Scalable Evaluation of Multi-Agent Reinforcement Learning with Melting Pot

Joel Z. Leibo [*1] **Edgar Duéñez-Guzmán** [*1] **Alexander Sasha Vezhnevets** [*1] **John P. Agapiou** [*1] **Peter Sunehag** [1] **Raphael Koster** [1] **Jayd Matyas** [1] **Charles Beattie** [1] **Igor Mordatch** [2] **Thore Graepel** [1]

Abstract

Existing evaluation suites for multi-agent reinforcement learning (MARL) do not assess generalization to novel situations as their primary objective (unlike supervised-learning benchmarks). Our contribution, Melting Pot, is a MARL evaluation suite that fills this gap, and uses reinforcement learning to reduce the human labor required to create novel test scenarios. This works because one agent's behavior constitutes (part of) another agent's environment. To demonstrate scalability, we have created over 80 unique test scenarios covering a broad range of research topics such as social dilemmas, reciprocity, resource sharing, and task partitioning. We apply these test scenarios to standard MARL training algorithms, and demonstrate how Melting Pot reveals weaknesses not apparent from training performance alone.

1. Introduction

No broadly accepted benchmark test set for multi-agent reinforcement learning (MARL) research yet exists. This lack of a standardized evaluation protocol has impeded progress in the field by making it difficult to obtain like-for-like comparisons between algorithms. The situation in MARL is now in stark contrast to the status quo in single-agent reinforcement learning (SARL) where a diverse set of benchmarks suitable for different purposes are available (e.g. Brockman et al. (2016); Fortunato et al. (2019); Machado et al. (2018); Osband et al. (2019); Tassa et al. (2018); Torrado et al. (2018)). Further afield, the comparison to the evaluation landscape in other machine learning subfields is even more unfavorable. Datasets like ImageNet (Deng et al., 2009) and their associated evaluation methodology achieve a level of rigor and community acceptance unparalleled in reinforcement learning.

Within the SARL research community there have been several recent calls to import the methodological stance concerning the primacy of generalization from supervised learning (Cobbe et al., 2019; Farebrother et al., 2018; Juliani et al., 2019; Zhang et al., 2018a;b). However, MARL research is still lagging behind. No one has yet attempted to build a benchmark with the explicit aim of pushing standards for judging multi-agent reinforcement learning research toward making generalization the first-and-foremost goal.

Supervised learning research benefits immensely from having a clear experimental protocol and set of benchmarks that explicitly measure how well methods generalize outside the data to which they were exposed in training (Chollet, 2019; Deng et al., 2009; LeCun et al., 2015). This facilitates clear like-for-like comparison between methods, channeling competition between research groups, and driving progress. One problem that arises when trying to import these ideas to reinforcement learning however is that generating a test set of environments is a lot more labor intensive than labeling a set of images. The engineering challenge of creating just a single test environment is akin to designing and implementing a new computer game. Thus calls to appreciate the primacy of generalization in SARL appear sometimes to justify a Sisyphean struggle to create ever more clever and more diverse intelligence tests.

This obstacle to scalability turns out to be much less severe for research aimed at multi-agent intelligence. In fact, multi-agent approaches have a natural advantage over single-player approaches in the measurement arena. In multi-agent systems, agents naturally pose tasks to one another. Any change to the policy of one agent changes the environment experienced by a set of interdependent others. For instance, if a focal agent learns an effective policy against a fixed set of co-players, it could be rendered useless if the co-players change. This aspect of multi-agent learning is more commonly associated with proposals for "training time" ideas like autocurricula (Baker et al., 2019; Bansal et al., 2017; Leibo et al., 2019a; Sukhbaatar et al., 2017) and open-endedness (Clune, 2019). Here however, we propose to make a different use of it. We can take advantage of multi-agent interaction to create large and diverse sets of generalization tests by pre-training "background populations" of agents to use for subsequent evaluation *only*, never training on them—much like the test images in the

[*]Equal contribution [1]DeepMind [2]Google Brain. Correspondence to: Joel Z. Leibo <jzl@google.com>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

ImageNet challenge for supervised learning.

Our proposal, Melting Pot, consists of an evaluation methodology and a suite of specific test environments. Its essence is embodied in its central "equation":

Substrate + Background Population = Scenario

A *scenario* is a multi-agent environment that we use only for testing; we do not allow agents to train in it. The term *substrate* refers to the physical part of the world, it includes: the layout of the map, where the objects are, how they can move, the rules of physics, etc. The term *background population* refers to the part of the simulation that is imbued with agency—excluding the *focal population* of agents being tested.

The Melting Pot research protocol aims to assess and compare multi-agent reinforcement learning algorithms. It is only concerned with test-time evaluation, and so is mostly agnostic to training method. That is, training-time access to each test's substrate is allowed but we do not mandate how to use it. The suite consists of a collection of zero-shot—i.e. not allowing for test-time learning—test scenarios that preserve a familiar substrate while substituting a new and unfamiliar background population.

Our intention is for Melting Pot to cover the full breadth of different types of strategic situations commonly studied in multi-agent reinforcement learning[1]. As such, we have included purely competitive games, games of pure common interest, team-based competitive games, and a range of different kinds of mixed-motivation games including prisoner's dilemma-like social dilemmas and games that stress coordination. The numbers of simultaneous players in each game range from two to sixteen and most substrates have around eight.

Finally, we provide benchmark results on Melting Pot for several different MARL models. Intriguingly, we find that maximizing collective reward often produces policies that are less robust to novel social situations than the policies obtained by maximizing individual reward.

2. What does Melting Pot evaluate?

We use the term *multi-agent population learning algorithm* (MAPLA) to refer to any training process that produces a decentralized population of agents capable of simultaneous interaction with one another. Melting Pot evaluates MAPLAs on their fulfillment of three desiderata. They are best introduced by way of an example. Consider the following problem faced by a manufacturer of self-driving cars. The goal is to build a population of agents that will act simultaneously in the world as decentralized individuals. They do not necessarily know in advance whether their cars will be a small minority of the overall number of vehicles on the road, interacting with large numbers of human drivers and self-driving cars from competing manufacturers, or whether their product might prove so popular that it rapidly becomes a majority of the cars on the road. This fraction may even change dynamically from day to day (consider: a competitor might recall their product, or human driving could become illegal). The self-driving fleet (a multi-agent population) must then satisfy the following:

Individual agents and sets of agents sampled from the population must:

1. perform well across a range of social situations where individuals are interdependent,
2. generalize to interact effectively with unfamiliar individuals not seen during training (who may well be human), and
3. pass a universalization test: answering positively to the question "what if everyone behaved like that?".

The class of MAPLA algorithms is very broad. Most multi-agent reinforcement learning approaches can be made to produce populations. For instance self-play schemes like those used for AlphaGo (Silver et al., 2016; 2017), AlphaZero (Silver et al., 2018), FTW (Capture the Flag) (Jaderberg et al., 2019), hide and seek (Baker et al., 2019), and AlphaStar (Vinyals et al., 2019) fit in the class of MAPLAs, as does recent work on DOTA (Berner et al., 2019) and MOBA (Ye et al., 2020) games, as well as algorithms like MADDPG (Lowe et al., 2017), LOLA (Foerster et al., 2018), PSRO (Lanctot et al., 2017), PSRO$_{rN}$ (Balduzzi et al., 2019), and Malthusian reinforcement learning (Leibo et al., 2019b).

