Bootstrap Your Own Teacher: Online Policy Distillation for Multi-Game Reinforcement Learning

1st Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

2nd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

3rd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

4th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

5th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

6th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

***Abstract*—Training generalist agents capable of performing well across diverse environments is a significant goal of reinforcement learning (RL). Current state-of-the-art methods for multi-game RL rely on offline datasets, and often discard the policy used to gather trajectories despite its potential to provide a rich learning signal. In this paper, we revisit policy distillation (PD) for multi-game RL and introduce a new framework called Bootstrap Your Own Teacher (BYOT) that extends policy distillation to the online-RL setting. BYOT alternates between two phases: (i) game-specific fine-tuning and (ii) distilling bootstrapped teachers back into a shared multi-game policy. By directly regulating the multi-game learning dynamics in policy space, BYOT balances training without explicit gradient adjustments or reward normalization, whilst being highly parameter efficient. Our framework is empirically validated for both online and offline multi-game learning on the Atari-40 benchmark. BYOT outperforms all prior online Atari-40 multi-game agents, achieving an IQM human-normalized-score (HNS) of 152.7%. Moreover, by adopting state-of-the-art PPO teacher agents—contrasting the widely-used datasets from weaker DQN agents—and policy distillation, we more than triple the leading IQM-HNS in offline settings to 369.5%, whilst using significantly fewer parameters. Overall, our results emphasise the power of distillation in multi-game settings.**

I. Introduction

Multi-game reinforcement learning (RL) is a promising approach to enable a single agent to learn to perform across multiple games and challenging scenarios simultaneously. The potential of this paradigm lies in the prospect of utilizing shared knowledge and structural similarities among games, thereby reducing the computational and memory requirements.

Multi-game RL methodologies are predominantly divided into two primary categories: offline and online learning. State-of-the-art offline methods—such as multi-game decision transformer (MGDT) [1], scaled Q-learning (Scaled QL) [2], and GATO [3]—learn from a pre-existing dataset of trajectories specific to each game. Conversely, online methods attempt to directly optimize a joint multi-game policy by directly interacting with the game environments.

However, both approaches come with challenges. Offline approaches depend heavily on a well-curated dataset of experiences, with their effectiveness often constrained by the quality of these expert trajectories. On the other hand, while online techniques can balance their own exploration and exploitation, they frequently face hindrances like negative transfer across games and conflicting gradients. Given these issues, offline RL methods are currently the state-of-the-art for multi-game RL on the Atari 40 benchmark.

In this work, we leverage the performance and stability of single-task learning to improve online multi-task agents. Concretely, we propose the Bootstrap Your Own Teacher (BYOT) framework that extends multi-game policy distillation (PD) [4], [5] to the online RL setting. BYOT repeatedly bootstraps game-specific teachers from its current parameters through fine-tuning steps, and then updates a shared multi-game policy via policy distillation. By minimizing the KL divergence between bootstrapped game-specific teachers and the shared multi-game policy, BYOT effectively balances the learning dynamics among games and reduces risks of negative transfer. Empirically, we validate BYOT in experiments using 40 Atari games. We demonstrate that BYOT utilizing PPO [6] bootstrapped teachers outperforms all previously reported online multi-game agents with an IQM human-normalized-score (HNS) of 152.7%.

Furthermore, the IQM-HNS of BYOT also surpasses those reported for the leading offline multi-game algorithms on the Atari 40 benchmark, such as scaled Q-Learning [2] and MGDT [1]. These baseline methods rely on a curated offline dataset of trajectories collected using a weaker DQN agent at various checkpoints during its training. However, noting the success of BYOT it is striking that these methods utilize only the trajectories, whilst discarding the policies themselves.

Exploring these constraints, we show that an enhanced offline dataset (comprising both trajectories and policies from state-of-the-art PPO single-task experts), combined with policy

distillation results in a substantial score improvement in the offline Atari 40 benchmark. Specifically, our offline multi-game PD registers an IQM-HNS of 369.5%, tripling the performance of Scaled QL [2] and MGDT [1], whilst using an order of magnitude fewer parameters ($\sim 5\,\text{M}$).

Overall, this work evidences our intuition that policy distillation is a powerful tool for multi-game learning. Specifically, the contributions of our paper are:

- Introduction of Bootstrap Your Own Teacher (BYOT) a multi-game RL framework that advances multi-game policy distillation in the online RL setting.
- Empirical evidence that BYOT excels over baseline multi-game agents in the online RL setting, achieving an IQM-HNS of 152.7% in the Atari 40 benchmark.
- Demonstration that the performance of offline RL methods on this benchmark can be significantly improved by considering both the trajectories and policies of state-of-the-art (PPO) teacher experts.

II. Related Work

a) Distillation for RL: Distilling knowledge from one neural network to another [7], [8] has its origins in model compression [9], where the aim is to leverage a pre-trained model in order to transfer to a smaller, equally performant model that utilises less compute.

Policy distillation was subsequently introduced by [4] and [5], which similarly seeks to transfer the policy of an expert teacher to a student. These early versions considered DQN agents and distilled the Q-value network using a supervised objective on experience gathered by the expert. Distillation can naturally be extended to policy-based or actor-critic methods [10], [11]. [12] further applied this concept by introducing reincarnating reinforcement learning, a policy-to-value distillation approach that scales and reuses computation from prior RL agents for improved training efficiency Our policy distillation implementation uses state-of-the-art PPO [6] expert teachers, scales offline-RL experiments beyond previous demonstrations and extends PD to the online RL setting.

PD distills experience collected by the teacher agent. Although this allows the student to mimic the teacher, it does so only on the teacher's state-distribution. This might introduce a distribution shift with respect to the student's behaviour. This has led to a trend to incorporate more online learning signal into the offline objective to allow the student to experience the state-distribution it is required to ultimately act under.

b) Multi-game RL: In the multi-game RL setting, a single agent is trained to play multiple games at the same time. In the online-RL setting this includes approaches such as MT-IMPALA [13], PopArt [14] and others [15], [16]. Recently, similar approaches have been developed in the offline domain, which train the agent entirely from offline trajectories such as multi-game decision transformer (MGDT) [1], scaled Q-learning (Scaled QL) [2], and GATO [3]. These methods all rely on curated offline datasets consisting of trajectories, often mixing trajectories collected at numerous stages during the training of a DQN agent, whilst noticeably discarding the policy that generated them. In contrast, our policy distillation implementation stores the policy (or logits) of the final expert teacher. Similarly, there has been a particular focus using large sequence models [1], [3] that predict optimal actions and rewards. These models commonly use large transformer architectures, often with separate policy heads for each game.

Multi-game RL introduces many potential complications such as catastrophic forgetting [17] or negative interference between games [18], [19]. Seminal works such the proposed PCGrad [20], uses gradient manipulation techniques to alleviate conflicting gradients between games. In contrast, BYOT is highly parameter efficient and by directly distilling policies, it does not require reward normalization or gradient manipulations. When gradients conflict, the model faces difficulty in optimizing its policy for all games simultaneously, as improving the performance in one task may hinder the performance in another. [21] revealed that in multi-task RL, pretraining on multiple game variants significantly enhances generalization capabilities and helps to limit negative transfer. Contrasting with our methodology, their approach focuses on training across variants of a single task, rather than spanning multiple, distinct tasks. Previous work, such as Distral [22], has shown that it is possible to provide positive transfer between related tasks via distillation. Distral learns multiple single-game agents simultaneously whilst using policy distillation to-and-from a master policy that acts as a regularizer for common strategies to be shared. However, their goal is to share learned knowledge between agents, rather than learning a shared multi-game agent.

Our focus on Multi-Game RL is distinct from recent advances in Meta RL. Meta RL emphasizes algorithms that swiftly adapt to new tasks via previous learning experiences, diverging from Multi-Game RL's aim of training a single agent to proficiently manage multiple, distinct games or tasks concurrently [23].

For example, Ada [24] utilizes Meta RL methods such as learning an automated curricula and in-context learning for fast adaptation to new tasks, and [25] improves Meta RL via adversarial environments, enhancing out-of-distribution task performance.

III. Preliminaries

a) Multi-Game RL: In RL, we can represent an environment using a finite-horizon Markov decision process (MDP) [26], $M = (S, A, P, R, \gamma)$, where $s \in S$ represents the possible state space, $a \in A$ the possible actions available to the agent, $P(s_{t+1}|s_t, a_t)$ is the transition probability from state s_t to s_{t+1} after taking action a_t, $R(s, a)$ is the reward function, and γ is the discount factor. The objective of a single-task agent is to maximise the expected discounted return, given by:

$$\mathbb{E}_{a_t \sim \pi(\cdot|s_t)}[\sum_{t=0}^{H} \gamma^t R(s_t, a_t)], \tag{1}$$

where H is the horizon. Commonly, *multi-task* RL [16] describes an agent that can solve multiple tasks within a single environment. This can be formulated as a single MDP, with a single shared state space, shared action space, and multiple

reward functions, e.g. one per task. In this paper, we make the distinction to *multi-game* RL, i.e. where a single agent is required to perform tasks with different environment dynamics also. One popular multi-game benchmark is the ALE suite [27], where each game can have a distinct state space, action space *and* reward function.

