A Survey of Zero-shot Generalisation in Deep Reinforcement Learning

Robert Kirk ROBERT.KIRK.20@UCL.AC.UK
University College London, Gower St, London
WC1E 6BT, United Kingdom

Amy Zhang AMYZHANG@FB.COM
University of California, Berkeley, Berkeley
CA, United States
Meta AI Research,

Edward Grefenstette E.GREFENSTETTE@UCL.AC.UK
University College London, Gower St, London
WC1E 6BT, United Kingdom

Tim Rocktäschel TIM.ROCKTASCHEL@UCL.AC.UK
University College London, Gower St, London
WC1E 6BT, United Kingdom

Abstract

The study of zero-shot generalisation (ZSG) 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 rely on a unifying formalism and terminology for discussing different ZSG problems, building upon previous works. We go on to categorise existing benchmarks for ZSG, as well as current methods for tackling these problems. 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 ZSG, we suggest fast online adaptation and tackling RL-specific problems as some areas for future work on methods for ZSG, and we recommend building benchmarks in underexplored problem settings such as offline RL ZSG and reward-function variation.

1. Introduction

Reinforcement Learning (RL) has the potential to be used in a wide range of applications from autonomous vehicles (Filos et al., 2020) and algorithm control (Biedenkapp et al., 2020) to robotics (OpenAI et al., 2019a), but to fulfil 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 variations 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 (Bellemare et al., 2013) and MuJoCo (E. Todorov et al., 2012; Brockman et al., 2016), which do not have the attributes described above: they evaluate the policy on exactly the same environment it was

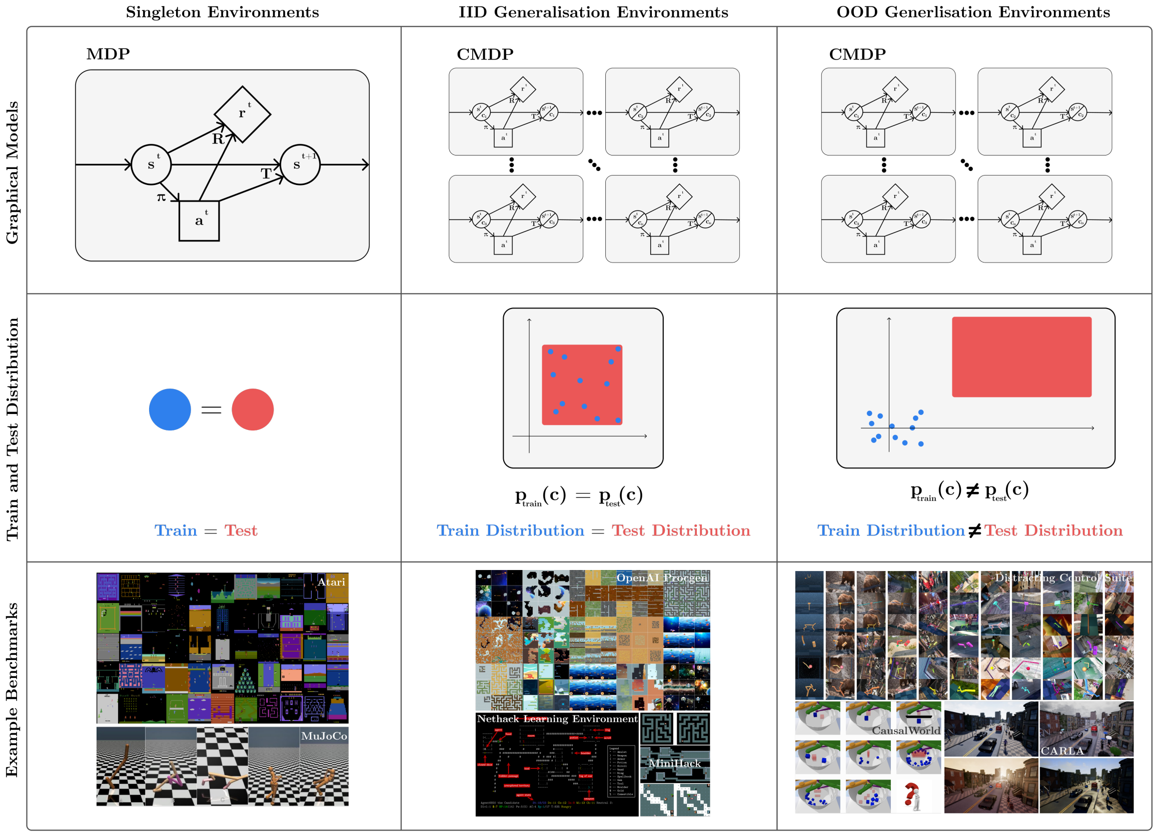
Singleton Environments
IID Generalisation Environments
OOD Generlisation Environments
Graphical Models
Train and Test Distribution
Example Benchmarks
MDP
CMDP
CMDP
r^t
R
s^t
T
s^{t+1}
π
a^t
Train = Test
$p_{train}(c) = p_{test}(c)$
Train Distribution = Test Distribution
$p_{train}(c) \neq p_{test}(c)$
Train Distribution ≠ Test Distribution
Atari
MuJoCo
OpenAI Procgen
Nethack Learning Environment
MiniHack
Distracting Control Suite
CausalWorld
CARLA

Figure 1: **Zero-shot 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). We focus on an underexplored setting, inspired by likely real-world scenarios, where training and testing environments will be different, with environment instances either from the same distribution (Independent and Identically Distributed (IID) ZSG Environments, second column) or from different distributions (OOD ZSG Environments, third column). The split between the second and third columns is just one example of the way in which ZSG is a class of problems rather than an individual problem. The top row visualises the differences in graphical models between singleton environments and environments where ZSG is required. For more information on the CMDP formalism see Section 3.3.

trained on, which often 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 (Whiteson et al., 2011). 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 (Zhang et al., 2018, 2018b; Farebrother et al., 2020; Gamrian & Goldberg, 2019).

In this work we survey the recent literature studying zero-shot generalisation in deep RL, a field focused on producing algorithms with the robustness, transfer and adaptation properties required to perform well in the real world. We offer a unifying framework that builds on previous work (Ghosh et al., 2021; Hallak et al., 2015; Harrison et al., 2017; Higgins et al., 2017b; Song et al., 2020; Perez et al., 2020) that formalises the problem of ZSG in RL as a *class* of problems, rather than a single problem. While prior work (Ghosh et al., 2021; Hallak et al., 2015; Perez et al., 2020) uses the contextual MDP framework or related frameworks to describe how an agent can encounter new, unseen states at test time to generalise to, we further extend and break down the types of generalisation that can be possible, e.g. combinatorial, interpolation vs. extrapolation, single-factor vs. multi-factor (Fig. 3). We formalise more fully the ZSG problem, formally specifying the policy class, making clear the choice of whether the context is observed or not, and including cases where the context distribution is controllable during training Section 3.4 This breakdown enables us to clearly and crisply compare previous works, as well as understand how to choose future research directions. For example, improving "generalisation" without any additional assumptions is inherently 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 (Wolpert & Macready, 1997) applies: improving generalisation in some settings could harm generalisation in others. Two broad categories of ZSG problem are shown in Fig. 1 in the centre and right columns.

Using this formalism, we survey and examine the range of benchmarks available for ZSG in RL, and go on to discuss methods aimed at tackling different ZSG problems. Finally, we propose several settings within ZSG 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. Throughout, we critically review the state of the field and provide recommendations for ensuring future research is robust and useful. 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 by providing a common reference and framework. This new clarity can improve the field, and enable robust progress towards more general RL methods.

Scope. Generalisation in RL captures a large amount of research, so to make this survey feasible we limit the scope of our review in several ways. First, we focus on the specific problem setting of zero-shot generalisation (ZSG), where the policy is evaluated zero-shot on a collection of environment instances different to those it was trained on. Crucially, this setting disallows any additional training in or data from the test environment instances, meaning methods such as domain adaptation and many meta-RL approaches are not applicable. This is different from classical RL, which has historically focused on online learning in a single MDP, where generalisation refers to the notion of generalising to novel states in the same MDP. While this setting can be useful to study, we instead focus on situations which require policies that can be deployed in situations they haven't been trained in, and generalise well zero-shot to those situations. This setting is especially relevant for current deep RL algorithms, which often aren't sample-efficient or safe enough to be deployed online without significant offline or in-simulation training first. This motivates our focus on ZSG. We discuss and motivate this setting and its restrictions more in Section 3.7.

Second, 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 (OpenAI et al., 2019b; Vinyals et al., 2019) and generalising to new team-mates in cooperative games (Hu et al., 2021, 2020), but we do not cover any work in this area here. While mathematically these problems could be modelled equivalently (if co-players are modelled as parts of the dynamics function, rather than as agents as in Games-based formulations (Shapley, 1953)), approaches to these problems tend to be quite different, explicitly utilising the fact that variation and generalisation challenges come from other co-players rather than other parts of the environment. 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 (Team et al., 2021), which we do cover.

Finally, we do not cover theoretical work on generalisation in RL. While there is recent work in this area (Du et al., 2020; Malik et al., 2021) that is valuable, we focus on empirical research as it's more widely studied.

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 ZSG in RL in Section 3, including the relevant background. We then proceed to use this formalism to describe current benchmarks for ZSG 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 ZSG 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 unified formalism and terminology for discussing the broad class of ZSG problems and breaking down the assumptions necessary to achieve ZSG, building on formalisms and terminology presented in multiple previous works (Ghosh et al., 2021; Hallak et al., 2015; Harrison et al., 2017; Higgins et al., 2017b; Song et al., 2020; Perez et al., 2020). Our contribution here is the unification of these prior works into *a clear formal description of the class of problems referred to as ZSG in RL*, which captures the full space of problems, which wasn't done by any one existing formalism.

- We propose a taxonomy of existing benchmarks that can be used to test for ZSG, splitting the discussion into categorising environments and evaluation protocols. Our formalism allows us to cleanly describe weaknesses of the purely Procedural Content Generation (PCG) approach to ZSG 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 ZSG 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 ZSG issues, novel architectures, model-based RL and environment instance generation.

- We critically discuss the current state of ZSG in RL research, recommending future research directions. In particular, we argue that *building benchmarks would enable progress in 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 zero-shot generalisation in RL explicitly. Khetarpal et al. (2020) motivated and surveyed continual reinforcement learning (CRL), which is closely related to ZSG in RL as both settings require adaptation to unseen tasks or environments; however, they explicitly do not discuss the zero-shot setting that is the concern of this paper (for more discussion of CRL see Section 6.1). Chen and Li (2020) gave a brief overview of Robust RL (RRL) (Morimoto & Doya, 2000), 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 Chen and Li (2020) only briefly survey the field. Albrecht and Stone (2018) survey methods for modelling other agents, which can be seen as a form of generalisation problem (even in single-agent RL), if the environment contains a distribution over agents.

Zhao et al. (2020) survey 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 Zhao et al. (2020), 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. (2021) and Zhu et al. (2021) 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 (2020) who survey multi-task deep RL, Amin et al. (2021) who survey exploration in RL, and Narvekar et al. (2020) who survey 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 ZSG problems which will enable research in this field to discuss the differences between different problems.

