Indexing and Searching Chinese, Japanese, and Korean text in Solr

Hector Correa
hector_correa@brown.edu
September 30, 2020
Disclaimer

I do not speak Chinese, Japanese, or Korean…

...this session is about how to index CJK text in Solr from the perspective of a software developer that knows a little bit of Solr but nothing about CJK languages.
Agenda

- The problem we were facing
- Indexing CJK text in Solr
- Searching for CJK text
- Questions and Answers
The problem: precision
### Precision

<table>
<thead>
<tr>
<th>Search</th>
<th>Expected Matches</th>
<th>Total Returned</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>莫言 (Mo Yan)</td>
<td>72</td>
<td>300</td>
<td>0.240</td>
</tr>
<tr>
<td>柳美里 (Yu Miri)</td>
<td>15</td>
<td>1000</td>
<td>0.015</td>
</tr>
<tr>
<td>ふくろうの本 (Fukuro no hon)</td>
<td>11</td>
<td>4800</td>
<td>0.002</td>
</tr>
</tbody>
</table>

- Precision = Number correct matches / Total results returned
- "When I issue a search, are the documents that come back the ones I was looking for?"
- Values close to zero are bad
The root of the problem

- Solr has many text fields
  - General: `text_general`
  - Language specific: `text_ar`, `text_cjk`, `text_en`, `text_es`, `text_fr`, ...

- `text_general` works OK-ish for several* languages

- But `text_general` field does not work for CJK languages

* some exceptions apply
text_general field with text in English
<table>
<thead>
<tr>
<th></th>
<th>For</th>
<th>Whom</th>
<th>The</th>
<th>Bell</th>
<th>Tolls</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST</td>
<td>For</td>
<td>Whom</td>
<td>The</td>
<td>Bell</td>
<td>Tolls</td>
</tr>
<tr>
<td>SF</td>
<td>Whom</td>
<td>Bell</td>
<td>Tolls</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LCF</td>
<td>whom</td>
<td>bell</td>
<td>tolls</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EPC</td>
<td>whom</td>
<td>bell</td>
<td>tolls</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SKMF</td>
<td>whom</td>
<td>bell</td>
<td>tolls</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSF</td>
<td>whom</td>
<td>bell</td>
<td>tolls</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Field Value (Index): **For Whom The Bell Tolls**

Field Value (Query): 

Analyze Fieldname / FieldType: **text_en**

Schema Browser: 

Verbose Output: 

Analyse Values:
text_general field with text in Chinese
胡志明 (Hồ Chí Minh)
text_cjk field with text in Chinese
胡志明 (Hồ Chí Minh)

Notice the bigrams
Nonsensical match with text_general
胡志明 (Hồ Chí Minh) vs 上海明心寺志 (Shanghai Ming xin si zhi)
Nonsensical match avoided with text_cjk
胡志明 (Hồ Chí Minh) vs 上海明心寺志 (Shanghai Ming xin si zhi)
text_cjk field with text in Chinese & Latin characters
胡志明 (Hồ Chí Minh)

Field Value (Index)
胡志明 was a Vietnamese revolutionary and politician

Field Value (Query)

Analyze Fieldname / FieldType: text_cjk  Schema Browser

<table>
<thead>
<tr>
<th>ST</th>
<th>胡志明</th>
<th>was</th>
<th>a</th>
<th>Vietnamese</th>
<th>revolutionary</th>
<th>and</th>
<th>politician</th>
</tr>
</thead>
<tbody>
<tr>
<td>CJKWF.</td>
<td>胡志明</td>
<td>was</td>
<td>a</td>
<td>Vietnamese</td>
<td>revolutionary</td>
<td>and</td>
<td>politician</td>
</tr>
<tr>
<td>LCF.</td>
<td>胡志明</td>
<td>was</td>
<td>a</td>
<td>vietnamese</td>
<td>revolutionary</td>
<td>and</td>
<td>politician</td>
</tr>
<tr>
<td>CJKBF.</td>
<td>胡志明</td>
<td>was</td>
<td>a</td>
<td>vietnamese</td>
<td>revolutionary</td>
<td>and</td>
<td>politician</td>
</tr>
</tbody>
</table>

bigrams for CJK

no bigrams for Latin alphabet
Bigrams

"Segmenting into character unigrams or bigrams is computationally easy and requires no knowledge of the language

[...]

Information retrieval research has generally found that simple approaches such as indexing overlapping character bigrams have comparable performance with more sophisticated word based approaches. As an example of overlapping bigrams, if the characters “ABCD” were Chinese characters the tokenizer would split them up into “AB” “BC” and “CD.”"

