The Web as a Parallel Corpus

- Philip Resnik and Noah Smith, CL'03

Presented by Kevin Duh

March 3, 2005

UW Machine Translation Reading Group

Parallel Corpora is Critical Resource

- Parallel corpora is essential for multilingual applications:
  - MT translation model
  - Multilingual NLP
  - Cross-lingual Information Retrieval …
- Parallel corpora can help monolingual applications:
  - Lexical acquisition
  - POS tagging (by projection)
  - Data sharing …
- Currently, our research is driven by our data:
  - E.g. Hansards, UN Proceedings.
  - Can we can obtain the data we desire to achieve our research objectives? (e.g. different styles and topics, different language pairs)

1

Mission statement: Extract Bitexts from the Web

The Web ***IS*** a parallel corpus

⇒ How can we mine it and extract useful bitexts?

Approaches:

1) STRAND: detect bitext based on similar structure
2) Content-based matching: use translation lexicon
3) Combination method

2

STRAND system

- Insight:

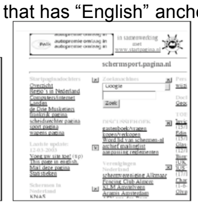
  Webpage authors often use same markup structure when presenting same content in different languages.

- 3 step process:
  1. Locate pages that might have parallel translations
  2. Generate candidate pairs
  3. Structural filtering to remove non-bitext

3

STRAND: Step 1

- Task: Locate pages that might have bitext
- Use AltaVista to find *parent* and *sibling* pages
  - Parent: page with both "English" and "French" anchors
  - Sibling: "French" page that has "English" anchor



4

STRAND: Step 2

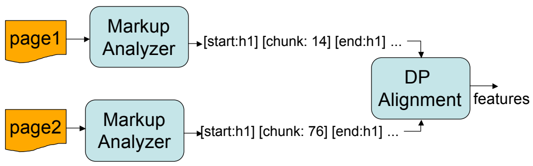
- Task: Generated candidate pairs
- Method 1:
  - Link up sibling pages
  - Link up children pages of parents
- Method 2:
  - Use crawler to consider entire websites
  - Then use URL matching:
    - http://mysite.com/english/home en.html

      => http://mysite.com/big5/home_ch.html
- Filter by document length: $length(E) \approx k * length(F)$

5

STRAND Step 3: Structural Filtering

- Step1&2 generates many non-bitexts. Step 3 filters these by exploiting HTML structural commonalities between true bitexts



Structural Filtering: Markup analysis

- Markup analyzer creates *linear* sequence of tokens
  - Types of tokens:
    - start/end token for each HTML element
    - Chunk token contains # of non-whitespace bytes in text segment

<TITLE>Sortie de Secours </TITLE>
<BODY>
→
[START:TITLE]
[Chunk:15]
[END:TITLE]
[START:BODY]

  - Embedded structure not used:
    - Tree alignment is computationally expensive
    - Many embedded markups are for text formatting, not structure

Structural Filtering: DP alignment

- Dynamic programming aligns token sequences

[START:TITLE]	[START:TITLE]
[Chunk:15]	[Chunk:13]
[END:TITLE]	[END:TITLE]
[START:BODY]	[START:BODY]
	[START:H1]
	[Chunk:13]
	[END:H1]
[Chunk:122]	[Chunk:112]

- Compute 4 scalars to quantify alignment quality
  - Percentage of non-aligned tokens
  - Number of aligned CHUNKs of unequal length
  - Correlation of lengths in CHUNKs
  - Significance level of correlation

  => Build classifier based on these features to filter bitext

STRAND Evaluation

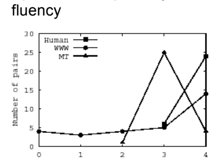
- Precision/Recall, but true Recall is impossible
  - Defines recall relative to the set of candidate pairs generated by STRAND Step 1 and 2.
- “Truth” is based on human judgment
  - Bilingual speakers are asked: “Was this pair intended to provide same content in two different languages”
  - In practice:
    - Select subset of Step2 output for human judgment; base recall on agreed human judgments
    - 16763 step2 outputs -> 326 for human judgment -> 261 pairs agreed (86 good/ 175 bad)

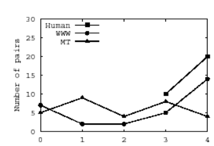
STRAND Results 1

- Precision/Recall results:
  - Set thresholds of DP alignment values based on held-out set. E.g:
    - If dp < 20% and p < 0.05, then pick candidate bitext
  - English-French: precision-100%, recall-68.6%
  - English-Chinese: precision-98%, recall-61%
    - Question: Is precision more important? Why is recall so low?
- CLIR: extract translation lexicon from STRAND
  - Backing off from bilingual dictionary to STRAND lexicon accounts for 8% token match
  - 12% relative improvement in precision over bilingual dictionary alone