3. Formalising Zero-shot Generalisation In Reinforcement Learning

In this section, we present a formalism for understanding and discussing the class of zero-shot generalisation (ZSG) problems in RL. We first review the relevant background in supervised learning and RL before motivating the formalism itself. Formalising ZSG in this way shows that it refers to a *class* of problems, rather than a specific problem, and hence research on ZSG needs to specify which group of ZSG 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 ZSG 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 independently and identically distributed (IID) from the same underlying distribution, although this is not always the case (see for example the Domain Generalisation literature (Zhou et al., 2022)). 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 normally used as a measure of generalisation specifically, independently of the train or test performance: for a given level of training performance, a smaller gap means a model generalises better. This metric isn't perfect, as a model that performs at random chance in both training and testing will get a gap of 0. Further, if the train and test datasets aren't drawn IID, then it's possible that the test dataset is easier (or harder), and hence a gap of zero doesn't necessarily imply perfect generalisation. However, it can be used to measure generalisation performance across benchmarks where absolute performance may not be comparable, or to motivate improvement from methods that may improve generalisation by lowering the gap without changing the test performance (in effect by lowering the training performance). These methods may then be combined with ones that improve training performance to improve the overall test performance, assuming that the methods don't conflict. We introduce this metric for completeness, as it has often been used in the literature in addition to test performance. In general, we think it's useful as a metric in addition to test performance, but not as a replacement for it. For more discussion, especially in the RL setting, see Section 3.4.

One specific type of generalisation examined frequently in supervised learning which is relevant to RL is compositional generalisation (Hupkes et al., 2020; Keysers et al., 2020). We explore a categorisation of compositional generalisation here introduced by Hupkes et al. (2020). 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 we discuss how assumptions of compositional structure in the RL environment 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:

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

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 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. Note that we formalise the policy here as Markovian (i.e. that only takes the previous state as input) for simplicity, but the policy can take in the full history $(s_1, a_1, r_1, \ldots s_{t-1}, a_{t-1}, r_{t-1}, s_t)$ as input, for example using a recurrent neural network. We define the set of possible histories for a state and action space as $H[S, A] = \{(s_1, a_1, r_1, \ldots s_{t-1}, a_{t-1}, r_{t-1}, s_t) | t \in \mathbb{N}\}$, similarly for an observation space. A policy being non-Markovian allows it to be adaptive (for further discussion see Section 3.7).

3.3 Contextual Markov Decision Processes

To talk about zero-shot generalisation, we desire a way of reasoning about a *collection* of tasks, environment instances or levels: the need for generalisation emerges from the fact we train and test the policy on different collections of environment instances. Consider as a didactic example OpenAI Procgen (Cobbe et al., 2020a): 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 (Stone et al., 2021), 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 a 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 by Hallak et al. (2015), but using the alternative formalism from Ghosh et al. (2021). This formalism also builds on those presented by Doshi-Velez and Konidaris (2016), Perez et al. (2020), but we extend them to consider different distributions over context parameters; we include both observed and unobserved contexts settings; we include cases where the context is controllable; and we define formally how to produce subset CMDPs (see the end of this section for more discussion comparing the formalism we present here and existing works).

Definition 1. *A contextual MDP (CMDP) is a tuple*

$$\mathcal{M} = \left(S', A, O, R, T, C, \phi : S' \times C \to O, p(s'|c), p(c)\right).$$

A, O, R, T, ϕ are as in the definition of the POMDP in Section 3.2. C is the context space (a set over which it is possible to have a distribution). The CMDP is a POMDP with state space $S := S' \times C$, initial state distribution $p((s', c)) = p(c)p(s'|c)$, that is the POMDP $(S' \times C, A, O, R, T, \phi, p(s'|c)p(c))$. Hence, R has type $R : S' \times C \to \mathbb{R}$ and $T((s, c), a)$ is the form of the transition probability distribution. For the tuple to be a CMDP, the transition function must be factored such that the context doesn't change within an episode, that is $T((s, c), a)((s', c')) = 0$ if $c' \neq c$. We call S' the underlying state space, and $p(c)$ the context distribution.

To give an intuition for this definition, the context takes the role of the seed, ID or parameter vector which determines the level. Hence why it should not change within an episode, only between episodes. The CMDP is the entire collection of tasks or environment instances; in Procgen, each game (e.g. starpilot, coinrun, etc.) is a separate CMDP. The context distribution $p(c)$ is what is used to determine the training and testing collections of levels, tasks or environment instances; in Procgen this distribution is uniform over the fixed 200 seeds at training time, and uniform over all seeds at testing time.

Note that this definition leaves it unspecified whether the context is observed by the agent: if $O = O' \times C$ for some underlying observation space O' and $\phi((s', c)) = (\phi'(s), c)$ for some underlying observation function $\phi' : S' \to O'$ then we say the context is observed, otherwise it isn't. The context needs to be observed for the CMDP to be an MDP (and not a POMDP), but the opposite isn't true - even if the context is observed, ϕ' could not be the identity, in which case the POMDP isn't likely to be an MDP. Note that 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 resulting 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* $\mathcal{M}_{c^*}$. Formally, this is a new CMDP with $p(c) := 1$ if $c = c^*$ else 0. This is a specific task or environment instance, 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 environment instances, 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 context sets, as the context uniquely determines the MDP.

First, we need to describe how to use training and testing context sets to create new CMDPs.

Definition 2. *For any CMDP $\mathcal{M} = (S', A, O, R, T, C, \phi, p(s'|c), p(c))$, we can choose a subset of the context set $C' \subseteq C$, and then produce a new CMDP*

$$\mathcal{M}|_{C'} = \left(S', A, O, R, T, C', \phi, p(s'|c), p'(c)\right)$$

where $p'(c) = \frac{p(c)}{Z}$ if $c \in C'$ else 0 and Z is a renormalisation term $Z = \sum_{c \in C'} p(c)$ that ensures $p'(c)$ is a probability distribution.

This allows us to split the total collection of context-MDPs 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 the objective, we use the expected return of a policy:

Definition 3. *For any CMDP $\mathcal{M}$ we can define the expected return of a policy in that CMDP as*

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

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

We can now formally define the Zero-Shot Policy Transfer (ZSPT) problem class.

Definition 4 (Zero Shot Policy Transfer)**.** *A ZSPT problem is defined by a choice of CMDP $\mathcal{M}$ with context set C and a choice of training and testing context sets $C_{train}, C_{train} \subseteq C$.*

The objective is to produce a non-Markovian policy $\pi : H[O, A] \rightarrow A$ which maximises the expected return in the testing CMDP $\mathcal{M}|_{C test}$:

$$J(\pi) := \boldsymbol{R}(\pi, \mathcal{M}|_{C test}).$$

This policy can be produced through interaction with the training CMDP $\mathcal{M}|_{C_{train}}$ for a fixed number of environment and episode samples N_s, N_e respectively.

ZSG research is generally concerned with developing algorithms that can solve a variety of ZSPT problems. For example, in Procgen we aim to produce an algorithm that can solve the ZSPT problem for every game. Specifically, 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 ($N_s = 25 \times 10^6, N_e = \infty$) on the training distribution of levels (which is a fixed set of 200 levels). The name *Zero-Shot Policy Transfer* comes from prior works (Harrison et al., 2017; Higgins et al., 2017b).

Some algorithms assume that the context distribution can be adjusted during interaction with the training CMDP, as long as sampled contexts are only within the fixed training context set:

Definition 5 (ZSPT controllable context)**.** *A* controllable context *ZSPT problem is the same as a ZSPT problem above, except the learning algorithm can adjust the context distribution of the training CMDP $p_{train}(c)$ during training, so long as it maintains the property of only sampling from the training context set: $p_{train}(c) = 0$ if $c \notin C_{train}$*

Note that this formalism defines a class *of problems, each determined by a choice of CMDP, training and testing context sets and whether the context is controllable.* 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 (either implicit or explicit) will likely be required for learning to occur (Section 3.6), but we do not believe there is a unifying assumption behind all ZSG problems apart from those stated here.

Evaluating Zero-Shot Generalisation. As in supervised learning, we can consider the gap between training and testing performance as a measure of generalisation. We define this analogously to how it's defined in supervised learning (Eq. (1)), swapping the ordering between training and testing (as we maximise reward, rather than minimise loss):

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

This metric is frequently used in the literature in addition to test performance to evaluate ZSG algorithms (Jiang et al., 2021b; Raileanu et al., 2021; Raileanu & Fergus, 2021). In general, it's not clear what the best evaluation metric for ZSG algorithms is. In supervised learning, the 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 (Henderson et al., 2018). In this setting, we may care more about the zero-shot generalisation *potential* of algorithms by decoupling generalisation from training performance and evaluating using the generalisation gap instead. For example, if algorithm A has higher testing performance than algorithm B, but A also has a much larger generalisation gap, we may prefer to use algorithm B in a new setting, so we have better assurance that the deployment performance won't deviate as much from the training performance, and the algorithm may be more robust. This is the reason the previous literature has often reported this metric alongside test performance.

However, the generalisation gap in RL has the same problems as discussed for the supervised learning generalisation gap: a gap of zero doesn't necessarily imply good performance (i.e. a random policy is likely to get a gap of 0), and if the reward functions aren't comparable across training and testing, then the magnitude of the gap my not be informative (and it could then only be used to compare between different algorithms). This means using it as the only metric for improved performance will likely not lead to robust progress in ZSG. Further, given how broad the current set of assumptions is, it is unlikely there is a single general measure of progress towards tackling ZSG: across such a broad problem class, objectives may even be conflicting (Wolpert & Macready, 1997).

Therefore, our recommendation is first and foremost 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 ZSG capability (for further discussion see Section 4.3 and Section 6.2). As in the literature, generalisation gap can be used as an additional auxiliary metric to evaluate the performance of ZSG algorithms as well as test performance, to either break ties between algorithms with very similar test time performance, or to inform users in situations where it is more important to have strong assurances on test time performance than to have expected test time performance as high as possible.

3.5 Real World Examples of This 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 real-world scenarios where this formalism naturally applies:

- Sim-to-real is a classic problem of ZSG and one which can be captured in this framework. Here the outer CMDP is a union between the simulation and real-world MDPs. The context set will be split into those contexts which correspond to simulation, and those that correspond to reality. The context generally conditions the dynamics, observation function and state distribution, but likely not the reward (so $\forall s', c : R((s', c)) = R'(s')$). Domain randomisation approaches are motivated by the idea that producing a wide range of possible contexts in simulation (the training CMDP) will make it more likely that the testing distribution of contexts is closer to the expanded training distribution. In a simulation setting, we'd normally assume access to the context distribution, and that the context could be made observable, but that the context won't be observable at testing time, and so it may not be useful to have a policy that explicitly conditions on the context.

- 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). 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 the patient's 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, which means that the context will likely control the reward function and that the part of the context that controls the read function will be observable (so the policy knows what task to perform). Driving in different locations (different contexts changing the initial state distribution), under different weather and lighting conditions due to the time of day (observation functions) and on different road surfaces (transition functions) 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 zero-shot to the unseen contexts well.

