A SURVEY OF GENERALISATION IN DEEP REINFORCEMENT LEARNING

Robert Kirk* University College London
Amy Zhang UC Berkeley
Edward Grefenstette University College London
Tim Rocktäschel University College London

ABSTRACT

The study of generalisation in deep Reinforcement Learning (RL) aims to produce RL algorithms whose policies generalise well to novel unseen situations at deployment time, avoiding overfitting to their training environments. Tackling this is vital if we are to deploy reinforcement learning algorithms in real world scenarios, where the environment will be diverse, dynamic and unpredictable. This survey is an overview of this nascent field. We provide a unifying formalism and terminology for discussing different generalisation problems, building upon previous works. We go on to categorise existing benchmarks for generalisation, as well as current methods for tackling the generalisation problem. Finally, we provide a critical discussion of the current state of the field, including recommendations for future work. Among other conclusions, we argue that taking a purely procedural content generation approach to benchmark design is not conducive to progress in generalisation, we suggest fast online adaptation and tackling RL-specific problems as some areas for future work on methods for generalisation, and we recommend building benchmarks in underexplored problem settings such as offline RL generalisation and reward-function variation.

Keywords Generalisation · Reinforcement Learning · Survey · Review

1 Introduction

Reinforcement Learning (RL) could be used in a range of applications such as autonomous vehicles [1] and robotics [2], but to fulfill this potential we need RL algorithms that can be used in the real world. Reality is dynamic, open-ended and always changing, and RL algorithms will need to be robust to variation in their environments, and have the capability to transfer and adapt to unseen (but similar) environments during their deployment.

However, much current RL research works on benchmarks such as Atari [3] and MuJoCo [4, 5] which do not have the attributes described above: they evaluate the policy on exactly the same environment it was trained on, which does not match with real-world scenarios (Fig. 1 left column). This is in stark contrast to the standard assumptions of supervised learning where the training and testing sets are disjoint, and is likely to lead to strong evaluation overfitting [6]. This has resulted in policies that perform badly on even slightly adjusted environment instances (specific levels or tasks within an environment) and often fail on unseen random seeds used for initialisation [7, 8, 9, 10].

Many researchers have taken these criticisms seriously, and now focus on improving generalisation in RL (as can be seen from the content of this survey). This research is focused on producing algorithms whose policies have the desired robustness, transfer and adaptation properties, challenging the basic assumption that train and test will be identical (Fig. 1 middle and right columns). While this research is valuable, currently it often lacks clarity or coherence between papers. We argue that this is partly because generalisation (especially in RL) is a *class* of problems, rather than a specific problem. Improving "generalisation" without specifying the kind of generalisation that is desired is underspecified; it is unlikely that we can generically improve generalisation, given this class of problems is so broad that some analogy of the No Free Lunch theorem [11] applies: improving generalisation in some settings could harm generalisation in others. Two broad categories of generalisation problem are shown in Fig. 1 in the centre and right columns.

We address the confusion stemming from conceptualising generalisation as a single problem. We present a formalism for understanding this class of problems (built on previous works [12, 13, 14, 15, 16]), and what choices there are in

*Correspondence to robert.kirk.20@ucl.ac.uk. For details on author contributions see Author Contributions.

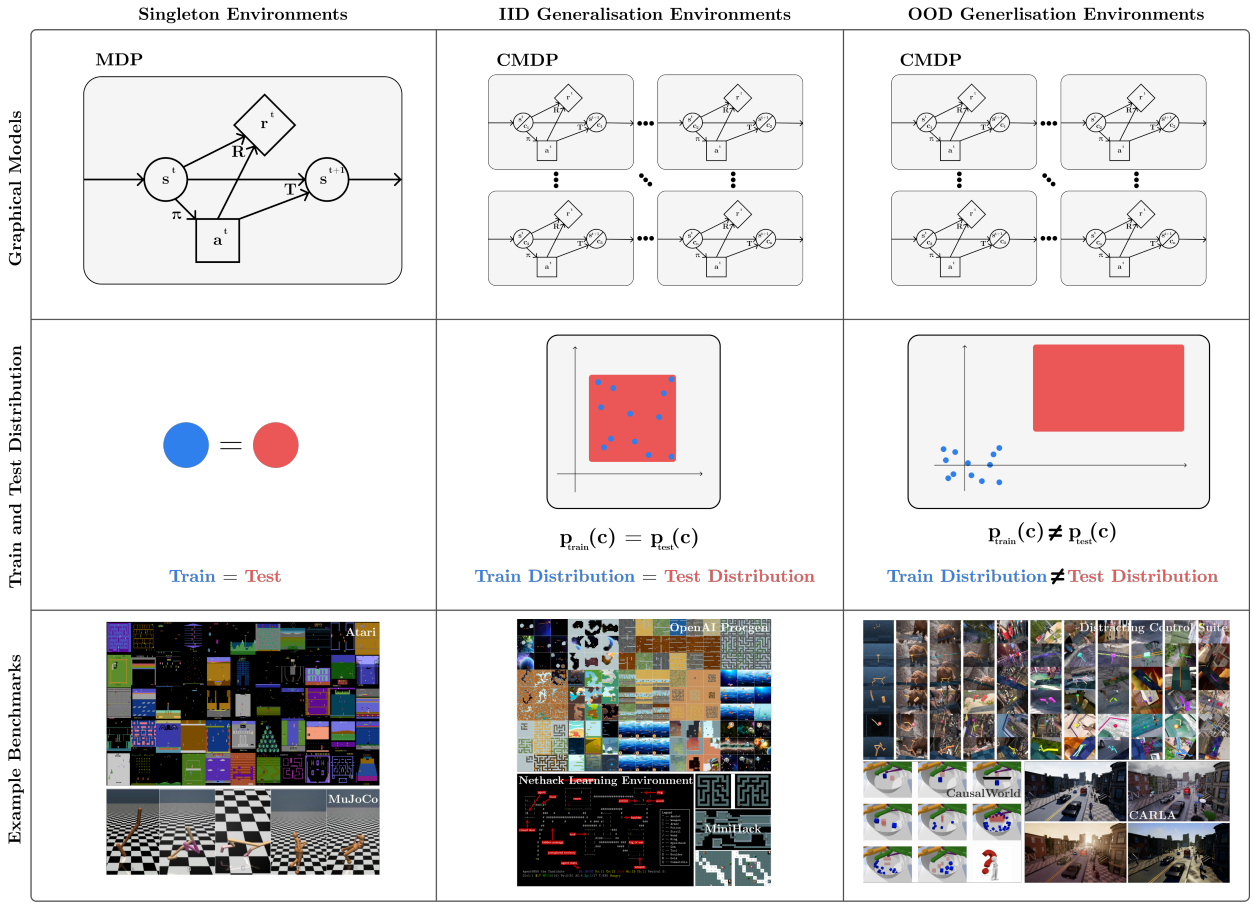


Figure 1: **Generalisation in Reinforcement Learning**. A visualisation of three types of environment (columns) with respect to their graphical model, training and testing distribution and example benchmarks (rows). Classical RL has focused on environments where training and testing are identical (singleton environments, first column) but in the real world training and testing environments will be different, either from the same distribution (IID Generalisation Environments, second column) or from different distributions (OOD Generalisation Environments, third column). The split between the second and third column is just one example of the way in which generalisation is a class of problems rather than an individual problem. For more information on the CMDP formalism see Section 3.3.

specifying a generalisation problem. This is then grounded in choices made by specific benchmarks, and assumptions made to justify specific methods, which we discuss next. Finally, we propose several settings within generalisation which are underexplored but still vital for various real-world applications of RL, as well as many avenues for future work on methods that can solve different generalisation problems. We aim to make the field more legible to researchers and practitioners both in and out of the field, and make discussing new research directions easier. This new clarity can improve the field, and enable robust progress towards more general RL methods.

Scope. We focus on a problem setting called zero-shot policy transfer. Here, a policy is evaluated zero-shot on a collection of environments different to those it was trained on. We discuss this setting and its restrictions more in Section 3.7, *Motivating Zero-Shot Policy Transfer*, but note here that this setting disallows any additional training in or data from the test environments, meaning methods such as domain adaptation and many meta RL approaches are not applicable.

We only cover single agent RL in this work. There are generalisation problems within multi-agent reinforcement learning (MARL), such as being general enough to defeat multiple different opponent strategies [17, 18] and generalising to new team-mates in cooperative games [19, 20], but we do not cover any work in this area here. Relatedly, there is work on using multiple agents in a single-agent setting to increase the diversity of the environment and hence the generality of the policy [21], which we do cover.

We do not cover theoretical work on generalisation in RL. While there is recent work in this area [22, 23], it is often quite disconnected from the current empirical results and practices in deep RL, and hence is not as applicable.

Overview of the Survey. The structure of the survey is as follows. We first briefly describe related work such as other surveys and overviews in Section 2. We introduce the formalism and terminology for generalisation in RL in Section 3, including the relevant background. We then proceed to use this formalism to describe current benchmarks for generalisation in RL in Section 4, discussing both environments (Section 4.1) and evaluation protocols (Section 4.2). We categorise and describe work producing methods for tackling generalisation in Section 5. Finally, we present a critical discussion of the current field, including recommendations for future work in both methods and benchmarks, in Section 6, and conclude with a summary of the key takeaways from the survey in Section 7.

Contributions. To summarise, our key contributions are:

- We present a formalism and terminology for discussing the broad class of generalisation problems, building on formalisms and terminology presented in multiple previous works [12, 13, 14, 15, 16]. Our contribution here is the unification of these prior works into *a clear formal description of the class of problems referred to as generalisation in RL.*
- We propose a taxonomy of existing benchmarks that can be used to test for generalisation, splitting the discussion into categorising environments and evaluation protocols. Our formalism allows us to cleanly describe weaknesses of the purely PCG approach to generalisation benchmarking and environment design: *having a completely PCG environment limits the precision of the research that can be done on that environment*. We recommend that *future environments should use a combination of PCG and controllable factors of variation*.
- We propose a categorisation of existing methods to tackle various generalisation problems, motivated by a desire to make it easy both for practitioners to choose methods given a concrete problem and for researchers to understand the landscape of methods and where novel and useful contributions could be made. *We point to many under-explored avenues for further research, including fast online adaptation, tackling RL-specific generalisation issues, novel architectures, model-based RL and environment generation*.
- We critically discuss the current state of generalisation in RL research, recommending future research directions. In particular, we point that *building benchmarks would enable progress offline RL generalisation and reward-function variation*, both of which are important settings. Further, we point to several different settings and evaluation metrics that are worth exploring: *investigating context-efficiency and working in a continual RL setting* are both areas where future work is necessary.

2 Related Work: Surveys In Reinforcement Learning Subfields

While there have been previous surveys of related subfields in RL, none have covered generalisation in RL explicitly. Khetarpal et al. [24] motivate and survey continual reinforcement learning (CRL), which is closely related to generalisation in RL as both settings require adaptation to unseen tasks or environments; however, they explicitly do not discuss zero-shot generalisation which is the concern of this paper (for more discussion of CRL see Section 6.1). Chen and Li [25] give a brief overview of Robust RL (RRL) [26], a field aimed at tackling a specific form of environment model misspecification through worst-case optimisation. This is a sub-problem within the class of generalisation problems we discuss here, and [25] only briefly survey the field.

Zhao et al. [27] surveys methods for sim-to-real transfer for deep RL in robotics. Sim-to-real is a concrete instantiation of the generalisation problem, and hence there is some overlap between our work and [27], but our work covers a much broader subject area, and some methods for sim-to-real transfer rely on data from the testing environment (reality), which we do not assume here. Müller-Brockhausen et al. [28] and Zhu et al. [29] survey methods for transfer learning in RL (TRL). TRL is related to generalisation in that both topics assume a policy is trained in a different setting to its deployment, but TRL generally assumes some form of extra training in the deployment or target environment, whereas we are focused on zero-shot generalisation. Finally, surveys on less related topics include Vithayathil Varghese and Mahmoud [30] on multi-task deep RL, Amin et al. [31] on exploration in RL, and Narvekar et al. [32] on curriculum learning in RL.

None of these surveys focuses on the zero-shot generalisation setting that is the focus of this work, and there is still a need for a formalism for the class of generalisation problems which will enable research in this field to discuss the differences between different problems.

3 Formalising Generalisation In Reinforcement Learning

In this section we present a formalism for understanding and discussing the class of generalisation problems in RL. We first review relevant background in supervised learning and RL before motivating the formalism itself. Formalising the generalisation in this way shows that generalisation refers to a *class* of problems, rather than a specific problem, and hence research on generalisation needs to specify which group of generalisation problems it is tackling. Having laid out this class of problems in Section 3.4, we discuss additional assumptions of structure that could make generalisation more tractable in Section 3.6; this is effectively specifying sub-problems of the wider generalisation problem.

3.1 Background: Generalisation In Supervised Learning

Generalisation in supervised learning is a widely studied area and hence is more mature than generalisation in RL (although it is still not well-understood). In supervised learning, some predictor is trained on a training dataset, and the performance of the model is measured on a held out testing dataset. It is often assumed that the data points in both the training and testing dataset are drawn iid from the same underlying distribution. Generalisation performance is then synonymous with the test-time performance, as the model needs to "generalise" to inputs it has not seen before during training. The generalisation gap in supervised learning for a model ϕ with training and testing data D_{train}, D_{test} and loss function $\mathcal{L}$ is defined as

$$\text{GenGap}(\phi) := \mathbb{E}_{(x,y)\sim D_{test}}[\mathcal{L}(\phi, x, y)] - \mathbb{E}_{(x,y)\sim D_{train}}[\mathcal{L}(\phi, x, y)]. \tag{1}$$

This gap is used as a measure of generalisation specifically: a smaller gap means a model generalises better.

One specific type of generalisation examined frequently in supervised learning which is relevant to RL is compositional generalisation [33, 34]. We explore a categorisation of compositional generalisation here from [33]. While this was designed for generalisation in language, many of those forms are relevant for RL. The five forms of compositional generalisation defined are:

1. **systematicity**: generalisation via systematically recombining known parts and rules,
2. **productivity**: the ability to extend predictions beyond the length seen in training data,
3. **substitutivity**: generalisation via the ability to replace components with synonyms,
4. **localism**: if model composition operations are local vs. global,
5. **overgeneralisation**: if models pay attention to or are robust to exceptions.

For intuition we will explore examples of some of these different types of compositional generalisation in a block-stacking environment. An example of *systematicity* is the ability to stack blocks in new configurations once the basics of block-stacking are mastered. Similarly, *productivity* can be measured by how many blocks the agent can generalise to, and the complexity of the stacking configurations. *Substitutivity* can be evaluated by the agent's ability to generalise to blocks of new colours, understanding that the new colour does not affect the physics of the block. In Section 4.3, *Compositional Generalisation in Contextual MDPs* we discuss how assumptions of compositional structure in the collection of RL environments can enable us to test these forms of generalisation. In Section 4.2 we will discuss how some of these forms of generalisation can be evaluated in current RL benchmarks.

3.2 Background: Reinforcement Learning

The standard formalism in RL is the Markov Decision Process (MDP). An MDP consists of a tuple $M = (S, A, R, T, p)$, where S is the state space; A is the action space; $R : S \times A \times S \to \mathbb{R}$ is the scalar reward function; $T(s'|s, a)$ is the possibly stochastic markovian transition function; and $p(s_0)$ is the initial state distribution. We also consider partially observable MDPs (POMDPs). A POMDP consists of a tuple $M = (S, A, O, R, T, \phi, p)$, where S, A, R, T and p are as above, O is the observation space, and $\phi : S \to O$ is the emission or observation function. In POMDPs, the policy only observes the observation of the state produced by ϕ.

The standard problem in an MDP is to learn a policy $\pi(a|s)$ which produces a distribution over actions given a state, such that the cumulative reward of the policy in the MDP is maximised:

$$\pi^* = \underset{\pi\in\Pi}{\text{argmax}}\, \mathbb{E}_{s\sim p(s_0)}\left[\mathcal{R}(s)\right],$$

where π^* is the optimal policy, Π is the set of all policies, and $\mathcal{R} : S \to \mathbb{R}$ is the *return* of a state, calculated as

$$\mathcal{R}(s) := \mathbb{E}_{a_t\sim\pi(a_t|s_t), s_{t+1}\sim T(s_{t+1}|s_t,a_t)}\left[\sum_{t=0}^{\infty} R(s_t, a_t, s_{t+1})|s_0 = s\right].$$

This is the total expected reward gained by the policy from a state s. The goal in a POMDP is the same, but with the policy taking observations rather than states as input. This sum may not to exist if the MDP does not have a fixed horizon, so we normally use one of two other forms of the return, either assuming a fixed number of steps per episode (a *horizon* H) or an exponential discounting of future rewards by a discount factor γ.

3.3 Contextual Markov Decision Processes

To discuss generalisation, we need a way of talking about a *collection* of tasks, environments or levels: the need for generalisation emerges from the fact we train and test the policy on different collections of tasks. Consider for example OpenAI Procgen [35]: in this benchmark suite, each game is a collection of procedurally generated levels. Which level is generated is completely determined by a level seed, and the standard protocol is to train a policy on a fixed set of 200 levels, and then evaluate performance on the full distribution of levels. Almost all other benchmarks share this structure: they have a collection of levels or tasks, which are specified by some seed, ID or parameter vector, and generalisation is measured by training and testing on different distributions over the collection of levels or tasks. To give a different example, in the Distracting Control Suite [36], the parameter vector determines a range of possible visual distractions applied to the observation of a continuous control task, from changing the colours of objects to controlling the camera angle. While this set of parameter vectors has more structure than the set of seeds in Procgen, both can be understood within the framework we propose. See Section 4.2 for discussion of the differences between these styles of environments.

To formalise the notion of a collection of tasks, we start with the Contextual Markov Decision Process (CMDP), as originally formalised in [13], but using the alternative formalism from [12]. Our formalism also builds on that presented in [37]. A contextual MDP (CMDP) $\mathcal{M}$ is an MDP where the state can be decomposed into a tuple $s = (c, s') \in \mathcal{S}_C$. Here, $s' \in \mathcal{S}$ is the underlying state, and we refer to $c \in \mathcal{C}$ as the *context*. This context takes the role of the seed, ID or parameter vector which determines the level. Hence, it does not change within an episode, only between episodes. The CMDP is the entire collection of tasks or environments; in Procgen, each game is a separate CMDP.

We factorise the initial state distribution as

$$p(s) = p((c, s')) := p(c)p(s'|c),$$

and we call $p(c)$ the *context distribution*. This distribution is what is used to determine the training and testing collections of levels, tasks or environments; in Procgen this distribution is uniform over the fixed 200 seeds at training time, and uniform over all seeds at testing time.

We often assume the context is not observed by the agent, making the CMDP a POMDP with observation space $\mathcal{S}$ and an emission function which discards the context information: $\phi((c, s')) := s'$. An analogous definition can be given for a contextual partially observable MDP (CPOMDP), where the emission function, as well as discarding the context, also transforms the state conditioned (on that context). We will generally use "MDP" to refer to environments that are either MDPs or POMDPs.

As the reward function, transition function, initial state distribution and emission function all take the context as input, the choice of context determines everything about the MDP apart from the action space, which we assume is fixed. Given a context c, we call the MDP resulting in the restriction of the CMDP $\mathcal{M}$ to the single context a *context-MDP* $\mathcal{M}_c$. This is a specific task or environment, for example a single level of a game in Procgen, as specified by a single random seed that is the context.

Some MDPs have stochastic transition or reward functions. When these MDPs are simulated, researchers often have control of this stochasticity through the choice of a random seed. In theory, these stochastic MDPs could be considered deterministic contextual MDPs, where the context is the random seed. We do not consider stochastic MDPs as automatically contextual in this way and assume that the random seed is always chosen randomly, rather than being modelled as a context. This more closely maps to real-world scenarios with stochastic dynamics where we cannot control the stochasticity.

3.4 Training And Testing Contexts

We now describe the class of generalisation problems we focus on, using the CMDP formalism. As mentioned, the need for generalisation emerges from a difference between the training and testing environments, and so we want to specify both a set of training context-MDPs and a testing set. We specify these sets of context-MDPs by their contexts, as the context uniquely determines the MDP.

