TeachMyAgent: a Benchmark for Automatic Curriculum Learning in Deep RL

Clément Romac [*] [1] **Rémy Portelas** [*] [1] **Katja Hofmann** [2] **Pierre-Yves Oudeyer** [1]

Abstract

Training autonomous agents able to generalize to multiple tasks is a key target of Deep Reinforcement Learning (DRL) research. In parallel to improving DRL algorithms themselves, Automatic Curriculum Learning (ACL) study how teacher algorithms can train DRL agents more efficiently by adapting task selection to their evolving abilities. While multiple standard benchmarks exist to compare DRL agents, there is currently no such thing for ACL algorithms. Thus, comparing existing approaches is difficult, as too many experimental parameters differ from paper to paper. In this work, we identify several key challenges faced by ACL algorithms. Based on these, we present *TeachMyAgent* (TA), a benchmark of current ACL algorithms leveraging procedural task generation. It includes 1) challenge-specific unit-tests using variants of a procedural Box2D bipedal walker environment, and 2) a new procedural Parkour environment combining most ACL challenges, making it ideal for global performance assessment. We then use *TeachMyAgent* to conduct a comparative study of representative existing approaches, showcasing the competitiveness of some ACL algorithms that do not use expert knowledge. We also show that the Parkour environment remains an open problem. We open-source our environments, all studied ACL algorithms (collected from open-source code or re-implemented), and DRL students in a Python package available at `https://github.com/flowersteam/TeachMyAgent`.

1. Introduction

When looking at how structured and gradual human-learning is, one can argue that randomly presenting tasks to a learning agent is unlikely to be optimal for complex learning problems. Building upon this, curriculum learning has long been identified as a key component for many machine learning problems (Selfridge et al., 1985; Elman, 1993; Bengio et al., 2009; Cangelosi & Schlesinger, 2015) in order to organize samples showed during learning. While such a curriculum can be hand-designed by human experts on the problem, the field of Automatic Curriculum Learning (Graves et al., 2017; Portelas et al., 2020a) focuses on designing teacher algorithms able to autonomously sequence learning problem selection so as to maximize agent performance (e.g. over a set of samples in supervised learning, or game levels in DRL).

Parallel to these lines of works, DRL researchers have been increasingly interested in finding methods to train generalist agents (Rajeswaran et al., 2017; Zhang et al., 2018; Vanschoren, 2018; Cobbe et al., 2019) to go beyond initial successes on solving single problems, e.g individual Atari games (Mnih et al., 2015) or navigation in fixed scenarios (Lillicrap et al., 2016; Haarnoja et al., 2018). Many works proposed novel DRL learning architectures able to successfully infer multi-purpose action policies when given an experience stream composed of randomly sampled *tasks* (Schaul et al., 2015; Hessel et al., 2018; Cobbe et al., 2019; Hessel et al., 2019). Here and thereafter *tasks* denote learning problems in general, for instance multiple mazes to solve (a.k.a environments) or, in the context of robotic manipulation, multiple state configuration to obtain (a.k.a. goals) (Portelas et al., 2020a). To compare existing and future Multi-task DRL agents, Cobbe et al. (2020) proposed a suite of 16 atari-like environments, all relying on Procedural Content Generation (PCG) to generate a wide diversity of learning situations. The high-diversity induced by PCG has been identified as particularly beneficial to foster generalization abilities to DRL agents (Justesen et al., 2018; Risi & Togelius, 2019; OpenAI et al., 2019).

An important aspect not covered by these prior works is that they all rely on proposing randomly selected tasks to their agent, i.e. they do not consider using curriculum in learning. One can argue that random task selection is inefficient, especially when considering complex continuous task sets, a.k.a task spaces, which can feature subspaces of varying difficulties ranging from trivial to unfeasible. Following this observation, many works attempted to train given multi-task

[*]Equal contribution [1]Inria, France [2]Microsoft Research, UK. Correspondence to: Clément Romac <clement.romac@inria.fr>, Rémy Portelas <remy.portelas@inria.fr>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

agents by pairing them with ACL algorithms (Portelas et al., 2020a). The advantages of ACL over random task sampling for DRL agents have been demonstrated in diverse experimental setups, such as domain randomization for sim2real robotics (OpenAI et al., 2019; Mehta et al., 2019), video games (Salimans & Chen, 2018; Mysore et al., 2019), or navigation in procedurally generated environments (Florensa et al., 2018; Portelas et al., 2019; Racanière et al., 2020).

While this diversity of potential application domains and implementations of ACL hints a promising future for this field, it also makes comparative analysis complicated, which limits large-scale adoption of ACL. For instance, depending on the ACL approach, the amount of required expert knowledge on the task space can range from close to none – as in Portelas et al. (2019) – to a high amount of prior knowledge, e.g. initial task sampling subspace and predefined reward range triggering task sampling distribution shifts, as in OpenAI et al. (2019). Additionally, some ACL approaches were tested based on their ability to master an expert-chosen target subspace (Klink et al., 2020) while others were tasked to optimize their performance over the entire task space (Baranes & Oudeyer, 2009; Florensa et al., 2018; Portelas et al., 2019). Besides, because of the large computational cost and implementation efforts necessary for exhaustive comparisons, newly proposed ACL algorithms are often compared to only a subset of previous ACL approaches (Mehta et al., 2019; Portelas et al., 2019; Racanière et al., 2020). This computation bottleneck is also what prevents most works from testing their ACL teachers on a diversity of DRL students, i.e. given a set of tasks, they do not vary the student's learning mechanism nor its embodiment. Designing a unified benchmark platform, where baselines would be shared and allow one to only run its approach and compare it to established results, could drive progress in this space.