3.6 Additional Assumptions For More Feasible Generalisation

In choosing the CMDP formalism we opted to formalise ZSG 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 ZSG 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 ZSG 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 (Cobbe et al., 2020a), 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 ZSG in RL do not assume that the train and test environment instances are drawn from the same distribution. This is often called *Domain Generalisation*, where we refer to the training and testing environments instances 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.

Note that while the testing context set could be a single context, this would likely lead to a not particularly robust algorithm - it's possible that it overfits to the problem of producing a policy that performs well on this specific context, which may not generalise to other similar contexts (which is the high-level goal of this research direction).

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 ZSG. 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 ZSG.

An example of a structured MDP that has been used to improve generalisation is the *block MDP* (Du et al., 2019). 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. (2019) 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. (2020) develop a representation learning method that disentangles relevant from irrelevant features, improving generalisation to environments instances where only the irrelevant features change, a simple form of *systematicity* (Section 3.1, Hupkes et al., 2020). This is a rare example of a method explicitly utilising additional assumptions of structure to improve generalisation. Block MDPs can be combined with contextual MDPs by introducing an emission mapping from state space to observation space that is also dependent on context, as defined by Sodhani et al. (2022).

Factored MDPs (Boutilier et al., 2000; Strehl et al., 2007) 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 (Hao et al., 2021) 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 demonstrated by Huang et al. (2021), although it does not target the zero-shot policy transfer setting directly. We note that contextual MDPs can be trivially represented by factored MDPs with two factors, the state and context. However, factored MDPs are capable of modelling more structure in a domain if present. Therefore, if an environment is capable of being modelled by a factored MDP in addition to a contextual MDP, better generalisation guarantees and results are likely possible if this structure is exploited. We hope to see more work applying these kinds of structural assumptions to the zero-shot generalisation problems discussed in this work.

3.7 Remarks And Discussion

Angles to Tackle the ZSPT Problem. While we aim to improve test-time performance Definition 3, we often do not have access to that performance metric directly (or will not when applying our methods in the real world). In research, to develop algorithms that improve the test-time performance, we could aim to produce algorithms that (when compared to existing work) 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.[1] Work on ZSG 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 (1) 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 (Harrison et al., 2017; Higgins et al., 2017b): a policy is learned from the training CMDP and evaluated zero-shot in the testing CMDP. This field is important for several reasons.

First, as mentioned above, current deep RL algorithms often aren't safe or sample-efficient enough to perform online learning in the real world. Hence, a significant amount of offline or in-simulation training is required, and policies need to generalise zero-shot to the real-world deployment setting. As access to more compute and richer simulations becomes available, we expect many successful real-world deployments of RL to follow this workflow at least in part: training offline or in simulation and then transferring the policy zero-shot to the deployment environment. Even if the policy will continue learning during deployment, it still needs to be reasonably good at deployment time (i.e. zero-shot), otherwise it wouldn't be safe to deploy it. In this way, we view work on ZSG as mostly complementary to work on continual RL, as we think both are important for enabling the deployment of robust and competent RL policies. Note that there may be tradeoffs between good zero-shot performance and good continual learning performance, and how to choose between these two desiderata will be determined by the specific problem setting being faced.

Second, from a safety, interpretability and verification perspective, ZSG may be preferable to a continually updated policy. It's likely in high-stakes scenarios that models will be verified (Katz et al., 2019) or audited with interpretability or explainability methods (Milani et al., 2022), and that this process will be expensive. In this scenario, it will be beneficial to have a single model which performs well zero-shot without having to be continually updated, as after each update these verification and auditing steps will likely have to be repeated.

[1] If the training environment instances are identical to the testing environment instances, then the generalisation gap will always be 0.

Finally, note that 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.[2] 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 can be used as a base which is then built upon with domain-specific knowledge and extra data.

Note that while "training" and "learning" are contentious terms, our definition grounds them in the production of a non-Markovian policy after some number of samples from the training context MDP. The fact that the objective is the expected testing return within a *single episode* of the non-Markovian policy means that online learning or adaptation across more than 1 episode isn't possible, and so the adaptation would have to happen within a single episode to be useful (i.e. using a recurrent policy). Several methods do take this approach, as described in Section 5.2.4. To be clear, we are grounding ourselves in this specific *objective*, and not placing any restrictions on the *properties* of methods as long as they satisfy the constraints: policies need not be Markovian, and can adapt within a single episode to the environment instances they're placed in if that improves performance.

Relationship to Previous Notions of Generalisation in RL. Historically, generalisation in RL has referred to the notion of generalising to novel states or state-action pairs within a single MDP. While this notion is useful in the online learning single-MDP setting that is the focus of that work, we here focus on a more recent and more realistic setting where the policy is trained on a collection of MDPs and then deployed on possibly unseen MDPs. This setting more closely mirrors the supervised learning notion of generalisation. We believe this type of workflow for deploying RL-trained policies in the real world is much more feasible than training a policy online from scratch, as current RL algorithms aren't sample-efficient or safe enough to train online in the real world. This means a large amount of off-line or in-simulation training will have to occur before the policy is deployed, and the deployed policy will still need to perform reasonably well as soon as it's deployed (i.e. zero-shot).

Relationship to Previous Formalisms of Collections or Distributions of MDPs. As mentioned previously, the formalism we presented above builds on multiple previous works. Here we briefly present the differences between our formalism and these works.

Doshi-Velez and Konidaris (2016) present Hidden-parameter MDPs (Hi-MDPs). These are MDPs with a hidden parameter which controls the transition function and a distribution over the set of these hidden parameters that implicitly defines a set of MDPs. Perez et al. (2020) builds on Hi-MDPs to present Generalised Hidden-parameter MDPs (GHP-MDPs), where the hidden parameters now also control the reward function in addition to the dynamics. Our work uses a context parameter which is analogous to the hidden parameter, which may be hidden or observed, and that controls the initial state distribution, transition function and reward function, rather than just the transition and reward functions. We also more formally discuss how different distributions of context parameters may be used during training and testing.

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

Harrison et al. (2017), Higgins et al. (2017b) introduce variations on the term "zero-shot policy transfer", which we use as the name for the formal class of problems we study. They both study problems within this class but don't formalise the entire class of problems as we do here, instead focusing on presenting methods for improving performance in the empirical settings they investigate.

Song et al. (2020) discusses a distribution over MDPs which are sampled from during training, and uses a kind of CMDP but where the context parameter adjusts only the observation function, rather than any other parts of the MDP. They also assume the distribution over MDPs is the same between training and testing.

Hallak et al. (2015), Ghosh et al. (2021) present Contextual MDPs (CMDPs), which is the formalism we use as the base of our definitions. Ghosh et al. (2021) present a formalism based on the context being part of an underlying state space, which is the one we use, while Hallak et al. (2015) present a formalism more similar to Hi-MDPs, where there is a collection of MDPs parameterised by a context variable. Both of these works don't consider a shift in distribution over contexts between training and testing apart from a shift to the full distribution in Ghosh et al. (2021). They don't present a formal definition of the policy class and don't define problem settings where the context distribution is controllable.

4. Benchmarks For Zero-shot Generalisation In Reinforcement Learning

In this section, we give a taxonomy of benchmarks for ZSG 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 the environment (a CMDP, covered in Section 4.1) and a 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), as well as discuss the range of difficulty among ZSG problems (Section 4.3). More discussion of future work on benchmarks for ZSG can be found in Sections 6.1, 6.2 and 6.4.

4.1 Environments

Categorising Environments That Enable Generalisation In Table 1, we list the available environments for testing ZSG in RL, as well as summarise 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

Name	Style	Contexts	Variation
Alchemy † (Wang et al.)	3D	PCG	D, R, S
Animal-AI (Crosby et al.)	3D	D-C, D-O	S, O
Atari Game Modes (Machado et al.)	Arcade	D-C	D, O, S
BabyAI (Chevalier-Boisvert et al.)	Grid, LC	D-C, D-O, PCG	R, S
CARL (Benjamins et al.)	Varied	Con, D-C, D-O	D, O, R, S
CARLA (Fan et al., Zhu et al.)	3D, Driving	D-C	O
CausalWorld † (Ahmed et al.)	3D, ConCon	Con, D-C, D-O	D, O, R, S
Construction (Bapst et al.)	2D, Structured	Con, D-C, D-O, PCG	R, S
Crafter (Hafner)	Arcade, Grid	PCG	S
Crafting gridworld (Chen et al.)	Grid, LC	D-C	R, S
DACBench (Eimer et al.)	Structured	PCG, D-C	D, R, S
DCS (Stone et al.)	ConCon	Con, D-C	O
DistractingCarRacing (Tang et al., Brockman et al.)	Arcade	D-C	O
DistractingVizDoom (Tang et al., OpenAI)	3D	D-C	O
DMC-GB (Hansen and Wang)	ConCon	Con, D-C	O
DMC-Remastered (Grigsby and Qi)	ConCon	Con, D-C	O
DM-Memory (Fortunato et al.)	Arcade, 3D	PCG, Con, D-O, D-C	R, D, S
GenAsses (Packer et al.)	ConCon	Con	D, S
GVGAI (Perez-Liebana et al.)	Grid	D-C	D, O, S
HALMA † (Xie et al.)	Grid	D-C, D-O	O, S
iGibson (Fan et al., Shen et al.)	3D	D-C	O, S
Jericho † (Hausknecht et al.)	Text	D-C	D, R, S
JumpingFromPixels (Tachet et al.)	Arcade	Con	S
KitchenShift (Xing et al.)	3D, ConCon	D-C	O, S, R
Malmo (Johnson et al.)	3D, Arcade	D-C, D-O	R, S
MarsExplorer (Koutras et al.)	Grid	PCG	S
MazeExplore (Harries et al.)	3D	PCG	O, S
MDP Playground † (Rajan et al.)	ConCon, Grid	Con, D-C, D-O	D, O, R, S
Meta-World † (Yu et al.)	3D, ConCon	Con, D-C	R, S
MetaDrive (Li et al.)	3D, Driving	D-C, D-O, PCG	D, S
MiniGrid (Chevalier-Boisvert)	Grid	PCG	S
MiniHack (Samvelyan et al.)	Grid	D-C, D-O, PCG	S
NaturalEnvs CV (Zhang et al.)	Grid	PCG	O, R, S
NaturalEnvs MuJoCo (Zhang et al.)	ConCon	D-C	O
NLE (Küttler et al.)	Grid	PCG	S
Noisy MuJoCo (Zhao et al.)	ConCon	Con, D-C	D, O
NovelGridworlds (Goel et al.)	Grid	D-C	D, S
Obstacle Tower (Juliani et al.)	3D	D-C, PCG	O, S
OffRoadBenchmark (Dosovitskiy et al., Han et al.)	3D, Driving	D-C,	O, S, R
OpenAI Procgen (Cobbe et al.)	Arcade	PCG	O, S
OverParam Gym (Song et al.)	ConCon	Con	O
OverParam LQR (Song et al.)	LQR	Con	O
ParamGen (Ke et al.)	3D, LC	D-C, D-O	R, S
RLBench † (James et al.)	3D, ConCon, LC	Con, D-C, D-O	R, S
RoboSuite (Fan et al., Zhu et al.)	3D, ConCon	D-C	O
Rogue-gym (Kanagawa and Kaneko)	Grid	PCG	S
RTFM (Zhong et al.)	Grid, LC	PCG	D, R, S
RWRL † (Dulac-Arnold et al.)	ConCon	Con	D
Sokoban (Racanière et al.)	Grid	PCG	S
TextWorld † (Côté et al.)	Text	Con, D-C, PCG	D, O, R, S
Toybox † (Tosch et al.)	Arcade	Con, D-C, D-O	D, O, S
TrapTube (Wenke et al.)	Grid	Con, D-C	D, O, S
WordCraft (Jiang et al.)	LC, Text	Con, D-C, D-O	R, S
XLand (Team et al.)	3D, LC	Con, D-C, D-O, PCG	D, O, R, S
Phy-Q (Xue et al.)	Arcade	D-C, PCG	S

Table 1: Categorisation of Environments for ZSG. 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, D-C for discrete cardinal and D-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 benchmarks but could be adapted to be. See main text for a more detailed description of the columns.

