Supplementary Material for Policy Consolidation for Continual Reinforcement Learning

A. Details of Implementation

Much of the code for the PC model was built on top of and adapted from the distributed PPO implementation in (Dhariwal et al., 2017).

A.1. Single agent experiments

For the baseline models, we mainly used the hyperparameters used for the training of Mujoco tasks in (Schulman et al., 2017). The value function network shared parameters with the policy network and no task-id input was given to the agents. As in (Dhariwal et al., 2017), the running mean and variance of the inputs was recorded and used to normalise the input to mean 0 and variance 1. The gradients are also clipped to a norm of 0.5 as in (Dhariwal et al., 2017). In (Schulman et al., 2017), different parameters were used for the Humanoid tasks as well as multiple actors - for simplicity we used the Mujoco parameters and a single actor. The hidden policies were all initialised with the same parameters as the visible policy for the PC agent, which means that the beginning of training can be slow as the agent is over-consolidated at the initial weights. This might be remedied in the future by introducing incremental flow from the deeper beakers as training progresses.

Table S1 shows a list of hyperparameters used for the experiments. In future, we would like to do a broader parameter search for both the baselines and the policy consolidation model. For this work, many more baselines were run than policy consolidation agents in the interest of fairness.

A.2. Self-play experiments

For the self-play experiments, the agents were trained for much longer than in the single agent tasks. For this reason, in order to speed up training, a number of changes were made, namely: using multiple environments in parallel to generate experience, increasing the trajectory length, increasing the minibatch size, reducing number of epochs per update. As a result of increasing the number of experiences trained on per update as well as the trajectory length, it was reasonable to expect that the variance of the updates should decrease and that short term non-stationarity is better dealt with. For this reason, we reduced $\omega_{1,2}$ and β in the PC model to allow larger updates per iteration. Additionally, we compared the PC model to a lower range of βs for the fixed-KL baselines for fairness.

The primary (sparse) reward for the RoboSumo agent was administered at the end of an episode, with 2000 for a win, -2000 for a loss and -2000 for a draw. To encourage faster learning, as in (Al-Shedivat et al., 2018) and (Bansal et al., 2018), we also trained all agents using a dense reward curriculum in the initial stages of training. We refer readers to (Al-Shedivat et al., 2018) for the details of the curriculum, which include auxiliary rewards for agents staying close to the centre of the ring and for being in contact with their opponent. Specifically, for the the first 15% of training episodes, the agent was given a linear interpolation of the dense and sparse rewards $\alpha r_{dense} + (1-\alpha) r_{sparse}$ with α being decayed linearly from 1 to 0 over the course of the first 15% of episodes until only the sparse reward was administered. Only the experiences from one of the players in each environment was used to update the agent.

B. Directionality of KL constraint

In our initial experiments we found that using a $D_{\mathrm{KL}}\left(\pi_k || \pi_{k_{old}}\right)$ constraint for each policy in the PC model, rather than the $D_{\mathrm{KL}}\left(\pi_{k_{old}} || \pi_k\right)$ constraint used in the KL versions of PPO (Schulman et al., 2017), resulted in better continual learning and so in the main results section we compared the PC model with KL baselines that also used the $D_{\mathrm{KL}}\left(\pi_k || \pi_{k_{old}}\right)$ constraint. Here we show in a few experiments that we get the same qualitative improvements from the PC agent if we use the original KL constraint from PPO for both the PC model and the baselines (Figure S1). As can be seen particularly in the HalfCheetah and Humanoid alternating task settings, the $D_{\mathrm{KL}}\left(\pi_k || \pi_{k_{old}}\right)$ version performs better.

The effect of the directionality of this KL constraint, as well as the directionality of the KL constraints between adjacent policies (of which there are four possible combinations) warrants further investigation and is an important avenue for future work.