Inspired by how the MNIST dataset (Lecun et al., 1998) or the ALE Atari games suite (Bellemare et al., 2013) respectively catalyzed supervised learning and single-task reinforcement learning research, we propose to perform this much-needed in-depth ACL benchmarking study. As such, we introduce *TeachMyAgent 1.0*[1], a teacher testbed featuring a) two procedural Box2D[2] environments with challenging task spaces, b) a collection of pre-defined agent embodiments, and c) multiple DRL student models. The combination of these three components constitutes a large panel of diverse teaching problems. We leverage this benchmark to characterize the efficiency of an ACL algorithm on the following key teaching challenges:

1. *Mostly unfeasible task spaces* - While using PCG systems to generate tasks allows to propose rich task spaces to DRL agents, which is good for generalization, such large spaces might contain a predominant amount of unfeasible (or initially unfeasible) tasks . A teacher algorithm must then have the ability to quickly detect and exploit promising task subspaces for its learner.
2. *Mostly trivial task spaces* - On the contrary, the task space might be mostly trivial and contain only few challenging subspaces, which is a typical scenario when dealing with a skilled student (e.g. that is already trained, or that has an advantageous embodiment). In that case the teacher has to efficiently detect and exploit the small portion of subspaces of relevant difficulty.
3. *Forgetting students* - DRL learners are prone to catastrophic forgetting (Kirkpatrick et al., 2017), i.e. to overwrite important skills while training new ones. This has to be detected and dealt with by the teacher for optimal curriculum generation.
4. *Robustness to diverse students* - Being able to adapt curriculum generation to diverse students is an important desiderata to ensure a given ACL mechanism has good chances to transfer to novel scenarios.
5. *Rugged difficulty landscapes* - Another important property for ACL algorithms is to be able to deal with task spaces for which the optimal curriculum is not a smooth task distribution sampling drift across the space but rather a series of distribution jumps, e.g. as in complex PCG-task spaces.
6. *Working with no or little expert knowledge* - Prior knowledge over a task space w.r.t. a given student is a costly information gathering process that needs to be repeated for each new problem/student. Relying on as little expert knowledge as possible is therefore a desirable property for ACL algorithms (especially if aiming for out-of-the-lab applications).

To precisely assess the proficiency of an ACL algorithm on each of these challenges independently, we extend a Box2D walker environment from Portelas et al. (2019) into multiple unit-test variants, one per challenge, inspired by the structure of *bsuite* (Osband et al., 2020), a recent benchmark for RL agents. The second environment of our benchmark is the *Parkour* environment, inspired by Wang et al. (2020). It features a complex task space whose parameters seed a neural network-based procedural generation of a wide diversity of environments, in which there exists drastically different learning curricula depending on the agent's embodiment (see fig. 1). To assess the ability of existing ACL methods to robustly adapt to diverse students, we consider a random black-box student scenario in the Parkour environment, i.e. the morphology (e.g. walker or climber) of the learner is randomly selected for each new training run.

[1] `http://developmentalsystems.org/TeachMyAgent/`

[2] 2D game engine, used in OpenAI gym (Brockman et al., 2016)

Scope More precisely, we conduct an in-depth comparative study of ACL approaches suited for generalist DRL agents such as SAC (Haarnoja et al., 2018) or PPO (Schulman et al., 2017) in single agent scenarios. We do not include works on self-play/multi-agent setups (Hernandez et al., 2019; Hernandez-Leal et al., 2018) nor single-agent population-based approaches (Forestier et al., 2017; Wang et al., 2020). Also, we are interested in the problem of task selection from a continuous parameter space encoding the procedural generation of tasks. We leave the analysis of ACL methods for discrete task sets (Matiisen et al., 2017; Mysore et al., 2019), sets of task spaces (Forestier et al., 2017; Colas et al., 2019), or intrinsic reward learning (Pathak et al., 2017; Burda et al., 2019) for future work. We assume this continuous space is given and relatively low-dimensional as it already poses strong teaching challenges: we therefore leave the analysis of approaches that autonomously learn task representations for subsequent work (Pong et al., 2020; Jabri et al., 2019; Kovač et al., 2020).

Our main contributions are:

- Identification of multiple challenges to be tackled by ACL methods, enabling multi-dimensional comparisons of these algorithms.
- *TeachMyAgent 1.0*, a set of teaching problems (based on PCG environments) to study and compare ACL algorithms when paired with DRL students.
- Comparative study of representative existing ACL approaches including both skill-specific unit-tests and global performance assessments, which highlights the competitiveness of methods not using expert knowledge and shows that our Parkour environment largely remains an open problem for current state-of-the-art ACL.
- Release of an open-source Python package, featuring 1) all environments, embodiments and DRL students from *TeachMyAgent*, 2) all studied ACL algorithms, that we either adapt to our API when code is available or re-implement from scratch if not open-sourced, 3) our experimental results as baselines for future works, and 4) tutorials & reproducibility scripts.

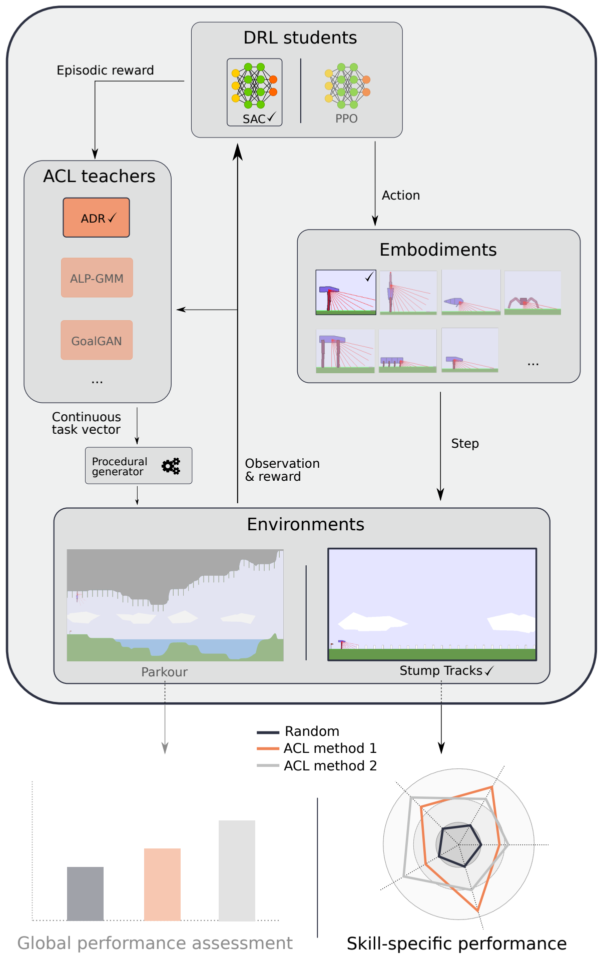

Figure 1. **TeachMyAgent**: A benchmark to study and compare teacher algorithms in continuous procedural environments.