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 Ke et al. (2021), 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 (Risi & Togelius, 2020), 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 for a 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 (Irpan & Song, 2019).

4.1.1 Trends In Environments

There are several trends and patterns shown in Table 1, which we draw the reader's attention to here. We describe 55 environments in total and have aimed to be fairly exhaustive.[3]

There are a range of different **Style**s that these environments have, which is beneficial as ZSG methods should themselves be generally applicable across styles if possible. While numerically there is a focus on gridworlds (14, 25%) and continuous control (13, 24%) there are well-established benchmarks for arcade styles (Cobbe et al., 2020a) and 3D environments (Juliani et al., 2019). Looking at **Context** sets, we see that PCG is heavily used in ZSG environments, featuring in 21 (38%) environments. Many environments combine PCG components with controllable variation (Chevalier-Boisvert et al., 2019; Côté et al., 2019; Juliani et al., 2019; Li et al., 2021b; Team et al., 2021; Xue et al., 2021; Fortunato et al., 2019; Eimer et al., 2021; Bapst et al., 2019). Most environments have several different kinds of factors of variation within their context set.

There are a lot of differences between environments when looking at the **Variation** they use. Numerically, state variation is most common (42, 76%) followed by observation (29, 53%), and then reward (20, 36%) and dynamics (19, 35%). Most environments have

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

multiple different types of variation (34, 62%), and while there are several environments targeted at just observation variation (10, 18%) or state variation (9, 16%), there is only a single environment with solely dynamics variation (RWRL, Dulac-Arnold et al., 2021), 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 parameterisable (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 Unchanged Optimality (Irpan & Song, 2019). 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 (Ahmed et al., 2020) enables easy testing of all forms of variation at once.[4]

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; Chevalier-Boisvert, 2021; Chevalier-Boisvert et al., 2019; Hafner, 2021; Kanagawa & Kaneko, 2019; Koutras et al., 2021; Küttler et al., 2020; Samvelyan et al., 2021), non-PCG observation-varying continuous control environments (RoboSuite, DMC-Remastered, DMC-GB, DCS, KitchenShift, NaturalEnvs MuJoCo; Fan et al., 2021; Grigsby & Qi, 2020; Hansen & Wang, 2021; Stone et al., 2021; Xing et al., 2021a; Zhang et al., 2018a), and multi-task continuous control benchmarks which could be adapted to ZSG (CausalWorld, RLBench, Meta-world; Ahmed et al., 2020; James et al., 2019a; Yu et al., 2019).

4.2 Evaluation Protocols For Zsg

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

[4] While CARL (Benjamins et al., 2021), MDP Playground (Rajan et al., 2021), XLand (Team et al., 2021) and TextWorld (Côté et al., 2019) are also categorised as containing all forms of variation, CARL is a collection of different environments, none of which have all variation types, XLand is not open-source, 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.

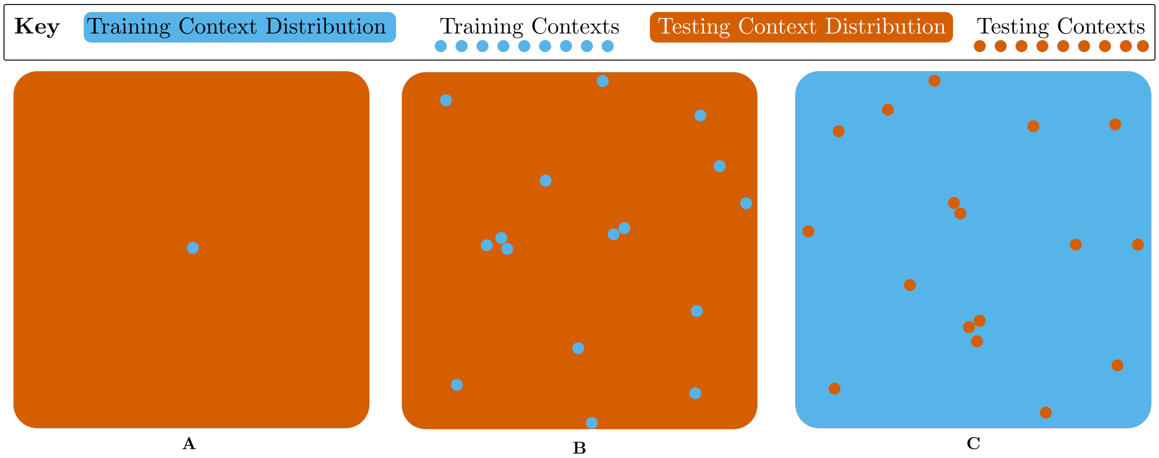

Figure 2: **Visualisation of Evaluation Protocols for PCG Environments**. **A** is a single training context, and the whole context set for testing. **B** uses a small collection of training contexts randomly sampled from the context set and the entire space for testing. **C** effectively reverses this, 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.

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).

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. 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 (Cobbe et al., 2020a), RogueGym (Kanagawa & Kaneko, 2019), some uses of JumpingFromPixels (Tachet et al., 2020) and MarsExplorer (Koutras et al., 2021).

Protocol **C** is commonly used among PCG environments that are not explicitly targeted at ZSG (MiniGrid, NLE, MiniHack, Alchemy; Chevalier-Boisvert, 2021; Küttler et al., 2020; Samvelyan et al., 2021; Wang et al., 2021). 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 zero-shot generalisation beyond avoiding memorising. While this protocol only tests for ZSG in a weak sense, it still matches a wider variety of real-world deployment scenarios than the previous standard in RL, where the evaluation of the policy is performed on the environment it is trained in, and so we believe it should be the standard evaluation protocol in RL (not just in ZSG), 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.

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

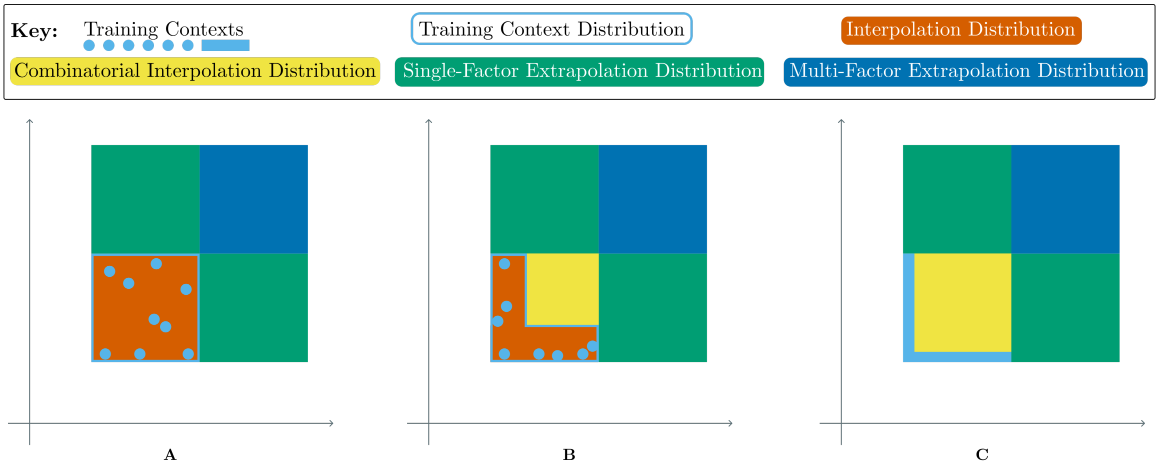
Key:
Training Contexts
Training Context Distribution
Interpolation Distribution
Combinatorial Interpolation Distribution
Single-Factor Extrapolation Distribution
Multi-Factor Extrapolation Distribution
A
B
C

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.

interpolation), or fully outside the convex hull (extrapolation). Combinatorial interpolation tests the ability of an agent to exhibit *systematicity*, a form of compositional ZSG 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 instances 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, ZSG 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 ZSG, 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 (Grigsby & Qi, 2020; Stone et al., 2021; Xing et al., 2021a). 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 ZSG benchmarks, which we raise here.

Non-visual Generalisation. If testing for non-visual types of generalisation, then visually simple domains such as MiniHack (Samvelyan et al., 2021) and NLE (Küttler et al., 2020) 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 ZSG benchmarks unto itself is the selection of DeepMind Control Suite (Tassa et al., 2018) variants: DMC-Remastered, DMC-Generalisation Benchmark, Distracting Control Suite, Natural Environments (Grigsby & Qi, 2020; Hansen & Wang, 2021; Stone et al., 2021; Zhang et al., 2018a). 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 (Stone et al., 2021) 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 ZSG benchmarks. For example, Tosch et al. (2019) presents three highly parameterisable 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 (Ahmed et al., 2020; Dulac-Arnold et al., 2021; James et al., 2019a; Wang et al., 2021; Yu et al., 2019)), 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 Zero-shot 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 zero-shot 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 ZSG and RL optimisation, but do not enable more targeted evaluation of specific types of ZSG. 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 (Juliani et al., 2019) has procedurally generated level layouts, but the visual features (and to some extent the layout complexity) can be controlled. Another example is MiniHack (Samvelyan et al., 2021), 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 ZSG.

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 (Hupkes et al., 2020): (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 environment instances 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 (Luketina et al., 2019), then the evaluations discussed by Hupkes et al. (2020) 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 difficult to validate in PCG environments like OpenAI Procgen (Cobbe et al., 2020a) or NLE (Küttler et al., 2020).

The other forms of compositional generalisation in (Hupkes et al., 2020) 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 ZSG 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 (Küttler et al., 2020) 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 ZSG 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 zero-shot 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 Ke et al. (2021), 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, zero-shot 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[5] such as transfer from related environments (Zhu et al., 2021); strong inductive biases (Zambaldi et al., 2019) or assumptions about the variation; or utilising online adaptation (Hansen et al., 2021a; Zintgraf et al., 2020) will be necessary to produce policies that generalise in this way.