First, we need to describe how to use training and testing context sets to create new CMDPs. We assume the context distribution is uniform over the entire context set if not otherwise specified. For any CMDP $\mathcal{M} = \langle S, A, R, T, C, p \rangle$,

we can choose a subset of the context set $C' \subseteq C$, and then produce a new CMDP $\mathcal{M}|_{C'}$ by setting the context set to C', and adjusting the context distribution to be uniform over C'. This allows us to split the total collection of tasks into smaller subsets, as determined by the contexts. For example, in Procgen any possible subset of the set of all seeds can be used to define a different version of the game with a limited set of levels.

For any CMDP $\mathcal{M}$ we can define the expected return of a policy in that CMDP as

$$\mathbf{R}(\pi, \mathcal{M}) := \mathbb{E}_{c \sim p(c)}[\mathcal{R}(\pi, \mathcal{M}|_c)],$$

where $\mathcal{R}$ is the expected return of a policy in an MDP and $p(c)$ is the context distribution as before.

To formally specify a generalisation problem, we choose a training context set C_{train} and a testing context set C_{test}. We train a policy using the training context-set CMDP $\mathcal{M}|_{C_{\text{train}}}$ for some number of training steps (possibly infinite), and the resulting policy is then evaluated on the testing context set CMDP $\mathcal{M}|_{C_{\text{test}}}$. The objective is for the expected return on the testing context set to be as high as possible:

$$J(\pi) := \mathbf{R}(\pi, \mathcal{M}|_{C_{\text{test}}}). \tag{2}$$

For example, in Procgen we want to achieve the highest return possible on the testing distribution (which is the full distribution over levels) after training for 25 million steps on the training distribution of levels (which is a fixed set of 200 levels). We call this *zero-shot policy transfer* [14, 15].

As in supervised learning, we can consider the gap between training and testing performance as a measure of generalisation. We define this analogously to in supervised learning (Eq. (1)), swapping the ordering between training and testing (as we maximise reward, rather than minimise loss):

$$\text{GenGap}(\pi) := \mathbf{R}(\pi, \mathcal{M}|_{C_{\text{train}}}) - \mathbf{R}(\pi, \mathcal{M}|_{C_{\text{test}}}). \tag{3}$$

This formalism defines a class *of generalisation problems, each determined by a choice of CMDP, training and testing context sets*. This means that we do not make any assumptions about shared structure within the CMDP between context-MDPs: for any specific problem some assumption of this kind will be required (Section 3.6), but we do not believe there is a unifying assumption behind all generalisation problems apart from those stated here.

3.5 Examples Of Applying The Formalism

We chose this formalism as it is simple to understand, captures all the problems we are interested in, and is based on prior work. To further justify why this formalism is useful, and to give intuition about how it can be used in a variety of settings, we give several examples of applying it to existing benchmarks and hypothetical real-world scenarios:

- As described throughout this section, OpenAI Procgen [35] is a popular benchmark for generalisation in RL. It is a suite of 16 procedurally generated arcade games, with variation in the state space and observation function: the games consist of different levels with different layouts or enemy numbers, as well as different visual styles which do not impact the dynamics or reward function. In this environment, the context is a single random seed that acts as input to the level generation. We can understand the context as mostly changing the initial state distribution to select the initial state for a given level, and then also conditioning the observation function. For some games (such as `starpilot`) enemies appear throughout the level, so the context conditions the transition function to spawn enemies at the specified time.
- Sim-to-real is a classic problem of generalisation, and one which can be captured in this framework. The context set will be split into those contexts which correspond to simulation, and those that correspond to reality. The context conditions the dynamics, observation function and state space. The CMDP can often be understood as effectively a union of two CMDPs, one for reality and one for simulation, with shared action space and observation space type. Domain randomisation approaches are motivated by the idea that producing a wide range of possible contexts in simulation will make it more likely that the testing distribution of contexts is closer to the expanded training distribution.
- Healthcare is a promising domain for deploying future RL methods, as there are many sequential decision-making problems. For example, the task of diagnosing and treating individual patients can be understood as a CMDP where the patient effectively specifies the context: patients will react differently to tests and treatments (dynamics variation) and may provide different measurements (state variation). The context can always condition the relevant MDP function to control the variation. Generalising to treating new patients is then exactly generalising to novel contexts. In this setting, we may be able to assume some part of the context (or some information about the context) is observable, as we will have access to patients' medical history and personal information.

- Autonomous vehicles are another area where RL methods could be applied. These vehicles will be goal-conditioned in some sense, such that they can perform different journeys (reward variation). Driving in different locations (state space variation), under different weather and lighting conditions due to the time of day (observation function variation) and on different road surfaces (dynamics variation) are all problems that need to be tackled by these systems. We can understand this in the CMDP framework, where the context contains information about the weather, time of day, location and goal, as well as information about the state of the current vehicle. Some of this context will be observed directly, and some may be inferred from observation. In this setting we may only be able to train in certain contexts (i.e. certain cities, or restricted weather conditions), but we require the policy to generalise to the unseen contexts well.

3.6 Additional Assumptions For More Feasible Generalisation

In choosing the CMDP formalism we opted to formalise generalisation in a way that captures the full class of problems we are concerned with, but this means that it is almost certainly impossible to prove any formal theoretical guarantees on learning performance using solely the CMDP structural assumptions. While we do not prove this, it is easy to see how one could design pathological CMDPs where generalisation to new contexts is entirely impossible without strong domain knowledge of the new contexts.

To have any chance of solving a specific generalisation problem then, further assumptions (either explicit or implicit) have to be made. These could be assumptions on the type of variation, the distributions from which the training and testing context sets are drawn, or additional underlying structure in the context set. We describe several popular or promising assumptions here, and note that the taxonomy in Section 4 also acts as a set of possible additional assumptions to make when tackling a generalisation problem.

Assumptions on the Training and Testing Context Set Distributions. One assumption which is often made is that while the training and testing context sets are not identical, the elements of the two sets have been drawn from the same underlying distribution, analogously to the iid data assumption in supervised learning. For example, this is the setup of OpenAI Procgen [35], where the training context set is a set of 200 seeds sampled uniformly at random from the full distribution of seeds, and the full distribution is used as the testing context set.

However, many works on generalisation in RL do not assume that the train and test environments are drawn from the same distribution. This is often called *Domain Generalisation*, where we refer to the training and testing environments as different *domains* that may be similar but are not from the same underlying generative distribution. Concrete examples occur in robotics such as the *sim-to-real* problem.

Further Formal Assumptions of Structure. Another kind of assumption that can be made is on the structure of the CMDP itself, e.g. the context space or transition function. There are several families of MDPs with additional structure which could enable generalisation. However, these assumptions are often not explicitly made when designing benchmarks and methods, which can make understanding why and how generalisation occurs difficult. A detailed discussion and formal definitions for these structures can be found in Appendix A, but we provide a high-level overview here, focusing on assumptions that have been used in practice, and those that hold particular promise for generalisation.

An example of a structured MDP that has been used to improve generalisation is the *block MDP* [38]. It assumes a *block structure* in the mapping from a latent state space to the given observation space, or that there exists another MDP described by a smaller state space with the same behaviour as the given MDP. This assumption is relevant in settings where we only have access to high-dimensional, unstructured inputs, but know that there exists a lower-dimensional state space that gives rise to an equivalent MDP. Du et al. [38] use this assumption for improved bounds on exploration that relies on the size of the latent state space rather than the given observation space. Zhang et al. [39] develop a representation learning method that disentangles relevant from irrelevant features, improving generalisation to environments where only the irrelevant features change, a simple form of *systematicity* (Section 3.1 [33]). This is a rare example of a method explicitly utilising additional assumptions of structure to improve generalisation.

Factored MDPs [40, 41] can be used to describe object-oriented environments or multi-agent settings where the state space can be broken up into independent factors, i.e. with sparse relationships over the one-step dynamics. This can be leveraged to learn dynamics models that explicitly ignore irrelevant factors in prediction or to compute improved sample complexity bounds for policy learning [42], and seems particularly relevant for generalisation as additional structure in the context set could map onto the factored structure in the transition and reward functions. An initial example of using a similar formalism to a factored MDP in a multi-domain RL setting is [43], although it does not target the zero-shot policy transfer setting directly. We hope to see more work applying these kinds of structural assumptions to generalisation problems.

3.7 Remarks And Discussion

On Metrics for Generalisation. There are two obvious ways of evaluating generalisation performance of models. One is to only look at absolute performance on evaluation tasks (Eq. (2)), and the other is to look at the generalisation gap (Eq. (3)). In supervised learning, generalisation capabilities of different algorithms are usually evaluated via final performance on an evaluation task. When the tasks used to evaluate a model are close to (or the same as) the tasks that the model will eventually be deployed on, it is clear that final performance is a good metric to evaluate on. However, in RL the benchmark tasks we use are often very dissimilar to the eventual real world tasks we want to apply these algorithms to. Further, RL algorithms are currently still quite brittle and performance can vary greatly depending on hyperparameter tuning and the specific task being used [44]. In this setting, we may care more about the generalisation *potential* of algorithms by decoupling generalisation from training performance and evaluating using generalisation gap instead.

However, using generalisation gap as a metric assumes that there exists a general measure of progress towards tackling generalisation in RL, but given how broad the current set of assumptions is, this seems unlikely: across such a broad class, objectives may even be conflicting [11]. Therefore, our recommendation is to focus on problem-specific benchmarks and revert to the SL standard of using overall performance in specific settings (e.g. visual distractors, stochastic dynamics, sparse reward, hard exploration). The generalisation performance of various RL algorithms is likely contingent on the type of environment they are deployed on and therefore careful categorisation of the type of challenges present at deployment is needed to properly evaluate generalisation capability (for further discussion see Section 4.3, *The Downsides of Procedural Content Generation for Generalisation* and Section 6.2).

Angles to Tackle the Generalisation Problem. While we aim to improve test-time performance Eq. (2), we often do not have access to that performance metric directly (or will not when applying our methods in the real world). To improve the test-time performance, we can either (1) increase the train-time performance while keeping the generalisation gap constant, (2) decrease the generalisation gap while keeping the train-time reward constant, or (3) do a mixture of the two approaches. Work in RL not concerned with generalisation tends to (implicitly) take the first approach, assuming that the generalisation gap will not change.[2] Work on generalisation in RL instead normally aims at (2) reducing the generalisation gap explicitly, which may reduce train-time performance but increase test-time performance. Some work also aims at improving train-time performance in a way that is likely to keep the generalisation gap constant.

Motivating Zero-Shot Policy Transfer. In this work, we focus on zero-shot policy transfer [14, 15]: a policy is transferred from the training CMDP to the testing CMDP, and is not allowed any further training in the test context-MDPs (hence zero-shot). While we do not cover methods that relax the zero-shot assumption, we believe that in a real-world scenario it will likely be possible to do so.[3] However, zero-shot policy transfer is still a useful problem to tackle, as solutions are likely to help with a wide range of settings resulting from different relaxations of the assumptions made here: zero shot policy transfer algorithms will be used as a base which is then built upon with domain-specific knowledge and data.

Note that while "training" and "learning" are difficult to define precisely, the fact that the evaluation is based on the expected testing return within a single episode means that online learning or adaptation is unlikely to be a tractable solution *unless* the adaptation happens within a single episode. Several methods do take this approach, as described in Section 5.2, *Adapting Online*. To be clear, we are grounding ourselves in this specific *objective*, and not placing any restrictions on the *properties* of methods.

4 Benchmarks For Generalisation In Reinforcement Learning

In this section we give a taxonomy of benchmarks for generalisation in RL. A key split in the factors of variation for a benchmark is those factors concerned with the environment and those concerned with the evaluation protocol. A benchmark task is a combination of a choice of environment (a CMDP, covered in Section 4.1) and suitable evaluation protocol (a train and test context set, covered in Section 4.2). This means that all environments support multiple possible evaluation protocols, as determined by their context sets.

Having categorised the set of benchmarks, we point out the limitations of the purely PCG approach to building environments (Section 4.3, *The Downsides of Procedural Content Generation for Generalisation*), as well as discuss

[2] If the training environment is identical to the testing environment, then the generalisation gap will always be 0.

[3] For example by using unsupervised data or samples in the testing environment, utilising some description of the contexts such that generalisation is possible, or enabling the agent to train in an online way in the testing context-MDPs.

the range of difficulty among generalisation problems (Section 4.3, *What Generalisation Can We Expect?*). More discussion of future work on benchmarks for generalisation can be found in Sections 6.1, 6.2 and 6.4.

4.1 Environments

4.1.1 Categorising Environments That Enable Generalisation

Table 1: Categorisation of Environments for Generalisation. In the **Style** column, LC stands for Language-Conditioned, ConCon for Continuous Control. In the **Contexts** column, PCG stands for Procedural Content Generation, Con for continuous, Dis-C for discrete cardinal and Dis-O for discrete ordinal. In the **Variation** column, S, D, O and R are respectively state, dynamics, observation or reward function variation. In the **Name** column, † refers to environments that were not originally designed as zero-shot policy transfer generalisation benchmarks but could be adapted to be. See main text for a more detailed description of the columns.

Name	Style	Contexts	Variation
Alchemy † [45]	3D	PCG	D, R, S
Atari Game Modes [46]	Arcade	Dis-C	D, O, S
BabyAI [47]	Grid, LC	Dis-C, Dis-O, PCG	R, S
CARL [48]	Varied	Con, Dis-C, Dis-O	D, O, R, S
CARLA [49, 50]	3D, Driving	Dis-C	O
CausalWorld † [51]	3D, ConCon	Con, Dis-C, Dis-O	D, O, R, S
Crafter [52]	Arcade, Grid	PCG	S
Crafting gridworld [53]	Grid, LC	Dis-C	R, S
DCS [36]	ConCon	Con, Dis-C	O
DistractingCarRacing [54, 5]	Arcade	Dis-C	O
DistractingVizDoom [54, 55]	3D	Dis-C	O
DMC-GB [56]	ConCon	Con, Dis-C	O
DMC-Remastered [57]	ConCon	Con, Dis-C	O
GenAsses [58]	ConCon	Con	D, S
GVGAI [59]	Grid	Dis-C	D, O, S
iGibson [49, 60]	3D	Dis-C	O, S
Jericho † [61]	Text	Dis-C	D, R, S
JumpingFromPixels [62]	Arcade	Con	S
KitchenSink [63]	3D, ConCon	Dis-C	O, S
Malmo [64]	3D, Arcade	Dis-C, Dis-O	R, S
MarsExplorer [65]	Grid	PCG	S
MazeExplore [66]	3D	PCG	O, S
MDP Playground † [67]	ConCon, Grid	Con, Dis-C, Dis-O	D, O, R, S
Meta-World † [68]	3D, ConCon	Con, Dis-C	R, S
MetaDrive [69]	3D, Driving	Dis-C, Dis-O, PCG	D, S
MiniGrid [70]	Grid	PCG	S
MiniHack [71]	Grid	Dis-C, Dis-O, PCG	S
NaturalEnvs CV [72]	Grid	PCG	O, R, S
NaturalEnvs MuJoCo [72]	ConCon	Dis-C	O
NLE [73]	Grid	PCG	S
Noisy MuJoCo [74]	ConCon	Con, Dis-C	D, O
NovelGridworlds [75]	Grid	Dis-C	D, S
Obstacle Tower [76]	3D	Dis-C, PCG	O, S
OpenAI Procgen [35]	Arcade	PCG	O, S
OverParam Gym [16]	ConCon	Con	O
OverParam LQR [16]	LQR	Con	O
ParamGen [77]	3D, LC	Dis-C, Dis-O	R, S
RLBench † [78]	3D, ConCon, LC	Con, Dis-C, Dis-O	R, S
RoboSuite [49, 50]	3D, ConCon	Dis-C	O
Rogue-gym [79]	Grid	PCG	S
RTFM [80]	Grid, LC	PCG	D, R, S
RWRL † [81]	ConCon	Con	D
TextWorld † [82]	Text	Con, Dis-C, PCG	D, O, R, S
Toybox † [83]	Arcade	Con, Dis-C, Dis-O	D, O, S
TrapTube [84]	Grid	Con, Dis-C	D, O, S
WordCraft [85]	LC, Text	Con, Dis-C, Dis-O	R, S
XLand [21]	3D, LC	Con, Dis-C, Dis-O, PCG	D, O, R, S
Phy-Q [86]	Arcade	Dis-C, PCG	S

In Table 1, we list the available environments for testing generalisation in RL, as well as summarising each environment's key properties. These environments all provide a non-singleton context set that can be used to create a variety of evaluation protocols. Choosing a specific evaluation protocol then produces a benchmark. We describe the meaning of the columns in Table 1 here.

Style. This gives a rough high-level description of the kind of environment.

Contexts. This describes the context set. In the literature there are two approaches to designing a context set, and the key difference between these approaches is whether the context-MDP creation is accessible and visible to the researcher. The first, which we refer to as Procedural Content Generation (PCG), relies on a single random seed to determine multiple choices during the context-MDP generation. Here the context set is the set of all supported random seeds. This is a black-box process in which the researcher only chooses a seed.

The second approach provides more direct control over the factors of variation between context-MDPs, and we call these *Controllable* environments. The context set is generally a product of multiple factor spaces, some of which may be discrete (i.e. a choice between several colour schemes) and some continuous (i.e. a friction coefficient in a physical simulation). Borrowing from [77], a distinction between discrete factors of variation is whether they are cardinal (i.e. the choices are just a set with no additional structure) or ordinal (i.e. the set has additional structure through an ordering). Examples of cardinal factors include different game modes or visual distractions, and ordinal factors are commonly the number of entities of a certain type within the context-MDP. All continuous factors are effectively also ordinal factors, as continuity implies an ordering.

Previous literature has defined PCG as any process by which an algorithm produces MDPs given some input [87], which applies to both kinds of context sets we have described. Throughout the rest of this survey we use "PCG" to refer to black-box PCG, which uses a seed as input, and "controllable" to refer to environments where the context set directly changes the parameters of interest in the context-MDPs, which could also be seen as "white-box PCG". We can understand (black-box) PCG settings as combinations of discrete and continuous factor spaces (i.e. controllable environments) where the choice of the value in each space is determined by the random generation process. However, only some environments make this more informative parametrisation of the context-MDPs available. In this table we describe environments where this information is not easily controllable as PCG environments. See Section 4.3, *The Downsides of Procedural Content Generation for Generalisation* for discussion of the downsides of purely PCG approaches.

Variation. This describes what varies within the set of context MDPs. This could be state-space variation (the initial state distribution and hence implicitly the state space), dynamics variation (the transition function), visual variation (the observation function) or reward function variation. Where the reward varies, the policy often needs to be given some indication of the goal or reward, so that the set of contexts is solvable by a single policy [88].

4.1.2 Trends In Environments

There are several trends and patterns shown in Table 1, which we draw the reader's attention to here. We describe 47 environments in total, and have aimed to be fairly exhaustive.[4]

There are a range of different **Style**s that these environments have, which is beneficial as generalisation methods should themselves be generally applicable across styles if possible. While numerically there is a focus on gridworlds (13, 28%) and continuous control (13, 28%), there are well established benchmarks for arcade styles [35] and 3D environments [76]. Looking at **Context** sets, we see that PCG is heavily used in generalisation environments, featuring in 17 (36%) environments. Many environments combine PCG components with controllable variation [47, 82, 76, 69, 21, 86]. Most environments have several different kinds of factors of variation within their context set.