2. Related work

Many environment suites already exist to benchmark DRL algorithms: some of them leverage video games, which provide challenging discrete action spaces, e.g. Atari 2600 Games as in Bellemare et al. (2013) or Sonic The Hedgehog levels in Nichol et al. (2018). To study and develop DRL agents suited for complex continuous control scenarios, the community predominantly used the MuJoCo physics engine (Todorov et al., 2012). The Deep Mind Lab (Beattie et al., 2016) provides customizable puzzle-solving environment, particularly well suited to study goal-conditioned policies learning from pixels in rich 3D environments. At the intersection of DRL and Natural Language Processing, benchmark environments such as TextWorld (Côté et al., 2018) or BabyAI (Chevalier-Boisvert et al., 2019) were also designed to provide a testbed to develop autonomous agent receiving linguistic goals and/or interacting using language. The *bsuite* benchmark (Osband et al., 2020) leverages unit-tests to assess the core capabilities of DRL methods (e.g. generalization, memory). In all these previous works, the DRL agent is learning in one or few environments presented randomly and/or intrinsically chooses goals within those predefined environments, and the long-term community objective is to find more efficient learning architectures. On the contrary, the objective of *TeachMyAgent* is to foster the development of new teacher algorithms whose objective is, given a task space and a DRL student, to most efficiently organize the learning curriculum of their DRL student such that its performance is maximized over the task set. In other words, it is not about finding efficient learning architectures but about finding efficient curriculum generators.

Perhaps closest to our work is the Procgen benchmark (Cobbe et al., 2020), which features several atari-like environments, all having unique procedural generation systems allowing to generate a wide diversity of learning situations, particularly well suited to assess the generalization abilities of DRL agents. While they rely on an uncontrolable, random procedural generation, we assume control over it, which enables the use of ACL methods to select parameters encoding task generation. An interesting future work, parallel to ours, would be to modify the Procgen benchmark to allow direct control over the procedural generation.

Because of the current lack of any ACL benchmark, most recently proposed ACL algorithms relied on designing their own set of test environments. Florensa et al. (2018) used a custom MuJoCo Ant maze in which the ACL approach is in control of which end-position to target. Klink et al. (2020) used another MuJoCo Ant maze and ball-catching environment featuring a simulated Barrett WAM robot. While these previous works studied how to control goal selection in a given fixed environment, we are interested in the arguably more challenging problem of controlling a rich parametric procedural generation. Portelas et al. (2019) already studied ACL in Stump Tracks, a procedural Box2D environment that we include and extend in *TeachMyAgent*, however it did not perform an extensive comparative study as what we propose in the present work. Racanière et al. (2020) also used procedural generation to test their ACL approach, however they only compared their ACL algorithm to Goal-GAN (Florensa et al., 2018), and did not open-source their environments. Additionally, in contrast with all previously cited ACL works, in *TeachMyAgent* we propose an in-depth analysis of each approaches through multiple unit-test experiments to fully characterize each teacher.

3. ACL baselines

In the following paragraphs we succinctly frame and present all the ACL algorithms that we compare using *TeachMyAgent*. More detailed explanations are left to appendix A.

Framework Given a DRL student s and a n-dimensional task-encoding parameter space $\mathcal{T} \in \mathbb{R}^n$ (i.e. a task space), the process of Automatic Curriculum Learning aims to learn a function $\mathcal{A} : \mathcal{H} \mapsto \mathcal{D}(\mathcal{T})$ mapping any information retained about past interactions with the task space to a distribution of tasks.

One can define the optimization objective of an ACL policy given an experimental budget of E episodic tasks as:

$$\max_{\mathcal{A}} \int_{\mathrm{T} \sim \mathcal{D}_{target}} P_{\mathrm{T}}^{E} \, \mathrm{dT}, \tag{1}$$

with $\mathcal{D}_{target}$ the distribution of test tasks over the task space and P the post-training performance (e.g. episodic reward, exploration score) of student s on task T after E episodes. Since it is usually difficult to directly optimize for this objective, various surrogate objectives have been proposed in the literature. See Portelas et al. (2020a) for a review and classification of recent ACL works.

Expert-knowledge To ease the curriculum generation process, multiple forms of expert knowledge have been provided in current ACL approaches. We propose to gather them in three categories: 1) use of initial task distribution $\mathcal{D}_{init}$ to bootstrap the ACL process, 2) use of a target task distribution $\mathcal{D}_{target}$ to guide learning, and 3) use of a function interpreting the scalar episodic reward sent by the environment to identify mastered tasks (*Reward mastery range*). For each implemented ACL method, we highlight its required prior knowledge over the task space w.r.t a given DRL agent in table 1. We hope that this classification will ease the process of selecting an ACL method for researchers and engineers, as available expert knowledge is (arguably) often what conditions algorithmic choices in machine learning scenarios.

Implemented baselines We compare seven ACL methods, chosen to be representative of the diversity of existing approaches, that can be separated in three broad categories. First, we include three methods relying on the idea of maximizing the Learning Progress (LP) of the student: RIAC (Baranes & Oudeyer, 2009), Covar-GMM (Moulin-Frier et al., 2014) and ALP-GMM (Portelas et al., 2019). We then add in our benchmark Goal-GAN (Florensa et al., 2018) and Setter-Solver (Racanière et al., 2020), both generating tasks using deep neural networks and requiring a binary reward for mastered/not mastered tasks, pre-defined using expert knowledge. Finally, we append to our comparison two ACL algorithms using the idea of starting from an initial distribution of tasks and progressively shifting it regarding the student's capabilities: ADR (OpenAI et al., 2019) (inflating a task distribution from a single initial task based on student mastery at each task distribution's border) and SPDL (Klink et al., 2020) (shifting its initial distribution towards a target distribution). We also add a baseline teacher selecting tasks uniformly random over the task space (called Random).

4. The *TeachMyAgent* benchmark

In the following section, we describe available environments and learners in *TeachMyAgent*. We propose two Box2D environments with procedural generation allowing to generate a

Table 1. Expert knowledge used by the different ACL methods. We separate knowledge required (REQ.) by algorithms, optional ones (OPT.), and knowledge not needed (empty cell).

ALGORITHM	$\mathcal{D}_{init}$	$\mathcal{D}_{target}$	REWARD MASTERY RANGE
ADR	REQ.		REQ.
ALP-GMM	OPT.		
COVAR-GMM	OPT.		
GOAL-GAN	OPT.		REQ.
RIAC			
SPDL	REQ.	REQ.	
SETTER-SOLVER		OPT.	REQ.

wide variety of terrains. Both our environments are episodic, use continuous action/observation spaces and return scalar rewards. In addition, we provide two DRL algorithms as well as multiple agent morphologies. An experiment is thus constituted of an ACL method, an environment and a learner (i.e. an embodied DRL algorithm).

4.1. Environments

Stump Tracks environment Stump Tracks is an extension of a parametric Box2D environment initially presented in Portelas et al. (2019). The learning policy is embodied into a walker agent whose motors are controllable with torque (i.e. continuous action space). The observation space is composed of lidar sensors, head position and joint positions. The walker is rewarded for going forward and penalized for torque usage. An episode lasts 2000 steps at most, and is terminated if the agent reaches the end of the track or if its head collides with the environment (in which case a -100 reward is received). A 2D parametric PCG is used for each new episode: it controls the height and spacing of stumps laid out along the track (see fig. 2 and app. B). We chose to feature this environment as its low-dimensional task space is convenient for visualizations and modifications. We derive multiple variants of Stump Tracks (e.g. by extending the task space boundaries or shuffling it) to design our unit-tests of ACL challenges (see sec. 1 and sec. 5).

