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

Presented by: Balázs Tóth

# Background

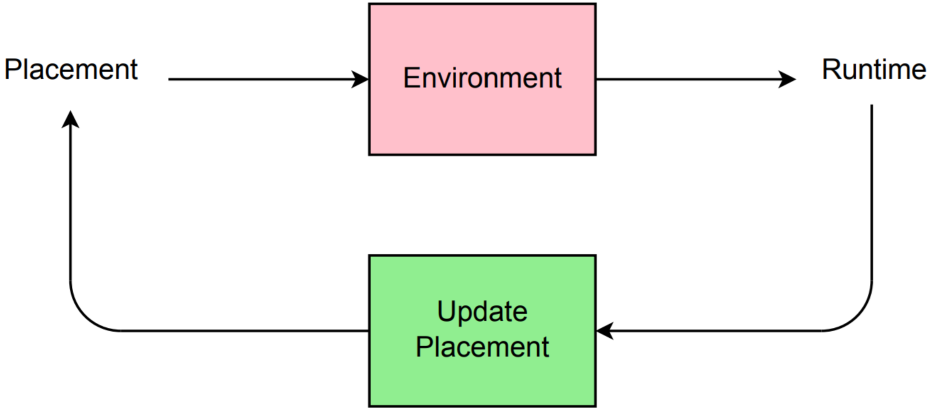
- Device placement important for deep learning models
  - Image classification
  - Speech recognition
  - Machine translation
- Can be framed as a graph partitioning problem
  - Scotch (Pellegrini, 2009), an open-source graph partitioner used as baseline

# 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

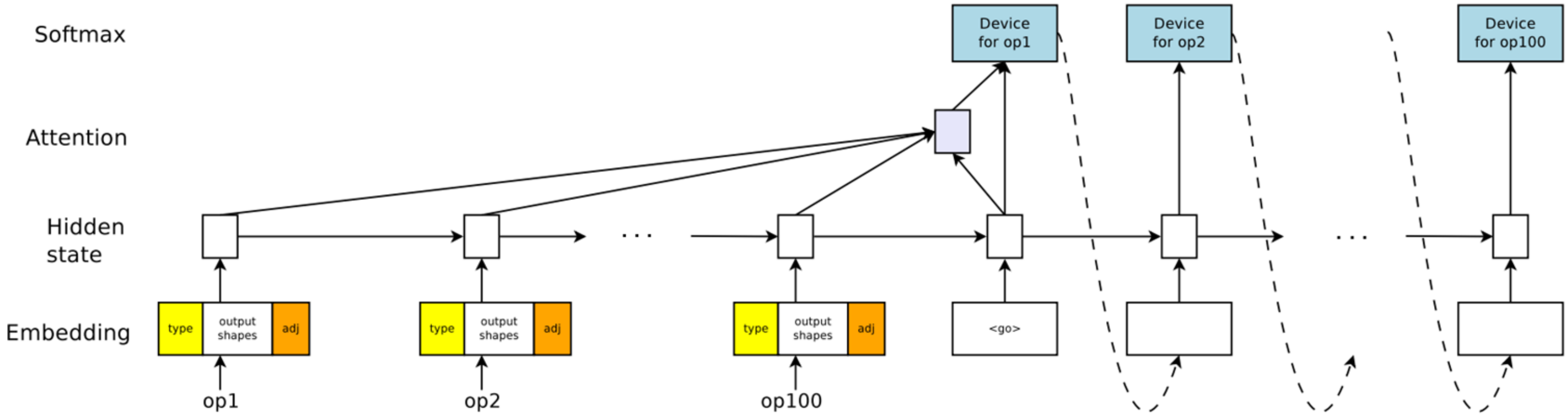
# A. Mirhoseini et al.: Device Placement Optimization with Reinforcement Learning, 2017

- RL based placement model
- Generates placement
- Executes it on hardware
- Updates policy based on running time based reward