STRAND Results 2

- Fluency and Adequacy Evaluation by bilingual speakers
  - Random sample of:
    - 30 human-translated bitext from FBIS
    - 30 Chinese sentence from FBIS, paired with Babelfish translation
    - 30 STRAND bitexts
  - Speakers rate pairs by translation adequacy and Chinese/English fluency





STRAND parameter tuning by machine learning

- Task: Binary classification
- Features: 4 structural values (dp,n,r,p)
- Decision tree with N-fold cross-validation using subset of data labeled by human judges

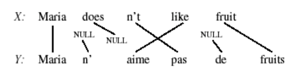
> If dp > 37 then BAD;
> Else if n > 11 then GOOD; …

Result: Precision-95.%, Recall-84.1%

Note: this process is language-independent and only requires some annotation

Content-based Matching

- Motivation
  - “Same structure” assumption doesn’t capture all bitexts
  - Many web documents don’t have much HTML markup
  - Techniques for translation detection exists
- Basic idea: Quantify Translational Similarity (tsim)
  - Computed from symmetric word-to-word model
  - Given link probability p(f,e). Find set of links that maximize total probability



Traslational Similarity (tsim)

- Definitions:
  - Link: pair (e,f), where e is English, f is French
  - Word-to-word model gives probability for each (e,f)
  - One of the pair (e,f) may be NULL
  - One word may only be linked once
- Compute probability of highest prob. set of links
  - Maximum Weighted Bipartite Matching (MWBM)
    - Given bipartite graph G=(V_1 U V_2, E) and weighted edges $C_{i,j}$ s.t. i in V_1, j in V_2
    - Find matching M in E s.t. each vertex has at most one edge in M and $\sum_{e \in M} C_{i,j}$ is maximized

Translation Similarity (tsim)

- Similarity score should be high when many link tokens in best matching M does not contain NULL.

$tsim_{MWBM}$ = log Pr(two-word links in M) / log Pr(all links in M)

- MWBM complexity is O(max(|E|,|F|)^3). If equal prob. links are assumed, MCBM can be used: O(|X|*|Y|*sqrt(|X|+|Y|))

$tsim_{MCBM}$ = # of two-word links in M) / # of all links in M

- Competitive Linking
  - greedily selects edge of highest weight, then remove from graph

Building the translation lexicon

- Several sources:
  - English-French dictionary
    - Lexicon is trained from it using EM (but most entries already have word-to-word mapping)
  - Cognate pairs
    - Learn language-specific character weights for computed weighted edit distance. Pairs are “cognates” if weight is high
    - Character weights trained from translation model built from the Bible. Resulting cognates are noisy (see Table 2).
- Combine dictionary and cognates to make Dirichlet prior
  - In EM, prior will increase expected counts for more probable word pairs
  - Train on versed-aligned Bible using MWBM
  - Final lexicon consists of all word pairs with nonzero probability

Tsim results

- Results from using Tsim only
  - Threshold: tsim = 0.44
  - Compute $tsim_{MCWB}$ on first 500 words of documents
  - Precision-83.3%, Recall-92.1%
- Combining Tsim feature with STRAND values
  - Precision-97.4%, Recall-98.0%
  - Combining structural and content features is beneficial

The Internet Archive: large-scale mining

- http://www.archive.org/web/researcher/
- (Temporal) Archive of entire web, available for researchers, etc. => enables mining on a large scale
- Estimated 120TB of data, 10 billion webpages (2003)
- Challenge with adapting STRAND: Scaling up to generate candidate page pairs
  - Subtract language-specific substrings from URL. Pages with same modified URL become candidate pairs.

Putting it all together: Building an English-Arabic Corpus

1. Search Internet Archive for country domains like .eg, .sa and .com domains originating from Arabic-speaking countries (e.g. emiratesbank.com) => 19M pages
2. Language-specific substring subtraction creates 786K types of URL (avg. 25 pages per URL)
3. 8K Arabic-English candidate pairs extracted
4. Apply structural + content-based matching
   - English is lemmatized; Arabic is converted to root form (48k types)
5. Use 149 human labeled pairs to train decision tree classifier. Apply this on entire 8K candidate set
6. Final result: 1821 bitexts
   - English: 1M tokens / Arabic: 1.3M tokens

Future Work

- Improving tsim
  - Can weights in dictionary be more robustly estimated?
  - Filter noisy translation lexicon (e.g. noisy cognates)
  - Rather than computing tsim from first 500 words, can sample content words, only words present in dictionary ..
- Bootstrapping
  - Iterative mine and enlarge training data for building lexicon and classifiers

Conclusions

- http://umiacs.umd.edu/~resnik/strand/
  - Distribute URLs rather than actual text => avoid legal issues
  - Internet Archive URLs are persistent
  - Currently available data:
    - English-French, English-Chinese, English-Basque, English-Arabic