Fast and Accurate Preordering for SMT using Neural Networks

Adrià de Gispert, Gonzalo Iglesias and Bill Byrne
SDL Research, Cambridge (UK)
Preordering for SMT

○ Transform the source sentence into target-like order

○ Doubly beneficial:
  – Better translation models → improved quality
  – Faster decoding is possible (less distortion required)

○ Particularly interesting for real-life commercial systems
Preordering for SMT

We have come to quite like Xi’an.

私たちは、すっかり西安が好きになりました。
We have come to quite like Xi’an.

We quite Xi’an like to come have.

私たちは、すっかり西安が好きになりました。
Approaches to Preordering

- Rule-based, using language-specific hand-crafted rules

- Statistical, learnt from:
  - Hand-aligned data (rarely available)
  - Automatically-aligned data
  - Automatically-aligned source-parsed data ⇔ this work
Approaches to Preordering (2)

○ From automatically-aligned source-parsed data:
  – Genzel (2010). Extracts corpus-level preordering rule sequences
    • all sentences get identical preordering treatment; no lexical info used
  – Jehl et al. (2014). Feature-rich Logistic Regression model
    • predicts when to swap two nodes in the source tree; incorporates lexical information; needs feature engineering

  – This work: use a Neural Network to learn the node-swapping model
    • superior modeling capabilities, better regression model
    • no need for feature engineering
    • faster decoding
**Preordering as node-pair swapping (Jehl et al. 2014)**

**PREORDERPARETRE**

1. for each node $n \in N$, $|C_n| > 1$
2. $F \leftarrow \text{GETFEATURES}(n)$
3. for each pair of nodes $i, j \in C_n, i \neq j$
4. $F \leftarrow F \cup \text{GETFEATURES}(i)$
5. $F \leftarrow F \cup \text{GETFEATURES}(j)$
6. $F_c \leftarrow \text{FEATURECOMBINATIONS}(F)$
7. $p_n(i, j) = \text{LOGREGPREDICT}(F, F_c)$
8. end for
9. $\pi_n \leftarrow \text{SEARCHPERMUTATION}(p_n)$
10. $\text{SORT}(C_n, \pi_n)$

**Features:** POS, dependency labels, identity and class of the head word (parent node), left/right-most word (children nodes)

**Combinations:** bigrams and trigrams of some of the above ← too many features with lexical info!
Preordering as node-pair swapping (Jehl et al. 2014)

We replace steps 6-7 by feed-forward Neural Network

- Trained with NPLM on ~100M labeled samples from aligned parallel corpus: swap if it decreases number of crossings
- 4 layers: in, 2 hidden, out softmax with 2 output values
- same basic features: no explicit combinations needed

**PreorderParseTree**

1. \textbf{for each} node \( n \in N, \vert C_n \vert > 1 \)
2. \( F \leftarrow \text{GetFeatures}(n) \)
3. \textbf{for each} pair of nodes \( i, j \in C_n, i \neq j \)
4. \( F \leftarrow F \cup \text{GetFeatures}(i) \)
5. \( F \leftarrow F \cup \text{GetFeatures}(j) \)
6. \( F_c \leftarrow \text{FeatureCombinations}(F) \)
7. \( p_n(i, j) = \text{LogRegPredict}(F, F_c) \)
8. \textbf{end for}
9. \( \pi_n \leftarrow \text{SearchPermutation}(p_n) \)
10. \( \text{Sort}(C_n, \pi_n) \)

\( F \) contains features including POS, dependency labels, identity and class of the head word (parent node), left/right-most word (children nodes) and bigrams and trigrams of some of the above \( \leftarrow \) too many features with lexical info!
Given a node with 3 children: $s_1 \ s_2 \ s_3$

Score each possible start:
- with $s_1$: $(1-p(s_1,s_2)) \cdot (1-p(s_1,s_3))$
- with $s_2$: $p(s_1,s_2) \cdot (1-p(s_2,s_3))$
- with $s_3$: $p(s_1,s_3) \cdot p(s_2,s_3)$

Continue exploring space of permutations with depth-first branch-and-bound search until global optimum is found.
Intrinsic Evaluation: Crossing score

- baseline
- Genzel'10
- Jehl'14
- this work

Languages:
- jpn
- kor
- hin
- por
- chi
- ara
- spa
Translation experiments

○ Phrase-based decoder, averaging 3 MERT runs

○ English into Japanese, Korean, Chinese, Arabic and Hindi
  – over 100M words training corpora (except Hindi: 9M)
  – General domain, web-crawled

○ Two test sets:
  – in domain: same as parallel data
  – mixed domain: equally represent 10 domains (news, health, sport, science, business, chat, ...)