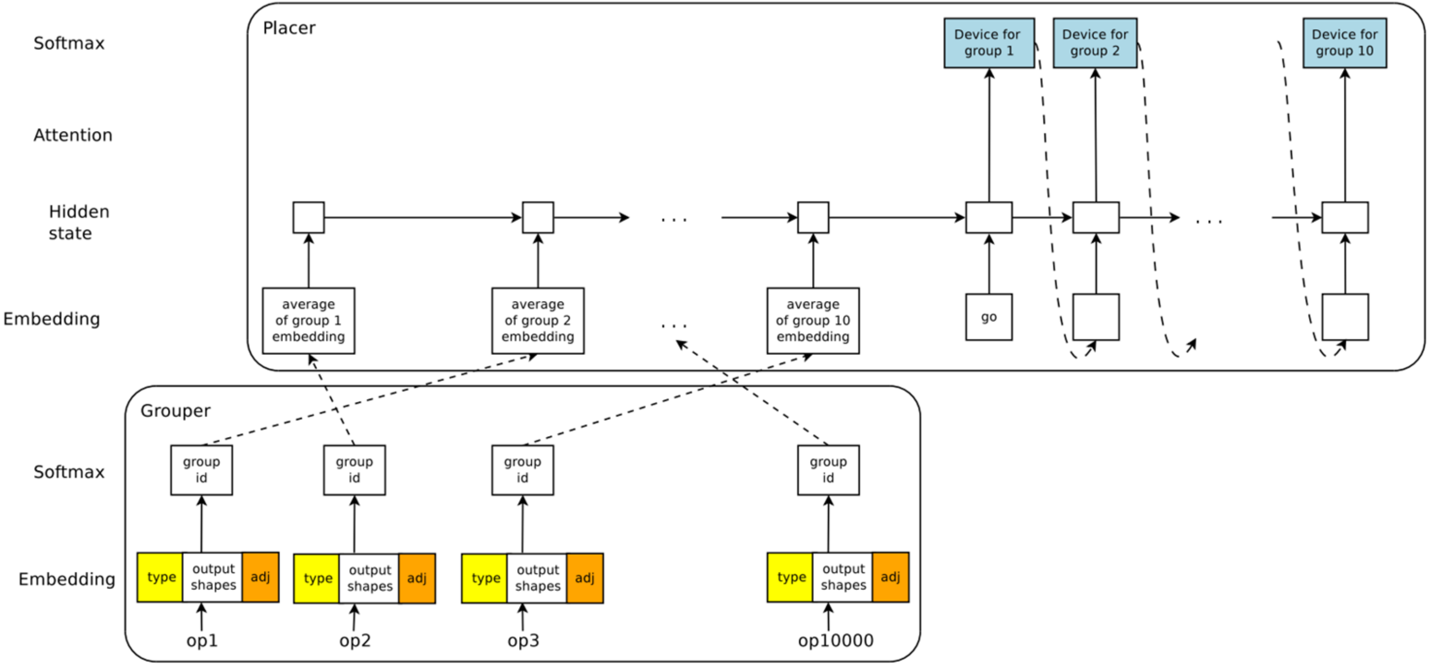
# A. Mirhoseini et al.: Device Placement Optimization with Reinforcement Learning, 2017

- 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.