In the multi-game setting, we are provided with a set of N environments $E = \{e_1, \ldots, e_N\}$ that we represent as a set of MDPs, $\{M_1, \ldots, M_N\}$, with a distribution over the possible MDPs, $p(M)$. The traditional multi-task RL objective extends Equation 1 to maximise the expected discounted return over the distribution of environments, i.e. $\mathbb{E}_{M\sim p(M)}\left[\mathbb{E}_{a_t\sim\pi(.|M)}\left[\sum_{t=0}^{H}\gamma^t R(s_t,a_t)\right]\right]$. However, this adds additional considerations of how to best handle the different reward scales of each game. Depending on the nature of the games, it may be beneficial to scale rewards according to difficulty or priority. Whilst a truly optimal joint policy will provide maximized rewards for each game independently, practical optimisation of the above objective will weight the relative importance of each game according to the reward scales. Whilst explicit crafting of these rewards can enable direct control over these trade-offs, BYOT instead uses a distillation loss (detailed in the next section) to automatically treat all games with equal importance without explicit reward scaling.

Offline-RL agents usually learn using previously collected trajectories stored in a dataset or replay buffer [2], [12], [28]. Whilst considered a safe method of training physical agents, this presents obvious limitations on the ability to explore the environment during training. Game-specific policies that already exhibit expert-level performance can be utilized or, as we introduce in Section IV, can be bootstrapped from a current policy during the training process and fine-tuned on a specific game in an online setting.

b) Policy Distillation: An expert-level teacher policy π_T can be used to generate trajectories and perform traditional policy distillation [4], [5] into a student policy, π_S. Traditionally, this has been a pre-trained DQN agent with expert-level Q-function. A supervised objective is introduced over the dataset D^M, to distill from one policy to another, i.e. $\mathcal{L}(\theta_{\pi^S}) = \mathbb{E}_{s_t\sim D^M}[\mathrm{KL}(\pi^T\|\pi^S)]$, where KL is the Kullback–Leibler divergence.

IV. Method: Bootstrap Your Own Teacher

We introduce Bootstrap Your Own Teacher (BYOT), a parameter-efficient, multi-game learning framework that extends multi-game policy distillation to the online-RL setting by repeatedly bootstrapping and fine-tuning single-task teachers during the training process. We demonstrate that direct policy distillation (in the offline-RL setting), is a special case of BYOT.

We define a *bootstrapped teacher* as: (i) a policy that is spawned during training and initialized using the current multi-game policy parameters; and (ii) fine-tuned on a specific game and then leveraged for policy distillation. The intuition is that these teachers return slightly improved rewards than the shared multi-game policy in each environment, which provides a consistent policy improvement mechanism to the shared multi-game agent.

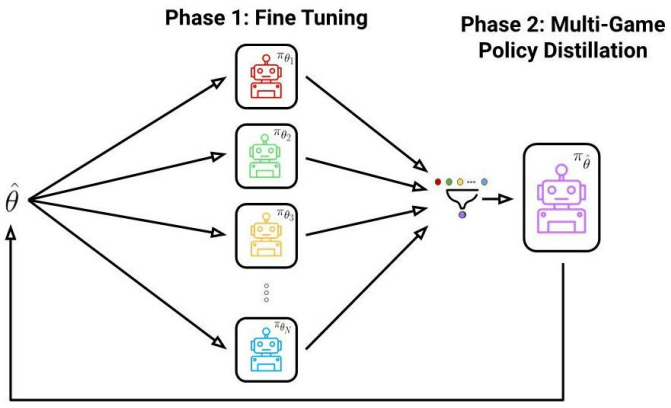


Fig. 1: Overview of the BYOT framework consisting of two alternating phases: **Fine-tuning** phase, in which the game-specific agents are fine tuned for their respective games, and **Multi Game Policy Distillation** phase, where the bootstrapped teacher agents are distilled back into a shared multi-game policy. The parameters of the multi-game policy are afterwards used to spawn the next iteration of game-specific agents.

Figure 1 illustrates a single policy improvement cycle in BYOT. Central to the BYOT framework is the repeated spawning, fine-tuning, and distilling of bootstrapped teacher agents. Algorithm 1 provides high-level pseudo-code of this process. We describe the details of each phase, corresponding to the *FineTuneTeacher* and *MultiGameDistillation* in Algorithm 1 in the following sections. We present a unified BYOT framework where the exact learning algorithms in either phase can be changed or switched in a plug-and-play manner, allowing BYOT to be extended for additional learning algorithms.

Considering just a single cycle of the BYOT algorithm, training the game-specific teacher agents independently to convergence, and then distilling into a multi-game policy can be considered equivalent to multi-game policy distillation [4], [5].

A. Game-specific Fine-tuning

During BYOT training, we denote a policy with parameters $\theta_i \in \mathbb{R}^d$ as π_{θ_i} and we denote the current shared multi-game policy as $\pi_{\hat{\theta}}$, with parameters $\hat{\theta} \in \mathbb{R}^d$. For each game in the fine-tuning phase, $e_i \in E$, we make a copy of the current multi-game policy's parameters, i.e. $\hat{\theta} \rightarrow \{\theta_1, \ldots, \theta_N\}$ (line 3 in Alg. 1), and then initialize N policies using these parameters, i.e. $\{\pi_{\theta_1}, \ldots, \pi_{\theta_N}\}$ (line 4 in Alg. 1). These policies become the bootstrapped teacher agents, i.e. they are passed along with an environment to the *FineTuneTeacher* function, where the game-specific agents are fine-tuned for N_F steps (line 6 in Alg. 1). In practice, rollouts are performed in batches of vectorized environments for each game, running n steps per environment rollout.

a) FineTuneTeacher: This function receives an initialised game-specific policy π_{θ_i} and an environment e_i. Game-specific experience is gathered, following trajectories by sampling actions $a_t \sim \pi_{\theta_i}(a_t \mid s_t, M_i)$ acting in the i'th game MDP M_i.

Algorithm 1: BYOT: Online Policy Distillation

Input : Set of environments: $E = \{e_1, \ldots, e_N\}$, where N is the number of games, fine_tuning_steps: N_F, distillation_steps: N_D, Hyperparameters: $\{\ldots\}$

Output : Shared multi-task policy $\pi_{\hat{\theta}}$ with parameters $\hat{\theta} \in \mathcal{R}^d$

1 **for** *n_iterations* **do**
2 **for** *Game:* $i \leftarrow 1$ **to** N **do**
3 Copy: $\theta_i \leftarrow \hat{\theta}$
4 Init: Single Game Proxy Expert: $\pi_{\theta_i} \leftarrow \pi_{\hat{\theta}}$
5 **for** *Step:* $\leftarrow 1$ **to** N_F **do**
6 $\pi_{\theta_i}, \mathcal{B}_i \leftarrow$ `FineTuneTeacher` (π_{θ_i}, e_i)
7 **end**
8 **end**
9 **for** *Step:* $\leftarrow 1$ **to** N_D **do**
10 $\pi_{\hat{\theta}} \leftarrow$ `MultiGameDistillation` $(\pi_{\theta_1}, \ldots, \pi_{\theta_N}, \mathcal{B})$
11 **end**
12 **end**

Any policy improvement update can be made to π_{θ_i} at this stage.

In our experiments we use a PPO actor-critic objective, i.e. a clipped surrogate policy gradient objective along with additional value-loss and entropy regularization, defined as:

$$L^{\text{PPO}}(\theta_i) = \mathbb{E}_{\pi_{\theta_i}}\big[L^{\text{CLIP}}(\theta_i) + c_1 L^{\text{VF}}(\theta_i) + c_2 H(\pi_{\theta_i})\big], \quad (2)$$

where: $L^{\text{CLIP}}(\theta_i) = \mathbb{E}_{\pi_{\theta_i}}\big[\min\big(r_{\theta_i}(s_t)A(s_t), \text{clip}(r_{\theta_i}(s_t), 1-\epsilon, 1+\epsilon)A(s_t)\big)\big]$ and $r_{\theta_i}(s_t) = \big[\frac{\pi_{\theta_i'}(a_t|s_t)}{\pi_{\theta_i}(a_t|s_t)}\big]$ is the ratio of new θ_i' and old θ_i; $A(s_t)$ is the Advantage function; $L^{\text{VF}}(\theta_i)$ is the mean squared error value loss; $H(\pi_{\theta_i})$ is the entropy bonus to promote exploration, and c_1 and c_2 are constants.

The game-specific fine-tuning phase is repeated for N_F number of steps for each game/environment. The output of this phase is:

1) Collection of game-specific agents, $\pi_{\theta_1}, \ldots \pi_{\theta_N}$, one per environment, where each agent is a *bootstrapped teacher* for that environment after game-specific policy improvements.
2) The final batch of rollouts from the current fine-tuning phase, stored as a mini-batch $\mathcal{B}_i$ on each environment i.e. $\mathcal{B} = \bigcup_{i=0}^{N} \mathcal{B}_i$).

Both π_{θ_i} and $\mathcal{B}_i$ are used in the Multi Game Distillation phase.

B. Multi-game Policy Distillation Phase

Once we have game-specific teachers, we use these agents in a multi-game distillation phase to distill back into a shared multi-game policy $\pi_{\hat{\theta}}$ (line 10 in Alg. 1).

a) MultiGameDistillation: The BYOT framework is flexible and allows for different supervised objective functions to be used in this phase: We use a standard multi-game policy distillation objective based on the distance between the game-expert policies with the current multi-game policy, that weighs each game with equal importance and removes the need for reward scaling, i.e.

$$\mathcal{L}(\hat{\theta}) = \mathbb{E}_{M_i \sim p(M)}\big[\mathbb{E}_{s_t \sim \mathcal{B}_i}\big[\text{KL}(s.g.(\pi_{\theta_i}) \| \pi_{\hat{\theta}}) + (V_{\hat{\theta}} - s.g.(V_{\theta_i}))^2\big]\big],$$

with a stop gradient (s.g.) applied around the game-specific policy and value functions, and s_t notation omitted from $\pi_\theta(s_t)$ and $V(s_t)$ for notational brevity. In practice the outer expectation is averaged over all games (i.e. environments), with the contribution for each calculated using the final batch of trajectories obtained by the game specific teacher.

