Problem

Question answering is a machine comprehension task where a passage and a question are provided, and the system must point to the set of words in the passage that answer the question. This task is an important research problem, as performance on these problems measures progress of AI systems toward natural language understanding. Applications include information retrieval and automated customer service dialog. Most approaches to this problem use RNNs to model the passage and question, perform attention to let these sequence representations interact, and then use a pointer network to point to the answer in the passage.

Data & Task

The SQuAD 2.0 dataset is a collection of passages from Wikipedia, and questions about the passages. The task is to identify the answer to the question as a span in the passage, or return "no answer" if the question has no answer. Example:

Steam engines are external combustion engines, where the working fluid is separate from the combustion products. Non-combustion heat sources such as solar power, nuclear power or geothermal energy may be used. The ideal thermodynamic cycle used to analyze this process is called the Rankine cycle. In the cycle, water is heated and transforms into steam within a boiler operating at a high pressure. When expanded through pistons or turbines, mechanical work is done. The reduced-pressure steam is then condensed and pumped back into the boiler.

Q: What ideal thermodynamic cycle analyzes the process by which steam engines work?
A: Rankine Cycle

Q: Along with geothermal and nuclear, what is a notable combustion heat source?
A: No answer

This dataset has 150,000 similar question-answer pairs.

Approach

Baseline model is BiDAF [1].
- RNN contextual encoders for passage/query
- Pointer network output

This model features:
- Bidirectional dot-product attention
- No character embedding

Final model has several added features from R-net [2]:
- More expressive additive gated attention
- Self-attention
- Condition pointer network on attention-pooled query
- Character embeddings

Results & Analysis

<table>
<thead>
<tr>
<th>Model</th>
<th>EM</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline (BiDAF)</td>
<td>55.99 (-)</td>
<td>59.29 (-)</td>
</tr>
<tr>
<td>+ Character Embedding</td>
<td>57.23 (+1.24)</td>
<td>61.34 (+2.05)</td>
</tr>
<tr>
<td>+ R-net attention</td>
<td>57.12 (-0.11)</td>
<td>61.31 (-0.03)</td>
</tr>
<tr>
<td>+ Self-Attention</td>
<td>59.56 (+2.44)</td>
<td>63.03 (+1.72)</td>
</tr>
<tr>
<td>+ R-net output (R-net)</td>
<td>60.11 (+1.65)</td>
<td>63.62 (+0.59)</td>
</tr>
</tbody>
</table>

Figure 1: The baseline model, BiDAF [1]. Character embeddings not used.

Figure 2: The final model, R-net [2].

Figure 3: Columns show attention coefficients at a position in the self-attention layer.

Conclusions

- More involved and expressive varieties of attention do bring enhanced performance.
- Character embeddings and self-attention bring the largest performance gains to the BiDAF model.
- Attention-heavy non-RNN-based models may be a better choice, as RNN's lead to substantially slower execution.

References