There is a lot of differences between environments when looking at the **Variation** they use. Numerically, state variation is most common (35, 74%) followed by observation (26, 55%), and then dynamics (17, 36%) and reward (15, 32%). Most environments have multiple different types of variation (28, 60%), and while there are several environments targeted at just observation variation (10, 22%) or state variation (8, 17%), there is only a single environment with solely dynamics variation (RWRL [81]), and none with solely reward variation. State and Observation variations are often the easiest to engineer, especially with the aid of PCG. This is because changing the rendering effects of a simulator, or designing multiple ways the objects in a simulator could be arranged, is generally easier than designing a simulator engine that is parametrisable (for dynamics variation). Creating an environment for reward variation requires further design choices about how to specify the reward function or goal such that the environment satisfies the Principle Of

[4] As such, if you think there are missing environments, please contact us using the details provided in the author list

Unchanged Optimality [88]. PCG is often the only good way of generating a large diversity in state variation, and as such is often necessary to create highly varied environments. Only CausalWorld [51] enables easily testing all forms of variation at once.[5]

There are several clusters that can be pointed out in the collection of benchmarks: There are several PCG state-varying gridworld environments (MiniGrid, BabyAI, Crafter, Rogue-gym, MarsExplorer, NLE, MiniHack [70, 47, 52, 79, 65, 73, 71]), non-PCG observation-varying continuous control environments (RoboSuite, DMC-Remastered, DMC-GB, DCS, KitchenSink, NaturalEnvs MuJoCo [49, 57, 56, 36, 63, 72]), and multi-task continuous control benchmarks which could be adapted to zero-shot generalisation (CausalWorld, RLBench, Meta-world [51, 78, 68]).

4.2 Evaluation Protocols For Generalisation

As discussed, a benchmark is the combination of an environment and an evaluation protocol. Each environment supports a range of evaluating protocols determined by the context set, and often there are protocols recommended by the environment creators. In this section we discuss the protocols, and the differences between them. An evaluation protocol specifies the training and testing context sets, any restrictions on sampling from the training set during training, and the number of samples allowed from the training environment.

An important first attribute that varies between evaluation protocols is *context-efficiency*. This is analogous to sample efficiency, where only a certain number of samples are allowed during training, but instead we place restrictions on the number of *contexts*. This ranges from a single context, to a small number of contexts, to the entire context set.

PCG Evaluation Protocols. In fact, in purely PCG environments, the only meaningful factor of variation between evaluation protocols is the context efficiency restriction. As we have no control over the factors of variation apart from sampling random seeds, the only choice we have is how many contexts to use for training. Further, the only meaningful testing context set is the full distribution, as taking a random sample from it (the only other option) would just be an approximation of the performance on the full distribution. This limitation of PCG environments is discussed further below (Section 4.3, *The Downsides of Procedural Content Generation for Generalisation*).

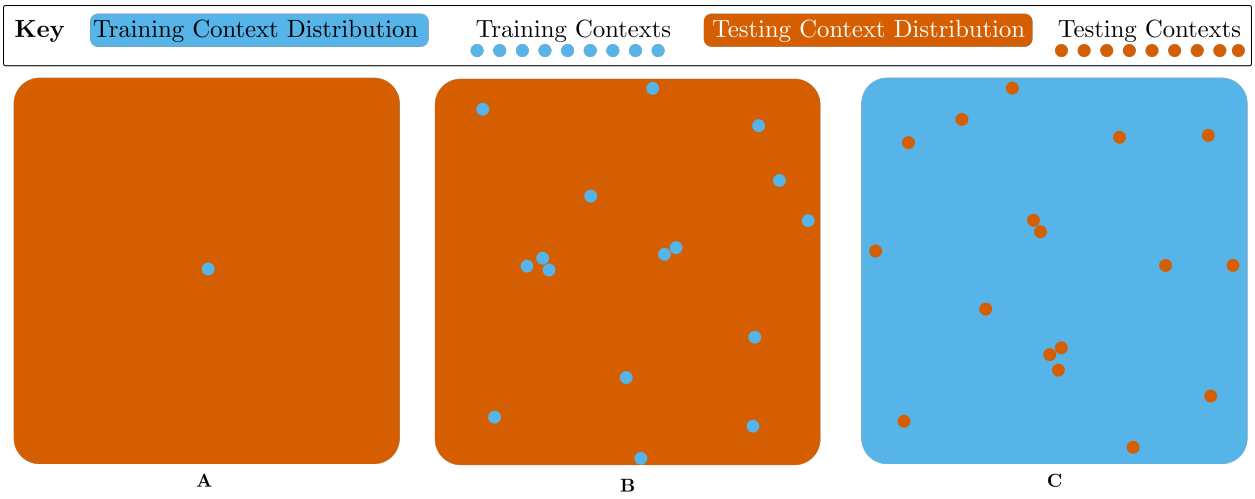


Figure 2: **Visualisation of Evaluation Protocols for PCG Environments**. **A** is a single training context, and the whole context set for testing. **B** uses a small collection of training contexts randomly sampled from the context set, and the entire space for testing. **C** effectively reverses this, using the entire context set for training apart from several randomly sampled held-out contexts that are used for testing. The lack of axes indicates that these sets have no structure.

This gives three classes of evaluation protocol for PCG environments, as determined by their training context set: A single context, a small set of contexts, or the full context set. These are visualised in Fig. 2 **A**, **B** and **C** respectively. There are not any examples of protocol **A** (for purely PCG environments), likely due to the high difficulty of such a challenge.

[5]While CARL [48], MDP Playground [67] and TextWorld [82] are also categorised as containing all forms of variation, CARL is a collection of different environments, none of which have all variation types, and MDP Playground and TextWorld would require significant work to construct a meaningful evaluation protocol which varies along all factors. Further, MDP Playground does not provide a method for describing the changed reward function to the agent, and there is uncertainty in how to interpret observation variation in text-based games like TextWorld.

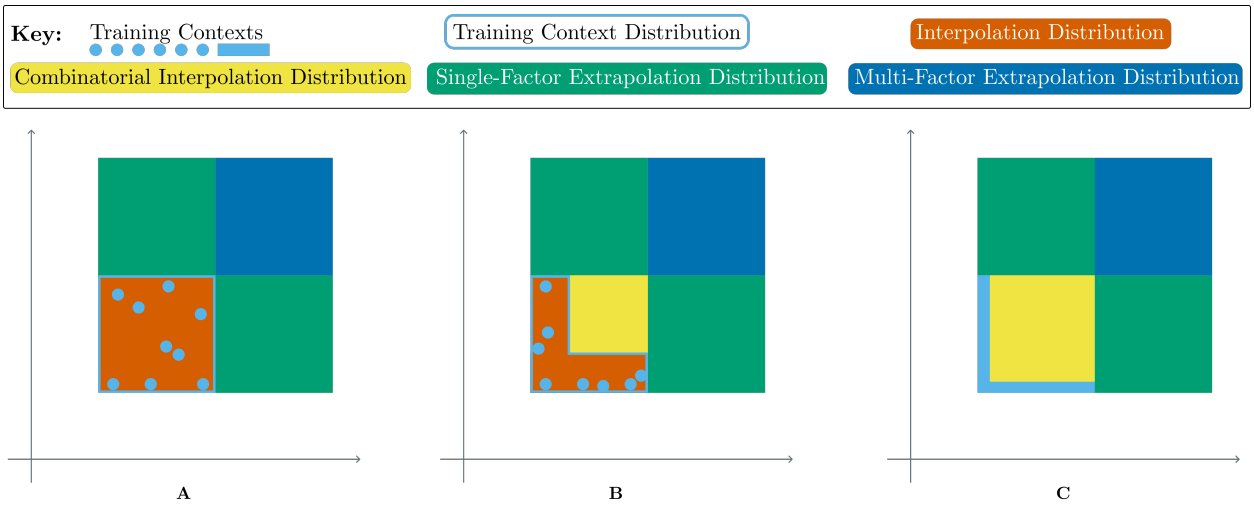
For protocol **B**, while "a small set of contexts" is imprecise, the relevant point is that this set is meaningfully different from the full context set: it is possible to overfit to this set without getting good performance on the testing set. Examples of this protocol include both modes of OpenAI Procgen [35], RogueGym [79], some uses of JumpingFromPixels [62] and MarsExplorer [65].

Protocol **C** is commonly used among PCG environments that are not explicitly targeted at generalisation (MiniGrid, NLE, MiniHack, Alchemy [70, 73, 71, 45]). The testing context set consists of seeds held out from the training set, and otherwise during training the full context set is used. This protocol effectively tests for more robust RL optimisation improvements, but does not test for generalisation beyond avoiding memorising. While this protocol only tests for generalisation in a very weak sense, it still matches a wider variety of real-world scenarios than the previous standard in RL of testing on the training set, and so we believe it should be the standard evaluation protocol in RL (not just in generalisation), and the previous standard should be considered a special case.

Controllable Environment Evaluation Protocols. Many environments do not use only PCG, and have factors of variation that can be controlled by the user of the environment. In these controllable environments, there is a much wider range of possible evaluation protocols.

The choice in PCG protocols – between a single context, a small set, or the entire range of contexts – transfers to the choice for each factor of variation in a controllable environment. For each factor we can choose one of these options for the training context set, and then choose to sample either within or outside this range for the testing context set. The range of options is visualised in Fig. 3.

Figure 3: **Visualisation of Evaluation Protocols for Controllable Environments**. Each diagram visualises one possible training context set (blue), and multiple possible testing context sets (all other colours). In **A** we choose the range for each factor of variation independently for the training distribution, resulting in a convex shape for this distribution. In this setting possible testing distributions can either be interpolation (red), extrapolation along a single factor (either green square) or extrapolation along both factors (blue). In **B** and **C** the ranges for each factor of variation are linked together, resulting in a non-convex shape for the training distribution. This allows an additional type of generalisation to be tested, combinatorial interpolation (yellow), where the factors take values seen during training independently, but in unseen combinations. We continue to have the previous interpolation and extrapolation testing distributions. The difference from **B** to **C** is in the width of the training distribution in the axes along which we expect the agent to generalise. In **C** the policy will not be able to learn that the two factors can vary independently at all, making all forms of generalisation harder. Note that in actual environments and real-world settings it is likely this space will be higher than two dimensions and contain non-continuous and non-ordinal axes. The axes indicate that in this setting we have control over these factors of variation, in contrast to Fig. 2.



Making this choice for each factor independently gives us a convex training context set within the full context set (Fig. 3 **A**). For testing, each factor can then be either inside or outside this convex set (often respectively referred to as interpolation and extrapolation). The number of factors chosen to be extrapolating contributes to the difficulty of the evaluation protocol.

However, if we create correlations or links between values of factors during training, we can get a non-convex training context set within the full context set (Fig. 3 **B**). Each possible testing context can either be within the training context set (fully interpolation), within the set formed by taking the convex hull of the non-convex training context set (combinatorial interpolation), or fully outside the convex hull (extrapolation). Combinatorial interpolation tests

the ability of an agent to exhibit *systematicity*, a form of compositional generalisation discussed in Section 3.1. For ordinal factors, we can also choose disjoint ranges, which allows us to test interpolation along a single axis (i.e. taking values between the two ranges). Note that when discussing convex hulls, this only applies to factors of variation that are continuous or discrete-ordinal; for cardinal factors of variation, the convex hull just includes those values sampled during training.

For example, consider a policy trained in a CMDP where the context set consists of values for friction and gravity strength. During training, the environments have either a friction coefficient between 0.5 and 1 but gravity fixed at 1, or gravity strength ranging between 0.5 and 1 but friction fixed at 1 (the light blue line in Fig. 3 **C**). Testing contexts which take friction and gravity values within the training distribution are full interpolation (e.g. $(f = 0.5, g = 1), (f = 1, g = 1)$), contexts which take values for friction and gravity which have been seen independently but not in combination are combinatorial interpolation (e.g. $(f = 0.5, g = 0.5), (f = 0.5, g = 0.9)$, the yellow area), and contexts which take values for friction and gravity which are outside the seen ranges during training are full extrapolation (e.g. $(f = 0.2, g = 0.5), (f = 1.1, g = 1.5)$, either the dark blue or green areas).

We can still consider the number of contexts within the training context set, which controls the density of the training context set, given its shape. When testing for extrapolation we can also vary the "width" of the training context set on the axis of variation along which extrapolation is being tested (Fig. 3 **B** vs **C**). These tests evaluate the agent's ability to exhibit *productivity* (Section 3.1). Of course, generalisation will be easier if there is a wide diversity of values for this factor at training time, even if the values at test time are still outside this set. For example, if we are testing whether a policy can generalise to novel amounts of previously seen objects, then we should expect the policy to perform better if it has seen different amounts during training, as opposed to only having seen a single amount of the object during training. To expand on the friction and gravity example, during training the policy never sees gravity and friction varying together, which makes it much more difficult to generalise to the testing contexts. If gravity and friction did vary together during training then this would make generalisation easier.

A notable point in this space is that of a single training context, and a wide variety of testing contexts. This protocol tests for a strong form of generalisation, where the policy must be able to extrapolate to unseen contexts at test time. Because of the difficulty of this problem, benchmarks with this evaluation protocol focus on visual variation: the policy needs to be robust to different observation functions on the same underlying MDP [57, 36, 63]. The protocol is often motivated by the sim-to-real problem, where we expect an agent trained in a single simulation to be robust to multiple visually different real-world settings at deployment time.

Beyond this single point, it is challenging to draw any more meaningful categorisation from the current array of evaluation protocols. Generally, each one is motivated by a specific problem setting or characteristic of human reasoning which we believe RL agents should be able to solve or have respectively.

4.3 Discussion

There are several comments, insights and conclusions that can be gained from surveying the breadth of generalisation benchmarks, which we raise here.

Non-visual Generalisation. If testing of non-visual types of generalisation, then visually simple domains such as MiniHack [71] and NLE [73] should be used. These environments contain enough complexity to test for many types and strengths of non-visual generalisation but save on computation due to the lack of complex visual processing required. There are many real-world problem settings where no visual processing is required, such as systems control and recommender systems. Representation learning is still a problem in these non-visual domains, as many of them have a large variety of entities and scenarios which representations and hence policies need to generalise across.

DeepMind Control Suite Variants. A sub-category of generalisation benchmarks unto itself is the selection of DeepMind Control Suite [89] variants: DMC-Remastered, DMC-Generalisation Benchmark, Distracting Control Suite, Natural Environments [57, 56, 36, 72]. All these environments focus on visual generalisation and sample efficiency, require learning continuous control policies from pixels and introduce visual distractors that the policy should be invariant to which are either available during training or only present at deployment. We believe that Distracting Control Suite [36] is the most fully-featured variant in this space, as it features the broadest set of variations, the hardest combinations of which are unsolvable by current methods.

Unintentional Generalisation Benchmarks. Some environments listed in Table 1 were not originally intended as generalisation benchmarks. For example, Tosch et al. [83] presents three highly parametrisable versions of Atari games, and uses them to perform post hoc analysis of agents trained on a single variant. Some environments are not targeted at zero-shot policy transfer (CausalWorld, RWRL, RLBench, Alchemy, Meta-world [51, 81, 78, 45, 68]), but

could be adapted to such a scenario with a different evaluation protocol. More generally, all environments provide a context set, and many then propose specific evaluation protocols, but other protocols could be used as long as they were well-justified. This flexibility has downsides, as different methods can be evaluated on subtly different evaluation protocols which may favour one over another. We recommend being explicit when using these benchmarks in exactly which protocol is being used and comparing with evaluations of previous methods. Using a standard protocol aids reproducibility.

The Downsides of Procedural Content Generation for Generalisation. Many environments make use of procedural content generation (PCG) for creating a variety of context-MDPs. In these environments the context set is the set of random seeds used for the PCG and has no additional structure with which to control the variation between context-MDPs.

This means that while PCG is a useful tool for creating a large set of context-MDPs, there is a downside to purely PCG-based environments: the range of evaluation protocols supported by these environments is limited to different sizes of the training context set. Measuring generalisation along specific factors of variation is impossible without significant effort either labelling generated levels or unravelling the PCG to expose the underlying parametrisation which captures these factors. Often more effort is required to enable setting these factors to specific values, as opposed to just revealing their values for generated levels. Hence, PCG benchmarks are testing for a "general" form of generalisation and RL optimisation, but do not enable more targeted evaluation of specific types of generalisation. This means making research progress on specific problems is difficult, as focusing on the specific bottleneck in isolation is hard.

An interesting compromise, which is struck by several environments, is to have some low-level portion of the environment procedurally generated, but still have many factors of variation under the control of the researcher. For example, Obstacle Tower [76] has procedurally generated level layouts, but the visual features (and to some extent the layout complexity) can be controlled. Another example is MiniHack [71], where entire MDPs can be specified from scratch in a rich description language, and PCG can fill in any components if required. These both enable more targeted types of experimentation. We believe this kind of combined PCG and controllable environment is the best approach for designing future environments; some usage of PCG will be necessary to generate sufficient variety in the environments (especially the state space), and if the control is fine-grained enough to enable precise scientific experimentation, then the environment will still be useful for disentangling progress in generalisation.

Compositional Generalisation in Contextual MDPs. Compositional generalisation is a key point of interest for many researchers (see Section 3.1). In *controllable* environments, different evaluation protocols enable us to test for some of the forms of compositional generalisation introduced in Section 3.1 [33]: (1) Systematicity can be evaluated using a multidimensional context set, and testing on novel combinations of the context dimensions not seen at training time (combinatorial interpolation in Fig. 3). (2) Productivity can be evaluated with ordinal or continuous factors, measuring the ability to perform well in environments with context values beyond those seen at training time (either type of extrapolation in Fig. 3). If dealing with CMDPs where the context space is partially language, as in language-conditioned RL [90], then the evaluations discussed in [33] can be directly applied to the language space. Crucially, a controllable environment with a structured context space is required to test these forms of compositional generalisation, and to ensure that the agent is seeing truly novel combinations at test time; this is difficulty to validate in PCG environments like OpenAI Procgen [35] or NLE [73].

The other forms of compositional generalisation in [33] require additional structure not captured by the choices of evaluation protocol, and we describe what testing these forms could entail here: (3) substitutivity through the use of synonyms (in language) or equivalent objects and tools; (4) locality through comparing the interpretation of an agent given command A and command B separately vs. the combination of A + B and if those interpretations are different; and (5) overgeneralisation through how the agent responds to exceptions in language or rules of the environment.

What Generalisation Can We Expect? In Section 4.2 we discussed a variety of different possible evaluation protocols for different styles of generalisation benchmark. However, it is a different question in which protocols we can expect reasonable performance. For example, it is unreasonable to expect a standard RL policy trained *tabula rasa* on a single level of NLE [73] to generalise to dissimilar levels, as it might encounter entirely unseen entities or very out-of-distribution combinations of entities and level layouts. Each evaluation protocol measures a different kind of generalisation strength, and they hence form a kind of partial ordering where "easier" evaluation protocols come before "harder" protocols. We outline this ordering here:

- Increasing the number of samples can make an evaluation protocol easier, but often only to a point: more samples are unlikely to bring greater variety which is needed for generalisation. Increasing the number of contexts (while keeping the shape of the context set the same) also makes an evaluation protocol easier. Even a small amount of additional variety can improve performance.

- The number of factors of variation which are extrapolating or combinatorially interpolating in the testing context set can also be varied. The more there are, the more difficult the evaluation protocol. Further, the width of the range of values that the extrapolating factors take at training time can vary. This is linked to the number of contexts but is also related to the variety available during training time along these axes of variation.
- Following [77], we consider the difficulty of interpolating and extrapolating along different types of factors of variation. Interpolation along ordinal axes is likely the easiest, followed by cardinal axes interpolation (which happens through unseen combinations of seen values for a cardinal axis combined with any other axis), and then extrapolation along ordinal axes. Finally, extrapolation along cardinal axes is the most difficult.

As the difficulty of an evaluation protocol increases, it becomes less likely that standard RL approaches will get good performance. In more difficult protocols which involve extrapolation of some form, generalisation is unlikely to occur at all with standard RL methods, as there is no reason to expect a policy to generalise correctly to entirely unseen values. That does not mean that this type of generalisation is impossible: it just makes clear the fact that to achieve it, more than standard RL methods will be needed. That is, methods incorporating prior knowledge[6] such as transfer from related environments [29]; strong inductive biases [91] or assumptions about the variation; or utilising online adaptation [92, 93] will be necessary to produce policies that generalise in this way.