3. Related work

The idea to use agents to create tasks for other agents appears in research on competitive games, such as Capture the Flag (Jaderberg et al., 2019), DOTA (Berner et al., 2019) and StarCraft II (Vinyals et al., 2019). There evaluation against held-out agents was successfully used to measure generalization, and ultimately beat human professionals. This idea also appears in contexts where agent interaction is used to drive learning (Racaniere et al., 2019; Wang et al., 2019). In these cases it was not used to create benchmarks for generalisation. Zero-shot transfer to new co-players in coordination games was investigated in Hu et al. (2020).

[1]The largest category of extant research that we left unrepresented is communication/language (e.g. Lazaridou & Baroni (2020); Lowe et al. (2019); Mordatch & Abbeel (2018)). We see no reason why scenarios engaging these ideas could not be added in the future. Turn-based games (e.g. Lanctot et al. (2019)) and games involving physics (e.g. Liu et al. (2018)) were also omitted.

Several papers Song et al. (2020b); Lowe et al. (2017) have introduced MARL benchmarks for specific domains, but do not measure generalisation and don't use learning agents to produce evaluation tasks. Another approach with a long history in game theory involves organizing a competition between strategies submitted by different research groups (e.g. Axelrod (1984)). Doing well in such a competition involves generalization since a submitted agent must play with the other researchers' submissions. Perez-Liebana et al. (2019) brought this competition approach to MARL. Differing from all these approaches, Melting Pot covers a broader range of multi-agent interactions and is more explicitly focused on providing a benchmark for generalization.

4. The Melting Pot protocol

Our term, *substrate*, refers to a partially observable general-sum Markov game (e.g. Shapley (1953); Littman (1994)). In each game state, agents take actions based on a partial observation of the state space and receive an individual reward. The rules of the game are not given to the agents; they must explore to discover them. Thus a Melting Pot substrate is simultaneously a game of *imperfect* information—each player possesses some private information not known to their coplayers (as in card-games like poker)—and *incomplete* information—lacking common knowledge of the rules (Harsanyi, 1967). We describe the various substrates available in Melting Pot in Section 5.

Formally, a substrate is an N-player partially observable Markov game $\mathcal{M}$ defined on a finite set of states $\mathcal{S}$, observations $\mathcal{X}$, and actions $\mathcal{A}$. The observation function $\mathcal{O} : \mathcal{S} \times \{1, \ldots, N\} \to \mathcal{X}$, specifies each player's view of the state space. In each state, each player i selects an individual action $a_i \in \mathcal{A}$. Following their joint action $\mathbf{a} = (a_1, \ldots, a_N) \in \mathcal{A}^N$, the state change obeys the stochastic transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A}^N \to \Delta(\mathcal{S})$, where $\Delta(\mathcal{S})$ denotes the set of discrete probability distributions over $\mathcal{S}$. After a transition, each player receives an individual reward defined by $\mathcal{R} : \mathcal{S} \times \mathcal{A}^N \times \{1, \ldots, N\} \to \mathbb{R}$.

A *policy* $\pi : \mathcal{X} \times \mathcal{A} \times \mathcal{X} \times \mathcal{A} \times \cdots \times \mathcal{X} \to \Delta(\mathcal{A})$ is a probability distribution over a single agent's actions, conditioned on that agent's history of observations and previous actions. Policies are not transferable between substrates, since $\mathcal{X}$ and $\mathcal{A}$ can differ between them. Let $\pi \in \Pi_{\mathcal{M}}$ indicate a policy defined on the substrate $\mathcal{M}$, and consider a *joint policy* formed by a tuple of individual policies $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_n) \in {\Pi_{\mathcal{M}}}^n$. We call this (factorized) joint policy *compatible* with the N-player substrate $\mathcal{M}$ if $n = N$. A compatible joint policy is necessary to sample episodes from the interaction between the individual policies and the substrate.

Given a compatible joint policy $\boldsymbol{\pi}$ on substrate $\mathcal{M}$, we measure the performance of each individual policy within this context as the *individual return* $R_i(\boldsymbol{\pi}|\mathcal{M})$—the expected total reward for player i. We then measure the performance of the joint policy using the *per-capita return*—the mean individual return:

$$\bar{R}(\boldsymbol{\pi}|\mathcal{M}) = \frac{1}{N}\sum_{i=1}^{N} R_i(\boldsymbol{\pi}|\mathcal{M})$$

A *population* for an N-player substrate $\mathcal{M}$ is a distribution $f(\Pi_{\mathcal{M}})$ over individual policies. A population for $\mathcal{M}$ can therefore create a compatible joint policy $\boldsymbol{\pi}$ for $\mathcal{M}$ by independently sampling N individual policies from f: $\boldsymbol{\pi} \sim f(\pi_1) \ldots f(\pi_N)$. Performance on a substrate by a population is measured by the expected per-capita return:

$$\bar{R}(f|\mathcal{M}) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{\pi_1 \sim f} \ldots \mathbb{E}_{\pi_N \sim f} R_i(\boldsymbol{\pi}|\mathcal{M})$$

4.1. Testing

Let a *scenario configuration* for an N-player substrate $\mathcal{M}$ be a binary vector $\mathbf{c} = (c_1, \ldots, c_N) \in \{0, 1\}^N$ of n zeros and m ones that indicates whether each player i is a *focal player* ($c_i = 1$), or a *background player* ($c_i = 0$). Let the *background population* for a substrate $\mathcal{M}$ be a distribution $g(\Pi_{\mathcal{M}})$ that is used to sample individual policies for the background players. Now, from the perspective of the focal players, the N-player substrate $\mathcal{M}$ is *reduced* to an equivalent m-player substrate $\mathcal{M}'$ (via marginalization of the transition and reward functions by the background player policies). We call this reduced m-player substrate —formed from a substrate, configuration, and background population—a *test scenario*.

Performance on a test scenario by a *focal population* f is measured by the expected per-capita return as before, except that background players are excluded from the mean:

$$\begin{aligned}\bar{R}(f|\mathcal{M}, \mathbf{c}, g) =& \bar{R}(f|\mathcal{M}') \\ =& \frac{1}{m}\sum_{i=1}^{N} c_i \mathbb{E}_{\pi_1 \sim h_1} \ldots \mathbb{E}_{\pi_N \sim h_N} R_i(\boldsymbol{\pi}|\mathcal{M})\end{aligned}$$

where $h_i(\pi) = f(\pi)^{c_i} g(\pi)^{1-c_i}$.

When focal players outnumber background players, we say the test scenario is in *resident* mode. These scenarios usually test the emergent cooperative structure of the population under evaluation for its robustness to interactions with a minority of unfamiliar individuals not encountered during training. When background players outnumber focal players, we say the test scenario is in *visitor* mode. One common use case for visitor-mode scenarios in Melting Pot is to test whether an individual from the focal population can observe

the conventions and norms of the dominant background population and act accordingly (without retraining).

