NAVER Machine Translation System for WAT 2015

Hyoung-Gyu Lee, Jae-Song Lee, Jun-Seok Kim and Chang-Ki Lee
2015-10-16
Contents

- Introduction
- English-to-Japanese MT Task
- Korean-to-Japanese MT Task
- Summary
Introduction
# Traditional SMT and Neural MT

<table>
<thead>
<tr>
<th>Traditional SMT</th>
<th>Traditional SMT + Neural Network</th>
<th>Neural MT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Sentence</td>
<td>Source Sentence</td>
<td>Source Sentence</td>
</tr>
<tr>
<td>Traditional SMT (PB/HPB/T2S/F2S)</td>
<td>Traditional SMT</td>
<td>Traditional SMT</td>
</tr>
<tr>
<td>Target Sentence</td>
<td>Target Sentence</td>
<td>Target Sentence</td>
</tr>
<tr>
<td>a few year ago</td>
<td>recently</td>
<td>more recently</td>
</tr>
</tbody>
</table>

- **Traditional SMT**
  - Source Sentence
  - Traditional SMT (PB/HPB/T2S/F2S)
  - Target Sentence
  - a few year ago

- **Traditional SMT + Neural Network**
  - Source Sentence
  - Traditional SMT
  - Target Sentence
  - recently

- **Neural MT**
  - Source Sentence
  - Neural Network
  - Target Sentence
  - more recently
Neural Machine Translation

- Proposed by Google and Montreal University in 2014
- Is called
  - Sequence-to-sequence model
  - End-to-end model
- Input sentence is encoded into fixed-length vector, and from the vector translated sentence is produced. That’s all
- Various extensions are emerged
  - LSTM, GRU, Bidirectional Encoding, Attention Mechanism, …
# Pros and Cons of NMT

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓ no need domain knowledge</td>
<td>✓ Is time consuming to train NMT model</td>
</tr>
<tr>
<td>✓ no need to store explicit TM and LM</td>
<td>✓ Is slow in decoding, if target vocab. is large</td>
</tr>
<tr>
<td>✓ Can jointly train multiple features</td>
<td>✓ Is weak to OOV problem</td>
</tr>
<tr>
<td>✓ Can implement decoder easily</td>
<td>✓ Is difficult to debug</td>
</tr>
</tbody>
</table>

Pros:
- No need domain knowledge
- No need to store explicit TM and LM
- Can jointly train multiple features
- Can implement decoder easily

Cons:
- Is time consuming to train NMT model
- Is slow in decoding, if target vocab. is large
- Is weak to OOV problem
- Is difficult to debug
At WAT 2015 ...

- Two tasks
  - English-Japanese MT
  - Korean-Japanese MT

- Methods of MT
  - Traditional SMT
  - Neural MT
  - Traditional SMT + Neural MT
English-to-Japanese Machine Translation Task
Outline of ENG-JPN MT Task

- Source Vocab.
- Training Corpus
- OOV Handling
- T2S Syntax-based MT
- NMT Re-ranking

English sentence → OOV Handling → T2S Syntax-based MT → NMT Re-ranking → Japanese sentence

n-best
Tree-to-String Syntax-based MT

- **Training Corpus**
  - Translation model:
    - 1 million sentence pairs (train-1.txt)
  - Language model:
    - 3 million Japanese sentences (train-1.txt, train-2.txt)

- **Tokenizer**
  - English: Moses tokenizer
  - Japanese: In-house tokenizer and POS tagger

- **T2S model**
  - Assign linguistic syntax label to X hole of HPB model
  - Use Berkeley parser
Tree-to-String Syntax-based MT 2/2

- **Rule Augmentation**
  - Proposed by CMU’s venugopal and Zollmann in 2006
  - Extract more rules by modifying parse trees
  - Use relax-parser in Moses toolkit (option: SAMT 2)

```
I                 love              you
(0)                (1)                (2)
PRP                VBP                PRP
NP                 NP
VP
S
```