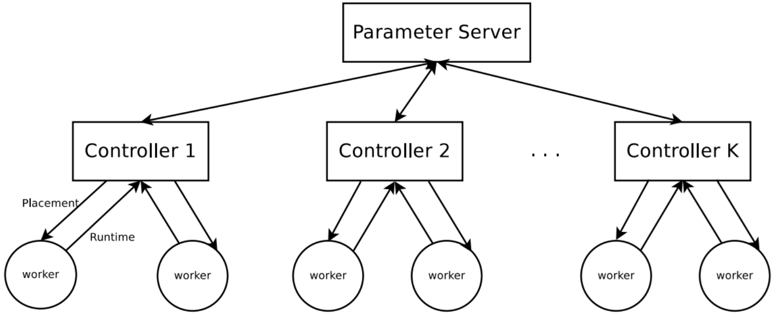


# Reward maximisation

- Goal is to maximise

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

- 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

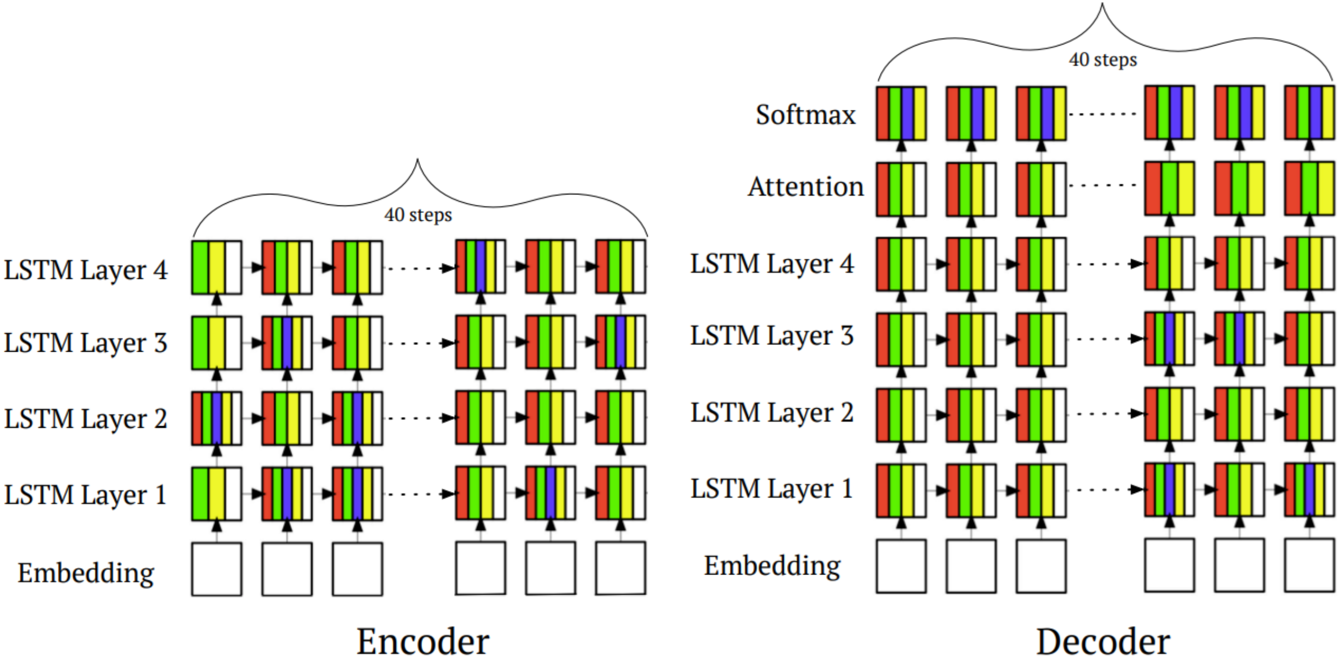


# Results

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

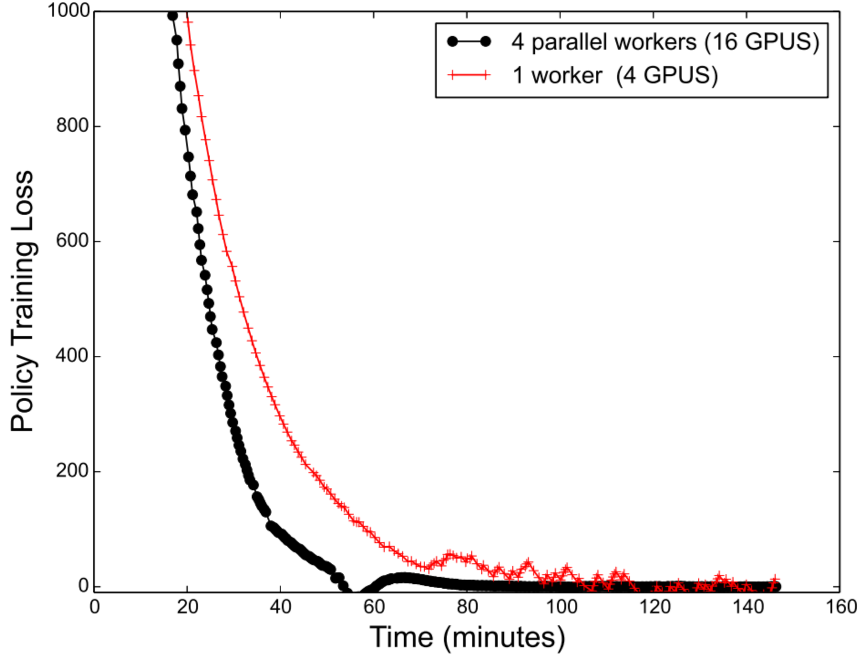
Model runtimes for different placements

# 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: Distillating knowledge about Scotch, 2009
- O. Vinyals et al.: Pointer networks, 2015
- I. Bello et al.: Neural combinatorial optimization with reinforcement learning, 2016
- H. Mao et al.: Resource Management with Deep Reinforcement Learning, 2016
- A. Mirhoseini et al.: Device Placement Optimization with Reinforcement Learning, 2017

# Discussion