</th>
<th>EQMF-All-NT</th>
<th>EQMF-wt-NE-NT</th>
<th>EQMF-wt-Mix-NT</th>
<th>EQMF-wt-Mix-n-NE-NT</th>
<th>EQMF-All</th>
<th>EQMF-wt-NE</th>
<th>EQMF-wt-Mix</th>
<th>EQMF-wt-Mix-n-NE</th>
<th>ETPM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run1</td>
<td>0.269</td>
<td>0.409</td>
<td>0.269</td>
<td>0.409</td>
<td>0.001</td>
<td>0.001</td>
<td>0.001</td>
<td>0.001</td>
<td>1616/2203</td>
</tr>
<tr>
<td>Run2</td>
<td>0.276</td>
<td>0.427</td>
<td>0.276</td>
<td>0.427</td>
<td>0.001</td>
<td>0.002</td>
<td>0.001</td>
<td>0.002</td>
<td>924/2251</td>
</tr>
<tr>
<td>Run3</td>
<td>0.276</td>
<td>0.427</td>
<td>0.276</td>
<td>0.427</td>
<td>0</td>
<td>0.002</td>
<td>0</td>
<td>0.002</td>
<td>1596/2251</td>
</tr>
<tr>
<td>Median</td>
<td>0.194</td>
<td>0.285</td>
<td>0.194</td>
<td>0.285</td>
<td>0.001</td>
<td>0.003</td>
<td>0</td>
<td>0.003</td>
<td>NA</td>
</tr>
<tr>
<td>MAX</td>
<td>0.276</td>
<td>0.427</td>
<td>0.276</td>
<td>0.427</td>
<td>0.005</td>
<td>0.01</td>
<td>0.005</td>
<td>0.01</td>
<td>NA</td>
</tr>
</tbody>
</table>

Figure 1: Comparison with Median and Max. Score