The output from the distillation phase is an updated shared multi-game policy. The two phases are repeated until convergence or a total number of environment steps limit is reached. In our experiments we report metrics of both the game-specific policies and the multi-game policy over the total number of environment steps. Lastly, one reported step is equivalent to one step in each of N game environments.

We highlight here that the addition of the Multi Game Distillation phase that occurs after the per-game fine tuning phase introduces a small additional computation cost compared to our baseline methods. This small increase is due to computing the distillation loss on the final batch of data returned from the Fine Tune Teacher subroutine, and includes no additional environment steps (the environment is stepped in the per-game fine tuning phases).

V. Experiments

We empirically validate the BYOT framework using the Atari-40 benchmark [27], [29]. The Atari 40 benchmark, encompassing 40 diverse games, serves as a robust and challenging test bed for validating multi-game RL environments, offering a wide array of complex and varied dynamics, as seen in prior works [1], [2], [13], [14], [30]. All 40 games are evaluated by our single, multi-game agent. In Section V-A we demonstrate that BYOT outperforms all leading online Atari-40 multi-game agents with an IQM-HNS of 152.7%. In Section V-B we evaluate that the combination of policy distillation with state-of-the-art PPO expert teachers (equivalent to BYOT offline) achieves an IQM-HNS of 369.5%, surpassing all previously reported scores in this setting. Finally, in Section V-C we investigate the per-game scores between our offline single-game teacher experts, multi-game online BYOT, and offline multi-game PD approaches.

a) Setup: For all experiments, the BYOT agent uses an actor-critic architecture with a shared convolutional encoder, single policy and critic heads, and uses a shared action space to play all 40 games without any one-hot game encoding. Our full experimental setup and the detailed architecture are presented in the supplementary material [1]. We present all

[1] Supplementary material is available at:

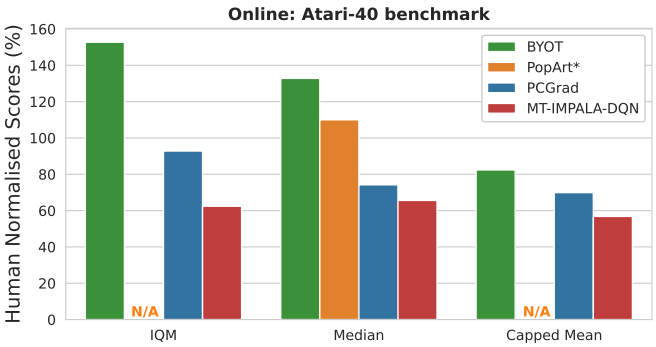


Fig. 2: IQM, median and capped mean HNS of multi-game agent's performance on the Atari-40 benchmark. Comparison between BYOT with online-RL algorithms PopArt*, A2C implementation using PCGrad optimizer and MT-IMPALA-DQN.

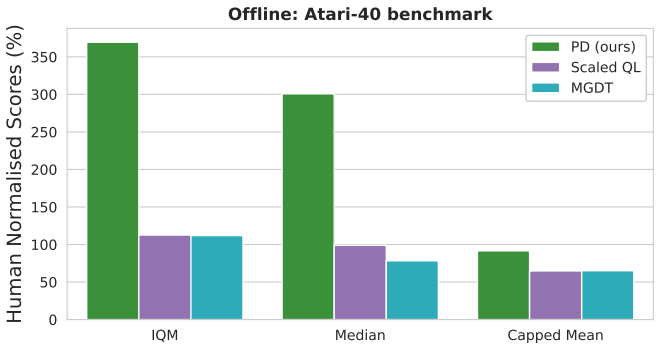


Fig. 3: IQM, median and capped mean HNS of multi-game agent's performance on the Atari-40 benchmark. Comparison between PD (ours) and existing state-of-the-art methods such as Scaled QL and MGDT.

our results after 200 million environment interaction steps, chosen to allow direct comparison to baselines. This results in 2 million iterations with 100 fine tuning steps and 100 distillation steps respectively. Where possible, we have used publicly available data for the baseline scores to avoid reproduction and expensive retraining costs; with the specific details and limitations provided in the relevant sections. As per in [31] we report the IQM-HNS as well as median score and capped mean human-normalized-score for each algorithm.

We provide all hyperparameters used for baseline experiments in order to reproduce our results in the supplementary material [1].

A. Multi-Game Online-RL

We evaluate BYOT; which alternates between bootstrapping teacher agents (copy then fine-tune phase), and distilling from teacher agents back into the shared multi-game policy (distillation phase).

a) Online Baselines: We compare against multi-game online-RL baselines that have direct access to the ALE environment during training, such as PCGrad [20], PopArt [14] and MT-IMPALA-DQN [2]. These leading online-RL Atari baselines do not share a standard experimental setup, therefore there exists an enormous computational burden to simultaneously match the exact setup of each baseline. For this reason, we train BYOT for 200 million environment interactions, and present a re-implementation of the A2C agent that utilizes the PCGrad optimizer from [20].

MT-IMPALA-DQN scores for this setting are publicly reported in [2]. The PopArt scores are publicly available and taken from [14]. However, to the best of our knowledge, PopArt scores are not available for the subset of 40 Atari games used in the other baselines. Therefore we include the results with the caveat (and asterisk in Fig. 2) that they are taken from a multi-game agent trained on all 57 Atari games, where all other baselines are trained on the 40 game subset.

b) Online Results: Results are presented in Fig. 2. BYOT achieves an IQM-HNS of 152.7%, outperforming all previously reported online multi-game RL methods on the subset of Atari games. As noted above, PopArt is trained on 57 games (not just the 40 game subset used by other baselines). However, they only report a HNS-Median score, making it impossible to compare using all metrics. While BYOT does shows an improved HNS-Median score, this metric is somewhat limited in that it disregards the relative performance of high and low performing games.

Overall, our results demonstrate the clear benefits of distilling multiple game-specific bootstrapped teacher policies into one shared multi-game policy, rather than manipulating the direction of multiple high dimensional gradient updates in parameter space, as seen in previous online methods. In the supplementary material [1] we provide an ablation experiment into the efficacy benefits of directly distilling the policy as opposed to the parameters of the bootstrapped teachers into the multi-game policy.

B. Multi-Game Offline-RL

Current SOTA models on the Atari-40 benchmark adopt an offline learning approach by utilizing prior trajectories to facilitate learning. In this section we evaluate our revisited implementation of policy distillation that leverages prior-trained state-of-the-art PPO expert agents in combination with the multi-game policy distillation supervised objective.

a) Offline Baselines: We compare against current SOTA baselines in this domain; Scaled Q-learning (Scaled QL) [2] and Multi Game Decision Transformer (MGDT) [1]. Both baseline methods use a near-optimal curated dataset of prior trajectories taken from [28]. The dataset comprises of 2 billion transition (50M per game) collected from numerous checkpoints during the training of a DQN agent [28]. The near-optimal dataset provides an IQM-HNS of 93.5% across the 2 billion transitions, (which is inferior to the newer PPO single-game experts). The baseline scores are publicly available and taken from [2].

b) Offline Results: Offline-RL results are presented in Fig. 3. The combination of policy distillation and state-of-the-art PPO expert teachers achieves an impressive performance of 369.4% IQM-HNS, despite having far fewer model parameters