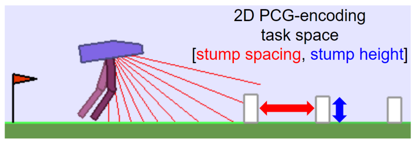

Figure 2. Stump Tracks, a simple parametric env. to study ACL algorithms with DRL students.

Parkour environment Inspired by both Stump Tracks and another Box2D environment from Wang et al. (2020), we present the parametric Parkour environment: a challenging task space with rugged difficulty landscape, few prior knowledge definable, and requiring drastically different learning curricula depending on the agent's embodiment.

It features an uneven terrain (see figure 1) composed of a ground and ceiling encoded through a Compositional Pattern-Producing Network (CPPN) (Stanley, 2007). This CPPN, whose weights and architecture are kept fixed, takes an additional input vector of bounded real numbers which acts as the parameters controlling terrain generation. This neural network based generation enables to create a task space with a rugged difficulty landscape (see appendix B), requiring time consuming exploration from an expert to seek trivial subspaces. We propose three versions of this task space (i.e. three possible bounds for the CPPN's input vector): easy, medium (used in the experiments of this paper) and hard. The Parkour environment also features graspable objects, called "creepers", creating a niche for climbing morphologies. Similarly to the stumps in Stump Tracks, the creepers' generation is controlled by their height and the space between them. The Parkour's task space also contains a dimension controlling the "water" level of the track, ranging from 0 (no water) to 1 (entire parkour under water). Water adds new physic rules aiming to imitate (in a simplified way) physics of water.

The resulting 6D task space (3 for the CPPN's input, 2 for creepers and 1 for water) creates a rich environment in which the optimal curriculum will largely depend on the agent's embodiment (e.g. swimming agents need high levels of water, while climbers and walkers need low levels). Note that, as in Stump Tracks, each episode lasts 2000 steps, agents are rewarded for moving forward (and penalised for using torque) and have access to lidars, head position, joint positions, and also additional information (see appendix B).

4.2. Learners

Embodiments As aforementioned, we introduce new morphologies using swimming and climbing locomotion (e.g. fish, chimpanzee, see figure 1). *TeachMyAgent* also features the short walker and quadrupedal walker from Portelas et al. (2019) as well as new walking morphologies such as the spider and the millipede (see figure 1).

DRL algorithms To benchmark ACL algorithms, we rely on two different state-of-the-art DRL algorithms: 1) Soft-Actor-Critic (Haarnoja et al., 2018) (SAC), a now classical off-policy actor-critic algorithm based on the dual optimization of reward and action entropy, and 2) Proximal Policy Optimization (PPO) (Schulman et al., 2017), a well-known on-policy DRL algorithm based on approximate trust-region gradient updates. We use OpenAI Spinningup's implementation[3] for SAC and OpenAI Baselines' implementation[4] for PPO. See appendix C for implementation details.

[3]https://spinningup.openai.com
[4]https://github.com/openai/baselines

5. Experiments

We now leverage *TeachMyAgent* to conduct an in-depth comparative study of the ACL algorithms presented in section 3. After discussing experimental details, we undergo two separate experiments, aiming to answer the following questions:

- How do current ACL methods compare on each teaching challenges proposed in sec. 1 ?
- How do current ACL methods scale to a complex task space with limited expert knowledge ?

5.1. Experimental details

For both our environments, we train our DRL students for 20 million steps. For each new episode, the teacher samples a new parameter vector used for the procedural generation of the environment. The teacher then receives the cumulative episodic reward that can be potentially turned into a binary reward signal using expert knowledge (as in GoalGAN and Setter-Solver). Additionally, SPDL receives the initial state of the episode as well as the reward obtained at each step, as it is designed for non-episodic RL setup. Every 500000 steps, we test our student on a test set composed of 100 pre-defined tasks and monitor the percentage of test tasks on which the agent obtained an episodic reward greater than 230 (i.e. "mastered" tasks), which corresponds to agents that were able to reach the last portion of the map (in both Stump Tracks and Parkour). We compare performance results using Welch's t-test as proposed in Colas et al. (2018), allowing us to track statistically significant differences between two methods. We perform a hyperparameter search for all ACL conditions through grid-search (see appendix A), while controlling that an equivalent number of configurations are tested for each algorithm. See appendix C for additional experimental details.

5.2. Challenge-specific comparison with Stump Tracks

First, we aim to compare the different ACL methods on each of the six challenges we identified and listed in section 1. For this, we propose to leverage the Stump Tracks environment to create five experiments, each of them designed to highlight the ability of a teacher in one the first five ACL challenges (see appendix C for details):

- *Mostly unfeasible task space*: growing the possible maximum height of stumps, leading to almost 80% of unfeasible tasks.
- *Mostly trivial task space*: allowing to sample stumps with negative height introducing 50% of new trivial tasks.
- *Forgetting student*: resetting the DRL model twice throughout learning (i.e. every 7 Millions steps).
- *Diverse students*: using multiple embodiments (short bipedal and spider) and DRL students (SAC and PPO).
- *Rugged difficulty landscape*: Applying a random transformation to the task space such that feasible tasks are scattered across the space (i.e. among unfeasible ones).

Additionally, in order to compare methods on the last challenge (i.e. the need of prior knowledge), we propose to perform each of our five experiments in three conditions:

- *No expert knowledge*: None of the prior knowledge listed in table 1 is given. Hence only methods not requiring it can run in this setup.
- *Low expert knowledge*: Only reward mastery range information is accessible. We consider this as low prior knowledge as, while it requires some global knowledge about the task space, it does not require assumptions on the difficulty of specific subspaces of the task space.
- *High expert knowledge*: All the expert knowledge listed in table 1 is given.

Note that in the *No expert knowledge* and *Low expert knowledge* setups, SPDL (and ADR in *Low expert knowledge*) uses an initial task distribution randomly chosen as a subset of the task space. Moreover, in order to make a fair comparison in the *High expert knowledge* condition, we modified the vanilla version of Covar-GMM and ALP-GMM such that they can use an expert-given initial task distribution.

