Question classification for Code-Mixed text

ARCHIT RATHORE
PRABUDDHA CHAKRABORTY
Problem Statement

- Given:
  - A code-mixed question
- To predict:
  - Expected type of the answer

Mein kaise ek web based business start kar sakta hun?

DESC: manner
Answer-type hierarchy [1]
Challenges

- No dataset available on the public domain
- Dearth of tools that handles code-mixed text - no POS-taggers, chunkers, dependency parsers, language identification tools etc

Proposed Solution

- Create a new dataset of code-mixed questions
- Make dataset in a format compatible with the existing tools
Dataset Format

- QueNo#1 CoarseLabel:FineLabel <QueString in English>
- QueNo#2 CoarseLabel:FineLabel <QueString in Hindi>
- QueNo#3 CoarseLabel:FineLabel <QueString in Codemixed_scriptPreserved>
- QueNo#4 CoarseLabel:FineLabel <QueString in Codemixed_romanized>

185#1 DESC:reason Why do horseshoes bring luck ?
185#2 DESC:reason घोड़े की नाल भाष्य क्यों लाती है?
185#3 DESC:reason घोड़े की नाल luck क्यों लाती है?
185#4 DESC:reason Ghode ki naal luck kyon laati hai?
Dataset Creation

- Get English annotated questions from dataset created by Li and Roth [5]
- Use python’s Goslate API to convert these questions to Hindi
- Manually do the following:
  - Fix translation errors in Goslate output
  - Repose the questions in a code mixed sense
- Transliterate the script-preserved code-mixed questions to roman script
  - Done using “Sanscript” API: has problems with schwa deletion [8]
  - Manually corrected as of now
Preliminary Results

- Data of 200 code-mixed sentences
- Used:
  - Linear kernel SVM (one-vs-rest approach)
  - Logistic regression classifier
- Performed 100 iterations of training and testing with both classifiers
- Split data randomly with 0.8 : 0.2 ratio
Confusion matrix for: LogisticRegression

<table>
<thead>
<tr>
<th></th>
<th>precision</th>
<th>recall</th>
<th>f1-score</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.64</td>
<td>0.65</td>
<td>0.64</td>
<td>727</td>
</tr>
<tr>
<td>1</td>
<td>0.40</td>
<td>0.41</td>
<td>0.40</td>
<td>804</td>
</tr>
<tr>
<td>2</td>
<td>0.74</td>
<td>0.65</td>
<td>0.69</td>
<td>845</td>
</tr>
<tr>
<td>3</td>
<td>0.59</td>
<td>0.65</td>
<td>0.62</td>
<td>581</td>
</tr>
<tr>
<td>4</td>
<td>0.60</td>
<td>0.61</td>
<td>0.60</td>
<td>643</td>
</tr>
<tr>
<td>avg / total</td>
<td>0.59</td>
<td>0.59</td>
<td>0.59</td>
<td>3600</td>
</tr>
</tbody>
</table>

--------------LogisticRegression--------------

[[470 91 38 71 57]
  [ 86 331 185 119 163]
  [ 65 137 547 67 29]
  [ 44 117 29 376 15]
  [ 66 158 25 2 392]]

CORRECT PREDICTIONS:

2116

TOTAL PREDICTIONS:

3600

ACCURACY:

0.587777777778
Confusion matrix for: OnevsRest SVC

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>540</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>480</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>480</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>540</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>540</td>
</tr>
</tbody>
</table>

Precision  | Recall  | F1-score  | Support |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.64</td>
<td>0.63</td>
<td>0.64</td>
</tr>
<tr>
<td>1</td>
<td>0.36</td>
<td>0.41</td>
<td>0.38</td>
</tr>
<tr>
<td>2</td>
<td>0.72</td>
<td>0.63</td>
<td>0.67</td>
</tr>
<tr>
<td>3</td>
<td>0.61</td>
<td>0.65</td>
<td>0.63</td>
</tr>
<tr>
<td>4</td>
<td>0.59</td>
<td>0.59</td>
<td>0.59</td>
</tr>
</tbody>
</table>

Average / Total: 0.59, 0.58, 0.58, 3600

------------------------ LinearSVC ------------------------

- [[461 102 40 73 52]]
- [[92 309 106 112 142]]
- [[60 163 555 61 36]]
- [[43 114 35 395 21]]
- [[62 162 34 2 368]]

Correct Predictions: 2088
Total Predictions: 3600
Accuracy: 0.58
Further Work

- Use a tree kernel for SVM as proposed by Collin and Duffy [3]
- Introduce adjacency features as proposed by Raghavi et al [2]
- Use DCNN for sentence modelling – Kalbrechhner et al [6]
References


7. Source - https://github.com/sanskrit/sanscript