We use another kind of test scenario to test universalization. In this case, we have no background players, but instead of independently sampling each focal policy π_i from f, we sample from f once and use this policy repeatedly. So the joint policy consists of N copies of the same policy $(\pi, \ldots, \pi) \in \Pi^N$ where $\pi \sim f$. We call this the *universalization* mode. It answers the rhetorical question "how would you like it if everyone behaved like you?". Such universalization is an important part of human moral psychology (Levine et al., 2020), at least in some cultures (Henrich, 2020). Because this universalization test scenario does not require a background population, it could easily be incorporated into a training process, in which case it would not be a test of generalization. However, we included it because it is useful in diagnosing failure cases.

4.2. Training

During training, MAPLA F is provided with unlimited access to a substrate $\mathcal{M}$, which it uses to train a population f. Thus, the whole training process may be represented as $F[\mathcal{M}] \mapsto f(\Pi_{\mathcal{M}})$. The purpose of Melting Pot is to evaluate the MAPLA by measuring the performance of the resulting *focal population* f.

We do this by measuring the per-capita return of the focal population when used to sample focal players in our test scenarios. Note that the learning algorithms only have access to the raw substrates and not the background populations. This means that policies sampled from the focal population must show good zero-shot generalization to unseen test scenarios.

Consider a MAPLA where N separate policies are trained together in a N-player substrate, and f selects a trained policy uniformly. In order for f to perform well in test scenarios, self-play should have adequately explored similar behavior to that present in the background population.

Our definition of population deliberately penalizes heterogeneous specialization. Since policies must be independently sampled at test-time, the training algorithm cannot control their joint distribution, and so cannot prescribe a fixed division of labor. To perform well on test scenarios, trained agents should be generalists. Division of labor is possible in Melting Pot, but it works best when it self-organizes at test time with individuals taking on their roles in response to the ongoing behavior of others. In the cases where specialization is most important, successful populations should feature significant redundancy in order to be robust enough to do well in Melting Pot.

Melting Pot focuses only on test-time evaluation, and is agnostic to the method of training. For example, during training, the substrate can be augmented to give agents privileged access to the rewards and observations of co-players. This privileged information is not present in test scenarios, so policies that rely on it will generalize poorly. But it can be useful for providing auxiliary targets to improve the training of internal representations (e.g. "centralized training and decentralized execution" (Kraemer & Banerjee, 2016; Lowe et al., 2017; Oliehoek & Amato, 2016)).

4.3. Secondary evaluation metrics

We propose the *focal-population per-capita return* as a primary evaluation metric, to test the performance of a learning algorithm in a novel social situation. This is because, first and foremost, we want Melting Pot to provide a rigorous and clearly interpretable evaluation metric that highlights unsolved problems and compares innovative algorithms to one another.

However, when evaluating the suitability of trained agents for a practical application, there will be additional considerations, which can be assessed from secondary evaluation metrics using the same test scenarios. For example, impacts on the background population may be an indicator of the impacts the trained agents might have on humans in the real world. So we can measure the *background-population per-capita return* to see if it is negatively impacted by the introduction of the focal population. This could be useful to study whether the joint policy of the focal agents produces negative externalities—"side effects" that impact the broader population while sparing the focal population, dovetailing well with research on value alignment and AI safety (Soares & Fallenstein, 2014; Amodei et al., 2016). Or following Perolat et al. (2017), we can measure the *inequality* of the background-population individual returns to see if any benefit or harm arising from having introduced the focal population is fairly shared, perhaps connecting fruitfully to beneficial and cooperative AI research agendas (Russell et al., 2015; Dafoe et al., 2020).

5. Description of the substrates

Fig. 2 provides a rough guide to the strategic and social intelligence concepts covered by the suite. See the appendix for additional details that describe the precise setup for all substrates and test scenarios. Some substrates share common game mechanics. All * *in the Matrix* games all share the same pattern: Players can collect items representing some number of choices (e.g. defect and cooperate or rock, paper, and scissors), and when they encounter each other their inventory is used to dispense rewards according to the payoff matrix of a classic matrix game.

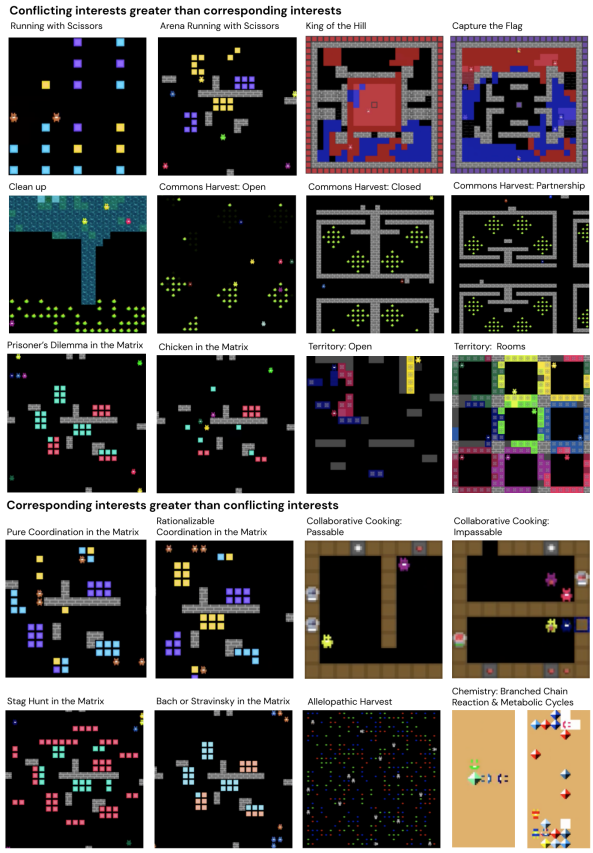

Figure 1. Overview of substrates.

5.1. Conflicting greater than corresponding interests

Running with Scissors in the Matrix first appeared in Vezhnevets et al. (2020). Two individuals gather rock, paper, or scissor resources in the environment, and can challenge others to a 'rock, paper scissor' game, the outcome of which depends on the resources they collected. It is possible (though not trivial) to observe the policy that one's partner is starting to implement, and to take countermeasures. This induces a wealth of possible feinting strategies. *Arena Running with Scissors in the Matrix* extends the game to eight players.

In *Capture the Flag* teams of players can expand their territory by painting the environment, which gives them an advantage in a confrontation with the competing team. The final goal is capturing the opposing team's flag. Payoffs are common to the entire winning team. *King of the Hill* has the same dynamics except the goal is to control the "hill" region in the center of the map. For both substrates there are scenarios where agents play with familiar teammates against unfamiliar opponents as well as scenarios where ad-hoc teamwork is needed (Stone et al., 2010).

Clean up is a social dilemma where individuals have to balance harvesting of berries for reward with cleaning a river that suppresses berry growth if it gets too dirty (Hughes et al., 2018). As cleaning the river is a public good, individuals are motivated to harvest instead of clean.

In *Commons Harvest: Open* individuals harvest apples that fail to regrow if a patch is exhausted. Preserving a patch requires all agents to show restraint in not harvesting the last apple (Perolat et al., 2017). *Commons Harvest: Closed* has the apples in rooms that can be defended by a single player, alleviating the risk of others over-harvesting. In *Commons Harvest: Partnership* it takes two players to defend a room, requiring effective cooperation both in defending and in not over-harvesting.

