Unsupervised Turkish Morphological Segmentation for Statistical Machine Translation

Coskun Mermer and Murat Saraclar

Workshop on Machine Translation and Morphologically-rich Languages
Haifa, 27 January 2011
Why Unsupervised?

- No human involvement
- Language independence
- Automatic optimization to task
Using a Morphological Analyzer

- Linguistic morphological analysis intuitive, but
  - language-dependent
  - ambiguous
  - not always optimal
    - manually engineered segmentation schemes can outperform a straightforward linguistic morphological segmentation
    - naive linguistic segmentation may result in even worse performance than a word-based system
Widely varying heuristics:

- Minimal segmentation
  - Only segment predominant & sure-to-help affixation
- Start with linguistic segmentation and take back some segmentations
  - Requires careful study of both linguistics, experimental results
- Trial-and-error
- Not portable to other language pairs
Adopted Approach

- Unsupervised learning from a corpus
- Maximize an objective function (posterior probability)
Probabilistic Segmentation Model

\[ P(M_f, f) = P(M_f)P(f \mid M_f) \]

- \( f \) : Observed corpus
- \( M_f \) : Hidden segmentation model for the corpus (≈ “morph” vocabulary)
MAP Segmentation

\[ \hat{M}_f = \arg\max_{M_f} P(M_f | f) \]

\[ = \arg\max_{M_f} P(M_f, f) \]

\[ = \arg\max_{M_f} P(M_f) P(f | M_f) \]
Probabilistic Model Components

\[ P(M_f) = P(\textit{frequencies}_{M_f})P(\textit{lengths}_{M_f}) \]

- \( P(\textit{frequencies}_{M_f}) \): Uniform probability for all possible morph vocabularies of size \( M \) for a given morph token count of \( N \) (i.e., frequencies do not matter)
- \( P(\textit{lengths}_{M_f}) \): For each morph, product of its character probabilities (including end-of-morph marker)
- \( P(f | M_f) \): Product of probabilities for each morph token
Original Search Algorithm

- Greedy

- Scan the current word/morph vocabulary

- Accept the best segmentation location (or non-segmentation) and update the model
Parallel Search

- Less greedy

- Wait until all the vocabulary is scanned before applying the updates
Sequential Search
Sequential Search
Sequential Search (different vocabulary scan orders)
Sequential Search vs. Parallel Search
Sequential Search vs. Parallel Search
Sequential Search vs. Parallel Search

[Graph showing the comparison between Sequential Search and Parallel Search, with the y-axis representing Model cost (logprob) and the x-axis representing Number of iterations. The graph displays the model cost decreasing over time for both search methods.]
Sequential Search vs. Parallel Search

Final model cost (logprob)

Search algorithm

Sequential

Parallel
Random Search

- Even less greedy

- Do not automatically accept the maximum probability segmentation, instead make a random draw proportional to the posteriors
  - cf. Gibbs sampling
Deterministic vs. Random Search
Deterministic vs. Random Search
Deterministic vs. Random Search
Deterministic vs. Random Search
Deterministic vs. Random Search

![Graph comparing deterministic and random search algorithms. The x-axis represents search algorithms: Sequential, Parallel, and Gibbs (2000). The y-axis represents the final model cost (logprob) ranging from 1.553 to 1.561 x 10^5. The graph shows the comparison of cost for each algorithm, with bars indicating variability.]
So far…

- We can obtain lower model costs by being less greedy in search

- Does it translate to BLEU scores?
Turkish-to-English

The diagram compares the performance of three methods: Sequential, Parallel, and Gibbs, in terms of development (Dev) and test BLEU scores. The Sequential method shows the lowest performance with a development BLEU score of approximately 55 and a test BLEU score of approximately 48. The Parallel method has a higher development BLEU score around 57 and a test BLEU score around 51. The Gibbs method has the highest development BLEU score around 58 and a test BLEU score around 51, with a notable variance indicated by the error bars.
English-to-Turkish

![Graph showing BLEU scores for English-to-Turkish translation.

**Development BLEU**
- Sequential: 38
- Parallel: 39
- Gibbs: 37

**Test BLEU**
- Sequential: 33
- Parallel: 35
- Gibbs: 34

The Gibbs method shows a slight improvement over the other two methods in both development and test settings.
Turkish-to-English (1 reference)
English-to-Turkish (1 reference)
On a Large Test Set (1512 sentences)
Turkish-to-English, No MERT
Optimizing Segmentation for Statistical Translation

- The best-performing segmentation is highly task-dependent
  - Could change when paired with a different language
  - Depends on size of parallel corpora

- For a given parallel corpus, what is the optimal segmentation in terms of translation performance?
Adding Bilingual Information

\[ \hat{M}_f = \arg \max_{M_f} P(M_f)P(f \mid M_f)P(e \mid f) \]

- \( P(e \mid f) \): Using IBM Model-1 probability
  - Estimated via EM
Results
Results
Results

![Bar chart showing BLEU scores for different methods: Words, Mono-Seq, Mono-Par, Mono-Gibbs, Bi-Par, Bi-Gibbs. The x-axis represents the methods, and the y-axis represents BLEU scores ranging from 50 to 54. The chart compares the performance of these methods, with Mono-Par and Mono-Gibbs showing the highest BLEU scores.](image-url)
Results
Evolution of the Gibbs Chain
Evolution of the Gibbs Chain

Model cost (logprob) vs Number of iterations
Evolution of the Gibbs Chain (BLEU)
Evolution of the Gibbs Chain (BLEU)
Conclusions

- Probabilistic model for unsupervised learning of segmentation
- Improvements to the search algorithm
  - Parallel search
  - Random search via Gibbs sampling
- Incorporated (an approximate) translation probability to the model
- So far, model scores do not correlate well with BLEU scores