Cross Lingual Information Retrieval for English and Hindi

Submitted by:

Sushant Narsale
Unmil Tambe
1. Motivation

Web search provides easiest way to access information on the internet. Although English is still the most dominant language on web, information in languages other than English has been increasing. The primary motivation behind choosing and implementing this project was to understand the intricacies involved in accessing information in language other than English. Since we were comfortable in reading and understanding Hindi, which is the most spoken language in India, we chose to build a CLIR for English to Hindi retrieval.

2. Introduction

We followed a dictionary based approach for Hindi – English translation. Dictionary based approaches have an inherent disadvantage of not accounting for out-of-vocabulary (OOV) words. To overcome this we implemented a generative statistical transliteration model using bi-gram probabilities. We also implemented a Hindi language stemmer to remove inflectional suffixes. The rest of the report is outlined as follows: We start with describing the main components of the system viz. Translation, Transliteration and Stemming modules, we then describe the retrieval model used, web interface designed and the complete flow diagram. In section 9 we give a brief insight into difficulties we faced in designing with a CLIR system and then conclude with results and our take on the entire project.
3. Cross-Language IR

Cross Lingual Information Retrieval refers to retrieval of and ranking of documents in one language in response to a query issued in a different language[1]. In this report we represent a system to retrieve Hindi documents given a English query. The detailed explanation of the translation, transliteration, Hindi-stemming and retrieval model used are described in following sections.

4. Translation

Translation refers to converting a word in one language to a word with similar meaning in the other language. For our project we had to translate queries given in English in Hindi. We used a dictionary based approach for translation, wherein for every English token in the user query a lookup was performed in a English-Hindi dictionary and the Hindi meanings for the word were added to the translated query. In case of English words with multiple Hindi meanings, we added all the Hindi meanings to the translated query. The decision to add all the Hindi in translated languages was taken on observation which showed that multiple meanings for a word yield better results. And since we were only using a dictionary look-up and not a probabilistic model to translate, we had an intuition about generating better results will when all meanings are used. Also since we had a limited dictionary of about 15,000 English words, wherein a word had at most 2-3 Hindi translations we decided against using a probabilistic model for translation, since we would neither have got good probability estimates, nor adding a couple of words for every query term would result into a major computational overhead. The dictionary used was made by combining a few freely available online resources.

<table>
<thead>
<tr>
<th>English</th>
<th>Hindi</th>
</tr>
</thead>
<tbody>
<tr>
<td>petty</td>
<td>नगणय, तुचछ, नीच, निदर्शी और मलबी</td>
</tr>
<tr>
<td>conveyance</td>
<td>वाहन, समर्पणपत्र</td>
</tr>
</tbody>
</table>

Table 1. Translation Example using our Model.
5. Transliteration

In dictionary-based CLIR, out-of-vocabulary (OOV) words are always a problem and result into poor performance [1]. It has been empirically observed that in Information Retrieval systems 50% of out-of-vocabulary words are names [David and Ogden 1998].

For transliteration, we built a generative statistical model, similar to one used for English to Arabic transliteration by Nasreen AbdulJaleel and Leah Larkey [2]. We used a bi-gram model for this purpose, the decision to use bi-gram models was based on our observation that most of the Hindi characters can be represented by two English characters. We also wanted to test a tri-gram model for transliteration, but given the size of training data (which consists of 400 Hindi names and their transliteration in English) we wouldn’t have got good results. Training the Bi-gram model requires alignment of characters in the two languages. We used GIZA++ for alignment, GIZA++ is designed for word alignment in parallel corpus in which sentences in two languages are aligned. We used a word-aligned corpus instead of a sentence aligned corpus by generating two parallel text files which one word in each line. The word was split into individual characters, so that GIZA++ could align individual characters.

| a → आ |
| b → भ |
| h → NULL |
| a → आ |

Table 2: Example of alignment given by GIZA++.

Transliteration Model:

We used a probabilistic bi-gram model for transliteration as mentioned above. Bi-grams counts were calculated using the aligned data form GIZA++ (table 2) where English characters $e_1, e_2$ were mapped into Hindi character $h_i$. For each bi-gram $e_1e_2$ the probabilities were calculated using MLE, by counting the number of times $e_1e_2$ was mapped to $h_i$ and dividing this by total number of occurrences of $e_1e_2$. $e_1e_2$ should be interpreted as probability of $e_2$ in context of $e_1$.

$$P(e_2 \rightarrow h_i | e_1) = \frac{\text{counts}(e_1, e_2 \rightarrow h_i)}{\text{counts}(e_1, e_2)} \quad (1)$$
We did not eliminate mappings with very low weights a suggested in [2] since we had a small training sets. In case where bi-gram counts were not present in training data we decided to back-off to unigram probabilities, and since all uni-grams were seen atleast once in taining data we did not have to back-off further.

<table>
<thead>
<tr>
<th>Kapil</th>
<th>क प ि ल</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sushant</td>
<td>सू ष न त</td>
</tr>
<tr>
<td>John</td>
<td>ज ओ ह ण</td>
</tr>
<tr>
<td>Sudarshan</td>
<td>सू द र श न</td>
</tr>
<tr>
<td>David</td>
<td>द व ि द</td>
</tr>
</tbody>
</table>

Table 4:- Example of transliteration generated by our code

This transliteration model was used to generate Hindi translations associated with English words which couldn't be translated using dictionary look-ups. Since the training data was small we decided to use top all possible transliterations for a given English word.