5 Methods For Generalisation In Reinforcement Learning

We now classify methods that tackle generalisation in RL. The problem of generalisation occurs when the training and the testing context sets are not identical. These context sets and their similarities and differences are what how to improve generalisation. There are many types of generalisation problems (as described in more detail in Section 4), and hence there are many different styles of methods. We categorise the methods into those that try and increase the similarity between training and testing data and objective (Section 5.1), those that explicitly aim to handle differences between training and testing environments (Section 5.2), and those that target RL-specific issues or optimisation improvements which aid generalisation performance (Section 5.3).

See Fig. 4 for a diagram of this categorisation, and Table 2 for a table classifying methods by their approach, the environment variation they were evaluated on, and whether they mostly change the environment, loss function or architecture. Performing this comprehensive classification enables us to see the under-explored areas within generalisation research, and we discuss future work for methods in Section 6.6.

5.1 Increasing Similarity Between Training And Testing

All else being equal, the more similar the training and testing environments are, the smaller the generalisation gap and the higher the test time performance. This similarity can be increased by designing the training environment to be as close to the testing environment as possible. Assuming this has been done, in this section we cover methods that make the data and objective being used to learn the policy during training closer to that which would be used if we were optimising on the testing environment.

Data Augmentation and Domain Randomisation. Two natural ways to make training and testing data more similar is data augmentation [94] and domain randomisation [95, 96, 97]. This is especially effective when the variation between the training and testing environments is known, as then a data augmentation or domain randomisation can be used which captures this variation. In practice there is only so far this approach can go, as stronger types of variation often cannot be captured by this simple method.

Data augmentation (DA) can be viewed in two ways. First, the augmented data points are seen as additional data to train the model. This interpretation is what causes us to classify DA techniques as trying to increase similarity between training and testing data. In the second view, DA can be used to enforce the learning of an invariance, by regularising the model to have the same output (or the same internal representations) for different augmented data points. In this view, DA is more with encoding inductive biases, which we cover in Section 5.2, *Encoding Inductive Biases*. We include all DA work in this section for clarity.

There are many examples of using DA in RL, although not all of them are targeted at generalisation performance. Raileanu et al. [98, UCB-DrAC] adapts the DA technique DrQ [99] to an actor-critic setting [100, PPO], and introduces a method for automatically picking the best augmentation during training. Wang et al. [101, Mixreg] adapts mixup [102] to the RL setting, which encourages the policy to be linear in its outputs with respect to mixtures of possible inputs. Zhang and Guo [103, PAADA] use adversarial DA combined with mixup. Lee et al. [104, RandFM] uses a

[6] `http://betr-rl.ml/2020/`

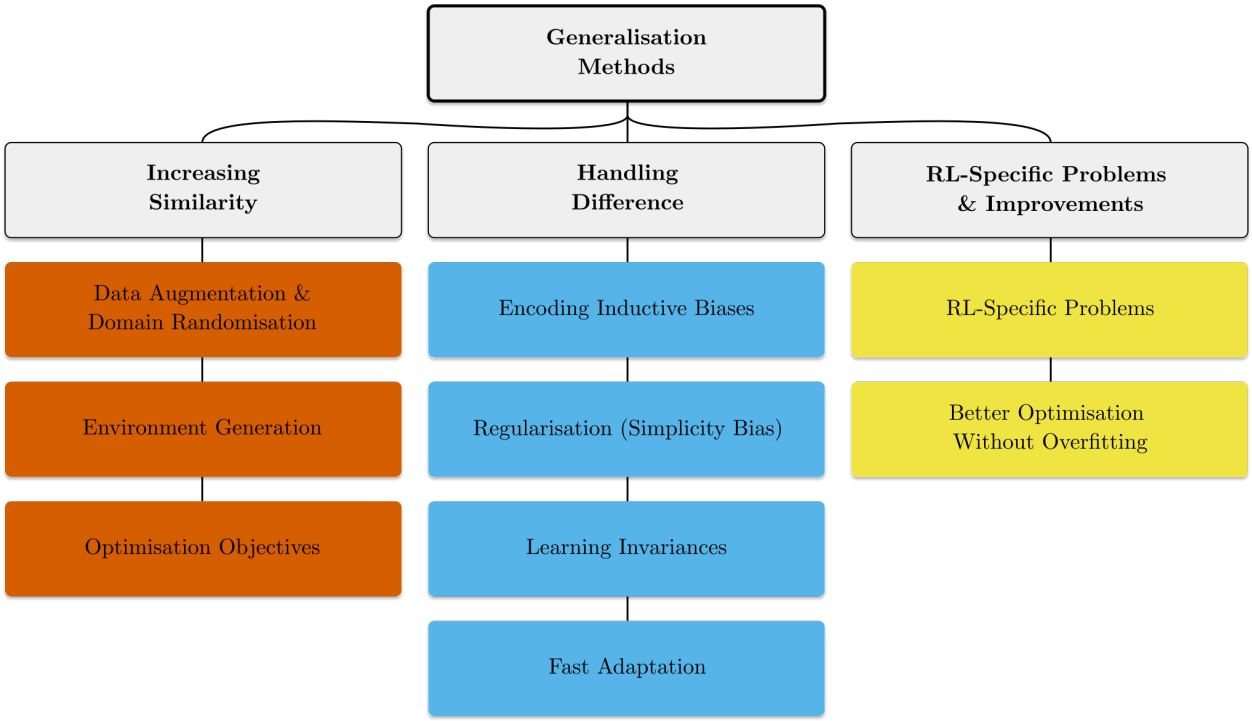


Figure 4: **Categorisation of methods for tackling generalisation in reinforcement learning**

randomised convolutional layer at the start of the network to improve robustness to a wide variety of visual inputs. All these methods [98, 101, 103, 104] show improved performance on CoinRun [105] or OpenAI Procgen [35] by improving both training and testing performance, and some also show gains on other benchmarks such as visually distracting DeepMind Control (DMC) variants. Hansen and Wang [56, SODA] uses similar augmentations as before but only to learn a more robust image encoder, while the policy is trained on non-augmented data, demonstrating good performance on DMC-GB [56]. James et al. [106, RCAN] use DA to learn a visual mapping from any different observation back to a canonical observation of the same state, and then train a policy on this canonical observation. They show improved sim-to-real performance on a robotic grasping task.

Ko and Ok [107, InDA,ExDA] show that the time at which the augmentations are applied is important for the performance: some augmentations help during training, whereas others only need to be applied to regularise the final policy. Fan et al. [49, SECANT] introduce a method for combining DA with policy distillation. As training on strong augmentations can hinder performance, they first train on weak augmentations to get an expert policy, which they then distil into a student policy trained with strong augmentations. Hansen et al. [108, SVEA] argues that DA when naively applied increases the variance of Q-value targets, making learning less stable and efficient. They introduce adjustments to the standard data-augmentation protocol by only applying augmentations to specific components at specific points during the calculation of the loss function, evaluating performance on DMC-GB [56]. These works show that in RL settings the choice of when to apply augmentations and what type of augmentations to apply is non-trivial, as the performance of the model during training impacts the final performance through changing the data the policy learns from.

Domain Randomisation (DR) is the practice of randomising the environment across a distribution of parametrisations, aiming for the testing environment to be covered by the distribution of environments trained on. Fundamentally DR is just the creation of a non-singleton training context set, and then randomly sampling from this set. Tobin et al. [95], Sadeghi and Levine [96] and Peng et al. [97] introduced this idea in the setting of sim-to-real transfer in robotics. Much work has been done on different types of DR, and so we cover just sample here. OpenAI et al. [2] describes Automatic Domain Randomisation: instead of sampling possible environment parametrisations uniformly at random, this method dynamically adjusts the distribution in response to the agent's current performance. Ren et al. [109, Minimax DSAC] use adversarial training to learn the DR for improved robustness. Zhao and Hospedales [110, P2PDRL] improve DR though peer-to-peer distillation. Wellmer and Kwok [111, DDL] learns a world model in which to train a policy and then applies dropout to the recurrent network within the world model, effectively performing DR in imagination. Finally, as procedural content generation [87] is a method for generating a non-zero context set, it can be seen as a form

of DR. Works on DR generally leverage the possibility of using a non-uniform context distribution, which possible varies during training.

In both DA and DR approaches, as the training environments are increasingly augmented or randomised, optimisation becomes increasingly difficult, which often makes these methods much less sample-efficient. This is the motivation behind [106, 56, 49] and other works, all of which train an RL policy on a non-randomised or only weakly randomised environment while using other techniques such as supervised or self-supervised learning to train a robust visual encoder.

Finally, note that most of the DA techniques are focused on visual variation in the context set, as that is the easiest variation to produce useful augmentations for. Some DR work focuses on dynamics variation as a way of tackling the sim-to-real problem, where it is assumed that the dynamics will change between training (simulation) and testing (reality).

Environment Generation. While DR and PCG produce context-MDPs within a pre-determined context set, it is normally assumed that all the context-MDPs are solvable. However, in some settings it is unknown how to sample from the set of all *solvable* context-MDPs. For example, consider a simple gridworld maze environment, where the context set consists of all possible block layouts on the grid; some configuration of block placements will result in unsolvable mazes. Further, many block configurations are not useful for training: they may produce trivially easy mazes. To solve these problems we can *learn* to generate new levels (sample new contexts) on which to train the agent, such that we can be sure these context-MDPs are solvable and useful training instances. We want a distribution over context-MDPs which is closer to the testing context set, which likely has only solvable context-MDPs. This is known as environment generation.

Wang et al. [112] introduces Paired Open-Ended Trailblazer (POET), a method for jointly evolving context-MDPs and policies which solve those MDPs, aiming for a policy that can solve a wide variety of context-MDPs. They produce policies that solve unseen level instances reliably and perform better than training from scratch or a naive curriculum. Wang et al. [113] builds on POET, introducing improvements to the open-ended algorithm including a measure of how novel generated context-MDPs are and a generic measure of how much a system exhibits open-ended innovation. These additions improve the diversity and complexity of context-MDPs generated.

Dennis et al. [114] introduces the framework of Unsupervised Environment Design (UED), similar to POET, in which the task is to generate context-MDPs in an unsupervised manner, which are then used to train a policy. The aim is to improve zero-shot generalisation to unseen tasks either within or outside the environment's context-MDP space. Their method PAIRED outperforms standard DR and a method analogous to POET in the grid-world setting described above, as measured by the zero-shot generalisation performance to unseen levels. Jiang et al. [115] extends the formal framework of UED, combining it with Prioritized Level Replay (PLR) [116], and motivates understanding PLR as an environment generation algorithm. This combined method shows improved performance in terms of generalisation to unseen out-of-distribution tasks in both gridworld mazes and 2D car-racing tracks. We summarise PLR in the next section.

Environment generation and DR are both methods for adjusting the context distribution over some context set provided by the environment. Environment generation tends to learn this sampling procedure, and is targeted at environments where the context set is unstructured such that not all context-MDPs are solvable or useful for training, whereas DR work often uses hard-coded heuristics or non-parametric learning approaches to adjust the context distribution, and focuses on settings where the domains are all solvable but possibly have different difficulties or learning potentials. Both can often also be seen as a form of automatic curriculum learning [117], especially if the context distribution is changing during training and adapting to the agent's performance.

This area is very new and we expect there to be more research soon. However, it does require access to an environment where context-MDPs can be generated at a fairly fine level of detail. Environment generation methods can target generalisation over any kind of variation, as long as that kind of variation is present in the output space of the context-MDP generator. Current methods focus on state-space variation, as that is the most intuitive to formulate as a context set within which the generator can produce specific MDPs.

Optimisation Objectives. It is sometimes possible to change our optimisation objective (explicitly or implicitly) to one which better aligns with testing performance.

Changing the distribution over which the training objective is calculated can be seen as implicitly changing the optimisation objective. An initial example in this area applied to improving generalisation is PLR [116], in which the sampling distribution over levels is changed to increase the learning efficiency and generalisation performance of the trained policy. They show both increased training and testing performance on OpenAI Procgen, and the method

effectively forms a rough curriculum over levels, enabling more sample-efficient learning while also ensuring that no context-MDP's performance is too low.

Methods for Robust RL (RRL) are also targeted at the zero-shot generalisation problem, and work by changing the optimisation objective of the RL problem. These methods take a worst-case optimisation approach, maximising the minimum performance over a set of possible environment perturbations (which can be understood as different context-MDPs), and are focused on improving generalisation to unseen dynamics. Chen and Li [25] gives an overview and introduction to the field. Abdullah et al. [118, WR^{2}L] optimises the worst-case performance using a Wasserstein ball around the transition function to define the perturbation set. Mankowitz et al. [119, SRE-MPO] incorporates RLL into MPO [120] and shows improved performance on the RWRL benchmark [81]. Pinto et al. [121, RARL] also (implicitly) optimises a robust RL objective through the use of an adversary which is trained to pick the worst perturbations to the transition function. This can also be seen as an adversarial domain randomisation technique.

5.2 Handling Differences Between Training And Testing

One way of conceptualising why policies do not transfer perfectly at test time is due to the differences between the two environments: the trained model will learn to rely on features during training that change in the testing environment, and generalisation performance then suffers. In this section we review methods that try and explicitly handle the possible differences between the features of the training and testing environments.

Encoding Inductive Biases. If we know how features change between the training and testing context-MDPs, we can use inductive biases to encourage or ensure the model does not rely on features that we expect to change: The policy should only rely on features which will behave similarly in both the training and testing environments. For example if we know colour varies between training and testing, and colour is irrelevant for the task, then we can remove colour from the visual input before processing. Simple changes like this tend not to be worthy of separate papers, but they are still important to consider in real-world problem scenarios.

IDAAC [122] adds an adversarial regularisation term which encourages the internal representations of the policy not to be predictive of time within an episode. This invariance is useful for OpenAI Procgen [35], as timestep is irrelevant for the optimal policy but could be used to overfit to the training set of levels. Higgins et al. [15, DARLA] uses β-VAEs [123] to encode the inductive bias of disentanglement into the representations of the policy, improving zero-shot performance on various visual variations. Vlastelica et al. [124, NAP] incorporates a black-box shortest-path solver to improve generalisation performance in hard navigation problems. Zambaldi et al. [125, 91] incorporate a relational inductive bias into the model architecture which aids in generalising along ordinal axes of variation, including extrapolation performance. Kansky et al. [126, SchemaNetworks] use an object-oriented and entity-focused architecture, combined with structure-learning methods, to learn logical schema which can be used for backwards-chaining-based planning. These schemas generalise zero-shot to novel state spaces as long as the dynamics are consistent. Wang et al. [127, VAI] use unsupervised visual attention and keypoint detection methods to enforce a visual encoder to only encode information relevant to the foreground of the visual image, encoding the inductive bias that the foreground is the only part of the visual input that is important.

Tang et al. [54] introduce AttentionAgent, which uses neuroevolution to optimise an architecture with a hard attention bottleneck, resulting in a network that only receives a fraction of the visual input. The key inductive bias here is that selective attention is beneficial for optimisation and generalisation. Their method generalises zero-shot to unseen backgrounds in CarRacing [5] and VizDoom [128, 55]. Tang and Ha [129, SensoryNeuron] build on AttentionAgent, adding an inductive bias of permutation invariance in the input space. They argue this is useful for improving generalisation, for a similar reason as before: the attention mechanism used for the permutation-invariant architecture encourages the agent to ignore parts of the input space that are irrelevant.

While methods in this area appear dissimilar, they all share the motivation of incorporating specific inductive biases into the RL algorithm. There are several ways of incorporating domain knowledge as an inductive bias. The architecture of the model can be changed to process the variation correctly. If the variation is one to which the policy should be invariant, it can either be removed entirely, or adversarial regularisation can be used to ensure the policy's representations are invariant. More broadly, regularisation or auxiliary losses which encourages the policy to handle this variation correctly can be used. A recommendation to make this body of work more systematic is to use the additional types of structural assumptions discussed in Section 3.6 as a starting point for developing algorithms that leverage those assumptions – many of the works discussed here rely on assumptions that can be classified in those introduced in Section 3.6. For a deeper discussion of inductive biases see [130], and for the original ideas surrounding inductive biases and No Free Lunch theorems see [11].

Regularisation and Simplicity. When we cannot encode a specific inductive bias, standard regularisation can be used. This is generally motivated by a paraphrased Occam's razor: the simplest model will generalise the best. Task-agnostic regularisation encourages simpler models that rely on fewer features or less complex combinations of features. For example, L2 weight decay biases the network towards less complex features, dropout ensures the network cannot rely on specific combinations of features, and the information bottleneck ensures that only the most informative features are used.

Cobbe et al. [105] introduced CoinRun, and evaluated standard supervised learning regularisation techniques on the benchmark. They investigate data augmentation (a modified form of Cutout [131]), dropout, batch norm, L2 weight decay, policy entropy, and a combination of all techniques. All the techniques separately improve performance, and the combination improves performance further, although the gains of the combination is minimal over the individual methods, implying that many of these methods address similar causes of worse generalisation. Early stopping can be seen as a form of regularisation, and Ada et al. [132] shows that considering training iteration as a hyperparameter (effectively a form of early stopping) improves generalisation performance on some benchmarks. Almost all methods report performance at the end of training, as often early stopping would not be beneficial (as can be seen from the training and testing reward curves), but this is likely an attribute of the specific benchmarks being used, and in the future we should keep early stopping in mind.

Several methods utilise information-theoretic regularisation techniques, building on the information bottleneck [133]. Igl et al. [134, IBAC-SNI] and Lu et al. [135, IB-annealing] concurrently introduce methods that rely on the information bottleneck, along with other techniques to improve performance, demonstrating improved performance on OpenAI Procgen, and a variety of random mazes and continuous control tasks, respectively. Eysenbach et al. [136, RPC] extend the motivation of the information bottleneck to the RL setting specifically, learning a dynamics model and policy which jointly minimises the information taken from the environment by using information from previous states to predict future states. This results in policies using much less information than previously, which has benefits for robustness and generalisation, although this method was not compared on standard generalisation benchmarks to other methods. Chen [137, SMIRL] use surprise minimisation to improve the generalisation of the trained policy, although more rigorous benchmarking is needed to know whether this method has a positive effect.

Song et al. [16] show that larger model sizes can lead to *implicit regularisation*: larger models, especially those with residual connections, generalise better, even when trained on the same number of training steps. This is also shown in [105, 35].

Learning Invariances. Sometimes we cannot rely on a specific inductive bias or standard regularisation. This is a very challenging setting for RL (and machine learning in general) to tackle, as there is a fundamental limit to performance due to a kind of "no free lunch" analogy: we cannot expect a policy to generalise to arbitrary contexts. However, several techniques can help, centred around the idea of using multiple training contexts to *learn* the invariances necessary to generalise to the testing contexts. If the factors of variation within the training contexts are the same as the factors of variation between the training and testing contexts, and the values which those factors take in testing are not far from those in training, then you can use that to learn these factors of variation and how to adapt or be invariant to them.

Much work draws on causal inference to learn invariant features from several training contexts. Zhang et al. [39, ICP] assume a block MDP structure [38] and leverage that assumption to learn a state abstraction that is invariant to irrelevant features, which aids in generalisation performance. Zhang et al. [138, DBC] use bisimulation metrics to learn a representation that is invariant to irrelevant visual features, and show that bisimulation metrics are linked to causal inference. Agarwal et al. [139, PSM] suggest limitations of bisimulation metrics, and instead propose a policy similarity metric, where states are similar if the optimal policy has similar behaviour in that and future states. They use this metric, combined with a contrastive learning approach, to learn policies invariant to observation variation.

Several approaches use multiple contexts to learn an invariant representation, which is then assumed to generalise well to testing contexts. Sonar et al. [140, IPO] apply ideas from Invariant Risk Minimization [141] to policy optimisation, learning a representation which enables jointly optimal action prediction across all domains, and show improved performance over PPO on several visual and dynamics variation environments. Bertrán et al. [142, IAPE] introduce the *Instance MDP*, an alternative formalism for the generalisation problem, and then motivate theoretically an approach to learn a collection of policies on subsets of the training domains, such that the aggregate policy is invariant to any individual context-specific features which would not generalise. They show improved performance on the CoinRun benchmark [105] compared to standard regularisation techniques. Ghosh et al. [12, LEEP] also produce an invariant policy across a collection of subsets of the training contexts but motivate this with Bayesian RL. Their approach learns separate policies on each subset and then optimistically aggregates them at test time, as opposed to [142] which uses

the aggregate policy during training for data collection and learns the collection of policies off-policy. While these approaches are similar there has been no direct comparison between them.