Using these 15 experiments (5 challenges in 3 expert knowledge setups), we here introduce what is, to our knowledge, the first unit-test like experiment of ACL methods, allowing one to compare teachers in each of the challenges we previously introduced. Moreover, performing each of the five experiments in three expert knowledge setups allows to show how the (un)availability of expert knowledge impacts performance for each method, which is hard to infer from each approach's original paper as they tend to focus only on the most ideal scenario. See appendix C for a detailed explanation of each experimental setup.

To conduct our analysis, each ACL method is used in 15 experiments with 32 seeds, except ADR, GoalGAN and Setter-Solver which cannot run in the *No expert knowledge* setup (i.e. only 10 experiments). We then calculate the aforementioned percentage of mastered test tasks on our test set (identical for all experiments), and average it over seeds. Performance results of all conditions can be visualized in figure 3 as a ratio of the Random teachers' performance, our lower-baseline.

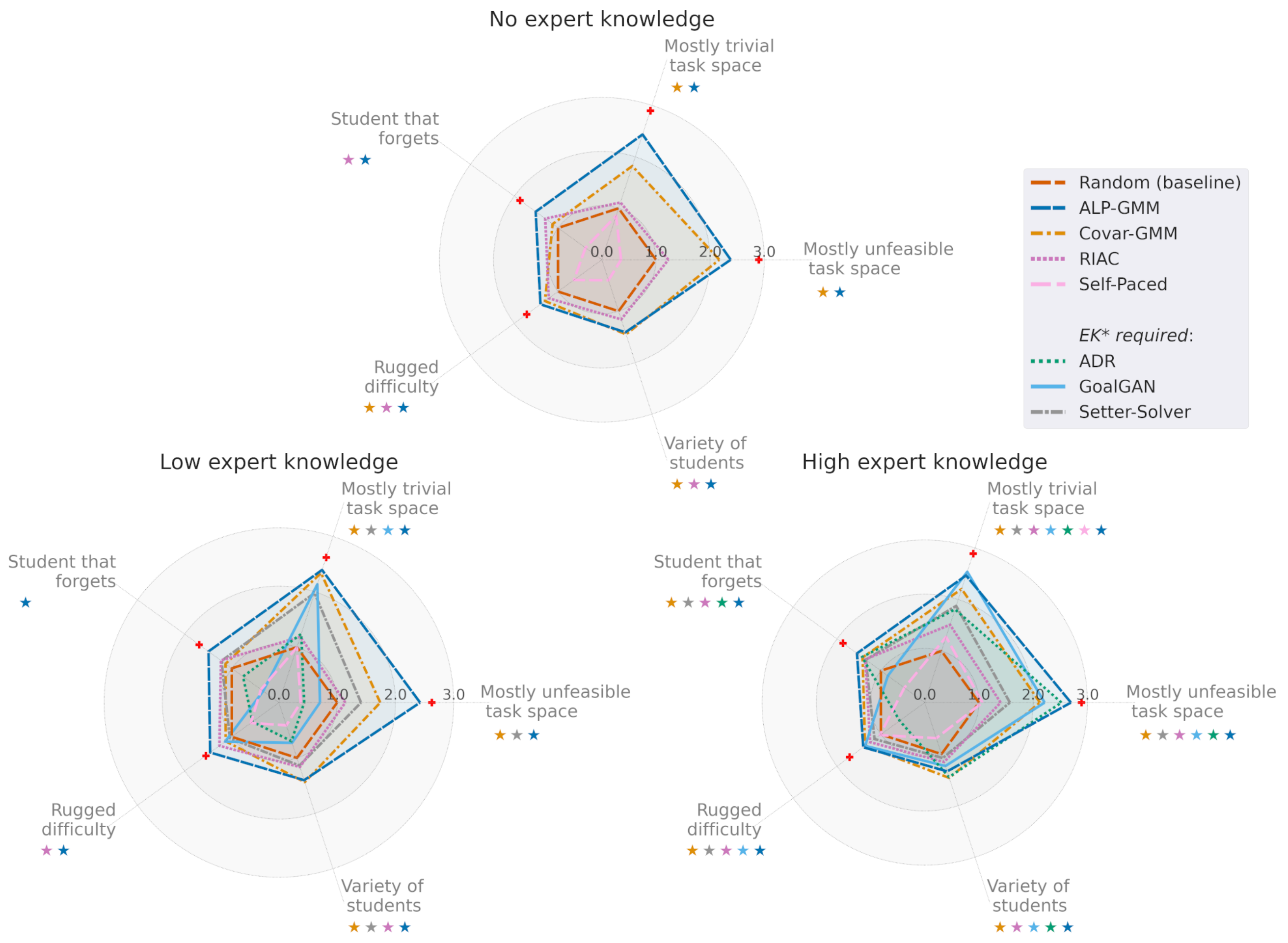

Figure 3. EK: Expert Knowledge. Post-training performance of each ACL method as a ratio of Random's results on multiple teaching challenges, done with 3 different expert knowledge levels. We use ✚ to show estimations of upper-bound performances in each challenge, except for *Variety of students* (see appendix D.1). On each axis, we indicate which method performed significantly better than Random ($p < 0.05$) using colored stars matching each method's color (e.g. ★ for Covar-GMM, ★ for ADR). See appendix D.2 for details.

Results We gather the results in figure 3 as well as in appendix D.2.

Expert-knowledge-free methods – Using these, one can see, first, that methods not requiring any expert knowledge (e.g. ALP-GMM or Covar-GMM) obtain very similar performances in *No expert knowledge* and in *High expert knowledge* setups (although expert knowledge does benefit them in terms of sample efficiency, see app. D.2 for details). Comparing their performance without prior knowledge to the results obtained by other teachers when they have access to high expert knowledge shows how competitive expert-knowledge-free methods can be.

Expert knowledge dependency – The *Low expert knowledge* setup highlights the dependence of methods relying on an initial distribution of easy tasks (e.g. ADR and GoalGAN), as it is not given in this scenario. As a result, in this setup, ADR obtains end performances not significantly different from Random in all challenges, and GoalGAN only outperforms Random in the mostly trivial task space ($p < 0.05$). This has to be compared with their performance on the *High expert knowledge* setup, in which both approaches reach the top 3 results on 3/5 challenges.

ADR & GoalGAN – Both ADR and GoalGAN have one strong weakness in a challenge (*Rugged difficulty* for ADR and *Forgetting student* for GoalGAN) that lead them to a performance worse than Random (significantly for ADR with $p < 0.05$) in all expert knowledge setups. For ADR, it can be explained by the fact that its expansion can get stuck by subspaces of very hard (or unfeasible) difficulty, and for GoalGAN, by its inability to adapt quickly enough to the student's regressing capabilities because of its inertia to update its sampling distribution (updating the buffer and training the GAN). We provide a more in-depth analysis of these two cases in appendix D.2.

