Language Model Adaptation for Statistical Machine Translation with Structured Query Models

Bing Zhao, Matthias Eck, Stephan Vogel
CMU
Coling 2004

presented by Sarah Schwarm, 11/10/2004
Goal: Language Model Adaptation

- Problem: Insufficient in-domain LM training data

- Approach: “unsupervised data augmentation by retrieval of relevant documents from large monolingual corpora...” and interpolation of model built from retrieved data with a background LM
Approach

Baseline SMT Decoder

Query Reformulator

IR System (Target Language)

First-pass Translation Hyps

Queries

Domain-specific Data

Small Domain-specific LM

Combined LM

Interpolate

Background LM

Test Data

Translation Model

Second-pass SMT Decoder

Final Translation Hypotheses
Questions to Address

• Should we use only 1-best, or n-best hyps for query generation?

• How should queries be constructed: bag-of-words, or more structured?

• How many documents should be retrieved, and what is the scope of a document?
Results from [Eck 2004]

- Used data retrieved from a local index (Lemur IR system) rather than the web
- Used Term Frequency /Inverse Document Frequency (tf/idf) for retrieval (outperformed two other IR techniques)
- Sentence-level retrieval outperforms story-level
- Big improvements in perplexity, smaller “actual” improvement
- Stemming and stopword removal were not helpful
Sentence Retrieval Process

- tf/idf queries built from translation hyps from first-pass decoder
- Consider each sentence as its own document
- Convert query and sentences in corpus into vectors
  - Assign term weight to each word
- Calculate cosine similarity between query and sentences in corpus
  - Select most similar 1-1000 sentences
I-best hyp as query model

- $w_i$ is a word in $V_{T1}$, the vocab of the top-1 hypothesis
- $f_i$ is the frequency of $w_i$

$$Q_{T1} = (w_1, w_2, \ldots, w_l) = \{(w_i, f_i) | w_i \in V_{T1}\}$$
Bag-of-words Query Models (2/3)

N-best hyps as query model

\[ Q_{TN} = (w_{1,1}, w_{1,2}, \ldots, w_{1,l_1}; \ldots; w_{N,1}, w_{N,2}, \ldots w_{N,l_N}) \]

\[ = \{(w_i, f_i) | w_i \in V_{TN}\} \]

- **Benefits of** \( Q_{TN} \)
  
  - Contains more translation candidates; more informative than \( Q_{T1} \)
  
  - Confident translations occur more, so they have a higher term frequency and more impact on retrieval
Bag-of-words Query Models (3/3)

Translation model as query model

- Extract n-grams from source sentence
- Collect all candidate translations from TM

\[ Q_{TM} = (w_{s1,1}, w_{s1,2}, \ldots w_{s1,n_1}; \ldots; w_{si,1}, w_{si,2}, \ldots, w_{si,n_i}) \]

\[ = \{(w_i, f_i) | w_i \in V_{TM}\} \]

- No decoding, no use of background LM
  - \( Q_{TM} \) is a generalization of \( Q_{T1} \) and \( Q_{TN} \)
    (subject to more noise)
Structured Query Models

• Word order and word proximity:
  • Ignored by bag-of-words models
  • Convey syntactic and semantic information
  • Can be extracted from 1-best/n-best hyps and translation lattices
Structured Query Language
InQuery (Lemur Toolkit)

• Four proximity operators (ordered and unordered windows) in queries
  • Sum: \textit{\#sum}(t_1, \ldots, t_n)
    • all terms have equal influence, avg. belief values
  • Weighted sum: \textit{\#wsum}(w_1 : t_1, \ldots, w_n : t_n)
  • Ordered distribution operator
    \textit{\#N}(t_1 \ldots t_n)
    • Terms must be within N word of each other
  • Unordered distribution operator
    \textit{\#uwN}(t_1 \ldots t_n)
    • Terms in any order within a window of N words
Structured Query Models (1/2)

- Collect target n-grams
  - For 1/n-best hyps, collect n-grams related to each source word
  - For TM, collect source n-grams and translate to target n-grams
- Model: collection of subsets of target n-grams
- \[ Q_{st} = \{ \vec{t}_{s_1}, \vec{t}_{s_2}, \ldots, \vec{t}_{s_I} \} \]
- \( \vec{t}_{s_i} \) is a set of target n-grams for the source word \( S_i \)
  \[ \vec{t}_{s_i} = \{ \{ t_i, \ldots \} \text{1-gram}; \{ t_i t_{i+1}, \ldots \} \text{2-gram}; \{ t_{i-1} t_i t_{i+1} \} \text{3-gram} \ldots \} \]
Structured Query Models (2/2)

- Example: sum of frequency-weighted sums
  \[ q = \sum \left( w \sum (2 \text{ eu} 2 \text{ phrase(european union)}) \right) \]
  \[ \sum (12 \text{ phrase(the united states)} \text{ american 1 phrase(an american)} \) \]
  \[ \sum (4 \text{ are 1 is}) \]
  \[ \sum (8 \text{ markets 3 market}) \]
  \[ \sum (7 \text{ phrase(the main) 5 primary}) \]
Experiments

- Test set: 878 sentences from NIST June 2002 Chinese to English MT evaluation
- Report NIST and BLEU scores with 4 refs for each sentence
- Baseline model:
  - TM training data: 284k parallel sentences
  - LM training data: 160 words of general English news text
- LM adaptation corpora: 4 collections from the GigaWord Corpora (English news text)
  - Preprocessing: lowercase, separate punctuation, no stopword removal
Results: Bag-of-words Models

- All adapted LMs outperformed the baseline
- Data from AFE corpus gave best improvement
- Used 100-best list for $Q_{TN}$ model - only 9 times bigger than $Q_{T1}$ (1-best)
- Retrieval of 100 sentences was best
- Overall, $Q_{TN}$ gave best results
  - More alternatives than $Q_{T1}$
  - $Q_{TM}$ probably contributed bad alternatives as well and good ones
Results: Structured Models

- Using more retrieved data (1000 sentences) gives better results
- $Q_{TM}$ performs best - the structured model appears to reduce noise in the retrieved data
Oracle Experiment

• Use reference translations to retrieve adaptation data (4000 sentences)

• Higher BLEU and NIST scores show room for improvement

• Better 1st pass translations lead to better retrieved data which leads to better 2nd pass translations - could we iterate?

• Results are still limited by TM and decoder
Summary and Future Work

- LM adaptation by retrieving sentences similar to initial translations results in improved performance
- Structured queries which capture word order outperform bag-of-words queries
- Future work:
  - Will larger corpora for retrieval of adaptation data improve performance?
  - Can translation probabilities be included in queries?