Prisoner's Dilemma in the Matrix mirrors the classic matrix game that exposes tension between individual and collective reward. In *Chicken in the Matrix*, both players attempting to defect leads to the worst outcome for both. These substrates target similar concepts to the Coins game of Lerer & Peysakhovich (2018), though they are somewhat more complex—in part because they have more players (eight versus two).

In *Territory: Open* individuals can claim a territory for reward by coloring it. They can find a peaceful partition, but also have the option of irreversibly destroying potentially rewarding territory rendering it useless for everyone. *Territory: Rooms* has segregated rooms that strongly suggest a partition individuals could adhere to.

5.2. Corresponding greater than conflicting interests

Collaborative Cooking: Impassable is inspired by (Carroll et al., 2019; Wang et al., 2020)'s work on an Overcooked-like environment. Players need to collaborate to follow recipes, but are separated by an impassable kitchen counter so no player can complete the objective alone. In *Collaborative Cooking: Passable*, players can have access to all sections of the kitchen, which allows individual players to sequentially perform all subtasks unilaterally (but less efficiently).

In *Pure Coordination in the Matrix* all individuals need to converge on the same color choice to gain reward when they encounter each other. Which convention emerges in a given population is entirely arbitrary, and all players are indifferent between the possible conventions. In *Rationalizable Coordination in the Matrix* the choices are of different values, suggesting an optimal color to converge on.

Bach or Stravinsky in the Matrix and *Stag Hunt in the Matrix* focus on coordination. In the former, coordination is tied to unfairness and "stubbornness" could play a role. In the latter, coordination is associated with risk for the individual. It engages similar concepts to Peysakhovich & Lerer (2017)'s Markov Stag Hunt game, though it is more complex—in part

		Substrates																			
		Running with Scissors in the Matrix	Arena Running with Scissors in the Matrix	Capture the Flag	King of the Hill	Clean Up	Commons Harvest: Open	Commons Harvest: Closed	Commons Harvest: Partnership	Prisoner's Dilemma in the Matrix	Chicken in the Matrix	Territory: Open	Territory: Rooms	Collaborative Cooking: Impassable	Collaborative Cooking: Passable	Pure Coordination in the Matrix	Rationizable Coordination in the Matrix	Bach or Stravinsky in the Matrix	Stag Hunt in the Matrix	Allelopathic Harvest	Chemistry: Branched Chain Reaction	Chemistry: Metabolic Cycles
Properties of game	Corresponding > Conflicting interests			x	x	x	x		x	x	x	x	x	x	x	x	x	x	x	x	x	x
	Conflicting > Corresponding Interests	x	x	x	x			x	x													
	Symmetric roles/teams	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x		x	x
	Asymmetric roles/teams					x		x	x		x			x	x			x		x		
	Reversible	x	x	x	x	x				x	x					x	x	x	x	x		
	Irreversible						x	x	x			x	x	x	x						x	x
	Near-perfect information							x	x					x	x							
	Far-from-perfect information	x	x	x	x	x	x			x	x	x	x			x	x	x	x	x		
	1:1 reward interdependence	x	x						x	x	x	x	x			x	x	x	x	x		
	1:Many reward interdependence					x	x	x	x			x	x	x	x							
	Many:Many reward interdependence			x	x									x	x					x	x	x
	Puzzle-like													x	x						x	x
Properties of potential solutions	Temporal coordination			x	x	x			x					x	x							
	Property/Ownership				x			x	x			x	x									
	Reciprocity					x	x		x	x	x	x	x						x	x	x	x
	Fair resource sharing					x	x	x	x			x	x							x	x	x
	Deception	x	x							x	x							x		x		
	Convention following					x					x	x	x	x	x	x	x	x	x	x		
	Task partitioning			x	x	x								x	x						x	x
	Trust & partnership								x	x	x								x		x	
	Free riding					x														x		x
	Stubbornness											x	x					x		x		

Figure 2. Multi-agent concepts engaged by each substrate. Gray ticks represent secondary characteristics. Properties highlighted here are only intended as a rough guide to the concepts at play in each substrate. They should not be taken as a serious theoretical attempt to classify multi-agent situations.

due to it being an eight player game (instead of two-player).

Combining the above dynamics, in *Allelopathic Harvest*, players can increase the growth-rate of berries by planting berries in the same color. However, for each player, a different berry color is intrinsically more rewarding. This creates tensions between groups of players and a free-rider problem between individuals who prefer to consume rather than plant (Köster et al., 2020).

In *Chemistry*, individuals control chemical reactions by transporting molecules around the space. *Chemistry: Branched Chain Reaction* requires alternation between two specific reactions. Combining molecules efficiently requires coordination, but can also lead to exclusion of players. In *Chemistry: Metabolic cycles*, individuals benefit from two different cyclic reaction networks and must coordinate to keep them both running.

6. Extending Melting Pot

We want to grow Melting Pot over time and ultimately create a comprehensive platform where most aspects of social intelligence can be assessed. To that end, we designed Melting Pot around the need to establish a scalable process through which it can be expanded. This led us to consider not just modular environment components (which we have), but also a modular process for contributing new scenarios.

A scenario consists of two parts: a substrate, and a background population. We built substrates on DMLab2D (Beattie et al., 2020) using an entity-component system approach similar to that of modern game engines like Unity (Unity Technologies, 2020). Members of the background population are RL agents. We call them *bots* to distinguish them from the agents in the focal population. A Melting Pot substrate emits events when interactions occur between agents, or agents and the environment, such as one player zapping another player or eating an apple. Events can be conditional on the identities of the players involved or the location where the interaction occurred.

Our approach to creating background populations involves three steps: (1) specification, (2) training, and (3) quality control (Fig. 3). We describe each in turn.

1. Specification: The designer typically starts with an idea of what they want the final bot's behavior to look like. Since

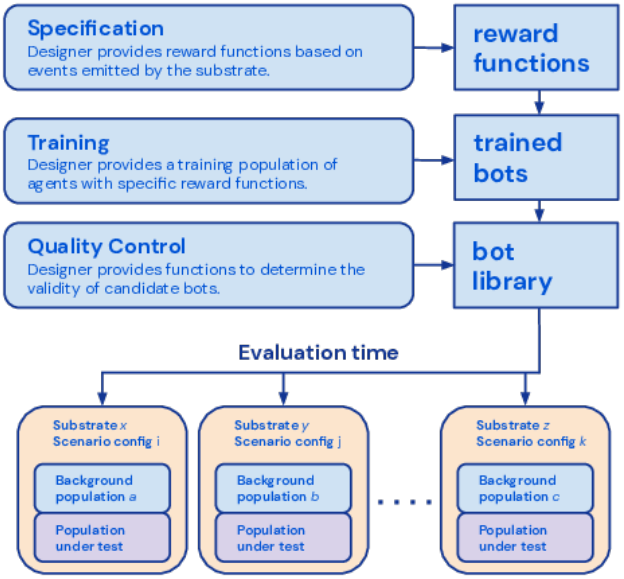

Figure 3. The process for extending Melting Pot.

substrate events provide privileged information about other agents and the substrate, we can often easily specify reward functions that induce the right behaviour. This is a much easier task than what focal agents need to solve—learning from only pixels and the final reward. However, sometimes the desired behavior is difficult to specify using a single reward function. In these cases, we generate background populations using techniques inspired by hierarchical reinforcement learning (Sutton et al., 2011; Schaul et al., 2015; Sutton et al., 1999); in particular reward shaping (Sutton & Barto, 2018) and "option keyboard" (Barreto et al., 2019). We create a basic portfolio of behaviors by training bots that use different environment events as the reward signal (as in Horde (Sutton et al., 2011)), and then chain them using simple Python code. This allows us to express complex behaviours in a "if this event, run that behaviour" way. For example, in *Clean Up* we created a bot that only cleans if other players are cleaning. These bots had a special network architecture based on FuN (Vezhnevets et al., 2017), with goals specified externally via substrate events rather than being produced inside the agent. See appendix for details.