SPDL – One can see that SPDL's performance seem very poor in our experimental setup: its end performance is significantly inferior to Random in 11/15 experiments ($p < 0.05$). This can be explained by the fact that SPDL, by design, optimizes performance over a Gaussian target distribution, while our test set is uniformly sampled over the task space. See appendix A for details and potential fixes.

5.3. Global performance analysis using the Parkour

The second experiment we propose aims to more broadly benchmark ACL methods' performance in the Parkour environment, which features most of the previously discussed ACL challenges: 1) most tasks are unfeasible, 2) before each run, unknown to the teacher, the student's embodiment is uniformly sampled among three morphologies (bipedal walker, fish and chimpanzee), requiring the teacher to adapt curriculum generation to a diversity of student profiles, and 3) tasks are generated through a CCPN-based PCG, creating a rich task space with rugged difficulty landscape and hardly-definable prior knowledge (see appendix B).

We perform 48 seeded experiments (i.e. 16 seeds per morphology). To evaluate performance, three test sets were hand-designed (one per embodiment) such that each contains an even distribution between easy, medium and hard tasks. In terms of expert knowledge for teachers, we only give reward mastery range. Without straightforward initial easy task distribution to give to teachers requiring such knowledge (ADR and SPDL), we set it randomly over the space for each new run. See appendix C for details.

Results We present the evolution of performance of each teacher averaged over all seeds (and thus all embodiments) in figure 4 and gather the detailed results in appendix D.3. Interestingly, one can observe that best-performing methods do not use expert knowledge. This is explained by the fact that few prior knowledge is provided to the teachers in these experiments and, as shown in the challenge-specific experiments, most methods using expert knowledge heavily rely on them to reach high-performance. However, one can see that, while SPDL and Setter-Solver remain at the performance level of Random, GoalGAN's performance along training is (mostly) not significantly different from those of Covar-GMM and RIAC, two methods not relying on expert knowledge, as opposed to GoalGAN. On his side, ADR seems to plateau very fast and finally reach an average performance significantly worse than Random ($p < 0.05$, see figure 14). Indeed, as the difficulty landscape of the Parkour environment is rugged, and the initial "easy" task distribution randomly set, ADR is unable to progressively grow its sampling distribution towards feasible subspaces. Finally, when looking specifically to each embodiment type, results show the incapacity of all teachers to make the DRL student learn an efficient policy with the climbing morphology (i.e. at most 1% of mastered tasks by the end of training across all teachers), although we are able to show that high-performing policies can be learned when considering a subspace of the task space (see our case study in appendix D.3). This might be due to the complexity of learning the climbing gait w.r.t walking or swimming, as it requires for instance good coordination skills between the arms and the grasping actions. For the two other morphologies (bipedal walker and fish), results obtained are also low (respectively less than 60% and 50%) and have a high variance (especially for the fish) considering that our test sets contain feasible tasks. This makes the Parkour environment an open challenge for future work on designing ACL algorithms.

6. Open-Source release of *TeachMyAgent*

With the open-source release of *TeachMyAgent* (version *1.0*), we hope to provide a tool that can be used as a step towards thorough comparison and better understanding of current and future ACL methods. *TeachMyAgent*'s documented repository features the code of our environments, embodiments, DRL students, as well as implementations of all ACL methods compared in this paper. All of these parts use APIs we provide such that one can easily add its ACL method, learning algorithm, and new embodiment or environment. We hope this will foster community-driven contributions to extend *TeachMyAgent* in order to broaden its impact and adapt it to the future of ACL. We also provide the code we used to reproduce our experiments, as well as Jupyter notebooks allowing to generate all the figures showed in this paper. Finally, we release the results of our benchmark, allowing one to load them and compare its ACL method against baselines without having to reproduce our large-scale experiments.

7. Discussion and Conclusion

In this article we presented *TeachMyAgent 1.0*, a first extensive testbed to design and compare ACL algorithms. It features unit-tests environments to assess the efficiency of a given teacher algorithm on multiple core skills and the Parkour environment, which provides a challenging teaching scenario that has yet to be solved. We used *TeachMyAgent* to conduct a comparative study of existing ACL algorithms. Throughout our experiments, we identified that 1) current ACL approaches not using expert knowledge matched and even outperformed (e.g. ALP-GMM) other approaches using high amounts of expert knowledge, and 2) the Parkour environment is far from solved, which makes it a good candidate as a testbed when designing new ACL approaches.

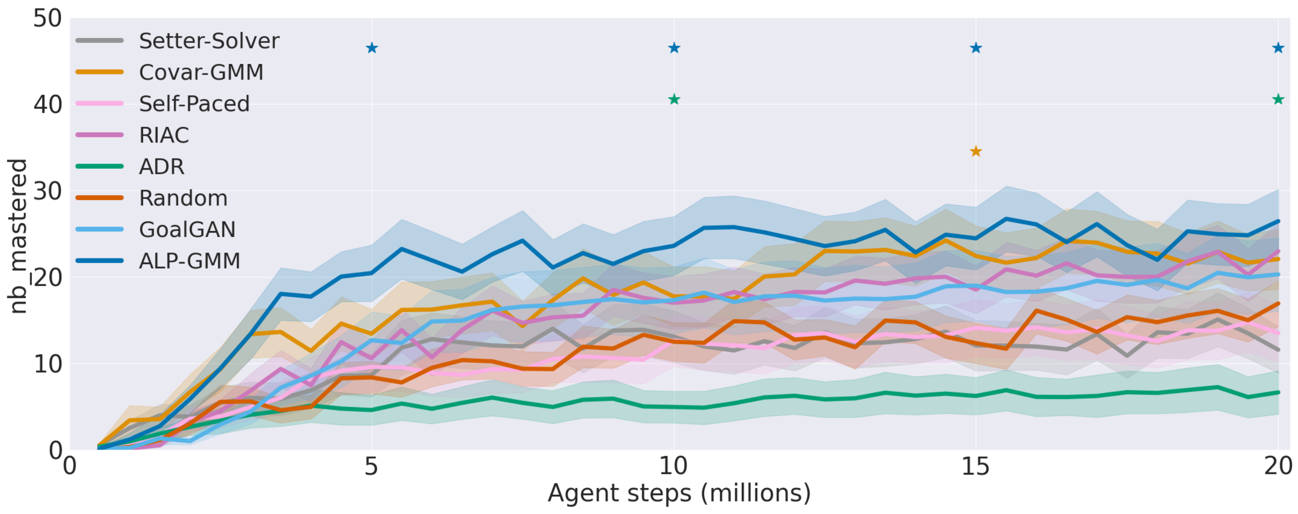

Figure 4. Averaged performance (48 seeds, with standard error of the mean) for each ACL method on Parkour. We calculate every 5 millions steps which method obtained statistically different ($p < 0.05$) results from Random and indicate it with a star.