Other approaches try to learn behaviourally similar representations with less theoretical motivation. Liu et al. [143, CSSC] use behavioural similarity (similarity of short future action sequences) to find positive and negative pairs for a contrastive learning objective. This auxiliary loss aids in generalisation and sample efficiency in several OpenAI Procgen games. Mazoure et al. [144, CTRL] uses clustering methods and self-supervised learning to define an auxiliary task for representation learning based on behavioural similarity, showing improved performance on OpenAI Procgen. Li et al. [145, DARL] uses adversarial learning to enforce the representations of different domains to be indistinguishable, improving performance on visually diverse testing contexts, even when only trained on simple training contexts. Fan and Li [146, DRIBO] uses contrastive learning combined with an information-theoretic objective to learn representations that only contain task-relevant information while being predictive of the future. They show improved performance on both visually diverse domains in DeepMind Control, and in OpenAI Procgen.

Adapting Online. A final way to handle differences between training and testing contexts is to adapt online to the testing contexts. This adaption has to happen within a single episode and be rapid enough to be useful for improved performance within that episode. While lots of work has been done on meta RL, the vast majority of it assumes access to multiple training episodes in the testing CMDP before evaluation and hence is not applicable in this setting.

Several methods use a hard-coded adaptation function of some kind to perform adaptation within an episode. Hansen et al. [92, PAD] uses a self-supervised objective to update the internal representations of the policy during the test episode. They improve performance over standard baselines on visually varying DeepMind Control environments as well as CRLMaze [147] (a VizDoom [128] 3D navigation environment with visual variation), and on generalisation to novel dynamics on both a real and simulated robot arm. Seo et al. [148, TW-MCL] leverages a multi-headed architecture and multiple-choice learning to learn an ensemble of dynamics models that are selected from during deployment by choosing the one with the highest accuracy. The chosen model plans with model-predictive control, and they show improvements in continuous control tasks. Kumar et al. [149, RMA] tackles the sim-to-real problem by training an agent using domain randomisation in simulation, but instead of producing an invariant policy, they use a domain inference module to condition the policy on the inferred context, which enables it to adapt zero-shot to a wide variety of downstream terrains in both simulation and the real world. Ball et al. [150, AugWM] take a similar idea but work in the offline RL setting, so first train a world model from offline data. They then perform domain randomisation of a specific form in the world model, and then use a hard-coded update rule (enabled by the specific form of domain randomisation) to determine the context to condition the policy, enabling it to adapt zero-shot to downstream tasks. These methods often make use of domain randomisation approaches but aim to have a context-conditioned adaptive policy, rather than a policy invariant to all possible contexts.

Other methods meta-learn an adaptation function during training. RL^2 [151, 152] is a meta RL method where a recurrent network is used, the hidden state of which is not reset at episode boundaries, allowing it to learn and adapt within the recurrent state over multiple episodes. While often compared to methods that require multiple training episodes, this method can often adapt and perform well within a single episode, due to the optimisation approach and architecture. Zintgraf et al. [93] introduce an extension to RL^2 called VariBAD based on bayesian RL, where the recurrent network learns to produce latent representations that are predictive of future rewards and previous transitions. This latent representation is used by the policy.

5.3 RL-Specific Problems And Improvements

The motivations in the previous two sections are mostly equally applicable to supervised learning. However, on top of the problems of generalisation from supervised learning, RL has additional problems which inhibit generalisation performance. In this section we discuss methods targeted at these problems, and also discuss methods that improve generalisation purely through more effective optimisation on the training set in a way that does not overfit (at least empirically).

RL-specific Problems. Optimisation in RL has additional issues on top of supervised learning, such as the non-stationarity of the data distribution, and the need to explore. These issues likely interact with generalisation in a non-trivial way. Igl et al. [153, ITER] shows that the non-stationarity of RL training means that policies learn features that do not generalise well, even if they achieve the same training performance. To address this, they introduce a method to iteratively distil the current policy network into a new policy network with reinitialised weights. This reduces the impact of non-stationarity on the new network, as it is being trained on a more stationary distribution. Other RL-specific optimisation issues likely interact with generalisation either positively or negatively, and this area deserves further attention if we are to go beyond techniques copied and adjusted from supervised learning.

Better Optimisation without Overfitting. Several works improve generalisation through improving the training performance without overfitting. Cobbe et al. [154] introduce Phasic Policy Gradient (PPG), which adjusts the training regime and architecture of PPO such that the policy and value functions use entirely separate networks (rather than just separate heads), which allows the value head to be optimised for longer while not causing the policy to overfit. To recover the benefits of a joint representation, the value network is distilled into an auxiliary value head on the policy network. Raileanu and Fergus [122] build on PPG and introduce Decoupled Advantage Actor Critic (DAAC). They distil an advantage function calculated with GAE into the policy instead of a value function, which further ensures that the policy does not overfit to details that may be predictive of value function but not optimal action selection. They both show improved performance on OpenAI Procgen, demonstrating that value functions can be optimised more strongly than policies. Singh and Zheng [155, Sparse DVE] adjusts the architecture of the value function to allow for a multi-modal output, more closely modelling the true value function given just a visual input. This novel architecture combined with sparsity losses to ensure the value function has the desired properties reduces the variance of value function prediction and improves performance in terms of both return and navigation efficiency in OpenAI Procgen.

Another angle on better optimisation is the use of model-based RL (MBRL). Very little work has applied MBRL to generalisation benchmarks, with [156] being an initial example. Anand et al. [156] apply MuZero Reanalyse [157, 158], a SOTA MBRL method, to OpenAI Procgen [35], showing much-improved performance over SOTA model-free methods at much lower sample complexity. This shows the potential of using MBRL to improve generalisation to varying state and observations. The authors also apply MuZero to the meta-learning tasks in Meta-World [68], although the performance there is not as impressive, showing that generalising to new rewards (as is necessary in Meta-World) is not currently as amenable to MBRL approaches.

While the methods described above do not target generalisation specifically, they improve test-time performance on generalisation benchmarks and so are included here. We hope the field will move towards benchmarks like Procgen being the standard for RL (and not just generalisation), such that in time this work is considered standard RL, rather the generalisation specifically.

5.4 Discussion

Having described existing methods for generalisation in RL, and categorised them in Table 2, we now draw some broader conclusions about the field, as well as discuss possible alternative classifications of methods.

Alternative Classifications. We have presented one possible classification of RL methods, but there are of course others. One alternative is to classify methods based on whether they change the architecture, environment or objective of the standard RL approach. This is useful from a low-level implementation perspective of what the differences are between approaches. This approach is not as useful for future researchers or practitioners who hope to choose a generalisation method for a concrete problem they are facing, as there is not a clear mapping between implementation details and whether a method will be effective for a specific problem. We do apply this classification through the colours in Table 2, to emphasise the current focus on adjusting the loss or objective function in current methods. Another approach would be to classify methods based on which benchmarks they attempt to solve, or what specific problem motivated their design. This goes too far in the other direction, grounding methods in exactly the benchmarks they tackle. While practitioners or researchers could try and see which benchmark is most similar to their problem, they might not understand which differences between benchmarks are most important, and hence choose a method that is not likely to succeed. This classification is also less useful in pointing out areas where there is less research being done. Our approach strikes a balance between these two approaches, describing the problem motivations and solution approaches at a high level which is useful for both practitioners and researchers in choosing methods and investigating future research directions.

Strong Generalisation Requires Inductive Biases. As described in Section 4.3, *What Generalisation Can We Expect?*, there are hard generalisation problems involving combinatorial interpolation or extrapolation. We want to tackle these problems, as they will occur in real-world scenarios when we have limited contexts to train on, or we know the type of variation but cannot create context-MDPs in the deployment context-MDP set (e.g. due to limited simulators). To tackle these problems, we need stronger inductive biases targeted towards specific types of extrapolation, as there is unlikely to be a general-purpose algorithm that can handle all types of extrapolation [11]. When doing research tackling extrapolative generalisation, researchers should be clear that they are introducing an inductive bias to help extrapolate in a specific way, and be rigorous in analysing how this inductive bias helps. This involves also discussing in which situations the bias may hinder performance, for example in a different setting where extrapolation requires something else.

Table 2: A table categorising all methods for tackling the generalisation problem in RL. The columns represent the type of variation (see Table 1) the method is evaluated on, and rows represent the classification in Fig. 4. Colours (and text styles) represent the main adjustment made by the method: *Green italic* methods mainly work by adjusting the training environment, `Red monospace-text` methods mainly work through adjusting the architecture, and Blue normal-text methods mainly work through adjusting the objective or loss function (including adding auxiliary losses). While changing the loss often requires an architectural adjustment, and often architectural changes require adjusted losses, we focus on the original motivation of the method.

Approach	**Evaluation Variation**				
	Observation	**State**	**Dynamics**	**Reward**	**All**
Data Augmentation	*SODA [56]*, `RCAN [106]`, *NetRand [104], InDA,ExDA [107], DrQ [99], SECANT [49], UCB-DrAC [98], PAADA [103], SVAE [108]*	*UCB-DrAC [98], PAADA [103]*			
Domain Randomisation		MD-SAC [109]	P2PDRL [110], DR [95, 97], CAD^{2}RL [96], ADR [2], DDL [111]		PCG [87]
Environment Generation		POET [112], PAIRED [114], E-POET [113]			
Optimisation Objectives		*PLR [116]*	*WR^{2}L [118], RARL [121], (S)(R)(E)-MPO[119]*		
Inductive Biases	`AttentionAgent [54]`, `VAI [127]`, `SensoryNeuron [129]`, *DARLA [15]*	`NAP [124]`, `RelationalRL [125, 91]`, `SchemaNetworks [126]`, *IDAAC [122]*	`RelationalRL [125, 91]`		
Regularisation	`Implicit Regularisation [16]`	*SMIRL [137], Mixreg [101], IBAC-SNI [134]*	*RPC [136], IB-annealing [135]*		*L2,Dropout,etc. [105], EarlyStopping [132]*
Learning Invariances	*DRIBO [146], DBC [138], DARL [145], PSM [139], MISA [39]*	*IAPE [142], DRIBO [146], CTRL [144], CSSC [143], PSM [139], LEEP [12], IPO [140]*	*IPO [140]*		
Fast Adaptation	`PAD [92]`	*VariBad [93], RMA [149], RL^2 [151, 152]*	*VariBad [93]*, `TW-MCL [148]`, *RMA [149], RL^2 [151, 152]*, `AugWM [150]`, `PAD [92]`	*VariBad [93], RL^2 [151, 152]*	
RL-specific Problems	`ITER [153]`	`ITER [153]`			
Better Optimisation	`Sparse DVE [155]`, `PPG [154]`, `DAAC [122]`, `MuZero++ [156]`	`Sparse DVE [155]`, `PPG [154]`, `DAAC [122]`, `MuZero++ [156]`		`MuZero++ [156]`	

Going Beyond Supervised Learning as Inspiration. Methods for improving generalisation from supervised learning have been a source of inspiration for many methods, particularly for visual variation. This is exactly the variation that happens in computer vision, and hence many methods from that field are applicable. However, non-visual forms of generalisation (i.e. dynamics, state and reward), while equally important are less studied. These challenges will be specific to RL and interact with other problems unique to RL such as the exploration-exploitation trade-off and the non-stationarity of the underlying data distribution. We hope to see more work in the area of non-visual generalisation, particularly when other hard RL problems are present.

6 Discussion And Future Work

In this section we highlight further points for discussion, building on those in Section 4.3 and Section 5.4, including directions for future research on new methods, benchmarks, evaluation protocols and understanding.

6.1 Generalisation Beyond Zero-Shot Policy Transfer

This survey focuses on zero-shot policy transfer. We believe this problem setting is a reasonable one that captures many challenges relevant for deploying RL systems. However, there are many important scenarios where zero-shot generalisation is impossible, or the assumptions can be relaxed. We will want to move beyond zero-shot policy transfer if we are to use RL effectively in a wider variety of scenarios.

The most sensible way of relaxing the zero-shot assumption is to move into a continual RL [24, CRL] setting: future RL systems will likely be deployed in scenarios where the environment is constantly changing, such that the system needs to adapt continually to these changes. Making progress on this setting will require benchmarks, and we agree with [24] that there are not enough good benchmarks for CRL. Most benchmarks do not enable testing all the different attributes desired of CRL methods, although [159] is a good first step. New benchmarks for CRL would also be excellent as benchmarks for zero-shot generalisation, especially if these benchmarks introduce new environments. We expect that methods built for domain generalisation in RL might be suitable in CRL, as the continual learning setting can be conceptualised as one in which the domain that tasks are within changes over time. This is in contrast to benchmarks that evaluate generalisation to levels sampled from the same distribution as during training, such as OpenAI Procgen [35]. Hence, we recommend work on building new CRL benchmarks, to make progress on CRL and zero-shot generalisation together.

Coupled with the idea of CRL as a more realistic setting for generalisation in RL, we can take inspiration from how humans generalise and what they transfer when generalising, to go beyond transferring a single policy. While humans may not always be able to achieve good results zero-shot on a new task, if the task is related to previously seen tasks, they can reuse previous knowledge or skills to learn the new task faster. This broader notion of generalisation of objects other than a complete policy (e.g. skills or environment knowledge) will become more relevant when we start to build more powerful RL systems. Hierarchical and multi-task RL are related fields, and methods in these settings often learn subcomponents, modules or skills on source tasks (possibly in an unsupervised manner) which they can then use to increase learning speed and performance when transferred to novel tasks [160, 161]. It is likely the capability to transfer components other than a single policy will be useful for future systems, and it would hence be beneficial to have benchmarks that enable us to test these kinds of capabilities. However, this is a challenging request to meet, as defining what these components are, and deciding on a performance metric for these subcomponent transfer benchmarks, are both difficult conceptual problems. We hope to see work on this area in the future.

A final assumption which is almost untouched in RL is that of a fixed action space between training and testing. Recently, Jain et al. [162] introduced a novel problem setting and framework centered around how to generalise to new actions in RL. They introduce several benchmarks for testing methods which generalise to new actions, and a method based on learning an action representation combined with an action-ranking network which acts as the policy at test time. There is very little work in this area, and we do not cover it in this survey, but it presents an interesting future direction for generalisation research.

6.2 Real World Reinforcement Learning Generalisation

In Dulac-Arnold et al. [81], the authors propose 9 properties that are characteristic of real-world RL.[7] In thinking about the current set of generalisation benchmarks, these properties are relevant in two ways. First, when applying our methods

[7]The 9 properties are: limited samples; sensor, action and reward delays; high-dimensional state and action spaces; reasoning about constraints; partial observability; multi-objective or poorly specified rewards; low action latency; offline training; and explainable policies

to the real world, we will have to tackle these problems. Hence, it would be beneficial if generalisation benchmarks had these properties, such that we can ensure that our generalisation methods work in real-world environments.

Second, several properties are particularly relevant for generalisation and the design of new benchmarks: (1) *high cost of new samples*, (2) *training from offline data* and (3) *underspecified or multi-objective reward functions*. We explain each of these and their relation to generalisation below.

Context Efficiency. Addressing (1), it is likely that the high cost of new samples will also mean a high cost of new environment tasks or contexts. This means we want methods that are *context efficient* as well as sample efficient, and hence we require benchmarks and evaluation protocols in which only a few contexts are allowed during training, rather than several 1000s. It is also worth investigating if there is an optimal trade-off (for different costs per sample and per context) between new training samples and new contexts. This line of work would revolve around different possible evaluation metrics based on how many contexts are needed to reach certain levels of generalisation performance. Further, there may be ways of actively selecting new contexts to maximise the generalisation performance while minimising the number of new contexts used, effectively a form of active learning. Evaluating context efficiency will be more computationally expensive, as models will need to be repeatedly trained on different numbers of training contexts, so work which figures out how to more efficiently evaluate this property is also beneficial.

Sim-to-Real and Offline RL. To tackle (2), two options emerge. The first is relying on good simulations, and then tackling the *sim-to-real* problem, and the second is tackling the offline RL problem directly [163]. These approaches might be more or less relevant or applicable depending on the scenario: for example in many robotics applications a simulator is available, whereas in healthcare settings it is likely learning from offline data is the only approach possible. Sim-to-real is a problem of domain generalisation. If this direction is most relevant, it implies we should focus on building environments that test for good domain generalisation. Existing work on sim-to-real does address this to some extent, but it would be beneficial to have a fully simulated benchmark for testing sim-to-real methods, as that enables faster research progress than requiring real robots. This is a difficult task and is prone to the possibility of overfitting to the simulation of the sim-to-real problem, but it would be useful as an initial environment for testing sim-to-real transfer.

Offline RL is also a problem of generalisation: a key issue here is generalising to state-action pairs unseen in the training data, and most current methods tackle this by conservatively avoiding such pairs [164]. If we had methods to reliably extrapolate to such pairs we could improve offline RL performance.

As well as generalisation improving offline RL, it is likely that future RL deployment scenarios will need to tackle the combination of offline RL and generalisation: training policies offline that then generalise to new contexts unseen in the offline dataset. Current offline RL benchmarks [165, 166] do not measure generalisation in this way, but we believe they should enable us to tackle this combined problem: for example, training on offline data from 200 levels in OpenAI Procgen, and evaluating on the full distribution of levels. If tackling this is infeasible with current methods (as both offline RL and generalisation are hard problems), then a good compromise is to first work on the offline-online setting, where offline RL is used for pretraining, followed by online fine-tuning. Some work has been done in this area [167], but this does not tackle the generalisation problem specifically. Creating benchmarks for evaluating these approaches, where the emphasis is on reducing the length of the online fine-tuning stage and evaluating generalisation after fine-tuning, would move us towards truly offline RL generalisation while still being tractable with current methods.

Reward Function Variation. It is likely future RL systems will be goal- or task-conditioned, as it will be more efficient to train a general system to do several related tasks than to train a different system for each task. Here, as well as generalising to new dynamics and observations, the trained policies will need to generalise to unseen goals or tasks. The policy will need to be able to solve novel problem formulations, and hence have a more generic problem-solving ability. Benchmarks which address this capability are hence necessary for progress towards more general RL systems.

Related to challenges of generalisation surrounding reward functions, for many real-world problems designing good reward functions is very difficult. A promising approach is using inverse RL [168, 169, IRL] to learn a reward function from human demonstrations [170, 171, 172, 173], rather than hand-crafting reward functions for each task. This is often more time-efficient, as demonstrating a task is easier than specifying a reward function for it. There are two generalisation problems here: ensuring the learned reward function generalises to unseen context-MDPs during policy training, and ensuring the trained policy generalises to unseen context-MDPs at test time. The first is an IRL generalisation problem, and the second is the standard problem of generalisation we have considered here. Solving both will be important for ensuring that this approach to training agents is effective.

Work building benchmarks and methods to solve these problems would be valuable future work. Some of these directions could be addressed by combining new evaluation protocols with pre-existing environments to create new benchmarks, rather than requiring entirely new environments to be designed and created.

6.3 Multi-Dimensional Evaluation Of Generalisation

Generalisation performance is usually reported using a single scalar value of test-time performance. However, this does not give us much information with which to compare and choose between methods. While better performance on a benchmark, all else being equal, probably means that a method is more useful, it is usually not clear to what extent the ordering of methods on a benchmark's leaderboard is representative of the hypothetical ordering of those methods on a real-world problem scenario for which we have to choose a method. To alleviate this, performance should be reported on multiple different testing context sets which evaluate different types of generalisation, and radar plots (such as those in [174]) can be used to compare methods. This will be more useful for comparing methods in a more realistic way, as well as for practitioners choosing between methods.

Very few environments have the context set required for this type of evaluation, and even those that do would require additional work to create the specific testing context sets. Hence, we recommend that future benchmarks for generalisation are designed to enable this type of evaluation. This requires building environments with controllable context sets as well as PCG components (Section 4.3, *The Downsides of Procedural Content Generation for Generalisation*), as well as careful thought to create the variety of testing context sets, ensuring they match important types of generalisation. A first way of splitting up testing context sets might be by the type of variation between training and testing, as well as whether interpolation or extrapolation are required to generalise to that context set. There are likely many other ways, which may be domain-specific or general across many domains.