2. *Training:* The decision at this stage is how to train the background population. The thing to keep in mind is that the bots must generalize to the focal population. To this end, we chose at least some bots—typically not used in the final scenario—that are likely to develop behaviors resembling that of the focal agent at test time. For instance, in *Running With Scissors in the Matrix*, we train rock, paper, and scissors specialist bots alongside "free" bots that experience the true substrate reward function.

3. *Quality control:* Bot quality control is done by running 10–30 episodes where candidate bots interact with other fixed bots. These other bots are typically a mixture of familiar and unfamiliar bots (that trained together or separately). We verify that agents trained to optimize for a certain event, indeed do. We reject agents that fail to do so.

7. Experiments

To demonstrate the use of Melting Pot, we provide benchmark MARL results for a number of agent architectures.

For each agent architecture, we performed 21 training runs—one for each substrate. Within each training run, we trained a group of N agents—one for each player in the N-player substrate. Every agent participated in every training episode, with each agent playing as exactly one player (selected randomly on each episode). Each agent was trained for 10^9 steps. At test time, we set the focal population to be the uniform distribution over the N agents.

The different architectures we trained are: A3C (Mnih et al., 2016), V-MPO (Song et al., 2020a), and OPRE (Vezhnevets et al., 2020). A3C is a well established, off-the-shelf RL method. V-MPO is relatively new and state-of-the-art on many single agent RL benchmarks. OPRE was specifically designed for MARL. We also trained *prosocial* variants of all three algorithms, which directly optimized the per-capita return (rather than individual return), by sharing reward between players during training. Optimizing for collective return as a surrogate objective has previously been used for collaborative games (substrates) and social dilemmas (Claus & Boutilier, 1998; Peysakhovich & Lerer, 2017), and our experiments here allow us to investigate whether it generalizes well.

All agent architectures had the same size convolutional net and LSTM. The OPRE agent had additional hierarchical structure in its policy as described in (Vezhnevets et al., 2020). V-MPO had a pop-art layer (Hessel et al., 2019) for normalizing the value function. A3C minimized a contrastive predictive coding loss (Oord et al., 2018) as an auxiliary objective (Jaderberg et al., 2016) to promote discrimination between nearby time points via LSTM state representations (a standard augmentation in recent work with A3C). See the appendix for additional implementation details.

We also use two special agents: "exploiters" and "random". Each exploiter is an A3C agent trained directly on a single test scenario, using their individual return as the reward signal without further augmentation. Exploiters trained for up to 10^9 steps. The random agent selects actions uniformly at random, and ignores input observations. Together, exploiters and the random agent provide a rough estimate of the upper and lower bounds (respectively) of performance on the test scenarios. To contextualize the range of agent

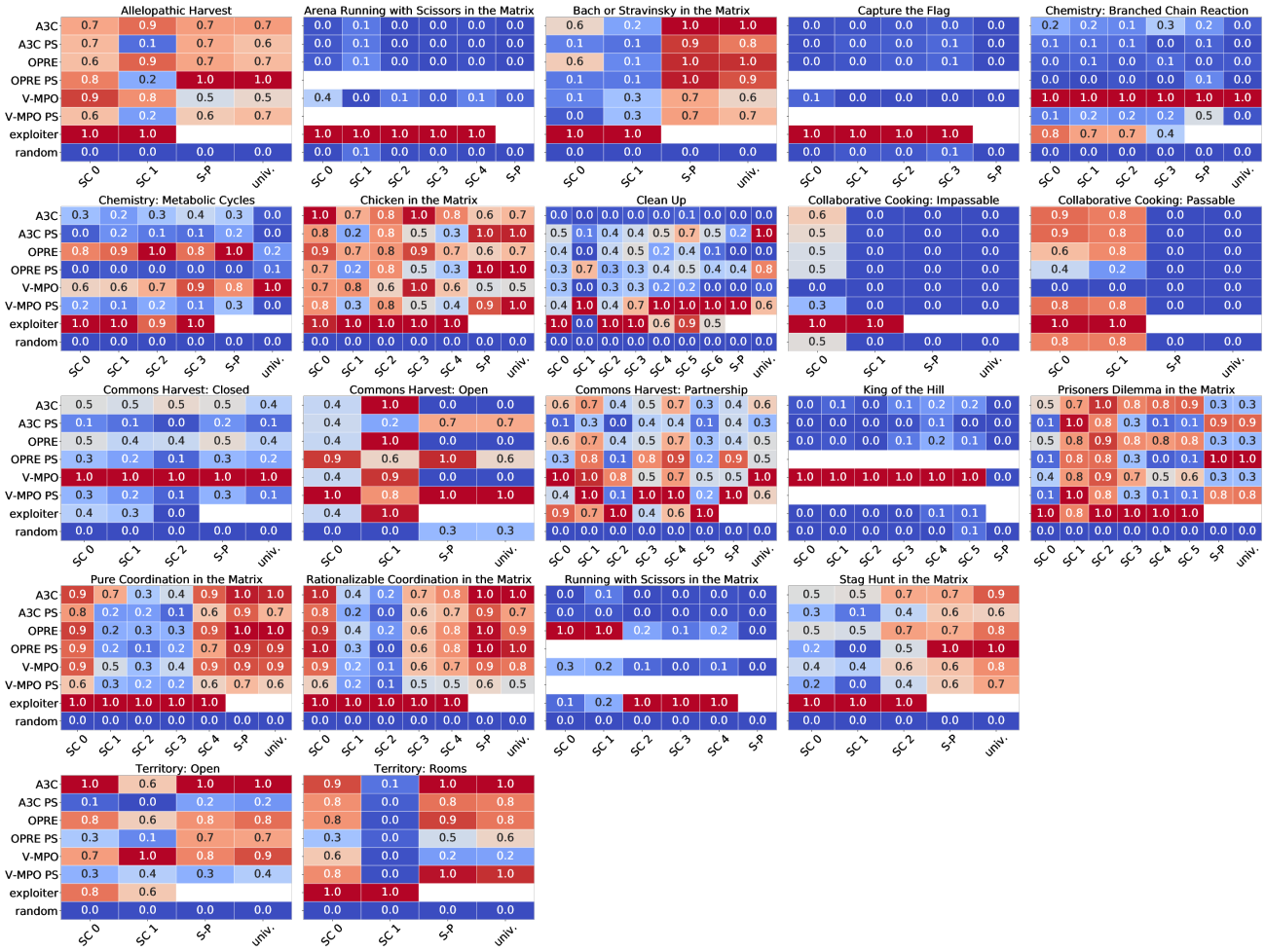

Figure 4. MAPLA performance scores on test scenarios for each substrate. Rows correspond to the agent training algorithms, columns to scenarios. PS stands for prosocial, SC for scenario, S-P for self-play score (i.e. the training performance), univ. for universalization scenario. Note: we did not train prosocial variants on zero-sum substrates since the per-capita return is zero by definition.

returns, we min-max normalize the focal per-capita returns to get a *performance score* that is between 0 (for the worst agent) and 1 (for the best agent). Often the score of 0 corresponds to the random agent and 1 to the exploiter. However, for some scenarios this is not the case. We discuss the reasons below.