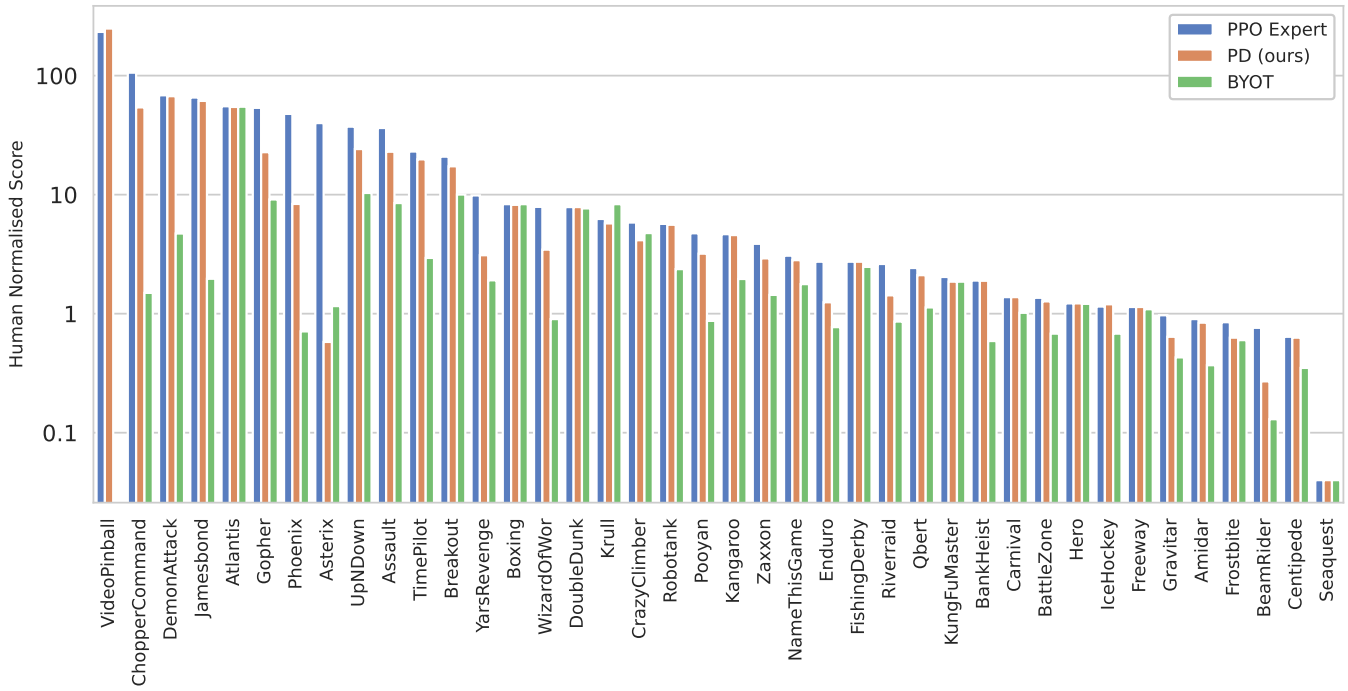


Fig. 4: Per-game performance of single-game PPO experts on Atari 40 benchmark, policy distillation and BYOT.

than both Scaled QL and MGDT: 5 million vs 80 million and 200 million parameters, respectively.

The results highlight the efficacy of directly distilling a policy to recover high per game performance across a diverse suite of games. By revisiting the policy distillation algorithm, in combination with modern state-of-the-art PPO expert teacher agents, we demonstrate offline-RL performance that surpasses the state-of-the-art in this domain, while using a comparatively small network architecture.

We note that the offline baseline methods, MGDT [1] and Scaled QL [2], rely on an older algorithm (DQN) to generate trajectories which achieves a lower IQM-HNS. Their offline dataset is curated to contain a mixture of expert, non-expert and human-level transitions. By contrast, we directly use our highest performant policies as teacher agents. With this in mind, MGDT and Scaled QL both report to outperform the IQM-HNS of their offline dataset. Our best hypothesis for this is the increased capacity in their respective network architecture in comparison to our policy distillation implementation. As seen in Scaled QL [2], MGDT [1] and ActorMimic [5], scaling network capacity has been shown to have a significant effect on the agents ability to generalize and perform across multiple tasks. We provide an initial study into different network architectures in the supplementary material[1], but leave scaling further to future work.

C. Per-Game Performance

We further investigate our per-game performance of BYOT (online) and policy distillation (offline) across the Atari-40 benchmark which can be seen in Fig. 4.

a) Online Results: The relative performance of BYOT is quite consistent with the single-game PPO expert agents across the majority of games. However, it is interesting to note that BYOT struggles on a handful of games where the single-game expert teachers have the highest HNS.

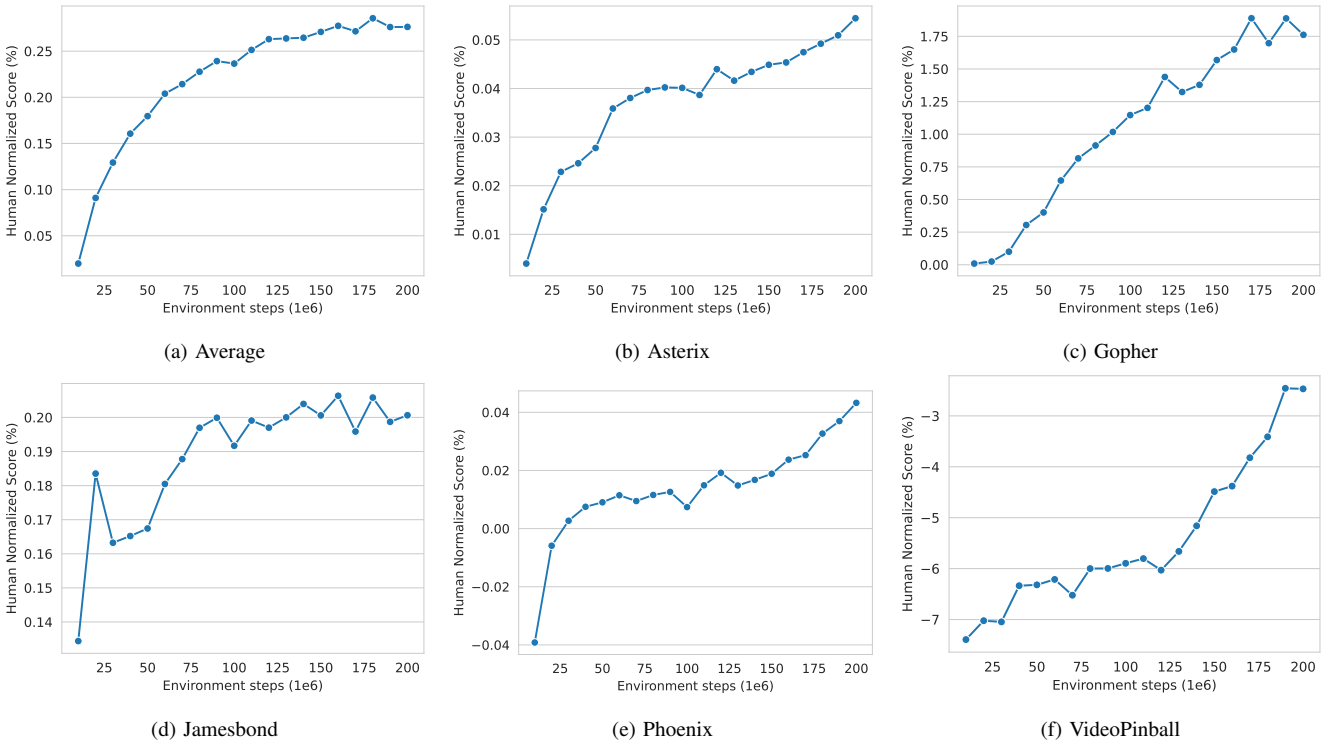
Whilst this could indicate a systematic effect, when we look at the per-game learning curves shown in the supplementary material[1], we see that in most cases performance is still increasing at the end of our training budget. Figure 5 shown that for the games on which online BYOT is furthest from the single-game agent performance the multi-game agent has not yet converged. This effect may simply represent that not all games are learned at the same rate during training, and that with continued optimisation even greater, and more consistent, performance is possible.

b) Offline Results: In the case of our policy distillation implementation, our PPO expert teacher agents have an IQM-HNS of 639.2%. Our multi-game agent recovers an average of 86.1% of the teacher performance. The multi-game agent achieves over 95% of the game-specific agent score on 15 games.

In general, we see that policy distillation performs consistently near the teacher performance across almost all games, suggesting that distillation in policy space effectively balances the learning across all games, despite the lack of any explicit reward scaling or normalization.

VI. Conclusion

Training multi-game agents capable of performing well across diverse environments still presents a significant challenge.



(a) Average (b) Asterix (c) Gopher

(d) Jamesbond (e) Phoenix (f) VideoPinball

Fig. 5: (a) Average training curve across all 40 Atari games. (b-f) Game-specific training curves for BYOT on games where it is noticeably outperformed by the single-game PPO expert.

Current state-of-the-art methods for multi-game RL often rely on offline datasets, and thus discard the policy used to gather the trajectories. In this paper we revisit multi-game policy distillation and demonstrate the potential of using bootstrapped teachers to provide a rich learning signal to tackle this problem.

We propose a new framework called Bootstrap Your Own Teacher (BYOT) that extends multi-game policy distillation to the online-RL setting. Central to the BYOT framework is the repeated spawning, fine-tuning and distilling of bootstrapped teachers. We evaluate our framework for both online and offline multi-game learning on the Atari-40 benchmark. BYOT outperforms all online Atari-40 multi-game agents with an IQM-HNS of 152.7%. Furthermore, in the offline-RL setting our implementation of policy distillation that utilizes state-of-the-art PPO expert teachers achieves a new leading IQM-HNS of 369.5%; surpassing all previously reported scores in this setting, whilst using an order of magnitude fewer parameters. Overall, our results demonstrate the power of distillation in multi-game settings.

a) Limitations: A specific design choice of BYOT, as with original policy distillation, is that the distillation phase requires action-values rather than simply the action taken by an expert. Whilst this is central to BYOT as it is designed to work in an online setting where RL agents are the natural paradigm, it restricts the use of human generated trajectories as is possible with some baseline approaches. As stated in Section V, due to computational constraints in this domain we are reliant on publicly reported scores for certain baselines.

A further limitation is that inline with current baselines, we are unable to run multiple seeds to produce statistically significant error bars on the Atari-40 benchmarks.

b) Future Work: We introduce BYOT as a flexible framework for online multi-game policy distillation. The highly configurable nature of BYOT leaves many directions to be explored in future work. In our experiments we observe that there can be a large gap between the single-game expert teacher performance and our master policy for some games. In order to understand and reduce this gap, further investigative work is warranted. Further to this, the baseline approaches use significantly larger models, with more capacity, than we do. We intend to further investigate the potential to scale BYOT.