6.4 Tackling Stronger Types Of Variation

Many of the methods for generalisation tackle observation function or state space variation. Both of these (or at least the practical implementations of them used in the corresponding benchmarks) tend to produce environment families in which it is much simpler to verify (at least intuitively) The Principle Of Unchanged Optimality, meaning that tackling the generalisation problem is tractable; generalisation problems with this style of variation tend to be easier to solve. These two types of variation will appear in real-world scenarios, but the other types of variation are equally important and often harder to tackle.

Work on dynamics is mostly focused on two specific settings: sim-to-real transfer, and multi-agent environments. In sim-to-real transfer [27], there is always dynamics variation between the simulator and reality, and much work has focused on how to train policies in this setting. More generally, work on robotics and continuous control tends to address some forms of dynamics variation either in how the robot itself is controlled (e.g. due to degrading parts) or in the environment (e.g. different terrain). In multi-agent environments, if the other agents are considered part of the environment (for example in a single-agent training setting), then varying the other agents varies the dynamics of the environment [21]. These both occur in the real world, but there are inevitably other forms of dynamics variation which are less well studied. Investigating what these other forms of dynamics variation are and whether studying them would be useful is promising future work.

Tackling reward-function variation will be required to train general-purpose policies that can perform a variety of tasks, and generalise to unseen tasks without further training data (as discussed in Section 6.2, *Reward Function Variation*). This variation is more difficult to tackle, and it is often difficult or impossible to verify The Principle Of Unchanged Optimality [88]. However, we must do work on research to tackle these problems, as otherwise RL approaches will be limited to less-ambitious problems or less-general applications. Further work building more benchmarks that enable testing for reward function variation, especially beyond simple styles of goal specification such as a target state, would be beneficial. Special attention needs to be paid to The Principle Of Unchanged Optimality [88] while building these benchmarks: current work tends to handle this by conditioning the policy on a goal or reward specification. Research on what approaches to goal specification are both tractable for policy optimisation and useful for real-world scenarios would be beneficial, as there is likely a trade-off between these two desirable attributes.

6.5 Understanding Generalisation In Reinforcement Learning

While beyond the scope of this survey, several works try to understand the problems underlying generalisation in RL. Works in this area include [16], which describes the notion of observational overfitting as one cause of the generalisation gap in RL [175], which analyses the relationship between gradient interference and generalisation in supervised and RL showing that temporal difference methods tend to have lower-interference training, which correlates with worse

generalisation; and [153] which studies transient non-stationarity in RL and shows that it negatively impacts RL generalisation.

This research is barely scratching the surface of understanding why generalisation in RL in particular is a challenge, and there is much future work to be done. This will enable us to build better methods, and understand any theoretical limits to the diversity of tasks that an RL agent can solve given a limited number of training contexts. The precise empirical experimentation required for this kind of research is exactly that which is enabled by having tight control over the factors of variation in the environments being used, which reinforces the conclusion made in Section 4.3, *The Downsides of Procedural Content Generation for Generalisation* that purely PCG environments are unsuited for a study of generalisation in RL.

6.6 Future Work On Methods For Generalisation

In this subsection we summarise directions for future work on methods for generalisation, informed by Section 5.

As described in Section 6.5, there are many RL-specific factors that interact with generalisation performance, often likely in a negative way. Examples of these factors include the non-stationarity of the data distribution used for training; bootstrapping and TD learning in general; and the need for exploration. Work to understand these factors and then build methods to tackle them as discussed in Section 5.3, *RL-specific Problems* is a fruitful direction for future work.

We often have a context space that is unstructured or contains many unsolvable context-MDPs. Methods that enable more effective sampling from these context spaces can alleviate this. Several methods were covered in Section 5.1, *Environment Generation* but more work in this area, tackling more challenging and realistic environments with different types of variation, would be beneficial.

While much work has been done on meta RL, most focuses on few-shot adaptation. However, work in this area could be adapted to tackle zero-shot policy transfer settings, if the environment has long episodes which require online adaptation. Enabling the policy to learn and adapt online, and learning this adaptation, would likely improve performance. These approaches would also be more suited to tackling stronger forms of variation (Section 6.4), as online adaptation may be necessary in these scenarios. Initial work in this area is described in Section 5.2, *Adapting Online*, but much more research should be done.

There are several under-explored approaches that cut across the categorisation in this work. As shown in Table 2, most methods focus on changing the loss function or algorithmic approach. Architectural changes informed by inductive biases are less well studied, with notable examples being [154, 155, 54, 124, 91]. More work can be done on investigating different architectures, either taking inspiration from supervised learning or creating RL-specific architectures. These architectures could encode inductive biases in ways that are difficult to encode through the use of auxiliary losses or regularisation. A second under-explored area is model-based reinforcement learning (MBRL) for generalisation. Most methods surveyed here are model-free, with notable exceptions being [150, 126, 148, 156]. Learning a world model and combining it with planning methods can enable stronger forms of generalisation, especially to novel reward functions (if the reward function is available during planning). As long as the model generalises well, it could also enable generalisation to novel state and observation functions. World models which can adapt to changing dynamics will be more challenging, but Seo et al. [148] give an initial example. Anand et al. [156] is the first example investigating how well standard MBRL approaches generalise, and we look forward to seeing more work in this area.

7 Conclusion

The study of generalisation in RL is still new but is of vital importance if we want to develop applicable and usable RL solutions to real-world problems. In this survey we have aimed to clarify the terminology and formalism concerning generalisation in RL, bringing together disparate threads of research together in a unified framework. We presented a categorisation of benchmarks for generalisation, splitting the taxonomy into environments and evaluation protocols, and we categorised existing methods for tackling the wide variety of generalisation problems.

Here we summarise the key takeaways of this survey (with pointers to the more in-depth discussion of the takeaway). The first two takeaways are concerned with the problem setting as a whole. The next four are focused on evaluating generalisation through benchmarks and metrics, and future work in these areas. The last two are focused on methods for tackling the generalisation problem.

- Zero-shot policy transfer is useful to study, even if in specific settings we may be able to relax the zero-shot assumption, as it provides base algorithms upon which domain-specific solutions can be built (Section 3.7, *Motivating Zero-Shot Policy Transfer*).

- However, more work should be done to look beyond zero-shot policy transfer, particularly at continual reinforcement learning, as a way to get around the restriction of the principle of unchanged optimality (Section 6.1).
- Purely black-box PCG environments are not useful for testing specific forms of generalisation and are most useful for ensuring robust improvements in standard RL algorithms. Combining PCG and controllable factors of variation is our recommended way to design new environments, having the best trade-off between high variety and the possibility of scientific experimentation (Section 4.3, *The Downsides of Procedural Content Generation for Generalisation*). This also enables a more multidimensional approach to evaluating generalisation performance (Section 6.3), and specific experimentation aimed at improving our understanding of generalisation in RL (Section 6.5).
- For real-world scenarios, we have to consider both sample-efficiency and context efficiency. Evaluating the performance of methods on different sizes of training context sets is a useful evaluation metric which gives us more information to choose between different methods (Section 6.2, *Context Efficiency*).
- Work on generalisation problems associated with offline RL is under-explored and would ensure that offline RL approaches are able to generalise effectively (Section 6.2, *Sim-to-Real and Offline RL*).
- While observation-function and state-space variation are commonly studied, dynamics variation is only tackled in limited settings and reward-function variation is very under-studied. These stronger forms of variation are still likely to appear in real-world scenarios, and hence should be the focus of future research (Section 6.4).
- For stronger forms of generalisation, stronger inductive biases are necessary, and research should be up-front about what the inductive bias they are introducing is, how it tackles the specific benchmark they are tackling, and how general they expect that inductive bias to be (Section 5.4, *Strong Generalisation Requires Inductive Biases*).
- There is much underexplored future work in developing new methods for improved generalisation, such as model-based RL, new architectures, fast online adaptation, solving RL-specific generalisation problems, and environment generation (Section 6.6).

We hope that this survey will help clarify and unify work tackling the problem of generalisation in RL, spurring further research in this area, and survey as a touch-point and reference for researchers and practitioners both inside and outside the field.

Acknowledgements

We thank (in alphabetical order) Flo Dorner, Jack Parker-Holder, Katja Hofmann, Laura Ruis, Maximilian Igl, Mikayal Samyvelyan, Minqi Jiang, Nicklas Hansen, Roberta Raileanu, Yingchen Xi and Zhengyao Jiang for discussion and comments on drafts of this work.

Author Contributions

Robert Kirk led the work, developed the formalism, benchmarks categorisation, methods categorisation, and discussion and future work, wrote the full manuscript of the survey, and wrote successive drafts with comments and feedback from the other authors. **Amy Zhang** wrote parts of Sections 3.1, 3.6, 3.7 and 4.3 and Appendix A, as well as providing improvements on the entire work through discussion and editing. **Tim Rocktäschel** and **Edward Grefenstette** advised Robert Kirk, providing discussion and feedback in developing the ideas behind the survey, and provided feedback and comments on the manuscript.

References

[1] Angelos Filos, Panagiotis Tigkas, Rowan McAllister, Nicholas Rhinehart, Sergey Levine, and Yarin Gal. Can autonomous vehicles identify, recover from, and adapt to distribution shifts? 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*, pages 3145–3153. PMLR, 2020. URL `http://proceedings.mlr.press/v119/filos20a.html`.

[2] OpenAI, Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, Jonas Schneider, Nikolas Tezak, Jerry Tworek, Peter Welinder, Lilian Weng, Qiming Yuan, Wojciech Zaremba, and Lei Zhang. Solving Rubik's Cube with a Robot Hand. *arXiv:1910.07113 [cs, stat]*, 2019. URL `http://arxiv.org/abs/1910.07113`.

[3] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The Arcade Learning Environment: An Evaluation Platform for General Agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013. ISSN 1076-9757. doi:10.1613/jair.3912. URL `https://www.jair.org/index.php/jair/article/view/10819`.

[4] E. Todorov, T. Erez, and Y. Tassa. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 7. ISBN 2153-0866. doi:10.1109/IROS.2012.6386109.

[5] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym. *arXiv:1606.01540 [cs]*, 2016. URL http://arxiv.org/abs/1606.01540.

[6] Shimon Whiteson, Brian Tanner, Matthew E. Taylor, and Peter Stone. Protecting against evaluation overfitting in empirical reinforcement learning. In *2011 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 120–127, 2011. doi:10.1109/ADPRL.2011.5967363.

[7] Amy Zhang, Nicolas Ballas, and Joelle Pineau. A Dissection of Overfitting and Generalization in Continuous Reinforcement Learning. *arXiv:1806.07937 [cs, stat]*, 2018. URL http://arxiv.org/abs/1806.07937.

[8] Chiyuan Zhang, Oriol Vinyals, Remi Munos, and Samy Bengio. A Study on Overfitting in Deep Reinforcement Learning. *arXiv:1804.06893 [cs, stat]*, 2018. URL http://arxiv.org/abs/1804.06893.

[9] Jesse Farebrother, Marlos C. Machado, and Michael Bowling. Generalization and Regularization in DQN. *arXiv:1810.00123 [cs, stat]*, 2020. URL http://arxiv.org/abs/1810.00123.

[10] Shani Gamrian and Yoav Goldberg. Transfer learning for related reinforcement learning tasks via image-to-image translation. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *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*, pages 2063–2072. PMLR, 2019. URL http://proceedings.mlr.press/v97/gamrian19a.html.

[11] D.H. Wolpert and W.G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997. ISSN 1941-0026. doi:10.1109/4235.585893.

[12] Dibya Ghosh, Jad Rahme, Aviral Kumar, Amy Zhang, Ryan P. Adams, and Sergey Levine. Why Generalization in RL is Difficult: Epistemic POMDPs and Implicit Partial Observability. *arXiv:2107.06277 [cs, stat]*, 2021. URL http://arxiv.org/abs/2107.06277.

[13] Assaf Hallak, Dotan Di Castro, and Shie Mannor. Contextual Markov Decision Processes. *arXiv:1502.02259 [cs, stat]*, 2015. URL http://arxiv.org/abs/1502.02259.

[14] James Harrison, Animesh Garg, Boris Ivanovic, Yuke Zhu, Silvio Savarese, Li Fei-Fei, and Marco Pavone. ADAPT: Zero-Shot Adaptive Policy Transfer for Stochastic Dynamical Systems. *arXiv:1707.04674 [cs]*, 2017. URL http://arxiv.org/abs/1707.04674.

[15] Irina Higgins, Arka Pal, Andrei A. Rusu, Loïc Matthey, Christopher Burgess, Alexander Pritzel, Matthew Botvinick, Charles Blundell, and Alexander Lerchner. DARLA: improving zero-shot transfer in reinforcement learning. In Doina Precup and Yee Whye Teh, editors, *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*, pages 1480–1490. PMLR, 2017. URL http://proceedings.mlr.press/v70/higgins17a.html.

[16] Xingyou Song, Yiding Jiang, Stephen Tu, Yilun Du, and Behnam Neyshabur. Observational overfitting in reinforcement learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=HJli2hNKDH.

[17] OpenAI, Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique P. d O. Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. Dota 2 with Large Scale Deep Reinforcement Learning. *arXiv:1912.06680 [cs, stat]*, 2019. URL http://arxiv.org/abs/1912.06680.

[18] Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander S. Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom L. Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019. ISSN 1476-4687. doi:10.1038/s41586-019-1724-z. URL https://www.nature.com/articles/s41586-019-1724-z.

[19] Hengyuan Hu, Adam Lerer, Brandon Cui, David Wu, Luis Pineda, Noam Brown, and Jakob Foerster. Off-Belief Learning. *arXiv:2103.04000 [cs]*, 2021. URL http://arxiv.org/abs/2103.04000.

[20] Hengyuan Hu, Adam Lerer, Alex Peysakhovich, and Jakob N. Foerster. "other-play" for zero-shot coordination. 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*, pages 4399–4410. PMLR, 2020. URL http://proceedings.mlr.press/v119/hu20a.html.

[21] Open Ended Learning Team, Adam Stooke, Anuj Mahajan, Catarina Barros, Charlie Deck, Jakob Bauer, Jakub Sygnowski, Maja Trebacz, Max Jaderberg, Michael Mathieu, Nat McAleese, Nathalie Bradley-Schmieg, Nathaniel Wong, Nicolas Porcel, Roberta Raileanu, Steph Hughes-Fitt, Valentin Dalibard, and Wojciech Marian Czarnecki. Open-Ended Learning Leads to Generally Capable Agents. *arXiv:2107.12808 [cs]*, 2021. URL http://arxiv.org/abs/2107.12808.

[22] Simon S. Du, Sham M. Kakade, Ruosong Wang, and Lin F. Yang. Is a good representation sufficient for sample efficient reinforcement learning? In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=r1genAVKPB.

[23] Dhruv Malik, Yuanzhi Li, and Pradeep Ravikumar. When Is Generalizable Reinforcement Learning Tractable? *arXiv:2101.00300 [cs, stat]*, 2021. URL http://arxiv.org/abs/2101.00300.

[24] Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards Continual Reinforcement Learning: A Review and Perspectives. *arXiv:2012.13490 [cs]*, 2020. URL http://arxiv.org/abs/2012.13490.

[25] Shiyu Chen and Yanjie Li. An Overview of Robust Reinforcement Learning. In *2020 IEEE International Conference on Networking, Sensing and Control (ICNSC)*, pages 1–6, 2020. doi:10.1109/ICNSC48988.2020.9238129.

[26] Jun Morimoto and Kenji Doya. Robust reinforcement learning. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*, pages 1061–1067. MIT Press, 2000. URL https://proceedings.neurips.cc/paper/2000/hash/e8dfff4676a47048d6f0c4ef899593dd-Abstract.html.

[27] Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: A Survey. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 737–744, 2020. doi:10.1109/SSCI47803.2020.9308468.

[28] Matthias Müller-Brockhausen, Mike Preuss, and Aske Plaat. Procedural Content Generation: Better Benchmarks for Transfer Reinforcement Learning. *arXiv:2105.14780 [cs]*, 2021. URL http://arxiv.org/abs/2105.14780.

[29] Zhuangdi Zhu, Kaixiang Lin, and Jiayu Zhou. Transfer Learning in Deep Reinforcement Learning: A Survey. *arXiv:2009.07888 [cs, stat]*, 2021. URL http://arxiv.org/abs/2009.07888.

[30] Nelson Vithayathil Varghese and Qusay H. Mahmoud. A survey of multi-task deep reinforcement learning. *Electronics*, 9(9), 2020. ISSN 2079-9292. doi:10.3390/electronics9091363. URL https://www.mdpi.com/2079-9292/9/9/1363.

[31] Susan Amin, Maziar Gomrokchi, Harsh Satija, Herke van Hoof, and Doina Precup. A Survey of Exploration Methods in Reinforcement Learning. *arXiv:2109.00157 [cs]*, 2021. URL http://arxiv.org/abs/2109.00157.

[32] Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E. Taylor, and Peter Stone. Curriculum Learning for Reinforcement Learning Domains: A Framework and Survey. *arXiv:2003.04960 [cs, stat]*, 2020. URL http://arxiv.org/abs/2003.04960.

[33] Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. Compositionality decomposed: How do neural networks generalise? *arXiv:1908.08351 [cs, stat]*, 2020. URL http://arxiv.org/abs/1908.08351.

[34] Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and Olivier Bousquet. Measuring compositional generalization: A comprehensive method on realistic data. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=SygcCnNKwr.

[35] Karl Cobbe, Christopher Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark 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*, pages 2048–2056. PMLR, 2020. URL http://proceedings.mlr.press/v119/cobbe20a.html.

[36] Austin Stone, Oscar Ramirez, Kurt Konolige, and Rico Jonschkowski. The Distracting Control Suite – A Challenging Benchmark for Reinforcement Learning from Pixels. *arXiv:2101.02722 [cs]*, 2021. URL http://arxiv.org/abs/2101.02722.

[37] Finale Doshi-Velez and George Dimitri Konidaris. Hidden parameter markov decision processes: A semi-parametric regression approach for discovering latent task parametrizations. In Subbarao Kambhampati, editor, *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 1432–1440. IJCAI/AAAI Press, 2016. URL http://www.ijcai.org/Abstract/16/206.

[38] Simon S. Du, Akshay Krishnamurthy, Nan Jiang, Alekh Agarwal, Miroslav Dudík, and John Langford. Provably efficient RL with rich observations via latent state decoding. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *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*, pages 1665–1674. PMLR, 2019. URL http://proceedings.mlr.press/v97/du19b.html.

[39] Amy Zhang, Clare Lyle, Shagun Sodhani, Angelos Filos, Marta Kwiatkowska, Joelle Pineau, Yarin Gal, and Doina Precup. Invariant causal prediction for block mdps. 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*, pages 11214–11224. PMLR, 2020. URL http://proceedings.mlr.press/v119/zhang20t.html.

[40] Craig Boutilier, Richard Dearden, and Moisés Goldszmidt. Stochastic dynamic programming with factored representations. *Artificial Intelligence*, 121(1):49–107, 2000. ISSN 0004-3702. doi:10.1016/S0004-3702(00)00033-3. URL https://www.sciencedirect.com/science/article/pii/S0004370200000333.

[41] Alexander Strehl, Carlos Diuk, and Michael Littman. Efficient Structure Learning in Factored-State MDPs. pages 645–650, 2007.

[42] Botao Hao, Tor Lattimore, Csaba Szepesvári, and Mengdi Wang. Online sparse reinforcement learning. In Arindam Banerjee and Kenji Fukumizu, editors, *The 24th International Conference on Artificial Intelligence and Statistics, AISTATS 2021, April 13-15, 2021, Virtual Event*, volume 130 of *Proceedings of Machine Learning Research*, pages 316–324. PMLR, 2021. URL http://proceedings.mlr.press/v130/hao21a.html.