Fig. 4 presents the full results, showing the score obtained by each agent on each test scenario. Fig. 5 presents an overview, showing the average test scenario score for each substrate. Alongside the total average score, we present Elo scores (Balduzzi et al., 2018; Hunter et al., 2004), which are computed separately for competitive substrates and the rest.

On average, the top performing agent was V-MPO (Song et al., 2020a), followed by OPRE (Vezhnevets et al., 2020), and A3C (Mnih et al., 2016). All three performed similarly in mixed-motivation games, but V-MPO outperforms in competitive games like *King of the Hill*. However, ranking agents' by skill is complicated (Balduzzi et al., 2018), and dependent on the contribution of each test scenario to the overall score. Here we did only a simple evaluation which is far from the only option.

There is scope to dramatically improve performance on the most challenging substrates. The *Collaborative Cooking* substrates proved impossible for these agents to learn, with no agent obtaining a self-play (training) score above random. In *Arena Running with Scissors* and *Capture The Flag*, agents are far below exploiter performance on the test scenarios.

Exploiters do not always achieve the highest score. In some cases (*Running with Scissors in the Matrix*, *King of the Hill*), exploiters fail because the scenario bots are strong opponents and learning to exploit them is hard. Here, the MAPLAs have the advantage of a gentler curriculum: in self-play training, all opponents are of a similar skill level, which

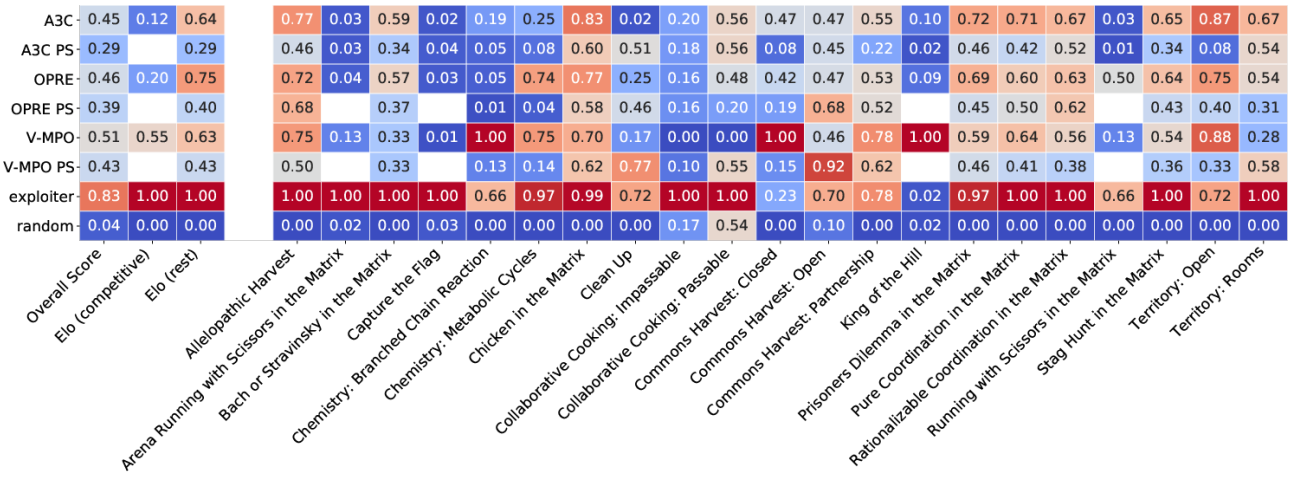

Figure 5. MAPLA performance scores averaged over each substrate's scenarios. Three columns on the left show: average score per algorithm; Elo per agent on competitive substrates, and Elo on the rest the suite. Elo is min-max normalised such that 0 corresponds to lowest performing agent, and 1 to the highest performing one.

ramps up as they all learn. In other cases (*Commons Harvest Open*, *Clean Up*), exploiters fail due to their selfish reward maximization in games where cooperation is required. In these cases, a better performance upper bound might be obtained by exploiters with within-group reward sharing.

Agents often exhibit a degree of overfitting. In Fig. 4 we see agents with high self-play scores (obtained during training), but low test-scenario scores. For example, in *Bach or Stravinsky in the Matrix* and *Stag Hunt in the Matrix*, prosocial OPRE and prosocial A3C achieve a similar self-play score to regular OPRE and A3C, but obtain a closer to random score in the test scenarios. This overfitting is due to prosocial agents only learning cooperation strategies during self-play training, and so becoming exploitable by defectors at test time.

Overall, prosocial agents underperformed their selfish counterparts, but the picture is nuanced. Optimizing for per-capita return can be difficult because it complicates credit assignment, and creates spurious reward "lazy agent" problems (Sunehag et al., 2018; Rashid et al., 2018). However, in the social dilemma *Clean Up*, only prosocial agent architectures managed to learn policies that were significantly better than random. This suggests that doing well on Melting Pot will require agents to be able to contingently balance selfishness and prosociality.

The universalization scenarios can diagnose issues with the of division of labour. For example, in *Chemistry: Metabolic Cycles*, OPRE performs well in self-play and other scenarios, but has low universalization scores. This means that some agents in the population learned specialized policies that expect other agents to behave in a particular way. Although such division of labour can create efficiency, it also makes populations less robust.

8. Conclusion

Here we have presented Melting Pot: an evaluation suite for MAPLAs that evaluates generalization to novel social situations. Melting Pot engages with concepts that have long been neglected by research in artificial intelligence. Solutions to the problems posed here seem to require agents that understand trust, generosity, and forgiveness, as well as reciprocity, stubbornness, and deception.

Melting Pot is scalable. We have used reinforcement learning to reduce human labor in environment design. This is how we rapidly created the diverse set of ~ 85 scenarios considered so far. Since Melting Pot will be openly released, it can be extended by any interested researchers. In addition, since the effectiveness of the bots in test scenarios is itself advanced by improvements in the performance of learning systems, Melting Pot will likewise improve over time by reincorporating the latest agent technology into new background populations and test scenarios.

9. Acknowledgements

The authors would like to thank: Richard Everett, DJ Strouse, Kevin McKee, and Edward Hughes for contributing to substrate development; Mary Cassin for contributing artwork; and Steven Wheelwright for help building an early version of the evaluation system.

References

Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., and Mané, D. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*, 2016.

Axelrod, R. *The Evolution of Cooperation*. Basic Books, 1984.

Baker, B., Kanitscheider, I., Markov, T., Wu, Y., Powell, G., McGrew, B., and Mordatch, I. Emergent tool use from multi-agent autocurricula. *arXiv preprint arXiv:1909.07528*, 2019.