Additionally, previous work has shown some promising results in the domain of fine tuning and transfer learning across tasks. This is outside the scope of this work and warrants further investigation. For additional data-efficiency, we also intend to adapt how much experience each bootstrapped teacher agent requires by implementing an adaptive strategy [32]. Finally, as per Distral [22] we intend to investigate smaller network architectures for the bootstrapped teachers. Distilling back to a larger model has been demonstrated to be particularly useful for scaling to open ended environments [24].

REFERENCES

[1] K.-H. Lee, O. Nachum, M. S. Yang, L. Lee, D. Freeman, S. Guadarrama, I. Fischer, W. Xu, E. Jang, H. Michalewski *et al.*, "Multi-game decision transformers," *Advances in Neural Information Processing Systems*, vol. 35, pp. 27 921–27 936, 2022.

[2] A. Kumar, R. Agarwal, X. Geng, G. Tucker, and S. Levine, "Offline q-learning on diverse multi-task data both scales and generalizes," in *International conference on learning representations*, 2023.

[3] S. Reed, K. Zolna, E. Parisotto, S. G. Colmenarejo, A. Novikov, G. Barth-maron, M. Giménez, Y. Sulsky, J. Kay, J. T. Springenberg, T. Eccles, J. Bruce, A. Razavi, A. Edwards, N. Heess, Y. Chen, R. Hadsell, O. Vinyals, M. Bordbar, and N. de Freitas, "A Generalist Agent," *Transactions on Machine Learning Research*, 2022.

[4] A. A. Rusu, S. G. Colmenarejo, C. Gulcehre, G. Desjardins, J. Kirkpatrick, R. Pascanu, V. Mnih, K. Kavukcuoglu, and R. Hadsell, "Policy distillation," in *International conference on learning representations*, 2016.

[5] E. Parisotto, J. L. Ba, and R. Salakhutdinov, "Actor-mimic: Deep multitask and transfer reinforcement learning," in *International conference on learning representations*, 2016.

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

[7] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *Deep Learning and Representation Learning Workshop at NeurIPS*, 2014.

[8] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *International Journal of Computer Vision*, vol. 129, pp. 1789–1819, 2021.

[9] C. Buciluǎ, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006, pp. 535–541.

[10] S. Green, C. M. Vineyard, and C. K. Koç, "Distillation strategies for proximal policy optimization," *arXiv preprint arXiv:1901.08128*, 2019.

[11] W. M. Czarnecki, R. Pascanu, S. Osindero, S. Jayakumar, G. Swirszcz, and M. Jaderberg, "Distilling policy distillation," in *The 22nd international conference on artificial intelligence and statistics*. PMLR, 2019, pp. 1331–1340.

[12] R. Agarwal, M. Schwarzer, P. S. Castro, A. Courville, and M. G. Bellemare, "Reincarnating reinforcement learning: Reusing prior computation to accelerate progress," *Advances in Neural Information Processing Systems*, 2022.

[13] L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning *et al.*, "Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures," in *International conference on machine learning*. PMLR, 2018, pp. 1407–1416.

[14] M. Hessel, H. Soyer, L. Espeholt, W. Czarnecki, S. Schmitt, and H. van Hasselt, "Multi-task deep reinforcement learning with PopArt," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019.

[15] Z. D. Guo, B. A. Pires, B. Piot, J.-B. Grill, F. Altché, R. Munos, and M. G. Azar, "Bootstrap latent-predictive representations for multitask reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 3875–3886.

[16] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," in *Conference on robot learning*. PMLR, 2020, pp. 1094–1100.

[17] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," *Psychology of Learning and Motivaiton*, vol. 24, pp. 109–165, 1989.

[18] M. T. Rosenstein, Z. Marx, L. P. Kaelbling, and T. G. Dietterich, "To transfer or not to transfer," in *Proc. NIPS 2005 Workshop on Transfer Learning*. NeurIPS, 2005, pp. 1–4.

[19] W. Zhang, L. Deng, and D. Wu, "Overcoming negative transfer: A survey," *CoRR*, 2020.

[20] T. Yu, S. Kumar, A. Gupta, S. Levine, K. Hausman, and C. Finn, "Gradient surgery for multi-task learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 5824–5836, 2020.

[21] A. A. Taiga, R. Agarwal, J. Farebrother, A. Courville, and M. G. Bellemare, "Investigating multi-task pretraining and generalization in reinforcement learning," in *The Eleventh International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=sSt9fROSZRO

[22] Y. Teh, V. Bapst, W. M. Czarnecki, J. Quan, J. Kirkpatrick, R. Hadsell, N. Heess, and R. Pascanu, "Distral: Robust multitask reinforcement learning," in *Advances in neural information processing systems*, vol. 30, 2017.

[23] J. Beck, R. Vuorio, E. Z. Liu, Z. Xiong, L. Zintgraf, C. Finn, and S. Whiteson, "A survey of meta-reinforcement learning," 2023.

[24] A. A. Team, J. Bauer, K. Baumli, S. Baveja, F. Behbahani, A. Bhoopchand, N. Bradley-Schmieg, M. Chang, N. Clay, A. Collister *et al.*, "Human-timescale adaptation in an open-ended task space," *arXiv preprint arXiv:2301.07608*, 2023.

[25] M. T. Jackson, M. Jiang, J. Parker-Holder, R. Vuorio, C. Lu, G. Farquhar, S. Whiteson, and J. N. Foerster, "Discovering general reinforcement learning algorithms with adversarial environment design," 2023.

[26] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 1994.

[27] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The arcade learning environment: An evaluation platform for general agents," *Journal of Artificial Intelligence Research*, vol. 47, pp. 253–279, 2013.

[28] R. Agarwal, D. Schuurmans, and M. Norouzi, "An optimistic perspective on offline reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 104–114.

[29] J. Fan, "A review for deep reinforcement learning in atari: Benchmarks, challenges, and solutions," *arXiv preprint arXiv:2112.04145*, 2021.

[30] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," in *Deep Learning Symposium at NeurIPS*, 2016.

[31] R. Agarwal, M. Schwarzer, P. S. Castro, A. C. Courville, and M. Bellemare, "Deep reinforcement learning at the edge of the statistical precipice," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[32] T. Schaul, D. Borsa, D. Ding, D. Szepesvari, G. Ostrovski, W. Dabney, and S. Osindero, "Adapting behaviour for learning progress," *arXiv preprint arXiv:1912.06910*, 2019.

[33] J. Weng, M. Lin, S. Huang, B. Liu, D. Makoviichuk, V. Makoviychuk, Z. Liu, Y. Song, T. Luo, Y. Jiang, Z. Xu, and S. Yan, "Envpool: A highly parallel reinforcement learning environment execution engine," in *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 22 409–22 421.

[34] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

Supplementary Material

APPENDIX

In this section, we provide details of the experimental setup used for the main paper training and evaluation. In Appendix A we include a table of all hyperparameters used for phase 1 fine-tuning of game specific agents as well as phase 2 distilling bootstrapped teachers into a shared multi-game policy. We further describe the ALE environment setup used in the main experiments in Appendix B, the hyperparameters used in the ablation experiments in Appendix C, and give details of the compute resources used in Appendix D.

A. Hyperparameters

In Table I we present the hyperparameters used during the online BYOT algorithm for the main experiments on the Atari-40 benchmark in Section V-A, including hyperparameters for both the fine-tuning phase and the distillation phase.

TABLE I: Online BYOT hyperparameters

Description	Value
Training steps	2.0×10^8
n-iterations	2.0×10^6
PPO epochs	4
PPO minibatches	4
PPO entropy coef	1×10^-2
PPO critic loss cost	0.5
PPO clip coef	0.1
Horizon	128
GAE lambda	0.95
Discount factor	0.99
Fine-tuning steps	100
Fine tuning learning rate	2.5×10^{-4}
Fine-tuning gradient clip	0.5
Fine-tuning steps used for buffer β	1
Distillation steps	100
Distillation minibatches	64
Distillation learning rate	5×10^{-4}

Offline BYOT uses a collection of pretrained single-game PPO experts trained for 200 million steps. These single-game agents take the place of the fine-tuned agents in Algorithm 1, and no further *FineTuning* updates take place. However, we collect a replay buffer of experience for each agent $\mathcal{B}_i$ using 1 finetuning step, where no update to the agents are made. This buffer is required for the multi-game distillation phase in Algorithm 1. We present the offline BYOT hyperparameters in Table II, which are used for the experiments in Section V-B.

TABLE II: Offline BYOT hyperparameters

Description	Value
Training steps	2×10^8
Fine-tuning steps	1
Fine tuning learning rate	0
Fine-tuning gradient clip	0.5
Fine-tuning steps used for buffer β	1
Distillation steps	1
Distillation minibatches	64
Distillation learning rate	5×10^{-4}

B. Environment Setup

For all experiments, we use the EnvPool [33] framework with the gym Atari games. EnvPool uses environments provided by the ale_py 0.7.5 library which have default settings consistent with OpenAI gym NoFrameskip-v4 wrapper and additionally uses the following wrappers: random-noops, fire-reset, episodic-life, frame-skip, action-repeat, image-resize, reward-clip.

Both online and offline BYOT were trained for 200 million steps per game (8.0×10^9 steps in total). The details of the enabled Atari wrappers and the initialisation setup is presented in Table III. Most notably, the *Full Action Space* is enabled, which extends the action space of each Atari game to the full 18 actions. This is done to unify the action space and avoid the use of game specific action mappings when using a single policy head for all games. The evaluation was done with greedy action selection, on 256 episodes for each game. This set of parameters was used for the experiments in Section V-B and Section V-A.

