Two methods for stabilizing MERT: NICT at IWSLT 2009

Masao Utiyama, Hirofumi Yamamoto, and Eiichiro Sumita

NICT
Our system

- Participated in the Challenge Task
- A baseline phrase-based SMT system

Outline

- Language resources
- Combination of Chinese segmentations
- Two methods for stabilizing MERT
- Official results
- What we tried but didn’t work
Language resources

- Training data: IWSLT09_BTEC.train.*, IWSLT09.devset*, IWSLT09_CT.train.*
- Expanded the devset data
- Two language models from the CT portion and rest
- Development data: Sample sentences from the training data. These sentences were excluded from the training data.
- Development test data:
  IWSLT09_CT.devset.*.with_interpreter.txt
- In-house tokenizers for Chinese and English
- Lowercased English sentences in EC
- Truecase in CE
Combination of Chinese segmentations

Method 1
Word segmentation with our in-house tokenizer

Method 2

1. Segmentation into characters with ‘⟨w⟩’ tags inserted between words.
2. Insertion of ‘⟨w⟩’ into English texts
3. Made a phrase (reordering) table from this data.

Combination of the two tables

1. Phrases in the first phrase (reordering) table were segmented into characters.
2. Removal of “⟨w⟩” from the output
Two methods for stabilizing MERT

• Devset sampling
• Averaged mert
Causes of instability in MERT

Mismatch between development and test data

- **Devset sampling** tries to sample sentences that are similar to the input data
- Best parameters on devset $\neq$ Best parameters on test
  
  *Averaged MERT* avoids over-fitting by using averaged parameters
Devset sampling

- Sampling similar sentences to the input texts
- Sampled sentences were excluded from the training
- For test each sentence in the 500 test sentences
  - Extracted the most similar 2 sentences
  - Average of BLEU1, ... BLEU4 scores as the similarity score
  - Input sentence was regarded as the reference
  - Used the most similar 1000 sentence development data
- After the tuning, the development data were added to the training data again to make the final model
## Results using devset sampling

<table>
<thead>
<tr>
<th></th>
<th>with sampling</th>
<th>w/o sampling</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC</td>
<td>32.16</td>
<td>30.34</td>
</tr>
<tr>
<td>CE</td>
<td>28.66</td>
<td>26.12</td>
</tr>
</tbody>
</table>
Similar sentences (testset // devset)

- hotel royal plaza may i help you // holiday inn crowne plaza may i help you
- yes can you tell me the number of people type of room and approximate budget please // yes would you tell me the address and phone number of the hotel please
- yes let me check for vacancies // yes let me check hold on a moment please
- sorry to keep you waiting // sorry to keep you waiting
- we have two types of rooms available in your budget range // we have two types of dressing japanese or french
Averaged MERT

- Run MERT several times on a development data
- Average tuned parameters to get final parameters

Why this works?

\[ \sum_{m=1}^{M} \lambda_m h_m(e, f) \]

- Average of parameters (weights) \( \rightarrow \) average of scores
- A kind of a system combination method
## Results of Averaged Mert

<table>
<thead>
<tr>
<th></th>
<th>average</th>
<th>max</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC</td>
<td>32.61</td>
<td>31.93</td>
</tr>
<tr>
<td>CE</td>
<td>29.24</td>
<td>28.49</td>
</tr>
</tbody>
</table>
Additional experiments on IWSLT-2007 Japanese-English translation task

- Bootstrap method
- Run MERT 100 times on devset1
- Obtained 100 parameter sets
- Calculated the averages and standard deviations of BLEU scores for 1, 2, 3, 5, 7, 10, 20, 30, 50, 70, and 100 parameter sets by sampling these parameter sets
- Sampled 100 parameter sets for each number of parameter sets.
Two methods

• Averaged parameters

• Parameters that obtained the maximum BLEU score on devset1
## Results

<table>
<thead>
<tr>
<th>No.</th>
<th>average av. (std.)</th>
<th>maximum av. (std.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>62.22 (0.54)</td>
<td>62.22 (0.54)</td>
</tr>
<tr>
<td>2</td>
<td>62.59 (0.41)</td>
<td>62.32 (0.42)</td>
</tr>
<tr>
<td>3</td>
<td>62.63 (0.37)</td>
<td>62.08 (0.59)</td>
</tr>
<tr>
<td>5</td>
<td>62.72 (0.38)</td>
<td>62.18 (0.53)</td>
</tr>
<tr>
<td>7</td>
<td>62.72 (0.29)</td>
<td>62.14 (0.56)</td>
</tr>
<tr>
<td>10</td>
<td>62.73 (0.27)</td>
<td>62.14 (0.54)</td>
</tr>
<tr>
<td>20</td>
<td>62.71 (0.21)</td>
<td>62.27 (0.52)</td>
</tr>
<tr>
<td>30</td>
<td>62.73 (0.21)</td>
<td>62.16 (0.55)</td>
</tr>
<tr>
<td>50</td>
<td>62.69 (0.19)</td>
<td>62.36 (0.45)</td>
</tr>
<tr>
<td>70</td>
<td>62.70 (0.16)</td>
<td>62.42 (0.41)</td>
</tr>
<tr>
<td>100</td>
<td>62.71 (0.15)</td>
<td>62.50 (0.33)</td>
</tr>
</tbody>
</table>
### BLEU scores for our official submissions

<table>
<thead>
<tr>
<th></th>
<th>EC</th>
<th>CE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>c+p</td>
<td>nc+np</td>
</tr>
<tr>
<td>ASR</td>
<td>35.83</td>
<td>35.44</td>
</tr>
<tr>
<td>CSR</td>
<td>38.42</td>
<td>38.15</td>
</tr>
<tr>
<td></td>
<td>26.67</td>
<td>25.80</td>
</tr>
<tr>
<td></td>
<td>29.70</td>
<td>28.72</td>
</tr>
</tbody>
</table>
What we tried but didn’t work

• Increasing the size of the CT corpus
• Alignment with lowercased prefixes
• Replacing numbers with a special symbol
Increasing the size of the CT corpus

- Adding several replications of each sentence of the CT corpus when we added them to the BTEC corpus

  with \ w/o
  EC  30.89  31.25
  CE  25.12  26.11
Alignment with lowercased prefixes

- Using lowercased 4-letter prefixes of English words in word alignment

  with \ w/o
  EC 29.58  32.22
  CE 26.73  26.91
Replacing numbers with a special symbol

with w/o
EC 29.56 32.22
CE 24.17 26.91

Examples failed

• a Chinese word sequence “0 0 0” was translated into “triple o”
Conclusions

• Participated in the Challenge Task

• Two methods for stabilizing MERT to reduce mismatch between development and test data
  – Devset sampling tries to sample sentences that are similar to the input data
  – Best parameters on devset ≠ Best parameters on test
    Averaged MERT avoids over-fitting by using averaged parameters