5. Methods For Zero-shot Generalisation In Reinforcement Learning

We now classify methods that tackle ZSG in RL. The problem of ZSG occurs when the training and the testing context sets are not identical. There are many types of ZSG 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

[5] http://betr-rl.ml/2020/

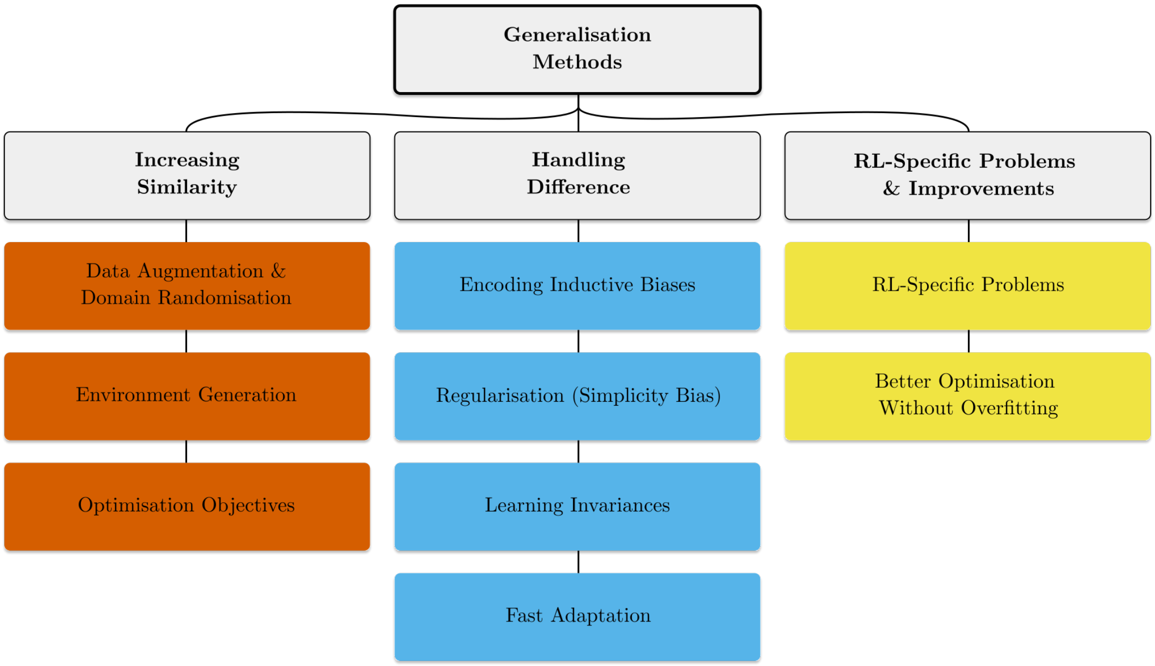
Generalisation Methods
Increasing Similarity
Handling Difference
RL-Specific Problems & Improvements
Data Augmentation & Domain Randomisation
Environment Generation
Optimisation Objectives
Encoding Inductive Biases
Regularisation (Simplicity Bias)
Learning Invariances
Fast Adaptation
RL-Specific Problems
Better Optimisation Without Overfitting

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

between training and testing environments (Section 5.2), and those that target RL-specific issues or optimisation improvements which aid ZSG performance (Section 5.3).

See Fig. 4 for a diagram of this categorisation, and Tables 2 and 3 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 ZSG 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.

5.1.1 Data Augmentation and Domain Randomisation

Two natural ways to make training and testing data more similar are data augmentation (Shorten & Khoshgoftaar, 2019) and domain randomisation (Tobin et al., 2017; Sadeghi & Levine, 2017; Peng et al., 2018). 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 the 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 5.2.1. 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 ZSG performance. Raileanu et al. (2021, UCB-DrAC) adapts the DA technique DrQ (Kostrikov et al., 2021) to an actor-critic setting (Schulman et al., 2017, PPO) and introduces a method for automatically picking the best augmentation during training. Wang et al. (2020, Mixreg) adapts mixup (Zhang et al., 2018) to the RL setting, which encourages the policy to be linear in its outputs with respect to mixtures of possible inputs. Zhang and Guo (2021, PAADA) use adversarial DA combined with mixup. Lee et al. (2020, RandFM) uses a randomised convolutional layer at the start of the network to improve robustness to a wide variety of visual inputs. Zhou et al. (2021, MixStyle) mixes style statistics across spatial dimensions in CNNs for increased data diversity. All these methods (Raileanu et al., 2021; Wang et al., 2020; Zhang & Guo, 2021; Lee et al., 2020; Zhou et al., 2021) show improved performance on CoinRun (Cobbe et al., 2019) or OpenAI Procgen (Cobbe et al., 2020a) 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 (2021, 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 (Hansen & Wang, 2021). James et al. (2019b, 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 (2021, 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. (2021, 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. (2021b, 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 (Hansen & Wang, 2021). 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 by 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. (2017), Sadeghi and Levine (2017) and Peng et al. (2018) introduced this idea in the setting of sim-to-real transfer in robotics. Much work has been done on different types of DR, so we cover just a sample here. OpenAI et al. (2019a) described 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. (2020, Minimax DSAC) use adversarial training to learn the DR for improved robustness. Zhao and Hospedales (2020, P2PDRL) improve DR though peer-to-peer distillation. Wellmer and Kwok (2021, 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 (Risi & Togelius, 2020) 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 possibly varies during training.

In both DA and DR approaches, as the training environment instances are increasingly augmented or randomised, optimisation becomes increasingly difficult, which often makes these methods much less sample-efficient. This is the motivation behind James et al.'s (2019b), Hansen and Wang's (2021), Fan et al.'s (2021) work 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).

5.1.2 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. (2019) introduced 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. (2020) built 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. (2020) 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 ZSG 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. (2021a) extended the formal framework of UED, combining it with Prioritized Level Replay (Jiang et al., 2021b, PLR), and motivates understanding PLR as an environment generation algorithm. This combined method shows improved performance in terms of zero-shot 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 (Portelas et al., 2020), 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 ZSG 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.

5.1.3 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 ZSG is PLR (Jiang et al., 2021b), in which the sampling distribution over levels is changed to increase the learning efficiency and zero-shot generalisation 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 ZSG 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 ZSG to unseen dynamics. Chen and Li (2020) gave an overview and introduction to the field. Abdullah et al. (2019, WR^{2}L) optimised the worst-case performance using a Wasserstein ball around the transition function to define the perturbation set. Mankowitz et al. (2020, SRE-MPO) incorporates RLL into MPO (Abdolmaleki et al., 2018) and shows improved performance on the RWRL benchmark (Dulac-Arnold et al., 2021). Pinto et al. (2017, 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 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.

5.2.1 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 to 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 (Raileanu & Fergus, 2021) 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 (Cobbe et al., 2020a), as timestep is irrelevant for the optimal policy but could be used to overfit to the training set of levels. Higgins et al. (2017b, DARLA) uses β-VAEs (Higgins et al., 2017a) to encode the inductive bias of disentanglement into the representations of the policy, improving zero-shot performance on various visual variations. Vlastelica et al. (2021, NAP) incorporates a black-box shortest-path solver to improve ZSG performance in hard navigation problems. Zambaldi et al. (2018, 2019) incorporate a relational inductive bias into the model architecture which aids in generalising along ordinal axes of variation, including extrapolation performance. Kansky et al. (2017, 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. (2021, 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. (2020) 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 ZSG. Their method generalises zero-shot to unseen backgrounds in CarRacing (Brockman et al., 2016) and VizDoom (Kempka et al., 2016; OpenAI, 2016). Tang and Ha (2021, SensoryNeuron) build on AttentionAgent, adding an inductive bias of permutation invariance in the input space. They argue this is useful for improving ZSG, 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. François-Lavet et al. (2019, CRAR) use a modular architecture combining dynamics learning and value estimation both in a low-dimensional latent space, and show improved performance on a simple maze task.

Hill et al. (2020b, SHIFTT) and Lynch and Sermanet (2021, TransferLanfLfP) both use large pretrained models (Devlin et al., 2019; Yang et al., 2020b) to encode natural language instructions for instruction following tasks, tackling reward function variation. They both show improved performance to novel instructions, leveraging the generalisation power of the large pretrained model. This can be seen as utilising domain knowledge to improve zero-shot generalisation to novel goal specifications by incorporating the inductive bias that all the goal specifications will be natural language.

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 that encourage 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 Battaglia et al. (2018), and for the original ideas surrounding inductive biases and No Free Lunch theorems see (Wolpert & Macready, 1997).

5.2.2 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. (2019) introduced CoinRun, and evaluated standard supervised learning regularisation techniques on the benchmark. They investigate data augmentation (a modified form of Cutout (DeVries & Taylor, 2017)), 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. (2021) shows that considering training iteration as a hyperparameter (effectively a form of early stopping) improves 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 (Tishby & Zaslavsky, 2015). Igl et al. (2019, IBAC-SNI) and Lu et al. (2020, 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. (2021, 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 MDP 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 zero-shot generalisation, although this method was not compared on standard ZSG benchmarks to other methods. Chen (2020, SMIRL) use surprise minimisation to improve the performance of the trained policy, although more rigorous benchmarking is needed to know whether this method has a positive effect.

Song et al. (2020) 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 (Cobbe et al., 2019, 2020a).

5.2.3 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 that 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. (2020, ICP) assume a block MDP structure (Du et al., 2019) and leverage that assumption to learn a state abstraction that is invariant to irrelevant features, which aids in generalisation performance. Zhang et al. (2021, 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. Kemertas and Aumentado-Armstrong (2021, DBC-normed-IR-ID) improve DBC with a norm on the representation space and the use of intrinsic rewards and regularisation. Agarwal et al. (2021, 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. (2020, IPO) apply ideas from Invariant Risk Minimization (Arjovsky et al., 2020) 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. (2020, IAPE) introduce the *Instance MDP*, an alternative formalism for the ZSG 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 (Cobbe et al., 2019) compared to standard regularisation techniques. Ghosh et al. (2021, 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 IAPE 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. (2020, 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 zero-shot generalisation and sample efficiency in several OpenAI Procgen games. Mazoure et al. (2021, 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. (2021a, DARL) uses adversarial learning to enforce the representations of different domains to be indistinguishable, improving performance in visually diverse testing contexts, even when only trained on simple training contexts. Fan and Li (2021, 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.

5.2.4 Adapting Online

A final way to handle differences between training and testing contexts is to adapt online to the testing contexts. Section 3.4 allows that the policy can adjust online within a single episode, as the policy class includes non-Markovian policies (which can equivalently be viewed as adaptation procedures for producing markovian policies). This adaption has to happen within a single episode and for it to be useful, the adaptation will have to be rapid enough to be useful for improved performance within that episode. Most work on Meta RL, which is traditionally concerned with fast adaptation, assumes access to multiple training episodes in the testing CMDP, violating the zero-shot assumption. However, there are works which can adapt zero-shot. While not being exhaustive, we give a brief overview of this body of work here, focusing on key examples where zero-shot generalisation is evaluated.

Many methods learn a context encoder or inference network, which then conditions either a policy or dynamics model for improved ZSG. The inference of this context at test-time can be seen as within-episode online adaptation. Yu et al. (2017, UP-OSI) uses Online System Identification to infer the context, which then conditions a policy. Yen-Chen et al. (2019, EVF) learns a context inference network end-to-end with a dynamics model and uses this to adapt to novel objects. Kumar et al. (2021, RMA) tackles the sim-to-real problem by training an agent using domain randomisation in simulation, and training a context inference model to condition the policy, similar to Yu et al.'s (2017) UP-OSI but with learned context inference. Ball et al. (2021, 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 all tackle dynamics variation in continuous control or visual tasks. Often, these methods make use of domain randomisation approaches but aim to have a context-conditioned adaptive policy or model, rather than a policy invariant to all possible contexts.