[43] Biwei Huang, Fan Feng, Chaochao Lu, Sara Magliacane, and Kun Zhang. AdaRL: What, Where, and How to Adapt in Transfer Reinforcement Learning. *arXiv:2107.02729 [cs, stat]*, 2021. URL http://arxiv.org/abs/2107.02729.

[44] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 3207–3214. AAAI Press, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16669.

[45] Jane X. Wang, Michael King, Nicolas Porcel, Zeb Kurth-Nelson, Tina Zhu, Charlie Deck, Peter Choy, Mary Cassin, Malcolm Reynolds, Francis Song, Gavin Buttimore, David P. Reichert, Neil Rabinowitz, Loic Matthey, Demis Hassabis, Alexander Lerchner, and Matthew Botvinick. Alchemy: A structured task distribution for meta-reinforcement learning. *arXiv:2102.02926 [cs]*, 2021. URL http://arxiv.org/abs/2102.02926.

[46] Marlos C. Machado, Marc G. Bellemare, Erik Talvitie, Joel Veness, Matthew Hausknecht, and Michael Bowling. Revisiting the Arcade Learning Environment: Evaluation Protocols and Open Problems for General Agents. *arXiv:1709.06009 [cs]*, 2017. URL http://arxiv.org/abs/1709.06009.

[47] Maxime Chevalier-Boisvert, Dzmitry Bahdanau, Salem Lahlou, Lucas Willems, Chitwan Saharia, Thien Huu Nguyen, and Yoshua Bengio. 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. URL https://openreview.net/forum?id=rJeXCo0cYX.

[48] Carolin Benjamins, Theresa Eimer, Frederik Schubert, André Biedenkapp, Bodo Rosenhahn, Frank Hutter, and Marius Lindauer. CARL: A Benchmark for Contextual and Adaptive Reinforcement Learning. 2021. URL https://openreview.net/forum?id=6D45bYP5MRP.

[49] Linxi Fan, Guanzhi Wang, De-An Huang, Zhiding Yu, Li Fei-Fei, Yuke Zhu, and Anima Anandkumar. SECANT: Self-Expert Cloning for Zero-Shot Generalization of Visual Policies. *arXiv:2106.09678 [cs]*, 2021. URL http://arxiv.org/abs/2106.09678.

[50] Yuke Zhu, Josiah Wong, Ajay Mandlekar, and Roberto Martín-Martín. Robosuite: A Modular Simulation Framework and Benchmark for Robot Learning. *arXiv:2009.12293 [cs]*, 2020. URL http://arxiv.org/abs/2009.12293.

[51] Ossama Ahmed, Frederik Träuble, Anirudh Goyal, Alexander Neitz, Yoshua Bengio, Bernhard Schölkopf, Manuel Wüthrich, and Stefan Bauer. CausalWorld: A Robotic Manipulation Benchmark for Causal Structure and Transfer Learning. *arXiv:2010.04296 [cs, stat]*, 2020. URL http://arxiv.org/abs/2010.04296.

[52] Danijar Hafner. Benchmarking the Spectrum of Agent Capabilities. *arXiv:2109.06780 [cs]*, 2021. URL http://arxiv.org/abs/2109.06780.

[53] Valerie Chen, Abhinav Gupta, and Kenneth Marino. Ask Your Humans: Using Human Instructions to Improve Generalization in Reinforcement Learning. *arXiv:2011.00517 [cs]*, 2021. URL http://arxiv.org/abs/2011.00517.

[54] Yujin Tang, Duong Nguyen, and David Ha. Neuroevolution of Self-Interpretable Agents. *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, pages 414–424, 2020. doi:10.1145/3377930.3389847. URL http://arxiv.org/abs/2003.08165.

[55] OpenAI Gym: The DoomTakeCover-v0 environment. URL https://gym.openai.com/envs/DoomTakeCover-v0.

[56] Nicklas Hansen and Xiaolong Wang. Generalization in Reinforcement Learning by Soft Data Augmentation. *arXiv:2011.13389 [cs]*, 2021. URL http://arxiv.org/abs/2011.13389.

[57] Jake Grigsby and Yanjun Qi. Measuring Visual Generalization in Continuous Control from Pixels. *arXiv:2010.06740 [cs]*, 2020. URL http://arxiv.org/abs/2010.06740.

[58] Charles Packer, Katelyn Gao, Jernej Kos, Philipp Krähenbühl, Vladlen Koltun, and Dawn Song. Assessing Generalization in Deep Reinforcement Learning. *arXiv:1810.12282 [cs, stat]*, 2019. URL http://arxiv.org/abs/1810.12282.

[59] Diego Perez-Liebana, Jialin Liu, Ahmed Khalifa, Raluca D. Gaina, Julian Togelius, and Simon M. Lucas. General Video Game AI: A Multi-Track Framework for Evaluating Agents, Games and Content Generation Algorithms. *arXiv:1802.10363 [cs]*, 2019. URL http://arxiv.org/abs/1802.10363.

[60] Bokui Shen, Fei Xia, Chengshu Li, Roberto Martín-Martín, Linxi Fan, Guanzhi Wang, Claudia Pérez-D'Arpino, Shyamal Buch, Sanjana Srivastava, Lyne P. Tchapmi, Micael E. Tchapmi, Kent Vainio, Josiah Wong, Li Fei-Fei, and Silvio Savarese. iGibson 1.0: A Simulation Environment for Interactive Tasks in Large Realistic Scenes. *arXiv:2012.02924 [cs]*, 2021. URL http://arxiv.org/abs/2012.02924.

[61] Matthew J. Hausknecht, Prithviraj Ammanabrolu, Marc-Alexandre Côté, and Xingdi Yuan. Interactive fiction games: A colossal adventure. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7903–7910. AAAI Press, 2020. URL https://aaai.org/ojs/index.php/AAAI/article/view/6297.

[62] Remi Tachet, Philip Bachman, and Harm van Seijen. Learning Invariances for Policy Generalization. *arXiv:1809.02591 [cs, stat]*, 2020. URL http://arxiv.org/abs/1809.02591.

[63] Eliot Xing, Abhinav Gupta, Sam Powers, and Victoria Dean. Evaluating Generalization of Policy Learning Under Domain Shifts. 2021. URL https://openreview.net/forum?id=11yRzZJsMwA.

[64] Matthew Johnson, Katja Hofmann, Tim Hutton, and David Bignell. The malmo platform for artificial intelligence experimentation. In Subbarao Kambhampati, editor, *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 4246–4247. IJCAI/AAAI Press, 2016. URL http://www.ijcai.org/Abstract/16/643.

[65] Dimitrios I. Koutras, Athanasios Ch Kapoutsis, Angelos A. Amanatiadis, and Elias B. Kosmatopoulos. MarsExplorer: Exploration of Unknown Terrains via Deep Reinforcement Learning and Procedurally Generated Environments. *arXiv:2107.09996 [cs]*, 2021. URL http://arxiv.org/abs/2107.09996.

[66] Luke Harries, Sebastian Lee, Jaroslaw Rzepecki, Katja Hofmann, and Sam Devlin. MazeExplorer: A Customisable 3D Benchmark for Assessing Generalisation in Reinforcement Learning. In *2019 IEEE Conference on Games (CoG)*, pages 1–4, 2019. doi:10.1109/CIG.2019.8848048.

[67] Raghu Rajan, Jessica Lizeth Borja Diaz, Suresh Guttikonda, Fabio Ferreira, André Biedenkapp, Jan Ole von Hartz, and Frank Hutter. MDP Playground: A Design and Debug Testbed for Reinforcement Learning. *arXiv:1909.07750 [cs, stat]*, 2021. URL http://arxiv.org/abs/1909.07750.

[68] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. *arXiv preprint arXiv:1910.10897*, 2019.

[69] Quanyi Li, Zhenghao Peng, Zhenghai Xue, Qihang Zhang, and Bolei Zhou. MetaDrive: Composing Diverse Driving Scenarios for Generalizable Reinforcement Learning. 2021. URL https://openreview.net/forum?id=9YyNOthj5Ex.

[70] Maxime Chevalier-Boisvert. Minimalistic Gridworld Environment (MiniGrid), 2021. URL https://github.com/maximecb/gym-minigrid.

[71] Mikayel Samvelyan, Robert Kirk, Vitaly Kurin, Jack Parker-Holder, Minqi Jiang, Eric Hambro, Fabio Petroni, Heinrich Kuttler, Edward Grefenstette, and Tim Rocktäschel. MiniHack the Planet: A Sandbox for Open-Ended Reinforcement Learning Research. In *Thirty-Fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021. URL https://openreview.net/forum?id=skFwlyefkWJ.

[72] Amy Zhang, Yuxin Wu, and Joelle Pineau. Natural Environment Benchmarks for Reinforcement Learning. *arXiv:1811.06032 [cs, stat]*, 2018. URL http://arxiv.org/abs/1811.06032.

[73] Heinrich Küttler, Nantas Nardelli, Alexander H. Miller, Roberta Raileanu, Marco Selvatici, Edward Grefenstette, and Tim Rocktäschel. The nethack learning environment. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/569ff987c643b4bedf504efda8f786c2-Abstract.html.

[74] Chenyang Zhao, Olivier Sigaud, Freek Stulp, and Timothy M. Hospedales. Investigating Generalisation in Continuous Deep Reinforcement Learning. *arXiv:1902.07015 [cs, stat]*, 2019. URL http://arxiv.org/abs/1902.07015.

[75] Shivam Goel, Gyan Tatiya, Matthias Scheutz, and Jivko Sinapov. NovelGridworlds: A benchmark environment for detecting and adapting to novelties in open worlds. 2021.

[76] Arthur Juliani, Ahmed Khalifa, Vincent-Pierre Berges, Jonathan Harper, Ervin Teng, Hunter Henry, Adam Crespi, Julian Togelius, and Danny Lange. Obstacle tower: A generalization challenge in vision, control, and planning. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 2684–2691. ijcai.org, 2019. doi:10.24963/ijcai.2019/373. URL https://doi.org/10.24963/ijcai.2019/373.

[77] Nan Rosemary Ke, Ishita Dasgupta, Silvia Chiappa, Anirudh Goyal, Theophane Weber, Jovana Mitrovic, Felix Hill, Stephanie C. Y. Chan, Michael Curtis Mozer, Danilo Jimenez Rezende, and Pushmeet Kohli. Parametric Generalization for Benchmarking Reinforcement Learning Algorithms. 2021. URL https://openreview.net/forum?id=zx36-ng_9U_.

[78] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J. Davison. RLBench: The Robot Learning Benchmark & Learning Environment. *arXiv:1909.12271 [cs]*, 2019. URL http://arxiv.org/abs/1909.12271.

[79] Yuji Kanagawa and Tomoyuki Kaneko. Rogue-Gym: A New Challenge for Generalization in Reinforcement Learning. *arXiv:1904.08129 [cs, stat]*, 2019. URL http://arxiv.org/abs/1904.08129.

[80] Victor Zhong, Tim Rocktäschel, and Edward Grefenstette. RTFM: Generalising to Novel Environment Dynamics via Reading. *arXiv:1910.08210 [cs]*, 2021. URL http://arxiv.org/abs/1910.08210.

[81] Gabriel Dulac-Arnold, Nir Levine, Daniel J. Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester. An empirical investigation of the challenges of real-world reinforcement learning. *arXiv:2003.11881 [cs]*, 2021. URL http://arxiv.org/abs/2003.11881.

[82] Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Ruo Yu Tao, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. TextWorld: A Learning Environment for Text-based Games. *arXiv:1806.11532 [cs, stat]*, 2019. URL http://arxiv.org/abs/1806.11532.

[83] Emma Tosch, Kaleigh Clary, John Foley, and David Jensen. Toybox: A Suite of Environments for Experimental Evaluation of Deep Reinforcement Learning. *arXiv:1905.02825 [cs, stat]*, 2019. URL http://arxiv.org/abs/1905.02825.

[84] Sam Wenke, Dan Saunders, Mike Qiu, and Jim Fleming. Reasoning and Generalization in RL: A Tool Use Perspective. *arXiv:1907.02050 [cs]*, 2019. URL http://arxiv.org/abs/1907.02050.

[85] Minqi Jiang, Jelena Luketina, Nantas Nardelli, Pasquale Minervini, Philip H.S. Torr, Shimon Whiteson, and Tim Rocktäschel. WordCraft: An environment for benchmarking commonsense agents. In *Workshop on Language in Reinforcement Learning (LaRel)*, 2020. URL https://github.com/minqi/wordcraft.

[86] Cheng Xue, Vimukthini Pinto, Chathura Gamage, Ekaterina Nikonova, Peng Zhang, and Jochen Renz. Phy-Q: A Benchmark for Physical Reasoning. *arXiv:2108.13696 [cs]*, 2021. URL http://arxiv.org/abs/2108.13696.

[87] Sebastian Risi and Julian Togelius. Increasing generality in machine learning through procedural content generation. *Nat Mach Intell*, 2(8):428–436, 2020. ISSN 2522-5839. doi:10.1038/s42256-020-0208-z. URL https://www.nature.com/articles/s42256-020-0208-z.

[88] Alex Irpan and Xingyou Song. The Principle of Unchanged Optimality in Reinforcement Learning Generalization. *arXiv:1906.00336 [cs, stat]*, 2019. URL http://arxiv.org/abs/1906.00336.

[89] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. DeepMind Control Suite. *arXiv:1801.00690 [cs]*, 2018. URL http://arxiv.org/abs/1801.00690.

[90] Jelena Luketina, Nantas Nardelli, Gregory Farquhar, Jakob N. Foerster, Jacob Andreas, Edward Grefenstette, Shimon Whiteson, and Tim Rocktäschel. A survey of reinforcement learning informed by natural language. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 6309–6317. ijcai.org, 2019. doi:10.24963/ijcai.2019/880. URL https://doi.org/10.24963/ijcai.2019/880.

[91] Vinícius Flores Zambaldi, David Raposo, Adam Santoro, Victor Bapst, Yujia Li, Igor Babuschkin, Karl Tuyls, David P. Reichert, Timothy P. Lillicrap, Edward Lockhart, Murray Shanahan, Victoria Langston, Razvan Pascanu, Matthew Botvinick, Oriol Vinyals, and Peter W. Battaglia. Deep reinforcement learning with relational inductive biases. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=HkxaFoC9KQ.

[92] Nicklas Hansen, Rishabh Jangir, Yu Sun, Guillem Alenyà, Pieter Abbeel, Alexei A. Efros, Lerrel Pinto, and Xiaolong Wang. Self-Supervised Policy Adaptation during Deployment. *arXiv:2007.04309 [cs, stat]*, 2021. URL http://arxiv.org/abs/2007.04309.

[93] Luisa M. Zintgraf, Kyriacos Shiarlis, Maximilian Igl, Sebastian Schulze, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: A very good method for bayes-adaptive deep RL via meta-learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=Hkl9JlBYvr.

[94] Connor Shorten and Taghi M. Khoshgoftaar. A survey on Image Data Augmentation for Deep Learning. *J Big Data*, 6(1):60, 2019. ISSN 2196-1115. doi:10.1186/s40537-019-0197-0. URL https://doi.org/10.1186/s40537-019-0197-0.

[95] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World. *arXiv:1703.06907 [cs]*, 2017. URL http://arxiv.org/abs/1703.06907.

[96] Fereshteh Sadeghi and Sergey Levine. CAD2RL: Real Single-Image Flight without a Single Real Image. *arXiv:1611.04201 [cs]*, 2017. URL http://arxiv.org/abs/1611.04201.

[97] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-Real Transfer of Robotic Control with Dynamics Randomization. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3803–3810, 2018. doi:10.1109/ICRA.2018.8460528. URL http://arxiv.org/abs/1710.06537.

[98] Roberta Raileanu, Max Goldstein, Denis Yarats, Ilya Kostrikov, and Rob Fergus. Automatic Data Augmentation for Generalization in Deep Reinforcement Learning. *arXiv:2006.12862 [cs]*, 2021. URL http://arxiv.org/abs/2006.12862.

[99] Ilya Kostrikov, Denis Yarats, and Rob Fergus. Image Augmentation Is All You Need: Regularizing Deep Reinforcement Learning from Pixels. *arXiv:2004.13649 [cs, eess, stat]*, 2021. URL http://arxiv.org/abs/2004.13649.

[100] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. *arXiv:1707.06347 [cs]*, 2017. URL http://arxiv.org/abs/1707.06347.

[101] Kaixin Wang, Bingyi Kang, Jie Shao, and Jiashi Feng. Improving Generalization in Reinforcement Learning with Mixture Regularization. *arXiv:2010.10814 [cs, stat]*, 2020. URL http://arxiv.org/abs/2010.10814.

[102] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=r1Ddp1-Rb.

[103] Hanping Zhang and Yuhong Guo. Generalization of Reinforcement Learning with Policy-Aware Adversarial Data Augmentation. *arXiv:2106.15587 [cs]*, 2021. URL http://arxiv.org/abs/2106.15587.

[104] Kimin Lee, Kibok Lee, Jinwoo Shin, and Honglak Lee. Network randomization: A simple technique for generalization in deep reinforcement learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=HJgcvJBFvB.

[105] Karl Cobbe, Oleg Klimov, Christopher Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *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*, pages 1282–1289. PMLR, 2019. URL http://proceedings.mlr.press/v97/cobbe19a.html.

[106] Stephen James, Paul Wohlhart, Mrinal Kalakrishnan, Dmitry Kalashnikov, Alex Irpan, Julian Ibarz, Sergey Levine, Raia Hadsell, and Konstantinos Bousmalis. Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 12627–12637. Computer Vision Foundation / IEEE, 2019. doi:10.1109/CVPR.2019.01291. URL http://openaccess.thecvf.com/content_CVPR_2019/html/James_Sim-To-Real_via_Sim-To-Sim_Data-Efficient_Robotic_Grasping_via_Randomized-To-Canonical_Adaptation_Networks_CVPR_2019_paper.html.

[107] Byungchan Ko and Jungseul Ok. Time Matters in Using Data Augmentation for Vision-based Deep Reinforcement Learning. *arXiv:2102.08581 [cs]*, 2021. URL http://arxiv.org/abs/2102.08581.

[108] Nicklas Hansen, Hao Su, and Xiaolong Wang. Stabilizing Deep Q-Learning with ConvNets and Vision Transformers under Data Augmentation. *arXiv:2107.00644 [cs]*, 2021. URL http://arxiv.org/abs/2107.00644.

[109] Yangang Ren, Jingliang Duan, Shengbo Eben Li, Yang Guan, and Qi Sun. Improving Generalization of Reinforcement Learning with Minimax Distributional Soft Actor-Critic. *arXiv:2002.05502 [cs, stat]*, 2020. URL http://arxiv.org/abs/2002.05502.

[110] Chenyang Zhao and Timothy Hospedales. Robust Domain Randomised Reinforcement Learning through Peer-to-Peer Distillation. *arXiv:2012.04839 [cs]*, 2020. URL http://arxiv.org/abs/2012.04839.

[111] Zac Wellmer and James T. Kwok. Dropout's Dream Land: Generalization from Learned Simulators to Reality. In Nuria Oliver, Fernando Pérez-Cruz, Stefan Kramer, Jesse Read, and Jose A. Lozano, editors, *Machine Learning and Knowledge Discovery in Databases. Research Track*, Lecture Notes in Computer Science, pages 255–270, Cham, 2021. Springer International Publishing. ISBN 978-3-030-86486-6. doi:10.1007/978-3-030-86486-6_16.

[112] Rui Wang, Joel Lehman, Jeff Clune, and Kenneth O. Stanley. Paired Open-Ended Trailblazer (POET): Endlessly Generating Increasingly Complex and Diverse Learning Environments and Their Solutions. *arXiv:1901.01753 [cs]*, 2019. URL http://arxiv.org/abs/1901.01753.

[113] Rui Wang, Joel Lehman, Aditya Rawal, Jiale Zhi, Yulun Li, Jeffrey Clune, and Kenneth O. Stanley. 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*, pages 9940–9951. PMLR, 2020. URL http://proceedings.mlr.press/v119/wang20l.html.