Balduzzi, D., Tuyls, K., Perolat, J., and Graepel, T. Re-evaluating evaluation. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 3268–3279, 2018.

Balduzzi, D., Garnelo, M., Bachrach, Y., Czarnecki, W., Perolat, J., Jaderberg, M., and Graepel, T. Open-ended learning in symmetric zero-sum games. In *International Conference on Machine Learning*, pp. 434–443, 2019.

Bansal, T., Pachocki, J., Sidor, S., Sutskever, I., and Mordatch, I. Emergent complexity via multi-agent competition. *arXiv preprint arXiv:1710.03748*, 2017.

Barreto, A., Borsa, D., Hou, S., Comanici, G., Aygün, E., Hamel, P., Toyama, D., Mourad, S., Silver, D., Precup, D., et al. The option keyboard: Combining skills in reinforcement learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 32:13052–13062, 2019.

Beattie, C., Köppe, T., Duéñez-Guzmán, E. A., and Leibo, J. Z. DeepMind Lab2D. *arXiv preprint arXiv:2011.07027*, 2020.

Berner, C., Brockman, G., Chan, B., Cheung, V., Debiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., Józefowicz, R., Gray, S., Olsson, C., Pachocki, J., Petrov, M., de Oliveira Pinto, H. P., Raiman, J., Salimans, T., Schlatter, J., Schneider, J., Sidor, S., Sutskever, I., Tang, J., Wolski, F., and Zhang, S. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. OpenAI gym. *arXiv preprint arXiv:1606.01540*, 2016.

Carroll, M., Shah, R., Ho, M. K., Griffiths, T. L., Seshia, S. A., Abbeel, P., and Dragan, A. On the utility of learning about humans for human-ai coordination. *arXiv preprint arXiv:1910.05789*, 2019.

Chollet, F. On the measure of intelligence. *arXiv preprint arXiv:1911.01547*, 2019.

Claus, C. and Boutilier, C. The dynamics of reinforcement learning in cooperative multiagent systems. *AAAI/IAAI*, 1998(746-752):2, 1998.

Clune, J. AI-GAs: AI-generating algorithms, an alternate paradigm for producing general artificial intelligence. *arXiv preprint arXiv:1905.10985*, 2019.

Cobbe, K., Klimov, O., Hesse, C., Kim, T., and Schulman, J. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning*, pp. 1282–1289. PMLR, 2019.

Dafoe, A., Hughes, E., Bachrach, Y., Collins, T., McKee, K. R., Leibo, J. Z., Larson, K., and Graepel, T. Open problems in cooperative AI. *arXiv preprint arXiv:2012.08630*, 2020.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Farebrother, J., Machado, M. C., and Bowling, M. Generalization and regularization in DQN. *arXiv preprint arXiv:1810.00123*, 2018.

Foerster, J., Chen, R. Y., Al-Shedivat, M., Whiteson, S., Abbeel, P., and Mordatch, I. Learning with opponent-learning awareness. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 122–130, 2018.

Fortunato, M., Tan, M., Faulkner, R., Hansen, S., Badia, A. P., Buttimore, G., Deck, C., Leibo, J. Z., and Blundell, C. Generalization of reinforcement learners with working and episodic memory. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Harsanyi, J. C. Games with incomplete information played by "bayesian" players, i–iii part i. the basic model. *Management science*, 14(3):159–182, 1967.

Henrich, J. *The weirdest people in the world: How the west became psychologically peculiar and particularly prosperous*. Farrar, Straus and Giroux, 2020.

Hessel, M., Soyer, H., Espeholt, L., Czarnecki, W., Schmitt, S., and van Hasselt, H. Multi-task deep reinforcement learning with popart. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3796–3803, 2019.

Hu, H., Lerer, A., Peysakhovich, A., and Foerster, J. "other-play" for zero-shot coordination. *arXiv preprint arXiv:2003.02979*, 2020.

Hughes, E., Leibo, J. Z., Philips, M. G., Tuyls, K., Duéñez-Guzmán, E. A., Castañeda, A. G., Dunning, I., Zhu, T., McKee, K. R., Koster, R., Roff, H., and Graepel, T. Inequity aversion improves cooperation in intertemporal social dilemmas. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 3330–3340. 2018.

Hunter, D. R. et al. Mm algorithms for generalized bradley-terry models. *The annals of statistics*, 32(1):384–406, 2004.

Jaderberg, M., Mnih, V., Czarnecki, W. M., Schaul, T., Leibo, J. Z., Silver, D., and Kavukcuoglu, K. Reinforcement learning with unsupervised auxiliary tasks. *arXiv preprint arXiv:1611.05397*, 2016.

Jaderberg, M., Czarnecki, W. M., Dunning, I., Marris, L., Lever, G., Castaneda, A. G., Beattie, C., Rabinowitz, N. C., Morcos, A. S., Ruderman, A., et al. Human-level performance in 3d multiplayer games with population-based reinforcement learning. *Science*, 364(6443):859–865, 2019.

Juliani, A., Khalifa, A., Berges, V.-P., Harper, J., Teng, E., Henry, H., Crespi, A., Togelius, J., and Lange, D. Obstacle tower: A generalization challenge in vision, control, and planning. *arXiv preprint arXiv:1902.01378*, 2019.

Köster, R., McKee, K. R., Everett, R., Weidinger, L., Isaac, W. S., Hughes, E., Duéñez-Guzmán, E. A., Graepel, T., Botvinick, M., and Leibo, J. Z. Model-free conventions in multi-agent reinforcement learning with heterogeneous preferences. *arXiv preprint arXiv:2010.09054*, 2020.

Kraemer, L. and Banerjee, B. Multi-agent reinforcement learning as a rehearsal for decentralized planning. *Neurocomputing*, 190:82–94, 2016.

Lanctot, M., Zambaldi, V., Gruslys, A., Lazaridou, A., Tuyls, K., Pérolat, J., Silver, D., and Graepel, T. A unified game-theoretic approach to multiagent reinforcement learning. In *Advances in neural information processing systems (NeurIPS)*, pp. 4190–4203, 2017.

Lanctot, M., Lockhart, E., Lespiau, J.-B., Zambaldi, V., Upadhyay, S., Pérolat, J., Srinivasan, S., Timbers, F., Tuyls, K., Omidshafiei, S., Hennes, D., Morrill, D., Muller, P., Ewalds, T., Faulkner, R., Kramar, J., De Vylder, B., Saeta, B., Bradbury, J., Ding, D., Borgeaud, S., Lai, M., Schrittwieser, J., Anthony, T., Hughes, E., Danihelka, I., and Ryan-Davis, J. Openspiel: A framework for reinforcement learning in games. *arXiv preprint arXiv:1908.09453*, 2019.

Lazaridou, A. and Baroni, M. Emergent multi-agent communication in the deep learning era. *arXiv preprint arXiv:2006.02419*, 2020.

LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *Nature*, 521(7553):436–444, 2015.

Leibo, J. Z., Hughes, E., Lanctot, M., and Graepel, T. Autocurricula and the emergence of innovation from social interaction: A manifesto for multi-agent intelligence research. *arXiv preprint arXiv:1903.00742*, 2019a.

