Multi Domain and Multi Task Deep Reinforcement Learning for Continuous Control

David C. Alvarez-Charris, Chenyang Zhao, Timothy Hospedales

University of Edinburgh

1 Research Problem and Motivation

Despite the recent success of Deep Reinforcement Learning (DRL) in areas of robotic manipulation, game playing and control (Mnih et al., 2015; Levine et al., 2016), most DRL agents are trained to maximize a unique reward function (solving a single *task*) or trained within a environment with a unique MDP structure (solving a single *domain*). This type of training is not only limited to solving one task / domain at a time but also ignores the potential benefit that training across parallel tasks and domains can have in improving learning speed, sample efficiency, and final (asymptotic) performance.

In this project we examine the research problem of how an agent trained through Reinforcement Learning can achieve high performance across multiple tasks or domains learnt in parallel (Zhang and Yang, 2017). We also analyze if this type of training poses any benefit when compared to single task learning agents. We propose a hard parameter sharing architecture with several branches capable of being trained in parallel for multi domain (MDL) and multi task (MTL) learning problems. [1]

2 Technical Contributions

The Multi Domain experiments were done in the 2D Bipedal Walker environment [2] were we explored three domains which consist in changing the magnitude of the ”wind” (constant negative horizontal force): no wind (wind=0), moderate wind (wind=1) and extreme wind (wind=2). The Multi Task experiments were carried in the Lunar Lander environment [3] were the reward function was modified in order to produce three different tasks: learning how to land in the center of the scenario (task ”goal”), landing away from the center of the scenario (task ”not goal”) and learning how to fly (task ”fly”).

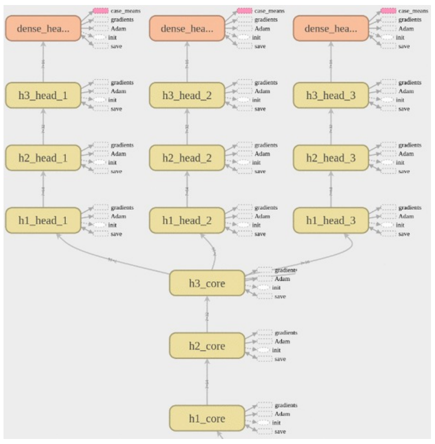

Figure 1: *Multi Headed Architecture*

The main contribution of this work is a hard parameter sharing (Ruder, 2017) neural network architecture that can be used as a policy and value function approximator when training a single agent across multiple tasks and domains. The proposed architecture denoted by Multi Headed Network is shown in **fig. 1** , the idea behind it is to have a block of layers shared across all task / domains (*Core Layers*) and several branches with layers specializing for specific feature extraction (*Head Layers*). Through this branched architecture we circumvent the common problem of catastrophic interference (McCloskey and Cohen, 1989), in which previously learnt knowledge is overwritten and forgotten when new tasks / domains are learnt. The Core layers who’s parameters are updated based on episodes from *all* tasks/domains act as a common encoder learning representations that are useful across all the different problems - for example a stable gate for walking, or a good controller

[1]The code for this project and a Demo video can be found in: `https://github.com/david1309/Multi_Task_RL` and `https://bit.ly/2OCqHsN`

[2]`https://gym.openai.com/envs/BipedalWalker-v2/`

[3]`https://gym.openai.com/envs/LunarLander-v2/`

that allows the spacecraft to be stable while in the air. The Head layers which are solely updated based on episodes sampled from each *specific* problem act as decoders which based on the common features extracted by the encoder produce appropriate actions for each individual problem - modifying the robots hull angle to produce different forward momentums for not falling under high winds or controlling the jet engines to fly or land. To produce the differentiated training of the Core and Head layers we introduce in the network's computational graph a switch-case operation controlling which gradients flow to each of the branches of the network.

To asses if the parallel task training of our proposed network had any benefits with respect to standard training methods, we created as baseline several Single Domain (SDL) and Single Task (STL) agents. Their Policy and Value function were parametrized by standard feed-forward neural networks. All the single and multi task/domain agents were trained using policy gradients based on Proximal Policy Optimization (Schulman et al., 2017) and its gradient estimators were computed using the Generalized Advantage Estimator approach (Schulman et al., 2016).

Our results shown in **Table. 1** demonstrate that the Multi Headed Network was capable of achieving a high performance across all domains and tasks in the Bipedal Walker and Lunar Lander training environments. In the first 2 domains our proposed architecture achieved a comparable performance to the SDL agents, and in the third domain it even produced a 4.981% improvement on the asymptotic reward (as measured by $\bar{R}_{10}$) and a 13.940 % improvement on the sample efficiency (as measured by $\bar{R}_{all}$) when compared to SDL agents. This demonstrates that parallel training was indeed able to produce benefits regarding performance and learning speed. For the multi task learning problem all though we did not observe any direct benefit of training across parallel tasks, the MTL agent achieved a comparable performance relative to each STL agent. Such comparable performance highlights the capability of the Multi Headed Architecture of achieving parallel learning and demonstrates that this agent is capable of learning to maximize multiple reward functions at the same time. On average, it performed 74.983% as good as the other single task agents.

Table 1: *Comparison of the proposed Multi Domain and Multi Task learning agent with respect to standard Single Domain/Task learning agents.* $\bar{R}_{all}$ *- average reward across entire training.* $\bar{R}_{10}$ *- average reward over last 10 network updates.* $\frac{\text{MDL}}{\text{SDL}}$ *- quantifies the relative final performance between agents based on the* $\bar{R}_{10}$ *score*

Domain	$\bar{R}_{all}$ (agent)	$\bar{R}_{10}$ (agent)	$\frac{\text{MDL}}{\text{SDL}}$	Best Agent
wind=0	**226.469** (SDL) 214.793 (MDL)	**287.735** (SDL) 261.712 (MDL)	90.955%	SDL
wind=1	**232.027** (SDL) 182.677 (MDL)	**263.074** (SDL) 242.836 (MDL)	92.307%	SDL
wind=2	170.746 (SDL) **194.548** (MDL)	262.385(SDL) **275.457** (MDL)	104.981%	MDL

Task	$\bar{R}_{all}$ (agent)	$\bar{R}_{10}$ (agent)	$\frac{\text{MTL}}{\text{STL}}$	Best Agent
Goal	**192.891** (STL) 134.495 (MTL)	**382.747** (STL) 298.388 (MTL)	77.959%	STL
Not Goal	**518.796** (STL) 377.908 (MTL)	**631.498** (STL) 305.070 (MTL)	48.308%	STL
Fly	**127.238** (STL) 93.731 (MTL)	**192.446** (STL) 189.911 (MTL)	98.682%	STL

References

Levine, S., Finn, C., Darrell, T., and Abbeel, P. (2016). End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373.

McCloskey, M. and Cohen, N. J. (1989). Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529.

Ruder, S. (2017). An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. (2016). High-dimensional continuous control using generalized advantage estimation. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Zhang, Y. and Yang, Q. (2017). A survey on multi-task learning. *CoRR*, abs/1707.08114.