TABLE III: Wrappers and initialisation parameters used for the Training and Evaluation process.

Description	Training	Evaluation
Number of parallel environments	128	32
Max episode steps	27000	27000
Image Height	84	84
Image Width	84	84
Stacked Frames	4	4
Gray Scale	True	True
Frame skip	4	4
Max noop number	30	30
Episodic Life	True	False
Zero discount on life loss	False	False
Reward Clip	True	False
Repeat action probability	0	0
Use inter area resize	True	True
Use fire reset	True	True
Full action space	True	True

C. Ablation Hyperparameters

For the ablation experiment in Section G and the ablation experiments in Appendix D, we used a smaller collection of 18 Atari games. The list of games is presented in Table IV. In general, we kept the hyperparameters consistent to the main experiments, except for the hyperparameters outlined in Table V.

TABLE IV: Set of games that were used in the Atari-18 benchmark.

Atari 18 environments					
Alien	Asterix	BeamRider	Boxing	Breakout	Centipede
Freeway	Gopher	Hero	Jamesbond	Pong	Rivverraid
RoadRunner	Robotank	Seaquest	SpaceInvaders	StarGunner	WizardOfWor

TABLE V: Subset of the parameters that were changed for the ablation experiments.

Description	Training	Evaluation
Training steps	5×10^7	-
Horizon	64	-
Number of parallel environments	64	32

D. Compute

All experiments were conducted on an internal compute cluster. All training experiments were run using a single A100 GPU with 80GB of memory, 64 CPUs and 192GB of RAM. Additionally, during our initial research and experimentation we utilised a TPUv3-8 pod for the purpose of rapid prototyping.

We present the main BYOT model architecture used for the experimental results in Section V-B and Section V-A in Figure 6 (left). This model uses a single CNN encoder, and a single decoder for all games. It uses separate fully connected layers in the policy and the critic and has 4 876 979 parameters in total.

We present the details of the ResNet BYOT model architecture used for the ablation experiment in Appendix E in Figure 6 (center). This model uses considerably more resources and has 10 881 891 total parameters. Furthermore, in Appendix E we investigate the effect of sharing the decoder-layer parameters and we present the shared-decoder BYOT model architecture in Figure 6 (right). This model is identical to the main experiments model in that it uses a single decoder for all games, but shares the fully-connected decoder layers between the policy and critic. It has 4 087 475 total parameters. Lastly, in the ablation experiment in Appendix F, we demonstrate the use of game-conditioning by concatenating a one-hot game encoding vector onto the flattened output of the encoder. This is only used in Appendix F, and demonstrated in the dashed box in Figure 6 (right).

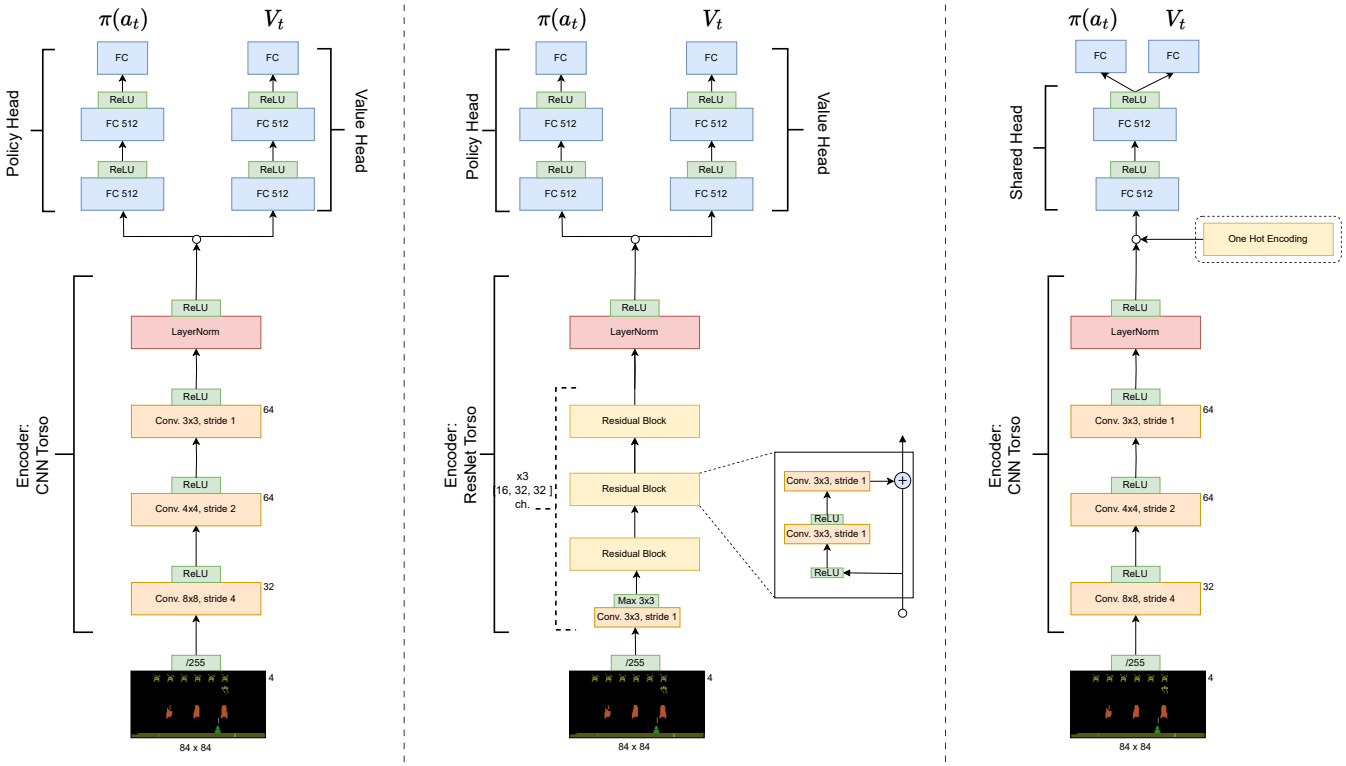


Fig. 6: **Left**. BYOT architecture used in the main paper experiments in Section V-B and Section V-A. **Center.** ResNet BYOT architecture used in Appendix E. **Right.** Shared-decoder BYOT architecture used in Appendix E. Game-conditioning via one-hot encoding used in Appendix F is demonstrated on the shared-decoder Architecture (right).

E. Architecture design choices

BYOT is a novel multi-game learning framework that uses an order of magnitude fewer parameters than recent state-of-the-art baselines. In this ablation, we investigate whether the performance of BYOT can be improved further by adjusting the model capacity, and including architectural design choices such as ResNet encoder [34].

We have seen that scaling the number of parameters within models has had a big impact on the performance of multi-game agents. Prior work such as multi game decision transformer [1] and scaled Q-learning [2] show that increasing the capacity of their models influences the final agent performance. Therefore, in-line with previous works, we present an ablation experiment looking at the impact that model design choices has on our online BYOT multi-game agent. We introduce three model architectures:

1) Original BYOT model used in the main paper experiments in Section V-B and Section V-B, uses a CNN encoder, a single decoder for all games with separate fully-connected layers for the policy and value heads. This model has 4 876 979 total parameters.
2) A ResNet model architecture that uses a ResNet encoder [34], and the same single decoder used for all games as the original model. In total uses 10 881 891 parameters.
3) A shared-decoder model, that uses a single decoder for all games and additionally shares parameters between the fully connected layers in the policy and the critic. This model has 4 087 475 total parameter.

We present an evaluation of these three models trained for 50 million steps on the Atari 18 collection of games in Figure 7 (left), with full details of experimental setup in Appendix C.

We find that in this setting, scaling the model to use a ResNet encoder achieves comparable results to our original CNN encoder. Whilst sharing additional parameters in the fully connected layers in the decoder drastically reduces performance.

F. Game conditioning via one-hot encoding

In this section, we present an ablation experiment to investigate the influence of game-conditioning via one-hot encoding on the performance of the multi-game BYOT agent. We demonstrate the model architecture of concatenating a one-hot encoding vector onto the flattened output of the encoder, shown in Figure 6 (right).