Leibo, J. Z., Perolat, J., Hughes, E., Wheelwright, S., Marblestone, A. H., Duéñez-Guzmán, E., Sunehag, P., Dunning, I., and Graepel, T. Malthusian reinforcement learning. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 1099–1107, 2019b.

Lerer, A. and Peysakhovich, A. Maintaining cooperation in complex social dilemmas using deep reinforcement learning. *arXiv preprint arXiv:1707.01068*, 2018.

Levine, S., Kleiman-Weiner, M., Schulz, L., Tenenbaum, J., and Cushman, F. The logic of universalization guides moral judgment. *Proceedings of the National Academy of Sciences*, 117(42):26158–26169, 2020.

Littman, M. L. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pp. 157–163. Elsevier, 1994.

Liu, S., Lever, G., Merel, J., Tunyasuvunakool, S., Heess, N., and Graepel, T. Emergent coordination through competition. In *International Conference on Learning Representations*, 2018.

Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., and Mordatch, I. Multi-agent actor-critic for mixed cooperative-competitive environments. *arXiv preprint arXiv:1706.02275*, 2017.

Lowe, R., Foerster, J., Boureau, Y.-L., Pineau, J., and Dauphin, Y. On the pitfalls of measuring emergent communication. *arXiv preprint arXiv:1903.05168*, 2019.

Machado, M. C., Bellemare, M. G., Talvitie, E., Veness, J., Hausknecht, M., and Bowling, M. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *Journal of Artificial Intelligence Research*, 61:523–562, 2018.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937. PMLR, 2016.

Mordatch, I. and Abbeel, P. Emergence of grounded compositional language in multi-agent populations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Oliehoek, F. A. and Amato, C. *A concise introduction to decentralized POMDPs*. Springer, 2016.

Oord, A. v. d., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Osband, I., Doron, Y., Hessel, M., Aslanides, J., Sezener, E., Saraiva, A., McKinney, K., Lattimore, T., Szepesvari, C., Singh, S., et al. Behaviour suite for reinforcement learning. *arXiv preprint arXiv:1908.03568*, 2019.

Perez-Liebana, D., Hofmann, K., Mohanty, S. P., Kuno, N., Kramer, A., Devlin, S., Gaina, R. D., and Ionita, D. The multi-agent reinforcement learning in malmö (marlö) competition. *arXiv preprint arXiv:1901.08129*, 2019.

Perolat, J., Leibo, J. Z., Zambaldi, V., Beattie, C., Tuyls, K., and Graepel, T. A multi-agent reinforcement learning model of common-pool resource appropriation. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 3643–3652, 2017.

Peysakhovich, A. and Lerer, A. Prosocial learning agents solve generalized stag hunts better than selfish ones. *arXiv preprint arXiv:1709.02865*, 2017.

Racaniere, S., Lampinen, A., Santoro, A., Reichert, D., Firoiu, V., and Lillicrap, T. Automated curriculum generation through setter-solver interactions. In *International Conference on Learning Representations*, 2019.

Rashid, T., Samvelyan, M., Schroeder, C., Farquhar, G., Foerster, J., and Whiteson, S. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 4295–4304. PMLR, 2018.

Russell, S., Dewey, D., and Tegmark, M. Research priorities for robust and beneficial artificial intelligence. *Ai Magazine*, 36(4):105–114, 2015.

Schaul, T., Horgan, D., Gregor, K., and Silver, D. Universal value function approximators. In *International conference on machine learning*, pp. 1312–1320. PMLR, 2015.

Shapley, L. S. Stochastic Games. *In Proc. of the National Academy of Sciences of the United States of America*, 1953.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

Soares, N. and Fallenstein, B. Aligning superintelligence with human interests: A technical research agenda. *Machine Intelligence Research Institute (MIRI) technical report*, 8, 2014.

Song, H. F., Abdolmaleki, A., Springenberg, J. T., Clark, A., Soyer, H., Rae, J. W., Noury, S., Ahuja, A., Liu, S., Tirumala, D., Heess, N., Belov, D., Riedmiller, M., and Botvinick, M. M. V-mpo: On-policy maximum a posteriori policy optimization for discrete and continuous control. In *International Conference on Learning Representations*, 2020a. URL https://openreview.net/forum?id=SylOlp4FvH.

Song, Y., Wojcicki, A., Lukasiewicz, T., Wang, J., Aryan, A., Xu, Z., Xu, M., Ding, Z., and Wu, L. Arena: A general evaluation platform and building toolkit for multi-agent intelligence. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7253–7260, 2020b.

Stone, P., Kaminka, G. A., Kraus, S., Rosenschein, J. S., et al. Ad hoc autonomous agent teams: Collaboration without pre-coordination. In *AAAI*, pp. 6, 2010.

Sukhbaatar, S., Lin, Z., Kostrikov, I., Synnaeve, G., Szlam, A., and Fergus, R. Intrinsic motivation and automatic curricula via asymmetric self-play. *arXiv preprint arXiv:1703.05407*, 2017.

Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W. M., Zambaldi, V., Jaderberg, M., Lanctot, M., Sonnerat, N., Leibo, J. Z., Tuyls, K., et al. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2085–2087, 2018.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Sutton, R. S., Precup, D., and Singh, S. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2): 181–211, 1999.

Sutton, R. S., Modayil, J., Delp, M., Degris, T., Pilarski, P. M., White, A., and Precup, D. Horde: A scalable

real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pp. 761–768, 2011.

Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., Casas, D. d. L., Budden, D., Abdolmaleki, A., Merel, J., Lefrancq, A., et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Torrado, R. R., Bontrager, P., Togelius, J., Liu, J., and Perez-Liebana, D. Deep reinforcement learning for general video game ai. In *2018 IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 1–8. IEEE, 2018.

Unity Technologies. The unity game engine, 2020. URL https://unity.com.

Vezhnevets, A., Wu, Y., Eckstein, M., Leblond, R., and Leibo, J. Z. Options as responses: Grounding behavioural hierarchies in multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 9733–9742. PMLR, 2020.

Vezhnevets, A. S., Osindero, S., Schaul, T., Heess, N., Jaderberg, M., Silver, D., and Kavukcuoglu, K. Feudal networks for hierarchical reinforcement learning. In *International Conference on Machine Learning*, pp. 3540–3549. PMLR, 2017.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019.

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. *arXiv preprint arXiv:1901.01753*, 2019.

Wang, R. E., Wu, S. A., Evans, J. A., Tenenbaum, J. B., Parkes, D. C., and Kleiman-Weiner, M. Too many cooks: Coordinating multi-agent collaboration through inverse planning. *arXiv preprint arXiv:2003.11778*, 2020.

Ye, D., Chen, G., Zhang, W., Chen, S., Yuan, B., Liu, B., Chen, J., Liu, Z., Qiu, F., Yu, H., et al. Towards playing full moba games with deep reinforcement learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 33, 2020.

Zhang, A., Ballas, N., and Pineau, J. A dissection of overfitting and generalization in continuous reinforcement learning. *arXiv preprint arXiv:1806.07937*, 2018a.

Zhang, A., Wu, Y., and Pineau, J. Natural environment benchmarks for reinforcement learning. *arXiv preprint arXiv:1811.06032*, 2018b.