SEARNN: Training RNNs with Global-Local Losses
Rémi Leblond*, Jean-Baptiste Alayrac*, Anton Osokin, Simon Lacoste-Julien
INRIA/ENS, Paris, France MILA/IRIDO UdeM
* equal contribution

Summary

Some known problems when training RNNs:
RNNs are usually trained using Maximum Likelihood Estimation (MLE) i.e., Teacher Forcing. Issues with this surrogate:
- Different from the test loss
- All-or-Nothing loss
- Teacher forcing Exposure bias

Can we build on existing approaches?
- Existing approaches use ideas from Reinforcement Learning to tackle these problems.
- Instead we use ideas from Structured Prediction: we revisit the Learning to Search literature, in particular the SEARN (or LOLS) approach [2, 7].

Contributions:
- Link between RNNs and Learning To Search
- Introduction of a new algorithm: SEARNN [3]
- Experiments on real structured prediction tasks with substantial improvement over MLE

Learning to Search

Structured prediction
Learn a mapping \( f \) between inputs \( X \) and structured outputs \( Y \) made of interrelated parts often subject to constraints.

Learning To Search (L2S)
Reduces the structured problem down to cost-sensitive classification with theoretical guarantees.

How does it work?
A unique shared classifier makes predictions one by one, conditioned on the input and the previous tokens. This classifier is trained on an intermediate dataset.

Links between Learning to Search and RNNs
- Decomposition of structured tasks in sequential predictions conditioned on the past
- Unique shared classifier for all decisions using predecessors

SEARNN [3]

Overview:
Integrate roll-outs in the decoder to compute the cost of every possible action at every step.
Leverage these costs to enable better training losses.

Algorithm
1. Compute costs with roll-in/outs
2. Derive a loss from the costs
3. Use the loss to take a gradient step
4. Rinse and repeat

The devil in the details [7]
Which cost-sensitive loss? How can we scale?

Why is it better than MLE?
Makes direct use of the evaluation metric, and leverages structured information by comparing costs, contrary to MLE.
Global-local losses, with global information at each local cell, vs.
MLE (local information) and RL approaches (global loss).

Experiments

Full algorithm:

<table>
<thead>
<tr>
<th>Dataset</th>
<th>A</th>
<th>T</th>
<th>Cost</th>
<th>MLE</th>
<th>roll-in/outs</th>
<th>learned</th>
<th>reference</th>
<th>learned</th>
<th>reference</th>
<th>learned</th>
<th>reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>OCR</td>
<td>26</td>
<td>15</td>
<td>2.9</td>
<td>1.9</td>
<td>2.5</td>
<td>1.8</td>
<td>1.9</td>
<td>2.4</td>
<td>1.9</td>
<td>2.4</td>
<td>1.9</td>
</tr>
<tr>
<td>CoNLL</td>
<td>22</td>
<td>70</td>
<td>4.2</td>
<td>3.7</td>
<td>4.6</td>
<td>4.1</td>
<td>4.1</td>
<td>4.6</td>
<td>4.5</td>
<td>4.5</td>
<td>4.5</td>
</tr>
<tr>
<td>Spelling</td>
<td>0.2</td>
<td>43</td>
<td>10.8</td>
<td>10.8</td>
<td>11.4</td>
<td>11.0</td>
<td>11.0</td>
<td>11.4</td>
<td>11.0</td>
<td>11.0</td>
<td>11.0</td>
</tr>
</tbody>
</table>

Scaling approach:

<table>
<thead>
<tr>
<th>Dataset</th>
<th>MLE</th>
<th>LL</th>
<th>LE</th>
<th>LUCAS</th>
</tr>
</thead>
<tbody>
<tr>
<td>OCR</td>
<td>2.4</td>
<td>1.9</td>
<td>1.9</td>
<td>1.9</td>
</tr>
<tr>
<td>CoNLL</td>
<td>3.3</td>
<td>2.9</td>
<td>2.9</td>
<td>2.9</td>
</tr>
<tr>
<td>Spelling</td>
<td>5.3</td>
<td>5.1</td>
<td>5.1</td>
<td>5.1</td>
</tr>
</tbody>
</table>

Takeaways:
- Significant improvements over MLE on all 4 tasks.
- The harder the task, the bigger the improvement.
- Learned/mixed is the best strategy for roll-in/out.
- The best performing losses (for now) are structurally close to MLE.
- SeaRNN does not require warm start.
- The proposed sampling strategy works, maintaining improvements at a fraction of the cost.

Machine Translation (in progress): ISWL 14 Ger/Eng

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>17.7</td>
<td>20.7</td>
<td>22.5</td>
<td>23.8</td>
<td>22.7</td>
<td>24.5</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>27.5</td>
<td>27.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>28.0</td>
</tr>
</tbody>
</table>

References


Check out our project webpage for code/data!