In multi-game RL, the goal is to learn a single multi-game policy. One way to achieve this is to learn a policy conditioned on the environment MDP, i.e. $\pi(a \mid \mathcal{M})$. We can encode a one-hot game identifier by conditioning the policy on the state and

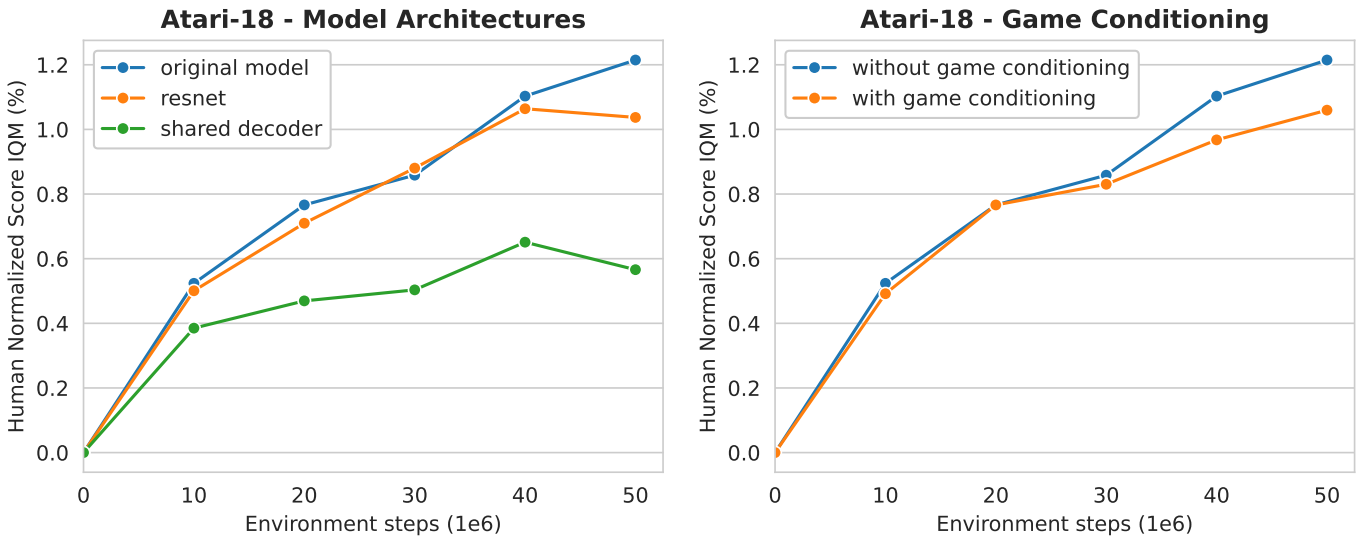


Fig. 7: **Left.** Comparison of IQM-HNS of the three architectures model. **Right.** Comparison of IQM-HNS of the original BYOT mode architecture with and without utilizing game conditioning via one-hot encoding.

the ID, i.e. $\pi(a \mid s, z)$, where z encodes the game ID. In experiments where we use game conditioning via one-hot encoding, we consider the game ID z to be part the MDP $\mathcal{M}$.

Figure 7 (right) presents an ablation investigation between the main BYOT architecture model (Figure 6 (left)) with and without the addition of game conditioning via one-hot encoding. We train the online BYOT agent for 50 million steps on the collection of 18 Atari games in Table IV. Further experimental setup details are outlined in Appendix C.

We see that the inclusion of game-conditioning via one-hot encoding does not significantly impact online BYOT results. This demonstrates that a shared encoder architecture is sufficient to learn a highly performant multi-game agent. We note that comparable baselines, such as Scaled QL [2], often use a separate decoder per game, which implicitly acts as conditioning on the game environment.

G. *Policies vs parameters*

a) Setup: In multi-game online-RL we have demonstrated improved results over methods such as PCGrad [20] and PopArt [14] that handle multi-game updates by directly manipulating the high dimensional parameters or gradients. To better understand the effectiveness of using policy distillation from bootstrapped teachers, we ablate our online BYOT algorithm by modifying the supervised objective used in the distillation phase (line 10 in Algorithm 1). Specifically, we investigate the effect of replacing the distillation loss (line 10) in the multi-game distillation phase. The distillation loss represents the distance between each bootstrapped teacher and the multi-game agent in policy space. We replace this with direct minimisation of the distance between teacher and multi-game parameters. Unlike policy distillation in Algorithm 1, which uses gradient descent to optimise the multi-game policy towards each of the teacher policies, this parameter-space optimisation can be performed directly by calculating the centroid of the bootstrapped teachers parameters.

b) Results: In Figure 8 we present the IQM-HNS for both approaches on the Atari-18 Benchmark, i.e. a subset of 18 Atari games trained for 50 million steps. The training and evaluation setup is fully outlined in Appendix A.

We can observe that direct distillation in parameter space is not able to perform as well as the one obtained by distilling the policies. Calculating the centroid of the bootstrapped teachers only reaches an IQM-HNS score of 8.0%. While still better than random, which has the IQM-HNS score of 0%, calculating the centroid of parameters is significantly worse than online multi-game policy distillation from bootstrapped teachers, that achieved an IQM-HNS score of 105.9%. This further validates our choice to operate in policy space.

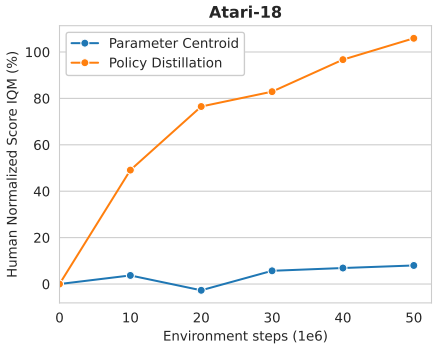


Fig. 8: Ablation comparison of IQM-HNS of online BYOT using multi-task policy distillation vs online BYOT that directly calculates the centroid of the bootstrapped teacher's parameters on the Atari-18 benchmark.

H. Atari Scores

The following section presents the raw scores on the Atari-40 benchmark. Table VI displays the per game scores using offline methods. Whilst, the per game scores using online methods are presented in Table VII.

TABLE VI: Raw per-game scores using offline methods on the Atari-40 benchmark. Bold scores outline the best offline multi-game agent performance on the specific game.

	Multi-Game			Single-Game		
Game	offline BYOT	MGDT	Scaled QL	PPO Expert	Human	Random
Amidar	**1,448.7**	101.5	21.0	1,553.9	1,719.0	5.8
Assault	**12,126.9**	2,385.9	3,809.0	19,088.3	742.0	222.4
Asterix	4,996.6	14,706.3	**34,278.0**	331,155.6	8,503.0	210.0
Atlantis	898,091.0	**3,105,342.3**	881,980.0	908,575.0	29,028.0	12,850.0
BankHeist	**1,407.5**	5.0	33.9	1,415.8	753.0	14.2
BattleZone	**46,692.1**	17,687.5	8,812.5	49,733.4	37,187.0	2,360.0
BeamRider	4,917.0	8,560.5	**10,301.0**	12,873.9	16,926.0	363.9
Boxing	98.8	95.1	**99.5**	99.9	12.1	0.1
Breakout	**500.6**	290.6	415.0	603.2	30.5	1.7
Carnival	**5,249.0**	2,213.8	926.0	5,249.4	3,800.0	0.0
Centipede	**8,348.9**	2,463.0	3,168.0	8,456.7	12,017.0	2,091.0
ChopperCommand	**357,207.0**	4,268.8	832.0	699,126.8	7,387.0	811.0
CrazyClimber	114,837.2	126,018.8	**140,500.0**	157,205.7	35,829.0	10,781.0
DemonAttack	**122,311.9**	23,768.4	56,318.0	124,681.8	1,971.0	152.1
DoubleDunk	**-1.3**	-10.6	-13.1	-1.3	-16.4	-18.6
Enduro	1,076.3	1,092.6	**2,345.0**	2,358.2	860.5	0.0
FishingDerby	**53.0**	11.8	23.8	53.0	-38.7	-91.7
Freeway	**33.7**	30.4	31.9	33.8	29.6	0.0
Frostbite	2,758.3	2,435.6	**3,566.0**	3,694.7	4,334.0	65.2
Gopher	**49,197.0**	9,935.0	3,776.0	115,812.0	2,412.0	257.6
Gravitar	**2,215.5**	59.4	262.3	3,246.4	3,351.0	173.0
Hero	**37,316.4**	20,408.8	20,470.0	37,303.2	30,826.0	1,027.0
IceHockey	**3.4**	-10.1	-1.5	2.7	0.9	-11.2
Jamesbond	**16,857.6**	700.0	483.6	17,968.4	302.8	29.0
Kangaroo	**13,682.5**	12,700.0	2,738.6	13,957.8	3,035.0	52.0
Krull	7,707.6	8,685.6	**10,176.9**	8,276.8	2,665.0	1,598.0
KungFuMaster	**42,059.7**	15,562.5	25,808.3	46,054.5	22,736.0	258.4
NameThisGame	**18,544.0**	9,056.9	11,647.0	20,018.1	8,049.0	2,292.0
Phoenix	**54,913.8**	5,295.6	5,264.0	310,815.6	7,242.0	761.4
Pooyan	**12,763.2**	2,859.1	202.1	18,989.7	4,000.0	0.0
Qbert	**28,169.5**	13,734.4	15,946.0	32,288.3	13,455.0	163.9
Riverraid	**23,883.4**	14,755.6	18,494.0	42,638.9	17,118.0	1,339.0
Robotank	56.5	**63.2**	53.2	57.5	11.9	2.2
Seaquest	1,910.9	**5,173.8**	414.1	1,920.0	42,054.0	68.4
TimePilot	**36,460.6**	2,743.8	4,220.5	41,820.5	5,229.0	3,568.0
UpNDown	**270,088.2**	16,291.3	55,512.9	416,604.4	11,693.0	533.4
VideoPinball	**367,374.7**	1,007.7	285.7	345,539.4	17,667.0	16,257.0
WizardOfWor	**15,047.0**	187.5	301.6	33,747.6	4,756.0	563.5
YarsRevenge	**162,552.5**	28,897.9	24,393.9	510,270.4	54,576.0	3,092.0
Zaxxon	**26,703.3**	275.0	2.1	35,348.4	9,173.0	32.5

TABLE VII: Raw per-game scores using online models on the Atari-40 benchmark. Bold scores outline the best online multi-game agent performance on the specific game.

