Improved Neural Machine Translation with a Syntax-Aware Encoder and Decoder

Huadong Chen¹, Shujian Huang¹, David Chiang², Jiajun Chen¹
{chenhd,huangsj,chenjj}@nlp.nju.edu.cn
dchiang@nd.edu

1. State Key Laboratory of Novel Software Technology
   (Nanjing University)
2. University of Notre Dame
Outline

- Motivation
- Approach
- Experiments
- Conclusion
Part I
Motivation
Neural Machine Translation

- Encoder-decoder framework

Cho et al., (2014)
Neural Machine Translation

• Attentional NMT

Bahdanau et al., (2015)
Their success depends on the representation they use to bridge the source and target language sentences.
• However, this representation, a sequence of fixed dimensional vectors, differs considerably from
  – most theories about mental representations of sentences;
  – and traditional natural language processing pipelines, in which semantics is built up compositionally using a syntactic structure.

• It neglecting the potentially useful structural information
  – perhaps as evidence of this, current NMT models still suffer from syntactic errors such as attachment (Shi et al., 2016).
Neural Machine Translation

- Encoder: building up representation at higher levels, such as phrases, may need structures
Neural Machine Translation

• Decoder: structures could act as the guidance or control for generation
Our Work

• We propose an encoder-decoder framework that takes the syntactic structures into consideration, which includes a bidirectional tree structure encoder and a tree coverage decoder.
Part 2

Syntax-aware Neural Machine Translation
• **Bottom-up tree encoder** (Tai et al., 2015, Eriguchi et al., 2016):
  
  – building the tree structure representations from the bottom, which form the representations of constituents from their children
We assume model consistency is important.
Our sequential model is based on bidirectional GRU.
We design tree-GRU instead of using tree-LSTM.
• **Drawbacks of bottom-up tree encoder:**
  – the learned representation of a node is based on its subtree only; it contains no information from higher up in the tree
  – the representation of leaf nodes is still the sequential one, thus no syntactic information is fed into words.
• Bi-directional tree encoder
  – also propagating information from the top, which includes information from outside the current constituent
  – bi-directional Tree-LSMT for classification (Teng and Zhang, 2016)
  – bi-directional Tree-GRU for sentiment (Kokkinos and Potamianos, 2017)
• The top-down encoder by itself would have no lexical information as input
  – we feed the hidden states of the bottom-up encoder to the top-down encoder as input
• The information propagated from parent node to left and right nodes is redundant
  – we use different weights for left and right nodes
• Treating the representation of tree nodes the same as word representations and performing attention (Eriguchi et al., 2016)
  – Pros: enabling attention at a higher level, i.e. the words in the same sub-tree could get attentions as a whole unit
  – Cons: still missing structural control, i.e. the attention for words and tree nodes may interfere with each other
• Two observations of translations:
  – a syntactic phrase in the source sentence is often incorrectly translated into discontinuous words in the output
  – the attention model prefers to attend to the non-leaf nodes, which may aggravate the over-translation problem
Tree-Coverage Model (2/5)

Attention with Tree Encoder
Tree-Coverage Model (3/5)

• Coverage model (Tu et al., 2016)
  – it could be interpreted as a control mechanism for the attention model

• Drawbacks
  – the coverage model sees the source-sentence annotations as a bag of vectors
  – it knows nothing about word order, still less about syntactic structure.
• We propose to use prior knowledge to control the attention mechanism
  – in our case, the prior knowledge is the source syntactic information
  – in particular, we build our model on top of the word coverage model proposed by Tu et al. (2016)
Tree-Coverage Model (5/5)

\[
C_{j,i} = \text{GRU}(C_{j-1,i}, \alpha_{j,i}, d_{j-1}, h_i).
\]

\[
C_{j,i} = \text{GRU}(C_{j-1,i}, \alpha_{j,i}, d_{j-1}, h_i, C_{j-1,L(i)}, \alpha_{j,L(i)}, C_{j-1,R(i)}, \alpha_{j,R(i)}).
\]
Part 3

Experiments
Data and Settings

• 1.6 million sentence pairs from LDC for training
• Using MT02 for held-out dev, MT03, 04, 05, 06 for test
• Implementation based on the dl4mt package
Tree-GRU v.s. Tree-LSTM

<table>
<thead>
<tr>
<th>Systems</th>
<th>#Params</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tree-GRU</td>
<td>130M</td>
<td>1x</td>
</tr>
<tr>
<td>SeqTree-LSTM</td>
<td>+ 1.6M</td>
<td>1.3x</td>
</tr>
</tbody>
</table>
Our tree-coverage model consistently improves performance further (rows 9–11)
Tree-Coverage Model (2/2)

Attention with Tree Encoder

+ Tree-Coverage Model
1. The proposed bidirectional tree encoder outperforms the sequential NMT system and the Tree-GRU encoder across all lengths.
2. The improvements become larger for sentences longer than 20 words, and the biggest improvement is for sentences longer than 50 words.
Conclusion

• We have investigated the potential of using explicit source-side syntactic trees in NMT.
• The improvement could come from:
  – the enrichment of the representation during encoding;
  – the structural control of attention during decoding.
• In this paper, we only use the binarized structure of the source side tree. For future work, it will be interesting to make use of target side structure information or the syntactic labels, as well.
Thanks!