[114] Michael Dennis, Natasha Jaques, Eugene Vinitsky, Alexandre M. Bayen, Stuart Russell, Andrew Critch, and Sergey Levine. Emergent complexity and zero-shot transfer via unsupervised environment design. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/985e9a46e10005356bbaf194249f6856-Abstract.html.

[115] Minqi Jiang, Michael Dennis, Jack Parker-Holder, Jakob Foerster, Edward Grefenstette, and Tim Rocktäschel. Replay-Guided Adversarial Environment Design. *arXiv:2110.02439 [cs]*, 2021. URL http://arxiv.org/abs/2110.02439.

[116] Minqi Jiang, Edward Grefenstette, and Tim Rocktäschel. Prioritized Level Replay. *arXiv:2010.03934 [cs]*, 2021. URL http://arxiv.org/abs/2010.03934.

[117] Rémy Portelas, Cédric Colas, Lilian Weng, Katja Hofmann, and Pierre-Yves Oudeyer. Automatic curriculum learning for deep RL: A short survey. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 4819–4825. ijcai.org, 2020. doi:10.24963/ijcai.2020/671. URL https://doi.org/10.24963/ijcai.2020/671.

[118] Mohammed Amin Abdullah, Hang Ren, Haitham Bou Ammar, Vladimir Milenkovic, Rui Luo, Mingtian Zhang, and Jun Wang. Wasserstein Robust Reinforcement Learning. *arXiv:1907.13196 [cs, stat]*, 2019. URL http://arxiv.org/abs/1907.13196.

[119] Daniel J. Mankowitz, Nir Levine, Rae Jeong, Abbas Abdolmaleki, Jost Tobias Springenberg, Yuanyuan Shi, Jackie Kay, Todd Hester, Timothy A. Mann, and Martin A. Riedmiller. Robust reinforcement learning for continuous control with model misspecification. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=HJgC60EtwB.

[120] Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Rémi Munos, Nicolas Heess, and Martin A. Riedmiller. Maximum a posteriori policy optimisation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=S1ANxQW0b.

[121] Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning. In Doina Precup and Yee Whye Teh, editors, *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*, pages 2817–2826. PMLR, 2017. URL http://proceedings.mlr.press/v70/pinto17a.html.

[122] Roberta Raileanu and Rob Fergus. Decoupling Value and Policy for Generalization in Reinforcement Learning. *arXiv:2102.10330 [cs]*, 2021. URL http://arxiv.org/abs/2102.10330.

[123] Irina Higgins, Loïc Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Sy2fzU9gl.

[124] Marin Vlastelica, Michal Rolínek, and Georg Martius. Neuro-algorithmic Policies enable Fast Combinatorial Generalization. *arXiv:2102.07456 [cs]*, 2021. URL http://arxiv.org/abs/2102.07456.

[125] Vinicius Zambaldi, David Raposo, Adam Santoro, Victor Bapst, Yujia Li, Igor Babuschkin, Karl Tuyls, David Reichert, Timothy Lillicrap, Edward Lockhart, Murray Shanahan, Victoria Langston, Razvan Pascanu, Matthew Botvinick, Oriol Vinyals, and Peter Battaglia. Relational Deep Reinforcement Learning. *arXiv:1806.01830 [cs, stat]*, 2018. URL http://arxiv.org/abs/1806.01830.

[126] Ken Kansky, Tom Silver, David A. Mély, Mohamed Eldawy, Miguel Lázaro-Gredilla, Xinghua Lou, Nimrod Dorfman, Szymon Sidor, D. Scott Phoenix, and Dileep George. Schema networks: Zero-shot transfer with a generative causal model of intuitive physics. In Doina Precup and Yee Whye Teh, editors, *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*, pages 1809–1818. PMLR, 2017. URL http://proceedings.mlr.press/v70/kansky17a.html.

[127] Xudong Wang, Long Lian, and Stella X. Yu. Unsupervised Visual Attention and Invariance for Reinforcement Learning. *arXiv:2104.02921 [cs]*, 2021. URL http://arxiv.org/abs/2104.02921.

[128] Michał Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski. ViZDoom: A Doom-based AI research platform for visual reinforcement learning. In *IEEE Conference on Computational Intelligence and Games*, pages 341–348, Santorini, Greece, 2016. IEEE. URL http://arxiv.org/abs/1605.02097.

[129] Yujin Tang and David Ha. The Sensory Neuron as a Transformer: Permutation-Invariant Neural Networks for Reinforcement Learning. *arXiv:2109.02869 [cs]*, 2021. URL http://arxiv.org/abs/2109.02869.

[130] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Caglar Gulcehre, Francis Song, Andrew Ballard, Justin Gilmer, George Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matt Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. Relational inductive biases, deep learning, and graph networks. *arXiv:1806.01261 [cs, stat]*, 2018. URL http://arxiv.org/abs/1806.01261.

[131] Terrance DeVries and Graham W. Taylor. Improved Regularization of Convolutional Neural Networks with Cutout. *arXiv:1708.04552 [cs]*, 2017. URL http://arxiv.org/abs/1708.04552.

[132] Suzan Ece Ada, Emre Ugur, and H. Levent Akin. Generalization in Transfer Learning. *arXiv:1909.01331 [cs, stat]*, 2021. URL http://arxiv.org/abs/1909.01331.

[133] Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pages 1–5, 2015. doi:10.1109/ITW.2015.7133169.

[134] Maximilian Igl, Kamil Ciosek, Yingzhen Li, Sebastian Tschiatschek, Cheng Zhang, Sam Devlin, and Katja Hofmann. Generalization in reinforcement learning with selective noise injection and information bottleneck. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13956–13968, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/e2ccf95a7f2e1878fcafc8376649b6e8-Abstract.html.

[135] Xingyu Lu, Kimin Lee, Pieter Abbeel, and Stas Tiomkin. Dynamics Generalization via Information Bottleneck in Deep Reinforcement Learning. *arXiv:2008.00614 [cs, stat]*, 2020. URL http://arxiv.org/abs/2008.00614.

[136] Benjamin Eysenbach, Ruslan Salakhutdinov, and Sergey Levine. Robust Predictable Control. *arXiv:2109.03214 [cs]*, 2021. URL http://arxiv.org/abs/2109.03214.

[137] Jerry Zikun Chen. Reinforcement Learning Generalization with Surprise Minimization. *arXiv:2004.12399 [cs]*, 2020. URL http://arxiv.org/abs/2004.12399.

[138] Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning Invariant Representations for Reinforcement Learning without Reconstruction. *arXiv:2006.10742 [cs, stat]*, 2021. URL http://arxiv.org/abs/2006.10742.

[139] Rishabh Agarwal, Marlos C. Machado, Pablo Samuel Castro, and Marc G. Bellemare. Contrastive Behavioral Similarity Embeddings for Generalization in Reinforcement Learning. *arXiv:2101.05265 [cs, stat]*, 2021. URL http://arxiv.org/abs/2101.05265.

[140] Anoopkumar Sonar, Vincent Pacelli, and Anirudha Majumdar. Invariant Policy Optimization: Towards Stronger Generalization in Reinforcement Learning. *arXiv:2006.01096 [cs, stat]*, 2020. URL http://arxiv.org/abs/2006.01096.

[141] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant Risk Minimization. *arXiv:1907.02893 [cs, stat]*, 2020. URL http://arxiv.org/abs/1907.02893.

[142] Martín Bertrán, Natalia Martínez, Mariano Phielipp, and Guillermo Sapiro. Instance-based generalization in reinforcement learning. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/82674fc29bc0d9895cee346548c2cb5c-Abstract.html.

[143] Guan Ting Liu, Pu-Jen Cheng, and GuanYu Lin. Cross-State Self-Constraint for Feature Generalization in Deep Reinforcement Learning. 2020. URL https://openreview.net/forum?id=JiNvAGORcMW.

[144] Bogdan Mazoure, Ahmed M. Ahmed, Patrick MacAlpine, R. Devon Hjelm, and Andrey Kolobov. Cross-Trajectory Representation Learning for Zero-Shot Generalization in RL. *arXiv:2106.02193 [cs]*, 2021. URL http://arxiv.org/abs/2106.02193.

[145] Bonnie Li, Vincent François-Lavet, Thang Doan, and Joelle Pineau. Domain Adversarial Reinforcement Learning. *arXiv:2102.07097 [cs]*, 2021. URL http://arxiv.org/abs/2102.07097.

[146] Jiameng Fan and Wenchao Li. DRIBO: Robust Deep Reinforcement Learning via Multi-View Information Bottleneck. *arXiv:2102.13268 [cs]*, 2021. URL http://arxiv.org/abs/2102.13268.

[147] Vincenzo Lomonaco, Karan Desai, Eugenio Culurciello, and Davide Maltoni. Continual Reinforcement Learning in 3D Non-stationary Environments. *arXiv:1905.10112 [cs, stat]*, 2020. URL http://arxiv.org/abs/1905.10112.

[148] Younggyo Seo, Kimin Lee, Ignasi Clavera, Thanard Kurutach, Jinwoo Shin, and Pieter Abbeel. Trajectory-wise Multiple Choice Learning for Dynamics Generalization in Reinforcement Learning. *arXiv:2010.13303 [cs]*, 2020. URL http://arxiv.org/abs/2010.13303.

[149] Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. RMA: Rapid Motor Adaptation for Legged Robots. *arXiv:2107.04034 [cs]*, 2021. URL http://arxiv.org/abs/2107.04034.

[150] Philip J. Ball, Cong Lu, Jack Parker-Holder, and Stephen Roberts. Augmented World Models Facilitate Zero-Shot Dynamics Generalization From a Single Offline Environment. *arXiv:2104.05632 [cs]*, 2021. URL http://arxiv.org/abs/2104.05632.

[151] Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl $2̂$: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

[152] Jane X. Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z. Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. *arXiv:1611.05763 [cs, stat]*, 2017. URL http://arxiv.org/abs/1611.05763.

[153] Maximilian Igl, Gregory Farquhar, Jelena Luketina, Wendelin Boehmer, and Shimon Whiteson. Transient Non-Stationarity and Generalisation in Deep Reinforcement Learning. *arXiv:2006.05826 [cs, stat]*, 2021. URL http://arxiv.org/abs/2006.05826.

[154] Karl Cobbe, Jacob Hilton, Oleg Klimov, and John Schulman. Phasic Policy Gradient. *arXiv:2009.04416 [cs, stat]*, 2020. URL http://arxiv.org/abs/2009.04416.

[155] Jaskirat Singh and Liang Zheng. Sparse Attention Guided Dynamic Value Estimation for Single-Task Multi-Scene Reinforcement Learning. *arXiv:2102.07266 [cs, stat]*, 2021. URL http://arxiv.org/abs/2102.07266.

[156] Ankesh Anand, Jacob Walker, Yazhe Li, Eszter Vértes, Julian Schrittwieser, Sherjil Ozair, Théophane Weber, and Jessica B. Hamrick. Procedural Generalization by Planning with Self-Supervised World Models. *arXiv:2111.01587 [cs]*, 2021. URL http://arxiv.org/abs/2111.01587.

[157] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020. ISSN 1476-4687. doi:10.1038/s41586-020-03051-4. URL https://www.nature.com/articles/s41586-020-03051-4.

[158] Julian Schrittwieser, Thomas Hubert, Amol Mandhane, Mohammadamin Barekatain, Ioannis Antonoglou, and David Silver. Online and Offline Reinforcement Learning by Planning with a Learned Model. *arXiv:2104.06294 [cs]*, 2021. URL http://arxiv.org/abs/2104.06294.

[159] Sam Powers, Eliot Xing, Eric Kolve, Roozbeh Mottaghi, and Abhinav Gupta. CORA: Benchmarks, Baselines, and a Platform for Continual Reinforcement Learning Agents. 2021. URL https://openreview.net/forum?id=Fr_KF_lMCMr.

[160] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=SJx63jRqFm.

[161] Ruihan Yang, Huazhe Xu, Yi Wu, and Xiaolong Wang. Multi-Task Reinforcement Learning with Soft Modularization. *arXiv:2003.13661 [cs, stat]*, 2020. URL http://arxiv.org/abs/2003.13661.

[162] Ayush Jain, Andrew Szot, and Joseph J. Lim. Generalization to new actions in 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*, pages 4661–4672. PMLR, 2020. URL http://proceedings.mlr.press/v119/jain20b.html.

[163] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems. *arXiv:2005.01643 [cs, stat]*, 2020. URL http://arxiv.org/abs/2005.01643.

[164] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/0d2b2061826a5df3221116a5085a6052-Abstract.html.

[165] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for Deep Data-Driven Reinforcement Learning. *arXiv:2004.07219 [cs, stat]*, 2021. URL http://arxiv.org/abs/2004.07219.

[166] Caglar Gulcehre, Ziyu Wang, Alexander Novikov, Tom Le Paine, Sergio Gomez Colmenarejo, Konrad Zolna, Rishabh Agarwal, Josh Merel, Daniel Mankowitz, Cosmin Paduraru, Gabriel Dulac-Arnold, Jerry Li, Mohammad Norouzi, Matt Hoffman, Ofir Nachum, George Tucker, Nicolas Heess, and Nando de Freitas. RL Unplugged: A Suite of Benchmarks for Offline Reinforcement Learning. *arXiv:2006.13888 [cs, stat]*, 2021. URL http://arxiv.org/abs/2006.13888.

[167] Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. AWAC: Accelerating Online Reinforcement Learning with Offline Datasets. *arXiv:2006.09359 [cs, stat]*, 2021. URL http://arxiv.org/abs/2006.09359.

[168] Saurabh Arora and Prashant Doshi. A Survey of Inverse Reinforcement Learning: Challenges, Methods and Progress. *arXiv:1806.06877 [cs, stat]*, 2020. URL http://arxiv.org/abs/1806.06877.

[169] Andrew Y. Ng and Stuart J. Russell. Algorithms for inverse reinforcement learning. In Pat Langley, editor, *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000), Stanford University, Stanford, CA, USA, June 29 - July 2, 2000*, pages 663–670. Morgan Kaufmann, 2000.

[170] Annie S. Chen, Suraj Nair, and Chelsea Finn. Learning Generalizable Robotic Reward Functions from "In-The-Wild" Human Videos. *arXiv:2103.16817 [cs]*, 2021. URL http://arxiv.org/abs/2103.16817.

[171] Karl Schmeckpeper, Oleh Rybkin, Kostas Daniilidis, Sergey Levine, and Chelsea Finn. Reinforcement Learning with Videos: Combining Offline Observations with Interaction. *arXiv:2011.06507 [cs]*, 2020. URL http://arxiv.org/abs/2011.06507.

[172] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, and Sergey Levine. Time-Contrastive Networks: Self-Supervised Learning from Video. *arXiv:1704.06888 [cs]*, 2018. URL http://arxiv.org/abs/1704.06888.

[173] Alessandro Sestini, Alexander Kuhnle, and Andrew D. Bagdanov. Demonstration-efficient Inverse Reinforcement Learning in Procedurally Generated Environments. *arXiv:2012.02527 [cs]*, 2020. URL http://arxiv.org/abs/2012.02527.

[174] Ian Osband, Yotam Doron, Matteo Hessel, John Aslanides, Eren Sezener, Andre Saraiva, Katrina McKinney, Tor Lattimore, Csaba Szepesvári, Satinder Singh, Benjamin Van Roy, Richard S. Sutton, David Silver, and Hado van Hasselt. Behaviour suite for reinforcement learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=rygf-kSYwH.

[175] Emmanuel Bengio, Joelle Pineau, and Doina Precup. Interference and generalization in temporal difference 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*, pages 767–777. PMLR, 2020. URL http://proceedings.mlr.press/v119/bengio20a.html.

[176] Ian Osband and Benjamin Van Roy. Near-optimal reinforcement learning in factored mdps. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 604–612, 2014. URL https://proceedings.neurips.cc/paper/2014/hash/0deb1c54814305ca9ad266f53bc82511-Abstract.html.

[177] Bernhard Schölkopf. Causality for Machine Learning. *arXiv:1911.10500 [cs, stat]*, 2019. URL http://arxiv.org/abs/1911.10500.

[178] Daniel S Weld. Solving Relational MDPs with First-Order Machine Learning. page 8.

[179] Carlos Diuk, Andre Cohen, and Michael L. Littman. An object-oriented representation for efficient reinforcement learning. In William W. Cohen, Andrew McCallum, and Sam T. Roweis, editors, *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, volume 307 of *ACM International Conference Proceeding Series*, pages 240–247. ACM, 2008. doi:10.1145/1390156.1390187. URL https://doi.org/10.1145/1390156.1390187.

A Other Structural Assumptions on MDPs

Other forms of structured MDPs have been defined beyond the contextual MDP [13] and leveraged to develop algorithms that exploit those structural assumptions. One type that holds promise for generalisation is the factored MDP. A **factored MDP** assumes a state space described by a set of discrete variables, denoted $S := \{S_1, S_2, \ldots, S_n\}$ [40, 41]. We follow the notation and definitions used by [176]. The transition function T has the following property:

Definition 1 (Factored transition functions)**.** *Given two states s, s' and action a in a factored MDP M, the transition function satisfies a conditional independence condition*

$$T(s'|s,a) = \prod_i P(s'_i|s,a), \tag{4}$$

where $P(s'_i|s,a)$ is the probability distribution for each factor S_i conditioned on the previous state and action.

Parallels to causal graphs can be drawn [177], where a causal graph is a DAG, each vertex is a variable, and the directed edges represent causal relationships. We can rewrite the conditional independence assumption as

$$T(s'|s,a) = \prod_i P(s'_i|\text{PA}(s'_i), a), \tag{5}$$

where the probability of each factor s_i only depends on its parent factors $\text{PA}(s_i)$ from the previous time step. Ideally, these parents are only a subset of all factors so this representation results in a reduction in size from the original MDP. Further, this enforces that there are no synchronous edges between factors in the same time step. Critically, the rewards can also be factored in the following way:

Definition 2 (Factored reward functions)**.** *Given two states s, s' and action a in a factored MDP M, the reward function satisfies a conditional independence condition*

$$\mathbb{E}\big[R(s,a)\big] = \sum_i \mathbb{E}\big[R_i(s_i,a)\big], \tag{6}$$

where $R_i(s_i, a)$ is the reward function for each factor S_i.

One can think of this factored MDP framework as an additional assumption for a single context-MDP, or the combination of one or more of these factors *as* the context, with the space of all possible combinations of factors being the context set. In the latter case, this formulation explicitly encodes how generalisation can be achieved to new contexts via *systematicity* (Section 3.1, [33]): the policy will be trained on contexts taking some values within the set of possible factors, and then be tested on unseen combinations of seen factors.

A more restricted form of structured MDP is the **relational MDP** [178]. A relational MDP is described by tuple $\langle C, F, A, D, T, R\rangle$. C is the set of object types, F is the set of fluent schemata that are arguments that modify each object type. A is the set of action schemata that acts on objects, D is the set of domain objects, each associated with a single type from C, and finally T is the transition function and R the reward function. An additional assumption is that objects that are not acted upon do not change in a transition. Note that the relational MDP can be expanded into a factored MDP that does not assume the additional structure of the form of object types with invariant relations. While this form of MDP is a Planning Domain Definition Language (PDDL) and therefore lends itself well to planning algorithms, it is overly complex for learning algorithms.

Object-oriented MDPs [179] are a simpler form of relational MDPs that are less constrained, and therefore hold more promise for learning methods. Objects, fluents, and actions are defined in the same away as in relational MDPs, but all transition dynamics are determined by a set of Boolean transition terms which consist of a set of pre-defined relation terms between objects and object attributes. In spite of this simplification, it is still significantly constrained compared to CMDPs, and can be difficult to use when describing complex systems.

A final example of a structured MDP is the **Block MDP**. Block MDPs [38] are described by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{X}, p, q, R\rangle$ with a finite, unobservable state space $\mathcal{S}$ and possibly infinite, but observable space $\mathcal{X}$. p denotes the latent transition distribution, q is the (possibly stochastic) emission function and R the reward function. This structured MDP is useful in rich observation environments where the given observation space is large, but a much smaller state space can be found that yields an equivalent MDP. This allows for improved exploration and sample complexity bounds that rely on the size of that latent state space rather than the given observation space.