Limitations & future work. An obvious extension of this work is the addition of recent ACL approaches proposed during or after our experimental campaign (Zhang et al., 2020; Jiang et al., 2020). So far, all studied ACL algorithms struggled to detect feasible task subspaces in Parkour, hinting that more research is needed to improve the "progress niche detection" ability of current teacher algorithms.

TeachMyAgent currently only features environments with low-dimensional PCG systems. Designing new environments with higher-dimensional PCG, that might require to learn low dimensional representations on which to apply ACL algorithms, is an interesting avenue. Besides, our current list of environments only studies 2D locomotion tasks inspired by ALP-GMM's original paper (Portelas et al., 2019) as well as other works on Deep RL and 2D locomotion (Ha, 2019; Song et al., 2018; Gaier & Ha, 2019; Wang et al., 2019; 2020). While we put maximal effort in building a thorough and fair analysis of ACL methods, we believe extending *TeachMyAgent* with other environments (e.g. ProcGen (Cobbe et al., 2020), robotic manipulation) would make the benchmark even more informative.

Additionally, extending the benchmark to consider environment-conditioned goal selection (Racanière et al., 2020; Campero et al., 2020) – i.e. where teachers have to observe the initial episode state to infer admissible goals – is also worth investigating. *TeachMyAgent* provides a distribution of diverse learners. To this respect, it could also serve as a testbed for *Meta ACL* (Portelas et al., 2020b), i.e. algorithms *learning to learn to teach* across a sequence of students.

Acknowledgments

This work was supported by Microsoft Research through its PhD Scholarship Programme. Experiments presented in this paper were carried out using 1) the PlaFRIM experimental testbed, supported by Inria, CNRS (LABRI and IMB), Université de Bordeaux, Bordeaux INP and Conseil Régional d'Aquitaine (see https://www.plafrim.fr/), 2) the computing facilities MCIA (Mésocentre de Calcul Intensif Aquitain) of the Université de Bordeaux and of the Université de Pau et des Pays de l'Adour, and 3) the HPC resources of IDRIS under the allocation 2020-[A0091011996] made by GENCI.

References

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein GAN. *arXiv:1701.07875 [cs, stat]*, December 2017. arXiv: 1701.07875.

Baranes, A. and Oudeyer, P. R-IAC: robust intrinsically motivated exploration and active learning. *IEEE Trans. Autonomous Mental Development*, 1(3):155–169, 2009.

Beattie, C., Leibo, J. Z., Teplyashin, D., Ward, T., Wainwright, M., Küttler, H., Lefrancq, A., Green, S., Valdés, V., Sadik, A., Schrittwieser, J., Anderson, K., York, S., Cant, M., Cain, A., Bolton, A., Gaffney, S., King, H., Hassabis, D., Legg, S., and Petersen, S. Deepmind lab. *CoRR*, abs/1612.03801, 2016.

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, Jun 2013. ISSN 1076-9757.

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. Curriculum learning. In *ICML*, 2009.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *CoRR*, abs/1606.01540, 2016.

Burda, Y., Edwards, H., Storkey, A. J., and Klimov, O. Exploration by random network distillation. *ICLR*, 2019.

Campbell, C. Buoyancy - Box2D tutorials - iforce2d, 2013. URL `https://www.iforce2d.net/b2dtut/buoyancy`.

Campero, A., Raileanu, R., Küttler, H., Tenenbaum, J. B., Rocktäschel, T., and Grefenstette, E. Learning with amigo: Adversarially motivated intrinsic goals. *CoRR*, abs/2006.12122, 2020.

Cangelosi, A. and Schlesinger, M. *Developmental robotics: From babies to robots*. MIT press, 2015.

Chevalier-Boisvert, M., Bahdanau, D., Lahlou, S., Willems, L., Saharia, C., Nguyen, T. H., and Bengio, Y. Babyai: A platform to study the sample efficiency of grounded language learning. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

Cobbe, K., Klimov, O., Hesse, C., Kim, T., and Schulman, J. Quantifying generalization in reinforcement learning. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1282–1289. PMLR, 2019.

Cobbe, K., Hesse, C., Hilton, J., and Schulman, J. Leveraging procedural generation to benchmark reinforcement learning. *ICLR*, 2020.

Colas, C., Sigaud, O., and Oudeyer, P.-Y. How Many Random Seeds? Statistical Power Analysis in Deep Reinforcement Learning Experiments. *arXiv:1806.08295 [cs, stat]*, July 2018. arXiv: 1806.08295.

Colas, C., Oudeyer, P.-Y., Sigaud, O., Fournier, P., and Chetouani, M. Curious: Intrinsically motivated modular multi-goal reinforcement learning. In *ICML*, 2019.

Côté, M., Kádár, Á., Yuan, X., Kybartas, B., Barnes, T., Fine, E., Moore, J., Hausknecht, M. J., Asri, L. E., Adada, M., Tay, W., and Trischler, A. Textworld: A learning environment for text-based games. In Cazenave, T., Saffidine, A., and Sturtevant, N. (eds.), *Computer Games - 7th Workshop, CGW 2018, Held in Conjunction with the 27th International Conference on Artificial Intelligence, IJCAI 2018, Stockholm, Sweden, July 13, 2018, Revised Selected Papers*, volume 1017 of *Communications in Computer and Information Science*, pp. 41–75. Springer, 2018.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using real NVP. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

Elman, J. L. Learning and development in neural networks: the importance of starting small. *Cognition*, 48(1):71 – 99, 1993. ISSN 0010-0277.

Florensa, C., Held, D., Geng, X., and Abbeel, P. Automatic goal generation for reinforcement learning agents. In *ICML*, 2018.

Forestier, S., Mollard, Y., and Oudeyer, P. Intrinsically motivated goal exploration processes with automatic curriculum learning. *CoRR*, abs/1708.02190, 2017.

Gaier, A. and Ha, D. Weight agnostic neural networks. In Wallach, H., Larochelle, H., Beygelzimer, A., Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative Adversarial Nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 27*, pp. 2672–2680. Curran Associates, Inc., 2014.

Graves, A., Bellemare, M. G., Menick, J., Munos, R., and Kavukcuoglu, K. Automated curriculum learning for neural networks. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1311–1320. PMLR, 2017.

Ha, D. Generating abstract patterns with tensorflow. *blog.otoro.net*, 2016.