Table S1. Hyperparameters

PARAMETER	MULTI-TASK	SINGLE TASK	SELF-PLAY
# TASK SWITCHES	19	0	0
# TIMESTEPS/TASK	1M	50M (HUMANOID) / 20M (OTHERS)	600M
DISCOUNT γ	0.99	0.99	0.995
GAE PARAMETER (λ)	0.95	0.95	0.95
HORIZON	2048	2048	8192
ADAM STEPSIZE (KTH POLICY)	$\omega^{1-k} \times 3 \times 10^{-4}$	$\omega^{1-k} \times 3 \times 10^{-4}$ OR $\omega^{1-k} \times 3 \times 10^{-5}$	$\omega^{1-k} \times 10^{-4}$
VF COEFFICIENT	0.5	0.5	0.5
# EPOCHS PER UPDATE	10	10	6
# MINIBATCHES	64	64	32
NEURON TYPE	RELU	RELU	RELU
WIDTH HIDDEN LAYER 1	64	64	64
WIDTH HIDDEN LAYER 2	64	64	64
ADAM β_1	0.9	0.9	0.9
ADAM β_2	0.999	0.999	0.999
# HIDDEN POLICIES	7	7	7
$\omega_{1,2}$	1	1	0.25
ω	4	4	4
β (POL.CONS.)	0.5	0.5	0.1
ADAPTIVE KL d_{targ}	0.01	0.01	0.01
# ENVIRONMENTS	1	1	16

C. Task switching schedule effects

Figure S2 shows the effects of changing the frequency of task switching in the alternating task setting for both the PC model and one of the baselines (fixed-KL with $\beta = 10$). An interesting point to note is that in the baseline runs with slower task-switching schedules, the performances on both tasks decrease over time, with the agent unable to reach previously attained highs. In other words, the agent not only catastrophically forgets, but learning one task puts the network in a state that it struggles to (re)learn the other task at all.