6. Stemming

Stemmers are used to convert inflected words into their root or stem. Stem does not necessarily correspond to linguistic root of a word. Stemming improve performance by reducing morphological variants into same words[3]. We build a Hindi stemmer based on [3]. Through our experiments we could identify 20 common suffixes which should be striped, these suffixes were a subset of suffixes mentioned in[3]. Since we were not very sure about our experiments and importantly about our training data we then decided to use all the 51 suffixes mentioned in [3] for stemming. Based on the suffix list each word was stemmed by matching the longest suffix.
Table 3:- Inflectional Hindi Suffixes.

7. Retrieval Models

We used a search based on tf-idf. Each term in the English query was translated into Hindi. All documents vectors used tf-idf weights, and we used cosine similarity as a measure of similarity between documents and queries.

8. Web Interface

Web search provides easiest way to access information on the Internet. Although English is still the most dominant language on web, information in languages other than English has been increasing. The primary motivation behind choosing and implementing this project was to understand the intricacies involved in accessing information in language other than English. Since we were comfortable in reading and understanding Hindi, which is the most spoken language in India, we chose to build a CLIR for English to Hindi retrieval.
9. Screenshots

IndicSearch

Keyword

Search

Predefined Query

Search

Inspired By This Site

18 results in 0.40393 seconds:

भे मे एक तस्क का बाहर होते है
Similarity: 0.0747 - Similar documents

दिन भर की टीम ने फिर फिरबार दम
Similarity: 0.0342 - Similar documents

राजस्थान रायस की कार्य हर
Similarity: 0.0223 - Similar documents

पूर्व छिमंगे से हर मैं टीम की दौड़
Similarity: 0.0098 - Similar documents

भारतीय क्रिकेट
Similarity: 0.0089 - Similar documents

पत्थरमाला ने निकाला धार्मिक
Similarity: 0.0078 - Similar documents

भारत में प्रमुख ग्रंथ अध्ययन का दृष्टिकोण
Similarity: 0.0050 - Similar documents

एक्स्क्यूज़ क्रिकेट
Similarity: 0.0048 - Similar documents
9. Results

To evaluate our system we calculated precision recall over a set of 28 queries and 100 tagged documents. The results observed are as shown in the table:

<table>
<thead>
<tr>
<th>Stemming Method</th>
<th>Mean Average Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>English and Hindi Stemming</td>
<td>0.2739</td>
</tr>
<tr>
<td>Only Hindi Stemming</td>
<td>0.3276</td>
</tr>
<tr>
<td>Without any stemming</td>
<td>0.4856</td>
</tr>
</tbody>
</table>

Table 4:- Results

The results observed are quite similar to [1]. Although we assume that the small size of training and test data used gives skewed results.

10. Problems and Difficulties

The project was a success, but we faced lot of difficulties in dealing with language peculiarities.

1. Dealing with proprietary encodings was a big task, we than restricted our work to 'utf-8'. This made it difficult to gather resources.
2. Finding corpus, dictionaries, data for transliteration, changing the data to a form which can be used for retrieval and training purpose was very time consuming.
3. The learning curve was really steep, we had to read a lot of literature to understand how a CLIR works, a list of references in this document is an indication of reading and understanding task.
4. Learning to use GIZA++ and Moses was a task in itself.

11. Conclusion

The project was a good learning experience for us. We were able to do a good job in translation, transliteration and Hindi stemming. The generative transliteration model and Hindi stemmer we wrote was the most satisfying part of work. With a little more training data and a better test-bed we are sure that the project would have been more than successful.

We would like to thank Dr. David Yarowsky for his guidance throughout the semester. We would also like to thank Prof. Chris Callison Burch for his inputs towards the end of the project.
12. List of Files

- common_words: List of Stop words for English.
- suffix: List of Hindi suffixes used for stemming.
- corpus: Training corpus
- corpus-stemmed: Stemmed training corpus
- titles: Titles for all training files.
- evaluation.py: Python code to evaluate the system. It calculates mean precision over a set of 28 queries and 100 documents.
- search_output.py: Retrieve Documents similar to a query or other documents.
- pstemmer.py: A stemmer based on porter stemmer. This was an online resource available on Dr. Porter's web-page.
- hindi_stemmer: Python code to stem Hindi tokens.
- transliteration_prob.py: Python code which reads in GIZA++ output and calculates bi-gram and uni-gram probabilities and dumps the values in two files.
- transliterate.py: transliteration Code, it uses bi-gram and uni-gram probabilities generated by transliteration_prob.py.
- Translate.py: Dictionary based translation code.
- run_server.py: CGI code for Web interface engine.

To run the system,
Untar the folder and start the server:

```
$ tar -xzvf sushant.unmil.tar.gz
$ cd sushant.unmil
$ python run_server.py
```

Now open a web browser and in the address bar type:

```
http://localhost:8000
```
13. References

1. Hindi CLIR in Thirty Days
   - Leah Larkey, Margaret Connell, Nasreen Abduljaleel (Umass)
2. English to Arabic Transliteration for Informational Retrieval – A Statistical Approach
   - Nasreen AbdulJaleel and Leah Larkey
3. An Unsupervised Hindi stemmer with Heuristic Improvements.
   - Amaresh Kumar Pandey, Tanveer J Siddiqui
4. English- Hindi Translation in 21 Days
   – Ondrej Bojar, Pavel Stranak, Daniel Zeman
5. Multilingual Information Access: an Indian Language Perspective
   – Prasenjit Majumder, Mandar Mitra, Kalyan Dutta.
6. WebKhoj: Indian language IR from Multiple Character Encodings
   - Prasad Pingali, Jagadeesh Jagarlamudi, Vasudeva Varma
7. Hindi and Telugu to English Cross Language Information Retrieval at CLEF 2006
   - Prasad Pingali, Vasudeva Varma