Ha, D. Reinforcement learning for improving agent design. *Artif. Life*, 25(4):352–365, 2019.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *ICML*, 2018.

Hernandez, D., Denamganaï, K., Gao, Y., York, P., Devlin, S., Samothrakis, S., and Walker, J. A. A generalized framework for self-play training. In *IEEE CoG*, 2019.

Hernandez-Leal, P., Kartal, B., and Taylor, M. E. Is multiagent deep reinforcement learning the answer or the question? A brief survey. *CoRR*, abs/1810.05587, 2018.

Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., and Silver, D. Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Hessel, M., Soyer, H., Espeholt, L., Czarnecki, W., Schmitt, S., and Van Hasselt, H. Multi-task deep reinforcement learning with popart. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:3796–3803, Jul 2019. ISSN 2159-5399.

Jabri, A., Hsu, K., Gupta, A., Eysenbach, B., Levine, S., and Finn, C. Unsupervised curricula for visual meta-reinforcement learning. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 10519–10530, 2019.

Jiang, M., Grefenstette, E., and Rocktäschel, T. Prioritized level replay. *CoRR*, abs/2010.03934, 2020.

Justesen, N., Torrado, R. R., Bontrager, P., Khalifa, A., Togelius, J., and Risi, S. Illuminating generalization in deep reinforcement learning through procedural level generation. *NeurIPS Deep RL Workshop*, 2018.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., and Hadsell, R. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. ISSN 0027-8424.

Klink, P., D'Eramo, C., Peters, J., and Pajarinen, J. Self-paced deep reinforcement learning. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Kovač, G., Laversanne-Finot, A., and Oudeyer, P.-Y. Grimgep: Learning progress for robust goal sampling in visual deep reinforcement learning. *CoRR*, abs/2008.04388, 2020.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pp. 2278–2324, 1998.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. In *ICLR*, 2016.

Matiisen, T., Oliver, A., Cohen, T., and Schulman, J. Teacher-student curriculum learning. *IEEE TNNLS*, 2017.

Mehta, B., Diaz, M., Golemo, F., Pal, C. J., and Paull, L. Active domain randomization. *CoRL*, 2019.

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. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529, 2015.

Moulin-Frier, C., Nguyen, S. M., and Oudeyer, P.-Y. Self-organization of early vocal development in infants and machines: The role of intrinsic motivation. *Front. in Psych. (Cog. Science)*, 2014.

Mysore, S., Platt, R., and Saenko, K. Reward-guided curriculum for robust reinforcement learning. *Workshop on Multi-task and Lifelong Reinforcement Learning at ICML*, 2019.

Nichol, A., Pfau, V., Hesse, C., Klimov, O., and Schulman, J. Gotta learn fast: A new benchmark for generalization in RL. *CoRR*, abs/1804.03720, 2018.

OpenAI, Akkaya, I., Andrychowicz, M., Chociej, M., Litwin, M., McGrew, B., Petron, A., Paino, A., Plappert, M., Powell, G., Ribas, R., Schneider, J., Tezak, N., Tworek, J., Welinder, P., Weng, L., Yuan, Q., Zaremba, W., and Zhang, L. Solving rubik's cube with a robot hand. *CoRR*, abs/1910.07113, 2019.

Osband, I., Doron, Y., Hessel, M., Aslanides, J., Sezener, E., Saraiva, A., McKinney, K., Lattimore, T., Szepesvari, C., Singh, S., Roy, B. V., Sutton, R., Silver, D., and Hasselt, H. V. Behaviour suite for reinforcement learning. *ICML*, 2020.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. In *CVPR*, 2017.

Peng, X. B., Andrychowicz, M., Zaremba, W., and Abbeel, P. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*, pp. 1–8. IEEE, 2018.

Pong, V., Dalal, M., Lin, S., Nair, A., Bahl, S., and Levine, S. Skew-fit: State-covering self-supervised reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 7783–7792. PMLR, 2020.

Portelas, R., Colas, C., Hofmann, K., and Oudeyer, P.-Y. Teacher algorithms for curriculum learning of deep rl in continuously parameterized environments. *CoRL*, 2019.

Portelas, R., Colas, C., Weng, L., Hofmann, K., and Oudeyer, P.-Y. Automatic curriculum learning for deep rl: A short survey. *IJCAI*, 2020a.

Portelas, R., Romac, C., Hofmann, K., and Oudeyer, P. Meta automatic curriculum learning. *CoRR*, abs/2011.08463, 2020b.

Racanière, S., Lampinen, A., Santoro, A., Reichert, D., Firoiu, V., and Lillicrap, T. Automated curricula through setter-solver interactions. *ICLR*, 2020.

Rajeswaran, A., Ghotra, S., Ravindran, B., and Levine, S. Epopt: Learning robust neural network policies using model ensembles. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

Risi, S. and Togelius, J. Procedural content generation: From automatically generating game levels to increasing generality in machine learning. *CoRR*, abs/1911.13071, 2019.

Salimans, T. and Chen, R. Learning montezuma's revenge from a single demonstration. *NeurIPS*, 2018.

Schaul, T., Horgan, D., Gregor, K., and Silver, D. Universal value function approximators. In *ICML*, 2015.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

Selfridge, O. G., Sutton, R. S., and Barto, A. G. Training and tracking in robotics. In *IJCAI*, 1985.

Song, D. R., Yang, C., McGreavy, C., and Li, Z. Recurrent deterministic policy gradient method for bipedal locomotion on rough terrain challenge. In *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pp. 311–318, 2018. doi: 10.1109/ICARCV.2018.8581309.

Stanley, K. O. Compositional pattern producing networks: A novel abstraction of development. *Genet. Program. Evolvable Mach.*, 8(2):131–162, 2007.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada, September 24-28, 2017*, pp. 23–30. IEEE, 2017.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012.

Vanschoren, J. Meta-learning: A survey. *CoRR*, abs/1810.03548, 2018.

Wang, R., Lehman, J., Clune, J., and Stanley, K. O. Paired open-ended trailblazer (POET): endlessly generating increasingly complex and diverse learning environments and their solutions. *CoRR*, abs/1901.01753, 2019.

Wang, R., Lehman, J., Rawal, A., Zhi, J., Li, Y., Clune, J., and Stanley, K. O. Enhanced POET: open-ended reinforcement learning through unbounded invention of learning challenges and their solutions. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9940–9951. PMLR, 2020.

Zhang, C., Vinyals, O., Munos, R., and Bengio, S. A study on overfitting in deep reinforcement learning. *CoRR*, abs/1804.06893, 2018.

Zhang, Y., Abbeel, P., and Pinto, L. Automatic curriculum learning through value disagreement. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.