○ 2014 WMT English-Hindi task
## Translation Quality Evaluation: BLEU

<table>
<thead>
<tr>
<th>$d$</th>
<th>system</th>
<th>speed ratio</th>
<th>eng-jpn in</th>
<th>eng-jpn mixed</th>
<th>eng-kor in</th>
<th>eng-kor mixed</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>baseline</td>
<td>1x</td>
<td>54.5 ± 0.2</td>
<td>26.2 ± 0.2</td>
<td>33.5 ± 0.3</td>
<td>9.7 ± 0.2</td>
</tr>
<tr>
<td>3</td>
<td>baseline</td>
<td>3.2x</td>
<td>50.9 ± 0.2</td>
<td>25.0 ± 0.2</td>
<td>28.7 ± 0.1</td>
<td>8.2 ± 0.1</td>
</tr>
<tr>
<td>3</td>
<td>Genzel (2010)</td>
<td>2.7x</td>
<td>54.0 ± 0.1</td>
<td>26.4 ± 0.2</td>
<td>30.5 ± 0.2</td>
<td>9.8 ± 0.2</td>
</tr>
<tr>
<td>3</td>
<td>Jehl et al (2014)</td>
<td>2.3x</td>
<td>55.0 ± 0.1</td>
<td>26.9 ± 0.2</td>
<td>33.1 ± 0.1</td>
<td>10.4 ± 0.1</td>
</tr>
<tr>
<td>3</td>
<td><em>this work</em></td>
<td>2.7x</td>
<td><strong>55.6 ± 0.2</strong></td>
<td><strong>27.2 ± 0.1</strong></td>
<td><strong>33.4 ± 0.1</strong></td>
<td><strong>10.6 ± 0.2</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$d$</th>
<th>system</th>
<th>eng-chi in</th>
<th>eng-chi mixed</th>
<th>eng-ara in</th>
<th>eng-ara mixed</th>
<th>eng-hin mixed</th>
<th>wmt14</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>baseline</td>
<td><strong>46.9 ± 0.5</strong></td>
<td>18.4 ± 0.6</td>
<td>25.1 ± 0.1</td>
<td>22.7 ± 0.2</td>
<td>10.1 ± 0.3</td>
<td>11.7 ± 0.1</td>
</tr>
<tr>
<td>3</td>
<td>baseline</td>
<td>44.8 ± 0.7</td>
<td>18.3 ± 0.4</td>
<td>24.6 ± 0.1</td>
<td>21.9 ± 0.2</td>
<td>8.3 ± 0.2</td>
<td>9.3 ± 0.3</td>
</tr>
<tr>
<td>3</td>
<td>Genzel (2010)</td>
<td>45.4 ± 0.2</td>
<td>17.9 ± 0.2</td>
<td>24.8 ± 0.1</td>
<td>21.6 ± 0.3</td>
<td>9.6 ± 0.2</td>
<td>11.4 ± 0.3</td>
</tr>
<tr>
<td>3</td>
<td>Jehl et al (2014)</td>
<td>45.8 ± 0.1</td>
<td>18.5 ± 0.3</td>
<td>25.1 ± 0.2</td>
<td>22.4 ± 0.2</td>
<td>10.0 ± 0.1</td>
<td>12.7 ± 0.3</td>
</tr>
<tr>
<td>3</td>
<td><em>this work</em></td>
<td><strong>46.5 ± 0.4</strong></td>
<td><strong>19.2 ± 0.2</strong></td>
<td><strong>25.5 ± 0.2</strong></td>
<td><strong>22.6 ± 0.1</strong></td>
<td><strong>10.6 ± 0.1</strong></td>
<td><strong>12.6 ± 0.3</strong></td>
</tr>
</tbody>
</table>