	Multi-Game			Single-Game		
Game	online BYOT	PCgrad	MT-IMPALA	PPO Expert	Human	Random
Amidar	633.1	**1,215.2**	629.8	1,553.9	1,719.0	5.8
Assault	4,826.1	**12,259.4**	1,338.7	19,088.3	742.0	222.4
Asterix	10,178.0	**17,211.8**	2,949.1	331,155.6	8,503.0	210.0
Atlantis	969,776.3	791,003.9	**976,030.4**	908,575.0	29,028.0	12,850.0
BankHeist	450.0	351.1	1,069.6	**1,415.8**	753.0	14.2
BattleZone	26,222.5	**44,615.4**	26,235.2	49,733.4	37,187.0	2,360.0
BeamRider	2,401.6	**3,805.6**	1,524.8	12,873.9	16,926.0	363.9
Boxing	100.0	**100.0**	68.3	99.9	12.1	0.1
Breakout	**293.1**	32.2	32.6	603.2	30.5	1.7
Carnival	**3,859.2**	2,291.6	2,021.2	5,249.4	3,800.0	0.0
Centipede	**5,570.2**	4,726.2	4,848.0	8,456.7	12,017.0	2,091.0
ChopperCommand	**10,336.2**	3,583.0	951.4	699,126.8	7,387.0	811.0
CrazyClimber	130,619.3	128,623.8	**146,362.5**	157,205.7	35,829.0	10,781.0
DemonAttack	10,414.0	**16,490.1**	446.8	124,681.8	1,971.0	152.1
DoubleDunk	-1.8	**-1.3**	-156.2	-1.3	-16.4	-18.6
Enduro	691.6	546.4	**896.3**	2,358.2	860.5	0.0
FishingDerby	**39.9**	22.0	-152.3	53.0	-38.7	-91.7
Freeway	32.1	**33.0**	30.6	33.8	29.6	0.0
Frostbite	2,631.6	**3,437.0**	2,748.4	3,694.7	4,334.0	65.2
Gopher	**19,943.0**	1,022.5	3,205.6	115,812.0	2,412.0	257.6
Gravitar	**1,541.5**	72.2	492.5	3,246.4	3,351.0	173.0
Hero	**37,025.0**	20,678.7	26,568.8	37,303.2	30,826.0	1,027.0
IceHockey	**-3.5**	-5.2	-10.4	2.7	0.9	-11.2
Jamesbond	583.5	**1,530.4**	264.6	17,968.4	302.8	29.0
Kangaroo	5,858.1	1,376.4	**7,997.1**	13,957.8	3,035.0	52.0
Krull	**10,627.4**	10,508.1	8,221.4	8,276.8	2,665.0	1,598.0
KungFuMaster	**42,888.1**	17,156.8	29,383.1	46,054.5	22,736.0	258.4
NameThisGame	**12,427.4**	5,847.5	6,548.8	20,018.1	8,049.0	2,292.0
Phoenix	5,339.9	**5,407.2**	3,932.5	310,815.6	7,242.0	761.4
Pooyan	3,459.4	1,686.5	**4,000.0**	18,989.7	4,000.0	0.0
Qbert	**15,296.2**	9,228.7	4,226.5	32,288.3	13,455.0	163.9
Riverraid	**14,784.3**	7,538.1	7,306.6	42,638.9	17,118.0	1,339.0
Robotank	26.1	**27.2**	9.2	57.5	11.9	2.2
Seaquest	**1,755.3**	900.4	1,415.2	1,920.0	42,054.0	68.4
TimePilot	**9,077.0**	6,483.0	-883.1	41,820.5	5,229.0	3,568.0
UpNDown	115,865.7	**160,559.5**	8,167.6	416,604.4	11,693.0	533.4
VideoPinball	7,018.3	0.0	**85,351.0**	345,539.4	17,667.0	16,257.0
WizardOfWor	4,324.4	**4,422.7**	975.9	33,747.6	4,756.0	563.5
YarsRevenge	**101,558.0**	39,758.8	18,889.5	510,270.4	54,576.0	3,092.0
Zaxxon	**13,910.4**	9,212.9	-0.1	35,348.4	9,173.0	32.5

I. Learning Curves

Average IQM-HNS training and evaluation curves for online BYOT on the Atari-40 benchmark are presented in Figure 9. Full training and evaluation curves for each individual game are presented in Figure 10. Training metrics are computed on the game-specific agents at the end of each fine-tuning phase. Evaluation metrics are computed on the shared multi-game agent after the distillation phase. As described in Appendix B, the environment setup during training uses the episodic life wrapper, while during evaluation episodic life wrapper is not used. This significantly changes the scale of the total rewards, as such we provide both sets of metrics on the same figure with separate y-axes in Figure 10. We notice significant correlation between the single-game agents' training metrics and the multi-game agent's performance, shown in the evaluation curves.

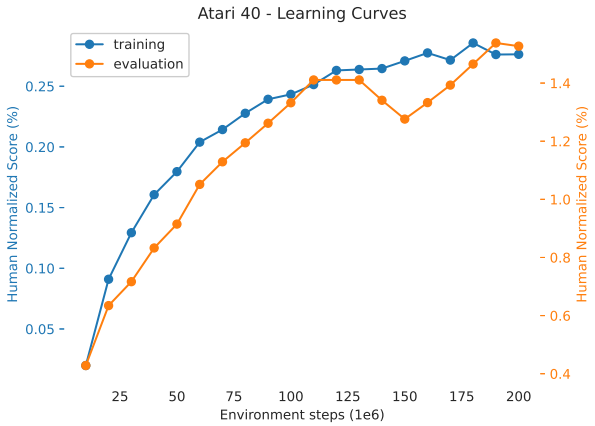


Fig. 9: Comparison of IQM-HNS of the online BYOT scores during training (blue) and evaluation (orange).

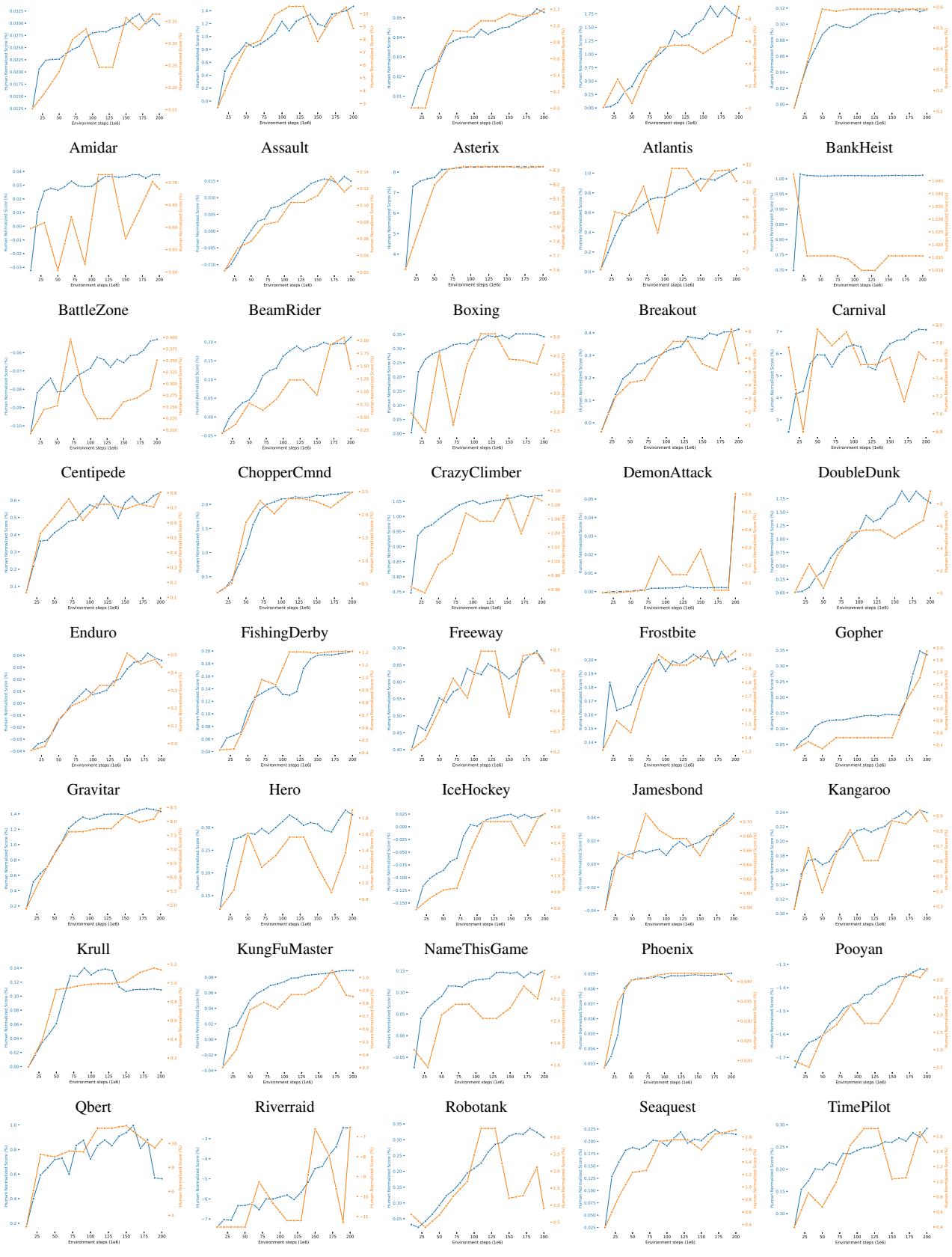


Fig. 10: Game-specific human normalized scores for training (blue) and evaluation (orange) curves for online BYOT.