Some approaches use hard-coded adaptation techniques which do not rely on explicitly inferring a context. Seo et al. (2020, 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. Nagabandi et al. (2019b, MOLe) tackles the online learning problem, keeping a continually updated and expanded collection of dynamics models which are chosen between using non-parametric task inference. In both these works the chosen model plans with model-predictive control, and they show improvements in continuous control tasks. Hansen et al. (2021a, 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 (Lomonaco et al., 2020) (a VizDoom (Kempka et al., 2016) 3D navigation environment with visual variation), and on zero-shot generalisation to novel dynamics on both a real and simulated robot arm. (Nagabandi et al., 2019a, GrBAL,ReBAL) use meta-RL methods (MAML Finn et al. (2017) and RL^2 Duan et al. (2016)) to learn to quickly adapt a dynamics model using gradients or recurrence, and then use the adapted model to plan. In this case, it becomes more difficult to draw the line between learned and hard-coded update rules: The gradient update itself is hard-coded, but the initialisation is learned, and in the recurrent case the update is fully learned.

Finally, several methods meta-learn an adaptation function during training. RL^2 (Duan et al., 2016; Wang et al., 2017) 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. (2020) 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. Dorfman et al. (2021, BOReL) adjusts VariBAD to be usable in an offline setting, improving zero-shot exploration using offline data. Zintgraf et al. (2021, HyperX) improves VariBAD with additional exploration bonuses to improve meta-exploration. Ni et al. (2021) shows that these simple recurrent methods such as these, with carefully tuned implementations, can compete be improved further, often competing with more specialised algorithms, including in Robust RL and Meta-RL settings. Mishra et al. (2018, SNAIL) modelled fast adaptation as a sequence-to-sequence problem and learns an attention-based architecture which encodes sequences of experience to condition the policy. These methods all show improved performance on ZSG tasks, even though their main focus is on adaptation over longer time horizons.

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 zero-shot generalisation performance. In this section, we discuss methods targeted at these problems, and also discuss methods that improve ZSG purely through more effective optimisation on the training set in a way that does not overfit (at least empirically).

5.3.1 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. (2021, 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 zero-shot generalisation either positively or negatively, and this area deserves further attention if we are to go beyond techniques copied and adjusted from supervised learning.

5.3.2 Better Optimisation without Overfitting

Several works improve ZSG by improving the training performance without overfitting. Cobbe et al. (2020b) 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 (2021) 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 (2021, 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 ZSG benchmarks, with Anand et al. (2021) being an initial example. Anand et al. (2021) apply MuZero Reanalyse (Schrittwieser et al., 2020, 2021), a SOTA MBRL method, to OpenAI Procgen (Cobbe et al., 2020a), showing much-improved performance over SOTA model-free methods at much lower sample complexity. This shows the potential of using MBRL to improve zero-shot generalisation to varying states and observations. The authors also apply MuZero to the meta-learning tasks in Meta-World (Yu et al., 2019), although the performance there is not as impressive, showing that generalising to new rewards (as is necessary for Meta-World) is not currently as amenable to MBRL approaches.

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

5.4 Discussion

Having described existing methods for ZSG in RL, and categorised them in Tables 2 and 3, 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 ZSG 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 Tables 2 and 3, 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, there are hard ZSG 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 (Wolpert & Macready, 1997). 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.

Approach	Evaluation Variation				
	Observation	State	Dynamics	Reward	All
Data Augmentation	*SODA (Hansen and Wang)*, `RCAN (James et al.)`, *RandFM (Lee et al.)*, *InDA,ExDA (Ko and Ok)*, *DrQ (Kostrikov et al.)*, *SECANT (Fan et al.)*, *UCB-DrAC (Raileanu et al.)*, *PAADA (Zhang and Guo)*, *MixStyle (Zhou et al.)*, *SVAE (Hansen et al.)*	*UCB-DrAC (Raileanu et al.)*, *PAADA (Zhang and Guo)*			
Domain Randomisation		MD-SAC (Ren et al.)	P2PDRL (Zhao and Hospedales), DR (Tobin et al., Peng et al.), CAD^{2}RL (Sadeghi and Levine), ADR (OpenAI et al.), DDL (Wellmer and Kwok)		PCG (Risi and Togelius)
Environment Generation		POET (Wang et al.), PAIRED (Dennis et al.), E-POET (Wang et al.)			
Optimisation Objectives		*PLR (Jiang et al.)*	*WR^{2}L (Abdullah et al.)*, *RARL (Pinto et al.)*, *(S)(R)(E)-MPO(Mankowitz et al.)*		
Inductive Biases	`AttentionAgent (Tang et al.)`, `VAI (Wang et al.)`, `SensoryNeuron (Tang and Ha)`, *DARLA (Higgins et al.)*	`NAP (Vlastelica et al.)`, `RelationalRL (Zambaldi et al.)`, `SchemaNetworks (Kansky et al.)`, *IDAAC (Raileanu and Fergus)*, *CRAR (François-Lavet et al.)*	`RelationalRL (Zambaldi et al.)`	`TransferLangLfP (Lynch and Sermanet)`, `SHIFTT (Hill et al.)`	

Table 2: A table categorising all methods for tackling the ZSG problem in RL, part 1 of 2. 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 normal-text methods mainly work by adjusting the training environment, `Red monospace-text` methods mainly work through adjusting the architecture, and *Blue italic* 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.

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

Approach	Evaluation Variation				
	Observation	State	Dynamics	Reward	All
Regularisation	`Implicit Regularisation (Song et al.)`	*SMIRL (Chen), Mixreg (Wang et al.), IBAC-SNI (Igl et al.)*	*RPC (Eysenbach et al.), IB-annealing (Lu et al.)*		*L2,Dropout,etc. (Cobbe et al.), EarlyStopping (Ada et al.)*
Learning Invariances	*DRIBO (Fan and Li), DBC (Zhang et al.), DBC-normed-IR-ID (Kemertas and Aumentado-Armstrong), DARL (Li et al.), PSM (Agarwal et al.), MISA (Zhang et al.)*	*IAPE (Bertrán et al.), DRIBO (Fan and Li), CTRL (Mazoure et al.), CSSC (Liu et al.), PSM (Agarwal et al.), LEEP (Ghosh et al.), IPO (Sonar et al.)*	*IPO (Sonar et al.)*		
Fast Adaptation	`PAD (Hansen et al.)`	*EVF (Yen-Chen et al.)*, `SNAIL (Mishra et al.)`, *HyperX (Zintgraf et al.), VariBAD (Zintgraf et al.), RMA (Kumar et al.), RL^2 (Duan et al., Wang et al.)*	*EVF (Yen-Chen et al.)*, `SNAIL (Mishra et al.)`, *HyperX (Zintgraf et al.), VariBAD (Zintgraf et al.)*, `TW-MCL (Seo et al.)`, *UP-OSI (Yu et al.), BOReL (Dorfman et al.), GrBAL,ReBAL (Nagabandi et al.), RMA (Kumar et al.), RL^2 (Duan et al., Wang et al.)*, `MOLe (Nagabandi et al.), AugWM (Ball et al.), PAD (Hansen et al.)`	`SNAIL (Mishra et al.)`, *HyperX (Zintgraf et al.), BOReL (Dorfman et al.), VariBAD (Zintgraf et al.), RL^2 (Duan et al., Wang et al.)*	
RL-specific Problems	`ITER (Igl et al.)`	`ITER (Igl et al.)`			
Better Optimisation	`Sparse DVE (Singh and Zheng), PPG (Cobbe et al.), DAAC (Raileanu and Fergus), MuZero++ (Anand et al.)`	`Sparse DVE (Singh and Zheng), PPG (Cobbe et al.), DAAC (Raileanu and Fergus), MuZero++ (Anand et al.)`		`MuZero++ (Anand et al.)`	

Table 3: A table categorising all methods for tackling the ZSG problem in RL, part 2 of 2. 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 normal-text methods mainly work by adjusting the training environment, `Red monospace-text` methods mainly work through adjusting the architecture, and *Blue italic* 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.

distribution. We hope to see more work in the area of non-visual ZSG, 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 to 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 (Khetarpal et al., 2020, 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 Khetarpal et al. (2020) that there are not enough good benchmarks for CRL. Most benchmarks do not enable testing all the different attributes desired of CRL methods, although Powers et al.'s (2021) CORA is a good first step. New benchmarks for CRL would also be excellent as benchmarks for ZSG, 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 (Cobbe et al., 2020a). Hence, we recommend work on building new CRL benchmarks, to make progress on CRL and ZSG 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 (Eysenbach et al., 2019; Yang et al., 2020a). 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 in 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. (2020) introduced a novel problem setting and framework centred 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 ZSG research.

6.2 Real World Reinforcement Learning Generalisation

Dulac-Arnold et al. (2021) propose 9 properties that are characteristic of real-world RL.[6] In thinking about the current set of ZSG benchmarks, these properties are relevant in two ways. First, when applying our methods to the real world, we will have to tackle these problems. Hence, it would be beneficial if ZSG benchmarks had these properties, such that we can ensure that our ZSG methods work in real-world environments.

Second, several properties are particularly relevant for zero-shot 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 ZSG below.

Context Efficiency. Addressing (1), it is likely that the high cost of new samples will also mean a high cost of new environment instances 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 evaluate this property more efficiently 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 (Levine et al., 2020). 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 (Kumar et al., 2020). 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 ZSG: training policies offline

[6] 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

that then generalise to new contexts unseen in the offline dataset. Current offline RL benchmarks (Fu et al., 2021; Gulcehre et al., 2021) 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 ZSG 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 (Nair et al., 2021), but this does not tackle the ZSG 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 (see Section 6.4).

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 (Arora & Doshi, 2020; Ng & Russell, 2000, IRL) to learn a reward function from human demonstrations (Chen et al., 2021; Schmeckpeper et al., 2020; Sermanet et al., 2018; Sestini et al., 2020), 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 demonstrated by Osband et al., 2020) 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 ZSG are designed to enable this type of evaluation. This requires building environments with controllable context sets as well as PCG components (Section 4.3), as well as careful thought to create a 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 is 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 ZSG 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 ZSG 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 (Zhao et al., 2020), 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 (Team et al., 2021). 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). This variation is more difficult to tackle, and it is often difficult or impossible to verify The Principle Of Unchanged Optimality (Irpan & Song, 2019). 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 (Irpan & Song, 2019) 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. Hill et al.'s (2020b), Lynch and Sermanet's (2021) work provide good examples of investigating natural language as goal specification, utilising pretrained models to improve ZSG, and we look forward to seeing more work in this area.

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 Song et al.'s (2020) results, which describe the notion of observational overfitting as one cause of the generalisation gap in RL; Bengio et al. (2020) 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; Igl et al. (2021) studies transient non-stationarity in RL and shows that it negatively impacts RL generalisation; and Hill et al. (2020a) investigates what environmental factors affect generalisation in an instruction-following task, finding for example that an egocentric viewpoint improves generalisation, as does a richer observation space.

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 that purely PCG environments are unsuited for a study of generalisation in RL.

