A. Mirhoseini et al.: A Hierarchical Mode for Device Placement, 2018

Presented by: Balázs Tóth
- Device placement important for deep learning models
  - Image classification
  - Speech recognition
  - Machine translation

- Can be framed as a graph partitioning problem
  - Scotch (Pellegrini, 2009), an open-source graph partitioner used as baseline
Prior work

- Neural networks and reinforcement learning for combinatorial optimization
  - Vinyals et al., (2015); Bello et al., (2016)
- Reinforcement learning to optimize system performance.
  - Mao et al. (2016) train a resource management algorithm with policy gradients
- First paper by Mirhoseini et al., (2017)
  - Uses a RNN policy network to predict operation placements
  - Only works for small (<1000 nodes) computation graphs
  - Requires manual human-expert co-locations
A. Mirhoseini et al.: Device Placement Optimization with Reinforcement Learning, 2017

- RL based placement model
- Generates placement
- Executes it on hardware
- Updates policy based on running time based reward
A. Mirhoseini et al.: Device Placement Optimization with Reinforcement Learning, 2017

- Sequence-to-sequence model with LSTM and a content-based attention mechanism to predict the placements
- Operations embedded, then at each time step a device is predicted
- Requires prior co-location
New architecture

- Run feed forward Grouper before the sequential Placer.
Reward maximisation

- Goal is to maximise

\[ J(\theta_g, \theta_d) = \mathbb{E}_{P(d; \theta_g, \theta_d)}[R_d] = \sum_{g \sim \pi_g} \sum_{d \sim \pi_d} p(g; \theta_g)p(d|g; \theta_d)R_d \]

- Use policy gradients achieved by drawing placement samples
  - \( m=1 \) Grouper samples
  - \( k=4 \) Placer samples
- Use Adam (Kingma & Ba, 2015) optimizer
- Use distributed training
# Results

## Model runtimes for different placements

<table>
<thead>
<tr>
<th>Tasks</th>
<th>CPU Only</th>
<th>GPU Only</th>
<th>#GPUs</th>
<th>Human Expert</th>
<th>Scotch</th>
<th>MinCut</th>
<th>Hierarchical Planner</th>
<th>Runtime Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inception-V3</td>
<td>0.61</td>
<td>0.15</td>
<td>2</td>
<td>0.15</td>
<td>0.93</td>
<td>0.82</td>
<td>0.13</td>
<td>16.3%</td>
</tr>
<tr>
<td>ResNet</td>
<td>-</td>
<td>1.18</td>
<td>2</td>
<td>1.18</td>
<td>6.27</td>
<td>2.92</td>
<td>1.18</td>
<td>0%</td>
</tr>
<tr>
<td>RNNLM</td>
<td>6.89</td>
<td>1.57</td>
<td>2</td>
<td>1.57</td>
<td>5.62</td>
<td>5.21</td>
<td>1.57</td>
<td>0%</td>
</tr>
<tr>
<td>NMT (2-layer)</td>
<td>6.46</td>
<td>OOM</td>
<td>2</td>
<td>2.13</td>
<td>3.21</td>
<td>5.34</td>
<td>0.84</td>
<td>60.6%</td>
</tr>
<tr>
<td>NMT (4-layer)</td>
<td>10.68</td>
<td>OOM</td>
<td>4</td>
<td>3.64</td>
<td>11.18</td>
<td>11.63</td>
<td>1.69</td>
<td>53.7%</td>
</tr>
<tr>
<td>NMT (8-layer)</td>
<td>11.52</td>
<td>OOM</td>
<td>8</td>
<td><strong>3.88</strong></td>
<td>17.85</td>
<td>19.01</td>
<td>4.07</td>
<td>-4.9%</td>
</tr>
</tbody>
</table>
Results

Hierarchical Planner’s placement of a NMT (4-layer) model
Results

Policy training results with 1 and 4 workers
Opinion

- **The placement takes significant time**
  - Newer approaches run considerably quicker producing comparable or better results
  - However, received a lot of citations and seems to have been novel in the field

- **Evaluation felt lacking**
  - Only tested on 1 architecture, using 1 CPU and 2/4/8 GPU
  - The optimal placement on a few of the models is the trivial GPU-only one
  - Missing comparison against their own previous paper?

- **Decisions made not explained well**
  - Arbitrary hyperparameters used
  - Embedding seems somewhat weird
References

- F. Pellegrini: Distilling knowledge about Scotch, 2009
- I. Bello et al.: Neural combinatorial optimization with reinforcement learning, 2016
- A. Mirhoseini et al.: Device Placement Optimization with Reinforcement Learning, 2017
Discussion