A Pruned RNNLM Lattice-Rescoring Algorithm for Automatic Speech Recognition

Hainan Xu\textsuperscript{1,2}, Tongfei Chen\textsuperscript{1}, Dongji Gao\textsuperscript{1}, Yiming Wang\textsuperscript{1}, Ke Li\textsuperscript{1}, Nagendra Goel\textsuperscript{3}, Yishay Carmiel\textsuperscript{2}, Daniel Povey\textsuperscript{1}, Sanjeev Khudanpur\textsuperscript{1}

\textsuperscript{1}Center for Language and Speech Processing, Johns Hopkins University; \textsuperscript{2}IntelligentWire, Seattle WA; \textsuperscript{3}Go-Vivace Inc., USA

Overview

- Usually lattice-rescoring uses \textit{n}-gram approximation to limit search space;
- We propose a heuristics that finds more promising arcs to expand, and use it for pruning;
- Complexity of the algorithm grows approximately (empirically) linear with \textit{n}-gram order, compared with exponential growth of the baseline algorithm;
- \textit{n}X and \textit{10}X faster for 4-gram and 5-gram;
- The heuristics also consistently improves WER;
- The evaluation is done with TensorFlow to Kaldi.

Lattice Rescoring

- In speech recognition, decoding is usually done on a static decoding graph compiled from an \textit{n}-gram;
- RNNLM rescoring helps further reduce WERs by (partially) replacing LMs weights on a decoded lattice;
- A naive implementation to rescore the lattice is lattice;
- \textit{n}-gram approximation algorithm is commonly used in order to limit the search space.

Pruned Algorithm

- For each arc to be expanded, we compute a score reflecting how likely this arc will become part of the best-path;
- Arcs that are not very promising (out of the beam) are not expanded;
- Arcs that are more promising get expanded first, so that output lattice states encode “better” history.

Heuristic

- The heuristic is computed as
  \[ H(c) = \alpha(c) + \beta(a) + \delta(c) \]  
  \( \alpha(c) \) is the forward-cost for \( c \) in the output lattice; 
  \( \beta(a) \) is the backward-cost for \( a \) in the input lattice; 
  \( \delta(c) \) is an “expectation” of \( \beta(c) - \beta(a) \)

\begin{equation}
\delta(c) = \frac{\beta(c) - \beta(a)}{\beta(\text{prev}(c))} = \begin{cases} 
\frac{\beta(c) - \beta(a)}{\beta(\text{prev}(c))} & \text{if } \beta(\text{prev}(c)) < +\infty \\
+\infty & \text{if } \beta(\text{prev}(c)) = +\infty
\end{cases}
\end{equation}

- \( \text{prev}(c) \) is the previous state of \( c \) on the best path from start to \( c \).

Word-error-rate

<table>
<thead>
<tr>
<th>Corpus</th>
<th>Test set</th>
<th>ARPA baseline</th>
<th>RNNLM rescoring with \textit{n}-gram approximation</th>
<th>2-gram</th>
<th>3-gram</th>
<th>4-gram</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMI-HM</td>
<td>dev</td>
<td>24.2</td>
<td>24.1</td>
<td>24.0</td>
<td>23.7</td>
<td>23.4</td>
</tr>
<tr>
<td>(0.5)</td>
<td>eval</td>
<td>25.4</td>
<td>25.8</td>
<td>25.0</td>
<td>24.6</td>
<td>24.4</td>
</tr>
<tr>
<td>SWBD</td>
<td>swbd</td>
<td>8.1</td>
<td>8.6</td>
<td>8.2</td>
<td>7.4</td>
<td>7.2</td>
</tr>
<tr>
<td>(0.8)</td>
<td>eval2000</td>
<td>12.4</td>
<td>12.9</td>
<td>12.3</td>
<td>11.7</td>
<td>11.5</td>
</tr>
<tr>
<td>WSJ</td>
<td>dev93</td>
<td>7.6</td>
<td>7.2</td>
<td>6.9</td>
<td>6.4</td>
<td>6.2</td>
</tr>
<tr>
<td>(0.8)</td>
<td>eval92</td>
<td>5.1</td>
<td>4.6</td>
<td>4.2</td>
<td>4.1</td>
<td>3.9</td>
</tr>
<tr>
<td>LIB</td>
<td>test-clean</td>
<td>6.0</td>
<td>5.5</td>
<td>5.1</td>
<td>4.9</td>
<td>4.8</td>
</tr>
<tr>
<td>(0.5)</td>
<td>dev-clean</td>
<td>5.7</td>
<td>5.0</td>
<td>4.8</td>
<td>4.4</td>
<td>4.3</td>
</tr>
<tr>
<td>dev-other</td>
<td>14.5</td>
<td>13.7</td>
<td>12.9</td>
<td>12.0</td>
<td>11.9</td>
<td>11.7</td>
</tr>
</tbody>
</table>

Table 1: WER of Lattice-rescoring of Different RNNLMs

Contact Information

- Web: http://www.hainanxv.com
- Email: hxxi31@jhu.edu

Acknowledgements

This work was partially supported by DARPA LORELEI award number HR0011-15-2-0024. NSF Grant No CRI-1513128 and IARPA MATERIAL award number FA8650-17-C-9115 and by IntelligentWire. The authors would also like to thank the TensorFlow team at Google for their help during the project.

Analysis of Old Algorithm

(c)(a) (b)

Lattice-rescoring Speed

Output Lattice Size

(arcs per frame)