6.6 Future Work On Methods For Zero-shot Generalisation

In this subsection, we summarise directions for future work on methods for ZSG, 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 5.3.1 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 5.1.2 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 work 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 that require or enable 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 5.2.4, but much more research should be done.

There are several under-explored approaches that cut across the categorisation in this work. As shown in Tables 2 and 3, most methods focus on changing the loss function or algorithmic approach. Architectural changes informed by inductive biases are less well studied, with notable examples coming from the work of Cobbe et al. (2020b), Singh and Zheng (2021), Tang et al. (2020), Vlastelica et al. (2021), Zambaldi et al. (2019). 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 ZSG. Most methods surveyed here are model-free, with notable exceptions being the work of Ball et al. (2021), Kansky et al. (2017), Seo et al. (2020), Anand et al. (2021). 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. (2020) give an initial example. Anand et al. (2021) 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 ZSG 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 ZSG in RL, bringing together disparate threads of research together in a unified framework. We presented a categorisation of benchmarks for ZSG, splitting the taxonomy into environments and evaluation protocols, and we categorised existing methods for tackling the wide variety of ZSG 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 ZSG through benchmarks and metrics, and future work in these areas. The last two are focused on methods for tackling the ZSG 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).
- 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). This also enables a more multidimensional approach to evaluating generalisation performance (Section 6.3),

and specific experimentation aimed at improving our understanding of ZSG 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).
- 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).
- 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 ZSG, 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).
- There is much underexplored future work in developing new methods for improved ZSG, 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 zero-shot generalisation in RL, spurring further research in this area, and serve 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, Mikayel Samvelyan, Minqi Jiang, Nicklas Hansen, Roberta Raileanu, Yingchen Xi and Zhengyao Jiang for discussion and comments on drafts of this work. We also thank (alphabetically) André Biedenkapp, Chelsea Finn, Eliot Xing, Jessica Hamrick, Pablo Samuel Castro, Sirui Xie, Steve Hansen, Theresa Eimer, Vincent François-Lavet and Zhou Kaiyang for pointing out missing references or work for an updated version of this survey. Finally, we thank Frans Oliehoek and the other reviewers at JAIR for all their helpful criticism and advice, which resulted in a much-improved paper.

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.

Appendix A. Other Structural Assumptions on MDPs

Other forms of structured MDPs have been defined beyond the contextual MDP (Hallak et al., 2015) 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 $S := \{S_1, S_2, \dots, S_n\}$ (Boutilier et al., 2000; Strehl et al., 2007). We follow the notation and definitions used by Osband and Roy (2014). The transition function T has the following property:

Definition 6 (Factored transition functions)**.** *Given two states s, s' and action a in a factored MDP M, the transition function satisfies a conditional independence condition*

$$T(s'|s,a) = \prod_i P(s'_i|s,a), \tag{3}$$

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 (Schölkopf, 2019), 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

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

where the probability of each factor s_i only depends on its parent factors $\mathrm{PA}(s_i)$ 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 7 (Factored reward functions)**.** *Given two states s, s' and action a in a factored MDP M, the reward function satisfies a conditional independence condition*

$$\mathbb{E}\big[R(s,a)\big] = \sum_i \mathbb{E}\big[R_i(s_i,a)\big], \tag{5}$$

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 extension of the single context-MDP, where the combination of one or more of these factors can be represented 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, Hupkes et al., 2020): 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** (Mausam, 2003). 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 is 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 (Diuk et al., 2008) 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 way 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 (Du et al., 2019) 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.

References

Abdolmaleki, A., Springenberg, J. T., Tassa, Y., Munos, R., Heess, N., & Riedmiller, M. A. (2018). 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.

Abdullah, M. A., Ren, H., Ammar, H. B., Milenkovic, V., Luo, R., Zhang, M., & Wang, J. (2019). Wasserstein Robust Reinforcement Learning.. arXiv:1907.13196 [cs, stat]..

Ada, S. E., Ugur, E., & Akin, H. L. (2021). Generalization in Transfer Learning.. arXiv:1909.01331 [cs, stat]., Comment: 23 pages, 36 figures.

Agarwal, R., Machado, M. C., Castro, P. S., & Bellemare, M. G. (2021). Contrastive Behavioral Similarity Embeddings for Generalization in Reinforcement Learning.. arXiv:2101.05265 [cs, stat]., Comment: ICLR 2021 (Spotlight). Website: https://agarwl.github.io/pse.

Ahmed, O., Träuble, F., Goyal, A., Neitz, A., Bengio, Y., Schölkopf, B., Wüthrich, M., & Bauer, S. (2020). CausalWorld: A Robotic Manipulation Benchmark for Causal Structure and Transfer Learning.. arXiv:2010.04296 [cs, stat]., Comment: The first two authors contributed equally, the last two authors avised jointly.

Albrecht, S. V., & Stone, P. (2018). Autonomous agents modelling other agents: A comprehensive survey and open problems.. Artificial Intelligence..

Amin, S., Gomrokchi, M., Satija, H., van Hoof, H., & Precup, D. (2021). A Survey of Exploration Methods in Reinforcement Learning.. arXiv:2109.00157 [cs]..

Anand, A., Walker, J., Li, Y., Vértes, E., Schrittwieser, J., Ozair, S., Weber, T., & Hamrick, J. B. (2021). Procedural Generalization by Planning with Self-Supervised World Models.. arXiv:2111.01587 [cs]..

Arjovsky, M., Bottou, L., Gulrajani, I., & Lopez-Paz, D. (2020). Invariant Risk Minimization.. arXiv:1907.02893 [cs, stat]..

Arora, S., & Doshi, P. (2020). A Survey of Inverse Reinforcement Learning: Challenges, Methods and Progress.. arXiv:1806.06877 [cs, stat]..

Ball, P. J., Lu, C., Parker-Holder, J., & Roberts, S. (2021). Augmented World Models Facilitate Zero-Shot Dynamics Generalization From a Single Offline Environment.. arXiv:2104.05632 [cs]., Comment: Accepted @ ICML 2021; Spotlight @ ICLR 2021 "Self-Supervision for Reinforcement Learning Workshop".

Bapst, V., Sanchez-Gonzalez, A., Doersch, C., Stachenfeld, K. L., Kohli, P., Battaglia, P. W., & Hamrick, J. B. (2019). Structured agents for physical construction. In Chaudhuri, K., & Salakhutdinov, R. (Eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, Vol. 97 of *Proceedings of Machine Learning Research*, pp. 464–474. PMLR.

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Gulcehre, C., Song, F., Ballard, A., Gilmer, J., Dahl, G., Vaswani, A., Allen, K., Nash, C., Langston, V., Dyer, C., Heess, N., Wierstra, D., Kohli, P., Botvinick, M., Vinyals, O., Li, Y., & Pascanu, R. (2018). Relational inductive biases, deep learning, and graph networks.. arXiv:1806.01261 [cs, stat]..

Bellemare, M. G., Naddaf, Y., Veness, J., & Bowling, M. (2013). The Arcade Learning Environment: An Evaluation Platform for General Agents.. Journal of Artificial Intelligence Research..

Bengio, E., Pineau, J., & Precup, D. (2020). 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*, Vol. 119 of *Proceedings of Machine Learning Research*, pp. 767–777. PMLR.

Benjamins, C., Eimer, T., Schubert, F., Biedenkapp, A., Rosenhahn, B., Hutter, F., & Lindauer, M. (2021). CARL: A Benchmark for Contextual and Adaptive Reinforcement Learning..

Bertrán, M., Martínez, N., Phielipp, M., & Sapiro, G. (2020). Instance-based generalization in reinforcement learning. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., & Lin, H. (Eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Biedenkapp, A., Bozkurt, H. F., Eimer, T., Hutter, F., & Lindauer, M. (2020). Dynamic Algorithm Configuration: Foundation of a New Meta-Algorithmic Framework.. ECAI 2020..

Boutilier, C., Dearden, R., & Goldszmidt, M. (2000). Stochastic dynamic programming with factored representations.. Artificial Intelligence..

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). OpenAI Gym.. arXiv:1606.01540 [cs]..

Chen, A. S., Nair, S., & Finn, C. (2021). Learning Generalizable Robotic Reward Functions from "In-The-Wild" Human Videos.. arXiv:2103.16817 [cs]., Comment: https://sites.google.com/view/dvd-human-videos.

Chen, J. Z. (2020). Reinforcement Learning Generalization with Surprise Minimization.. arXiv:2004.12399 [cs]., Comment: Inductive biases, invariances and generalization in RL Workshop, ICML 2020.

Chen, S., & Li, Y. (2020). An Overview of Robust Reinforcement Learning. In *2020 IEEE International Conference on Networking, Sensing and Control (ICNSC)*, pp. 1–6.

Chen, V., Gupta, A., & Marino, K. (2021). Ask Your Humans: Using Human Instructions to Improve Generalization in Reinforcement Learning.. arXiv:2011.00517 [cs]..

Chevalier-Boisvert, M. (2021). Minimalistic Gridworld Environment (MiniGrid)..

Chevalier-Boisvert, M., Bahdanau, D., Lahlou, S., Willems, L., Saharia, C., Nguyen, T. H., & Bengio, Y. (2019). 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.

Cobbe, K., Hesse, C., Hilton, J., & Schulman, J. (2020a). 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*, Vol. 119 of *Proceedings of Machine Learning Research*, pp. 2048–2056. PMLR.

Cobbe, K., Hilton, J., Klimov, O., & Schulman, J. (2020b). Phasic Policy Gradient.. arXiv:2009.04416 [cs, stat]..

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

Côté, M.-A., Kádár, Á., Yuan, X., Kybartas, B., Barnes, T., Fine, E., Moore, J., Tao, R. Y., Hausknecht, M., Asri, L. E., Adada, M., Tay, W., & Trischler, A. (2019). TextWorld: A Learning Environment for Text-based Games.. arXiv:1806.11532 [cs, stat]., Comment: Presented at the Computer Games Workshop at IJCAI 2018, Stockholm.

Crosby, M., Beyret, B., Shanahan, M., Hernández-Orallo, J., Cheke, L., & Halina, M. (2020). The Animal-AI Testbed and Competition. In *Proceedings of the NeurIPS 2019 Competition and Demonstration Track*, pp. 164–176. PMLR.

Dennis, M., Jaques, N., Vinitsky, E., Bayen, A. M., Russell, S., Critch, A., & Levine, S. (2020). Emergent complexity and zero-shot transfer via unsupervised environment design. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., & Lin, H. (Eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual.*

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

DeVries, T., & Taylor, G. W. (2017). Improved Regularization of Convolutional Neural Networks with Cutout.. arXiv:1708.04552 [cs]..

Diuk, C., Cohen, A., & Littman, M. L. (2008). An object-oriented representation for efficient reinforcement learning. In Cohen, W. W., McCallum, A., & Roweis, S. T. (Eds.), *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, Vol. 307 of *ACM International Conference Proceeding Series*, pp. 240–247. ACM.

Dorfman, R., Shenfeld, I., & Tamar, A. (2021). Offline Meta Learning of Exploration.. arXiv:2008.02598 [cs, stat]..

Doshi-Velez, F., & Konidaris, G. D. (2016). Hidden parameter markov decision processes: A semiparametric regression approach for discovering latent task parametrizations. In Kambhampati, S. (Ed.), *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pp. 1432–1440. IJCAI/AAAI Press.

