Convolutional over Recurrent Encoder for Neural Machine Translation

Praveen Dakwale and Christof Monz
Neural Machine Translation

- End to end neural network with RNN architecture where the output of an RNN (decoder) is conditioned on another RNN (encoder).

\[ p(y_i|y_1, \ldots, y_{i-1}, x) = g(y_{i-1}, s_i, c_i), \]

- \( c \) is a fixed length vector representation of source sentence encoded by RNN.

- Attention Mechanism:
  - (Bahdanau et al 2015): compute context vector as weighted average of annotations of source hidden states.

\[ c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j. \]
Convolutional over Recurrent Encoder for Neural Machine Translation

\[ C'_j = \sum_ia_{ji}C_i \]

\[ \alpha_{ji} \]

\[ X_1, X_2, X_3 \]

\[ Y_1, Y_2, Y_3 \]

\[ h_1^1, h_2^1, h_3^1 \]

\[ S_1^1, S_2^1, S_3^1 \]

\[ C_1, C_2, C_3 \]

\[ Z_1, Z_2, Z_3 \]

\[ S_{t-1}^2 \]
Why RNN works for NMT?

- Recurrently encode history for variable length large input sequences
- Capture the long distance dependency which is an important occurrence in natural language text
RNN for NMT:

- Disadvantages:
  - Slow: Doesn’t allow parallel computation within sequence
  - Non-uniform composition: For each state, first word is over-processed and the last one only once
  - Dense representation: each $h_i$ is a compact summary of the source sentence up to word ‘i’
  - Focus on global representation not on local features
CNN in NLP:

- Unlike RNN, CNN apply over a fixed size window of input
  - This allows for parallel computation
- Represent sentence in terms of features:
  - a weighted combination of multiple words or n-grams
- Very successful in learning sentence representations for various tasks
  - Sentiment analysis, question classification (Kim 2014, Kalchbrenner et al 2014)
Convolution over Recurrent encoder (CoveR):

- Can CNN help for NMT?
  - Instead of single recurrent outputs, we can use a composition of multiple hidden state outputs of the encoder
- Convolution over recurrent:
  - We apply multiple layers of fixed size convolution filters over the output of the RNN encoder at each time step
  - Can provide wider context about the relevant features of the source sentence
CoveR model

\[ y_1 \rightarrow y_2 \rightarrow y_3 \rightarrow y_j \]

\[ S_1^2 \rightarrow S_2^2 \rightarrow S_3^2 \rightarrow S_i^2 \]

\[ S_1^1 \rightarrow S_2^1 \rightarrow S_3^1 \rightarrow S_i^1 \]

\[ C_1' \rightarrow C_2' \rightarrow C_3' \rightarrow C_i' \]

\[ \alpha_{jn} \]

\[ C_j = \Sigma \alpha_{jn} CN_i \]

\[ S_{t-1}^2 \]

\[ Z_1 \rightarrow Z_2 \rightarrow Z_3 \rightarrow Z_i \]

\[ CN_1^2 \rightarrow CN_2^2 \rightarrow CN_3^2 \rightarrow CN_i^2 \]

\[ pad_0 \]

\[ CN_1^1 \rightarrow CN_2^1 \rightarrow CN_3^1 \rightarrow CN_i^1 \]

\[ pad_0 \]

\[ h_1^2 \rightarrow h_2^2 \rightarrow h_3^2 \rightarrow h_i^2 \]

\[ pad_0 \]

\[ h_1^1 \rightarrow h_2^1 \rightarrow h_3^1 \rightarrow h_i^1 \]

\[ X_1 \rightarrow X_2 \rightarrow X_3 \rightarrow X_i \]
Convolution over Recurrent encoder:

- Each of the vectors $CN_i$ now represents a feature produced by multiple kernels over $h_i$

$$CN_i^1 = \sigma(\theta \cdot h_{i-[\frac{w-1}{2}]:i+[(w-1)/2]} + b)$$

- Relatively uniform composition of multiple previous states and current state.
- Simultaneous hence faster processing at the convolutional layers
Related work:

- Gehring et al 2017:
  - Completely replace RNN encoder with CNN
  - Simple replacement doesn’t work, position embeddings required to model dependencies
  - Require 6-15 convolutional layers to compete 2 layer RNN