Source: https://www.hathitrust.org/blogs/large-scale-search/multilingual-issues-part-1-word-segmentation
Solr's CJK Bigram Filter

"Forms bigrams (overlapping 2-character sequences) of CJK characters that are generated from Standard Tokenizer or ICU Tokenizer.

By default, all CJK characters produce bigrams, but finer grained control is available by specifying orthographic type arguments han, hiragana, katakana, and hangul. When set to false, characters of the corresponding type will be passed through as unigrams, and will not be included in any bigrams."

Indexing our CJK text
CJK text in our data

- Source data is in MARC
- Sample record
- Author in MARC 100:
  - Lin, Quanzhong
- Subfield $6$ indicates author in original script in MARC 880:
  - 林泉忠
Indexing our data

● Specific CJK fields in addition to our existing fields
  ○ Existing: title_txt and author_txt for values in Latin alphabet
  ○ New: title_txt_cjk and author_txt_cjk for values using CJK characters

● Example
  ○ "Lin, Quanzhong" => author_txt (text_general)
  ○ "林泉忠" => author_txt_cjk (text_cjk)
Indexing author into **author_txt_cjk**

We use Traject (a Ruby gem) to process our MARC files

```ruby
# Authors for CJK languages
author_vern_lambda = extract_marc('100abcdq:110abcd:111abcd', :alternate_script=>:only)
to_field "author_txt_cjk" do |rec, acc, context|
  ...
  authors_cjk = []
  author_vern_lambda.call (rec,authors_cjk,nil)
  authors_cjk.each do |author|
    acc << author
  end
end
```
Searching for CJK text
CJK searches

- Since we created separated fields
  - author_txt is text_general
  - author_txt_cjk is text_cjk

- ...now we need to decide when to use each ¯\_(ツ)_/¯

- When searching for "Lin, Quanzhong" use field author_txt
- When searching for "林泉忠" use field author_txt_cjk
Is text in CJK?

- We are using a regular expression to detect CJK text
  
  ```ruby
  /\p{Han}|\p{Katakana}|\p{Hiragana}|\p{Hangul}/
  ```

- \p{} matches a character's Unicode script. (source)

  ```ruby
  if regex is a match
      use author_txt_cjk
  else
      use author_txt
  End
  ```

- Our code (Ruby): the controller and the regex
Works in PHP 7 too

```php
// outputs 0
echo preg_match_all("\p{Han}\u", "Lin, Quanzhong");

// outputs 3
echo preg_match_all("\p{Han}\u", "林泉忠");

?>

Notice: The /u modifier is required
Current results

Precision for CJK searches has improved significantly

<table>
<thead>
<tr>
<th>Search</th>
<th>Expected* Matches</th>
<th>Total Returned (before CJK)</th>
<th>Precision (before CJK)</th>
<th>Total Returned (with CJK)</th>
<th>Precision (with CJK)</th>
</tr>
</thead>
<tbody>
<tr>
<td>莫言 (Mo Yan)</td>
<td>72</td>
<td>300</td>
<td>0.240</td>
<td>56</td>
<td>1.285</td>
</tr>
<tr>
<td>柳美里 (Yu Miri)</td>
<td>15</td>
<td>1000</td>
<td>0.015</td>
<td>14</td>
<td>1.071</td>
</tr>
<tr>
<td>ふくろうの本 (Fukuro no hon)</td>
<td>11</td>
<td>4800</td>
<td>0.002</td>
<td>12</td>
<td>0.916</td>
</tr>
</tbody>
</table>

* Our original "Expected Matches" values were off. The current Total Returned values are in fact more accurate.
A few other notes...

- Bug SOLR-13336 in older versions of Solr causes "exponential expansion of naive queries" when creating bigrams
  - Fixed in latest versions of Solr.

- Other more robust CJK configurations
  - Stanford and Michigan's Solr configurations

- Chinese Simplified vs Chinese Traditional
  - E.g. author "Zhang, Ailing"
  - 张爱玲 (Simplified) and 張愛玲 (Traditional)
  - Not handled by text_cjk field
Thanks

Many people were involved in making this work possible

- Thanks to the people at Discovery Day 2018 for making us aware of the problem, and in particular to Nikitas Tampakis and Michael Gibney.
- And thanks to Jeanette Norris and Toshiyuki Minami for helping document, implement, and test the solution at Brown.
Source and other references

- Naomi Dushay's posts: CJK with Solr for Libraries (11 posts)
- An introduction to indexing Chinese
- HathiTrust post on word segmentation
- Podcast: The Wubi Effect (Radiolab)

- MARC field 880
- Book: Solr in Action by Trey Grainger and Timothy Potter
- Shameless plug for my workshop: Solr for newbies
Questions and (hopefully) Answers

slides: https://tinyurl.com/solr-and-cjk | email: hector_correa@brown.edu | twitter: @hectorjcorrea