<table>
<thead>
<tr>
<th>Baseline nodes</th>
<th>Additional nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-0 PRP</td>
<td>1-2 VBP+PRP</td>
</tr>
<tr>
<td>0-0 NP</td>
<td>0-2 PRP+VP</td>
</tr>
<tr>
<td>1-1 VBP</td>
<td>0-1 PRP++VBP</td>
</tr>
<tr>
<td>2-2 PRP</td>
<td></td>
</tr>
<tr>
<td>2-2 NP</td>
<td></td>
</tr>
<tr>
<td>1-2 VP</td>
<td></td>
</tr>
<tr>
<td>0-2 S</td>
<td></td>
</tr>
</tbody>
</table>

Handling OOV

1) Hyphen word split
   - Ex.) nano-laminate \(\rightarrow\) nano laminate

2) English spell correction
   - Use open source spell checker, ‘Aspell’

| Detection Phrase | ✓ Based on skip rules  
|                  | ✓ Skip the word containing capital, number or symbol |
| Correction Phrase| ✓ Based on edit distance  
|                  | ✓ Because large gap causes wrong correction  
|                  | ✓ Select one with shortest distance among top-3 suggestion |

VLSI … H2 … remarakable …

remakable detection \(\rightarrow\) remarakable correction \(\rightarrow\) remarkable

1. remarkable
2. remakable
3. reamarkable

[Suggestion by Aspell]
Neural Machine Translation (1/2)

- RNN with an attention mechanism [Bahdanau, 2015]

| Tokenization          | English: word-level  
<table>
<thead>
<tr>
<th></th>
<th>Japanese: char-level</th>
</tr>
</thead>
</table>
| # of vocab.           | English: 245k        
|                       | Japanese: 6k         |
| BI representation     | Use                  
|                       | Ex) 大学生 => 大/B 学/I 生/I |
| Dim. of word-embedding| 200                  |
| Size of recurrent unit| 1000                 |
| Optimization          | Stochastic gradient  
|                       | descent (SGD)        |
| Drop-out              | Don’t use            |
| Time of training      | 10 days (4 epoch)    |
Neural Machine Translation (2/2)

\[
h_t = [\tilde{h}_t; \tilde{h}_t]
\]

\[
\tilde{h}_t = f_{GRU}(W_{s\_we}x_t, \tilde{h}_{t+1})
\]

\[
\tilde{h}_t = f_{GRU}(W_{s\_we}x_t, \tilde{h}_{t-1})
\]

\[
c_t = \sum_{i=1}^{T} \alpha_{ti} h_i
\]

\[
\alpha_{ti} = \frac{\exp(e_{tj})}{\sum_{j=1}^{T} \exp(e_{tj})}
\]

\[
e_{ti} = f_{FFNN}(z_{t-1}, h_i, y_{t-1})
\]

- New hidden state of the decoder
  \[
  z_t = f_{GRU}(y_{t-1}, z_{t-1}, c_t)
  \]

- Prob. of the next target word
  \[
p(y_t|y_{<t}, x) = y_t^T f_{softmax}\{W_{z' y}z'_t + W_{zy}z_t + W_{cy}c_t + W_{yy}(W_{t\_we}y_{t-1}) + b_y\}
  \]
  \[
z'_t = f_{ReLU}(W_{z' z}z_t)
  \]

[ Modified RNN ]
## Experimental Results (T2S Syntax-based MT)

<table>
<thead>
<tr>
<th>SYS</th>
<th>BLEU</th>
<th>#Rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>T2S SB MT</td>
<td>31.34</td>
<td>250M</td>
</tr>
<tr>
<td>+ Rule augmentation</td>
<td>32.48</td>
<td>1950M</td>
</tr>
<tr>
<td>+ Parameter modification</td>
<td>32.63</td>
<td>1950M</td>
</tr>
<tr>
<td>+ OOV handling</td>
<td>32.76</td>
<td>1950M</td>
</tr>
</tbody>
</table>

- Rule augmentation increases both BLEU and #Rules
- OOV handling improves the performance
Experimental Results (Neural MT)