Best WMT14 constrained system: 11.1
Translation Quality Evaluation: BLEU

<table>
<thead>
<tr>
<th>d</th>
<th>system</th>
<th>speed ratio</th>
<th>eng-jpn in</th>
<th>mixed</th>
<th>eng-kor in</th>
<th>mixed</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>baseline</td>
<td>1x</td>
<td>54.5 ± 0.2</td>
<td>26.2 ± 0.2</td>
<td>33.5 ± 0.3</td>
<td>9.7 ± 0.2</td>
</tr>
<tr>
<td>3</td>
<td>baseline</td>
<td>3.2x</td>
<td>50.9 ± 0.2</td>
<td>25.0 ± 0.2</td>
<td>28.7 ± 0.1</td>
<td>8.2 ± 0.1</td>
</tr>
<tr>
<td>3</td>
<td>Genzel (2010)</td>
<td>2.7x</td>
<td>54.0 ± 0.1</td>
<td>26.4 ± 0.2</td>
<td>30.5 ± 0.2</td>
<td>9.8 ± 0.2</td>
</tr>
<tr>
<td>3</td>
<td>Jehl et al (2014)</td>
<td>2.3x</td>
<td>55.0 ± 0.1</td>
<td>26.9 ± 0.2</td>
<td>33.1 ± 0.1</td>
<td>10.4 ± 0.1</td>
</tr>
<tr>
<td>3</td>
<td>this work</td>
<td>2.7x</td>
<td>55.6 ± 0.2</td>
<td>27.2 ± 0.1</td>
<td>33.4 ± 0.1</td>
<td>10.6 ± 0.2</td>
</tr>
</tbody>
</table>

We achieve the best performance among all preorderers.

Scores with distortion 3 are:
- better or equal than baseline with distortion 10, but about 3 times faster
- much better than fast baseline with distortion 3
Faster decoding

- Preordering allows much faster decoding: narrower beams can be used as task is more monotonic

<table>
<thead>
<tr>
<th>$d$</th>
<th>system</th>
<th>speed ratio</th>
<th>eng-jpn</th>
<th>eng-kor</th>
<th>eng-chi</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>in</td>
<td>mixed</td>
<td>in</td>
</tr>
<tr>
<td>10</td>
<td>baseline</td>
<td>22x</td>
<td>53.6 (-0.9)</td>
<td>25.4 (-0.8)</td>
<td>32.8 (-0.7)</td>
</tr>
<tr>
<td>3</td>
<td>baseline</td>
<td>66x</td>
<td>50.5 (-0.4)</td>
<td>24.8 (-0.2)</td>
<td>28.8 (+0.1)</td>
</tr>
<tr>
<td>3</td>
<td>Genzel (2010)</td>
<td>64x</td>
<td>53.8 (-0.2)</td>
<td>26.3 (-0.1)</td>
<td>30.4 (-0.1)</td>
</tr>
<tr>
<td>3</td>
<td>Jehl et al (2014)</td>
<td>61x</td>
<td>55.0 (0.0)</td>
<td>26.5 (-0.4)</td>
<td>33.0 (-0.1)</td>
</tr>
<tr>
<td>3</td>
<td>this work</td>
<td>65x</td>
<td><strong>55.7 (+0.1)</strong></td>
<td><strong>27.2 (0.0)</strong></td>
<td><strong>33.2 (-0.2)</strong></td>
</tr>
</tbody>
</table>

Good preordering: same performance than with wide beams, but way faster
Example

We have come to quite like Xi’an
Example

We have come to quite like Xi’an like

[SENT] [TO] [LIKE]
We have come quite like Xi’an like to
Example

We [TO] have come [SENT] quite [LIKE] to Xi’an like

Syntactic Analysis

Preordering
Conclusions

We use a Neural Network to model the node-swapping Logistic Regression model for Preordering

○ Accurate:
  – automatically learns non-linear feature combinations
  – beats previous models in crossing score and translation performance

○ Fast:
  – feed-forward network is more efficient than explicit feature combination
  – BLEU scores improve at very fast decoding conditions

Now Hiring SDL Research

Research Scientists & Research Engineers
Los Angeles & Cambridge, UK

SDL.com/Research
Thank you for your attention!