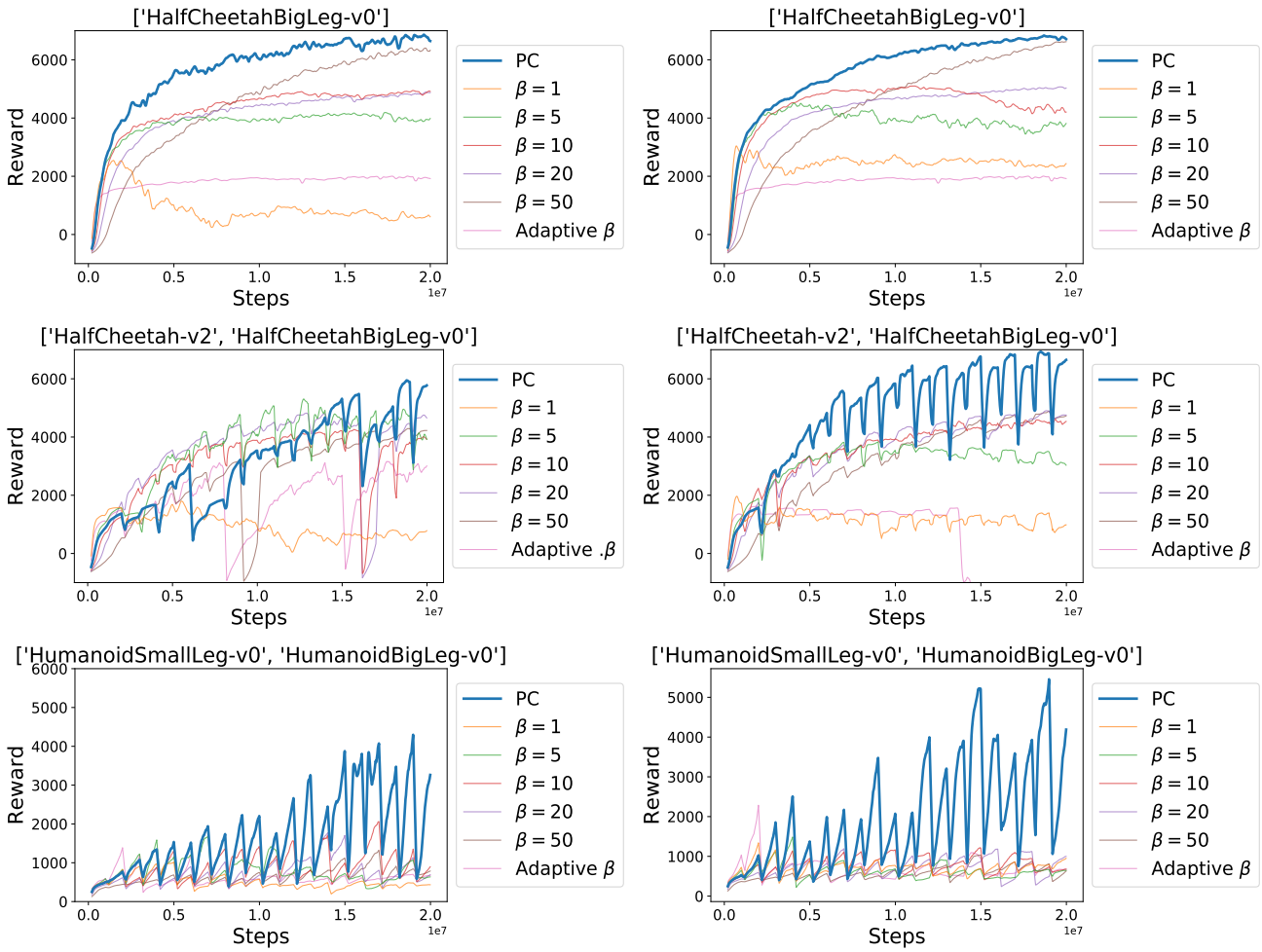
References

Al-Shedivat, M., Bansal, T., Burda, Y., Sutskever, I., Mordatch, I., and Abbeel, P. Continuous adaptation via meta-learning in nonstationary and competitive environments. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Sk2u1g-0-.

Bansal, T., Pachocki, J., Sidor, S., Sutskever, I., and Mordatch, I. Emergent complexity via multi-agent competition. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Sy0GnUxCb.

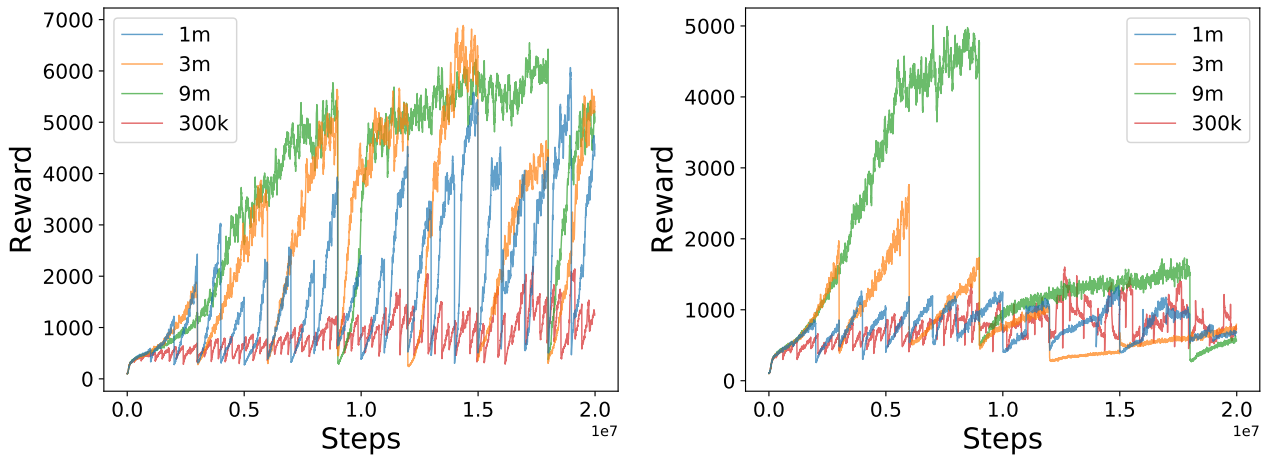
Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., Wu, Y., and Zhokhov, P. Openai baselines. https://github.com/openai/baselines, 2017.

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



(a) $D_{\mathrm{KL}}\left(\pi_{k_{old}}||\pi_k\right)$ (b) $D_{\mathrm{KL}}\left(\pi_k||\pi_{k_{old}}\right)$

Figure S1. Reward over time using the (a) $D_{\mathrm{KL}}\left(\pi_{k_{old}}||\pi_k\right)$ and (b) $D_{\mathrm{KL}}\left(\pi_k||\pi_{k_{old}}\right)$ constraints.



(a) PC model (b) Fixed-KL $\beta = 10$

Figure S2. Reward over time for (a) PC model and (b) fixed-KL baseline with $\beta = 10$ for different task-switching schedules between the HumanoidSmallLeg-v0 and HumanoidBigLeg-v0 tasks.