<table>
<thead>
<tr>
<th>NMT Model</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN (target word-level)</td>
<td>29.78</td>
</tr>
<tr>
<td>RNN (target char-level)</td>
<td>31.25</td>
</tr>
<tr>
<td>RNN (target char-level with BI)</td>
<td>32.05</td>
</tr>
<tr>
<td>Modified RNN (target char-level with BI)</td>
<td>33.14</td>
</tr>
</tbody>
</table>

- Char-level of target language is better than word-level
- BI representation is helpful
- Modified RNN is better than original RNN
## Experimental Results (/w Human evaluation)

<table>
<thead>
<tr>
<th>SYS</th>
<th>ENG-JPN</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BLEU</td>
<td>Human</td>
</tr>
<tr>
<td>T2S SB MT* only</td>
<td>32.76</td>
<td>-</td>
</tr>
<tr>
<td>NMT** only</td>
<td>33.14</td>
<td>48.50</td>
</tr>
<tr>
<td>T2S SB MT* + NMT** re-ranking</td>
<td>34.60</td>
<td>53.25</td>
</tr>
</tbody>
</table>

- NMT only outperform T2S SB MT
- NMT re-ranking gives the best

- T2S SB MT* : Rule augmentation + Parameter modification + OOV handling
- NMT** : Modified NMT using target char. seg. with B/I
Korean-to-Japanese Machine Translation Task
Outline of KOR-JPN MT Task

- Korean sentence
- Char-level PBMT Decoding
- Word-level PBMT Decoding
- NMT Re-ranking
- Japanese sentence

Training Corpus

N-best
Phrase-based MT system

- Training Corpus
  - Translation model & Language model
    - 1 million sentence pairs (JPO corpus)

- Word-level PB MT
  - use Mecab-ko and Juman for tokenization
  - 5-gram LM

- Char-level PB MT
  - tokenize Korean and Japanese into char-level
  - 10-gram LM
  - Max-phrase length: 10
Neural Machine Translation

- RNN using attention mechanism [Bahdanau, 2015]

| Tokenization | Korean: word-level  
Japanese: char-level |
|---------------|-------------------|
| # of vocab.   | Korean: 60k  
Japanese: 5k    |
| BI representation | Use  
Ex) 大学生 => 大/B 学/I 生/I |
| Dim. of word-embedding | 200 |
| Size of recurrent unit | 1000 |
| Optimization   | Stochastic gradient descent (SGD) |
| Drop-out       | Don’t use |
| Time of training | 10 days (4 epoch) |
Combination of PBMT+ NMT

- Rule-based
  - Choose the result of char-based PB if there is OOV in word-level
  - Choose the result of word-based PB, otherwise

- NMT-based
  - Re-rank simply by NMT score
## Experimental Results

<table>
<thead>
<tr>
<th>SYS</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word PB</td>
<td>70.36</td>
</tr>
<tr>
<td>Character PB</td>
<td>70.31</td>
</tr>
<tr>
<td>Word PB + Character PB</td>
<td>70.91</td>
</tr>
</tbody>
</table>

- Character-level PB is comparable to Word-level PB
- Combined system has the best result
## Experimental Results (/w human evaluation)

<table>
<thead>
<tr>
<th>SYS</th>
<th>KOR-JPN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BLEU</td>
</tr>
<tr>
<td>Word PB + Character PB</td>
<td>70.91</td>
</tr>
<tr>
<td>NMT only</td>
<td>65.72</td>
</tr>
<tr>
<td>Word PB + Character PB + NMT re-ranking</td>
<td>71.38</td>
</tr>
</tbody>
</table>

- NMT only doesn’t outperform PBMT
- NMT re-ranking gives the best
Summary

- We apply different MT models for each task
- T2S/PB SMT + NMT Re-ranking is best in both tasks
- Char-level tokenization of target language is useful for NMT
  - Speed up the time of training
  - Vanish OOV problem
  - Give the better BLEU score
- BI representation of char-level tokenization is helpful also for NMT
- In the future, we will apply our method to other language-pair; CHN-JPN