- Meng et al 2015:
  - For Phrase-based MT, use CNN language model as additional feature
Experimental setting:

- Data:
  - WMT-2015 En-De training data: 4.2M sentence pairs
  - Dev: WMT2013 test set
  - Test: WMT2014, WMT2015 test sets
- Baseline:
  - Two layer unidirectional LSTM encoder
  - Embedding size, hidden size = 1000
  - Vocab: Source: 60k, Target: 40k
Experimental setting:

- CoveR:
  - Encoder: 3 convolutional layers over RNN output
  - Decoder: same as baseline
  - Convolutional filters of size: 3
  - Output dimension: 1000
  - Zero padding on both sides at each layer, no pooling
  - Residual connection (He et al. 2015) between each intermediate layer
Experimental setting:

- Deep RNN encoder:
  - Comparing 2 layer RNN encoder baseline to CoveR is unfair
    - Improvement maybe just due to increased number of parameters
  - We compare with a deep RNN encoder with 5 layers
  - 2 layers of decoder initialized through a non-linear transformation of encoder final states
Result

<table>
<thead>
<tr>
<th>BLEU</th>
<th>Dev</th>
<th>wmt14</th>
<th>wmt15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>17.9</td>
<td>15.8</td>
<td>18.5</td>
</tr>
<tr>
<td>Deep RNN encoder</td>
<td>18.3</td>
<td>16.2</td>
<td>18.7</td>
</tr>
<tr>
<td>CoveR</td>
<td>18.5</td>
<td>16.9*</td>
<td>19.0*</td>
</tr>
</tbody>
</table>

* Compared to baseline:
  - +1.1 for WMT-14 and 0.5 for WMT-15

* Compared to deep RNN encoder:
  - +0.7 for WMT-14 and 0.3 for WMT-15
Convolutional over Recurrent Encoder for Neural Machine Translation

Result

<table>
<thead>
<tr>
<th>BLEU</th>
<th>#parameters (millions)</th>
<th>avg sec/sent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>174</td>
<td>0.11</td>
</tr>
<tr>
<td>Deep RNN encoder</td>
<td>283</td>
<td>0.28</td>
</tr>
<tr>
<td>CoveR</td>
<td>183</td>
<td>0.14</td>
</tr>
</tbody>
</table>

- CoveR model:
  - Slightly slower than baseline but faster than deep RNN
  - Slightly more parameter than baseline but less than deep RNN
  - Improvements not just due to increased number of parameters
Qualitative analysis:

* Increased output length

<table>
<thead>
<tr>
<th>Source</th>
<th>as the reverend martin luther king jr. said fifty years ago</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference</td>
<td>wie pastor martin luther king jr. vor fünfzig jahren sagte</td>
</tr>
<tr>
<td>Baseline</td>
<td>wie der martin luther king jr. sagte</td>
</tr>
<tr>
<td>Cover</td>
<td>wie der martin luther king jr. sagte <strong>vor fünfzig jahren</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BLEU</th>
<th>Avg sent length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>18.7</td>
</tr>
<tr>
<td>Deep RNN</td>
<td>19.0</td>
</tr>
<tr>
<td>CoveR</td>
<td><strong>19.9</strong></td>
</tr>
<tr>
<td>Reference</td>
<td>20.9</td>
</tr>
</tbody>
</table>

* With additional context, CoveR model generates complete translation
Qualitative analysis:

* More uniform attention distribution

Source: he said the itinerary is still being worked out.
Reference: er sagte, das genaue reiseroute werde noch ausgearbeitet.
Baseline: er sagte, dass die strecke noch <unk> ist.
Cover: er sagte, die reiseroute wird noch ausgearbeitet.

* Generation of correct composite word
Qualitative analysis:

- More uniform attention distribution

Baseline translates: ‘itinerary’ to ‘strecke’ (road, distance)

- Pays attention only to ‘itinerary’ for this position

Cover translates: ‘itinerary’ to ‘reiseroute’

- Also pays attention to final verb
Conclusion:

- CoveR: multiple convolutional layers over RNN encoder
- Significant improvements over standard LSTM baseline
- Increasing LSTM layers improves slightly, but convolutional layers perform better
- Faster and less parameters than fully RNN encoder of same size
- CoveR model can improve coverage and provide more uniform attention distribution
Thanks

Questions ?