Population Based Training of Neural Networks

M. Jaderberg, V. Dalibard, S. Osindero, W.M. Czarnecki
November 14, 2018

DeepMind, London, United Kingdom

Presented by: Devin Taylor
Problem statement

Neural networks suffer from sensitivity to empirical choices of hyperparameters

Solution

Asynchronous optimisation algorithm that jointly optimises a population of models
Figure 1: Overview of proposed approach
Population Based Training - Algorithm

- **step** - weight update
- **eval** - performance evaluation
- **ready** - current path limit
- **exploit** - compare to population
- **explore** - adjust hyperparameters

**Figure 2:** PBT algorithm

**Algorithm 1** Population Based Training (PBT)

```
1: procedure TRAIN(\mathcal{P})
2: for (\theta, h, p, t) \in \mathcal{P} \text{ (asynchronously in parallel)} do
3:     while not end of training do
4:         \theta \leftarrow \text{step}(\theta | h)
5:         p \leftarrow \text{eval}(\theta)
6:         if ready(p, t, \mathcal{P}) then
7:             h', \theta' \leftarrow \text{exploit}(h, \theta, p, \mathcal{P})
8:             if \theta \neq \theta' then
9:                 h, \theta \leftarrow \text{explore}(h', \theta', \mathcal{P})
10:                p \leftarrow \text{eval}(\theta)
11:            end if
12:        end if
13:    update \mathcal{P} \text{ with new } (\theta, h, p, t + 1)
14: end while
15: end for
16: return \theta \text{ with the highest } p \text{ in } \mathcal{P}
17: end procedure
```
Population Base Training - Core

- **exploit**
  - Replace weights and/or hyperparameters
  - T-test selection, truncation selection, binary tournament

- **explore**
  - Adjust hyperparameters
  - Perturb, resample

**Figure 3:** PBT dummy example
Implementation Notes

• Asynchronous
• No centralised orchestrator
• Only current performance information, weights, hyperparameters published
• No synchronisation of population
Experiments conducted in three areas:

- **Deep reinforcement learning** - Find policy to maximise expected episodic return
- **Neural machine translation** - Convert sequence of words from one language to another
- **Generative adversarial networks** - Generative models with competing components, *generator* and *discriminator*
Results - Spoiler

Figure 4: PBT result summary
Figure 5: PBT deep reinforcement learning result - DM Lab
Figure 6: PBT machine translation results
Results - Generative Adversarial Networks

Figure 7: PBT GAN results
Figure 8: PBT design space analysis
Figure 9: PBT lineage analysis
Analysis

Figure 10: PBT development as phylogenetic tree
Critique

Positives

• Well written
• Detailed analysis - although some questions left unanswered
• Result improvements without sacrificing on time
• Approximate complex paths for hyperparameter tuning
• Improved training stability

Negatives

• No results showing evidence of reduced time
• Added in additional hyperparameters (ready steps, perturb, etc)
• Is susceptible to local minima
• Minimum computational requirements (10 workers) quite large
Related Work

• Unique genetic algorithm approach to implementation - parallel and sequential
• Author: Max Jaderberg
  • Mix&Match: Agent Curricula for Reinforcement Learning - bootstrapping off simpler agents
Conclusion

• Presented algorithm that asynchronously and jointly optimises a population of models
• Obtained improved results on a range of different algorithms
• Still certain questions unanswered but still a good contribution