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., & Koltun, V. (2017). CARLA: An Open Urban Driving Simulator.. arXiv:1711.03938 [cs]., Comment: Published at the 1st Conference on Robot Learning (CoRL).

Du, S. S., Kakade, S. M., Wang, R., & Yang, L. F. (2020). 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.

Du, S. S., Krishnamurthy, A., Jiang, N., Agarwal, A., Dudík, M., & Langford, J. (2019). Provably efficient RL with rich observations via latent state decoding. In Chaudhuri, K., & Salakhutdinov, R. (Eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, Vol. 97 of *Proceedings of Machine Learning Research*, pp. 1665–1674. PMLR.

Duan, Y., Schulman, J., Chen, X., Bartlett, P. L., Sutskever, I., & Abbeel, P. (2016). Rl \$2̂\$: Fast reinforcement learning via slow reinforcement learning.. arXiv preprint arXiv:1611.02779..

Dulac-Arnold, G., Levine, N., Mankowitz, D. J., Li, J., Paduraru, C., Gowal, S., & Hester, T. (2021). An empirical investigation of the challenges of real-world reinforcement learning.. arXiv:2003.11881 [cs]., Comment: arXiv admin note: text overlap with arXiv:1904.12901.

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

Eimer, T., Biedenkapp, A., Reimer, M., Adriaensen, S., Hutter, F., & Lindauer, M. (2021). DACBench: A Benchmark Library for Dynamic Algorithm Configuration.. arXiv:2105.08541 [cs]., Comment: Accepted at IJCAI 2021.

Eysenbach, B., Gupta, A., Ibarz, J., & Levine, S. (2019). 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.

Eysenbach, B., Salakhutdinov, R., & Levine, S. (2021). Robust Predictable Control.. arXiv:2109.03214 [cs]., Comment: Project site with videos and code: https://ben-eysenbach.github.io/rpc.

Fan, J., & Li, W. (2021). DRIBO: Robust Deep Reinforcement Learning via Multi-View Information Bottleneck.. arXiv:2102.13268 [cs]., Comment: 27 pages.

Fan, L., Wang, G., Huang, D.-A., Yu, Z., Fei-Fei, L., Zhu, Y., & Anandkumar, A. (2021). SECANT: Self-Expert Cloning for Zero-Shot Generalization of Visual Policies.. arXiv:2106.09678 [cs]., Comment: ICML 2021. Website: https://linxifan.github.io/secant-site/.

Farebrother, J., Machado, M. C., & Bowling, M. (2020). Generalization and Regularization in DQN.. arXiv:1810.00123 [cs, stat]., Comment: Earlier versions of this work were presented both at the NeurIPS'18 Deep Reinforcement Learning Workshop and the 4th Multidisciplinary Conference on Reinforcement Learning and Decision Making (RLDM'19).

Filos, A., Tigkas, P., McAllister, R., Rhinehart, N., Levine, S., & Gal, Y. (2020). 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*, Vol. 119 of *Proceedings of Machine Learning Research*, pp. 3145–3153. PMLR.

Finn, C., Abbeel, P., & Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In Precup, D., & Teh, Y. W. (Eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, Vol. 70 of *Proceedings of Machine Learning Research*, pp. 1126–1135. PMLR.

Fortunato, M., Tan, M., Faulkner, R., Hansen, S., Badia, A. P., Buttimore, G., Deck, C., Leibo, J. Z., & Blundell, C. (2019). Generalization of reinforcement learners with working and episodic memory. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., & Garnett, R. (Eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 12448–12457.

François-Lavet, V., Bengio, Y., Precup, D., & Pineau, J. (2019). Combined reinforcement learning via abstract representations. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pp. 3582–3589. AAAI Press.

Fu, J., Kumar, A., Nachum, O., Tucker, G., & Levine, S. (2021). D4RL: Datasets for Deep Data-Driven Reinforcement Learning.. arXiv:2004.07219 [cs, stat]., Comment: Website available at https://sites.google.com/view/d4rl/home.

Gamrian, S., & Goldberg, Y. (2019). Transfer learning for related reinforcement learning tasks via image-to-image translation. In Chaudhuri, K., & Salakhutdinov, R. (Eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, Vol. 97 of *Proceedings of Machine Learning Research*, pp. 2063–2072. PMLR.

Ghosh, D., Rahme, J., Kumar, A., Zhang, A., Adams, R. P., & Levine, S. (2021). Why Generalization in RL is Difficult: Epistemic POMDPs and Implicit Partial Observability.. arXiv:2107.06277 [cs, stat]., Comment: First two authors contributed equally.

Goel, S., Tatiya, G., Scheutz, M., & Sinapov, J. (2021). NovelGridworlds: A benchmark environment for detecting and adapting to novelties in open worlds..

Grigsby, J., & Qi, Y. (2020). Measuring Visual Generalization in Continuous Control from Pixels.. arXiv:2010.06740 [cs]., Comment: A total of 20 pages, 8 pages as the main text.

Gulcehre, C., Wang, Z., Novikov, A., Paine, T. L., Colmenarejo, S. G., Zolna, K., Agarwal, R., Merel, J., Mankowitz, D., Paduraru, C., Dulac-Arnold, G., Li, J., Norouzi, M., Hoffman, M., Nachum, O., Tucker, G., Heess, N., & de Freitas, N. (2021). RL Unplugged: A Suite of Benchmarks for Offline Reinforcement Learning.. arXiv:2006.13888 [cs, stat]., Comment: NeurIPS paper. 21 pages including supplementary material, the github link for the datasets: https://github.com/deepmind/deepmind-research/rl_unplugged.

Hafner, D. (2021). Benchmarking the Spectrum of Agent Capabilities.. arXiv:2109.06780 [cs]., Comment: Website: https://danijar.com/crafter.

Hallak, A., Di Castro, D., & Mannor, S. (2015). Contextual Markov Decision Processes.. arXiv:1502.02259 [cs, stat]..

Han, I., Park, D.-H., & Kim, K.-J. (2021). A New Open-Source Off-Road Environment for Benchmark Generalization of Autonomous Driving.. IEEE Access..

Hansen, N., Jangir, R., Sun, Y., Alenyà, G., Abbeel, P., Efros, A. A., Pinto, L., & Wang, X. (2021a). Self-Supervised Policy Adaptation during Deployment.. arXiv:2007.04309 [cs, stat]., Comment: Website: https://nicklashansen.github.io/PAD/ Code: https://github.com/nicklashansen/policy-adaptation-during-deployment ICLR 2021.

Hansen, N., Su, H., & Wang, X. (2021b). Stabilizing Deep Q-Learning with ConvNets and Vision Transformers under Data Augmentation.. arXiv:2107.00644 [cs]., Comment: Code and videos are available at https://nicklashansen.github.io/SVEA.

Hansen, N., & Wang, X. (2021). Generalization in Reinforcement Learning by Soft Data Augmentation.. arXiv:2011.13389 [cs]., Comment: Website: https://nicklashansen.github.io/SODA/ Code: https://github.com/nicklashansen/dmcontrol-generalization-benchmark. Presented at International Conference on Robotics and Automation (ICRA) 2021.

Hao, B., Lattimore, T., Szepesvári, C., & Wang, M. (2021). Online sparse reinforcement learning. In Banerjee, A., & Fukumizu, K. (Eds.), *The 24th International Conference on Artificial Intelligence and Statistics, AISTATS 2021, April 13-15, 2021, Virtual Event*, Vol. 130 of *Proceedings of Machine Learning Research*, pp. 316–324. PMLR.

Harries, L., Lee, S., Rzepecki, J., Hofmann, K., & Devlin, S. (2019). MazeExplorer: A Customisable 3D Benchmark for Assessing Generalisation in Reinforcement Learning. In *2019 IEEE Conference on Games (CoG)*, pp. 1–4.

Harrison, J., Garg, A., Ivanovic, B., Zhu, Y., Savarese, S., Fei-Fei, L., & Pavone, M. (2017). ADAPT: Zero-Shot Adaptive Policy Transfer for Stochastic Dynamical Systems.. arXiv:1707.04674 [cs]., Comment: International Symposium on Robotics Research (ISRR), 2017.

Hausknecht, M. J., Ammanabrolu, P., Côté, M., & Yuan, X. (2020). 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*, pp. 7903–7910. AAAI Press.

Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., & Meger, D. (2018). Deep reinforcement learning that matters. In McIlraith, S. A., & Weinberger, K. Q. (Eds.), *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*, pp. 3207–3214. AAAI Press.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., & Lerchner, A. (2017a). 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.

Higgins, I., Pal, A., Rusu, A. A., Matthey, L., Burgess, C., Pritzel, A., Botvinick, M., Blundell, C., & Lerchner, A. (2017b). DARLA: improving zero-shot transfer in reinforcement learning. In Precup, D., & Teh, Y. W. (Eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, Vol. 70 of *Proceedings of Machine Learning Research*, pp. 1480–1490. PMLR.

Hill, F., Lampinen, A. K., Schneider, R., Clark, S., Botvinick, M., McClelland, J. L., & Santoro, A. (2020a). Environmental drivers of systematicity and generalization in a situated agent. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Hill, F., Mokra, S., Wong, N., & Harley, T. (2020b). Human Instruction-Following with Deep Reinforcement Learning via Transfer-Learning from Text.. arXiv:2005.09382 [cs]..

Hu, H., Lerer, A., Cui, B., Wu, D., Pineda, L., Brown, N., & Foerster, J. (2021). Off-Belief Learning.. arXiv:2103.04000 [cs]..

Hu, H., Lerer, A., Peysakhovich, A., & Foerster, J. N. (2020). "other-play" for zero-shot coordination. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, Vol. 119 of *Proceedings of Machine Learning Research*, pp. 4399–4410. PMLR.

Huang, B., Feng, F., Lu, C., Magliacane, S., & Zhang, K. (2021). AdaRL: What, Where, and How to Adapt in Transfer Reinforcement Learning.. arXiv:2107.02729 [cs, stat]..

Hupkes, D., Dankers, V., Mul, M., & Bruni, E. (2020). Compositionality decomposed: How do neural networks generalise?.. arXiv:1908.08351 [cs, stat]..

Igl, M., Ciosek, K., Li, Y., Tschiatschek, S., Zhang, C., Devlin, S., & Hofmann, K. (2019). Generalization in reinforcement learning with selective noise injection and information bottleneck. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., & Garnett, R. (Eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 13956–13968.

Igl, M., Farquhar, G., Luketina, J., Boehmer, W., & Whiteson, S. (2021). Transient Non-Stationarity and Generalisation in Deep Reinforcement Learning.. arXiv:2006.05826 [cs, stat]..

Irpan, A., & Song, X. (2019). The Principle of Unchanged Optimality in Reinforcement Learning Generalization.. arXiv:1906.00336 [cs, stat]., Comment: Published at ICML 2019 Workshop "Understanding and Improving Generalization in Deep Learning".

Jain, A., Szot, A., & Lim, J. J. (2020). 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*, Vol. 119 of *Proceedings of Machine Learning Research*, pp. 4661–4672. PMLR.

James, S., Ma, Z., Arrojo, D. R., & Davison, A. J. (2019a). RLBench: The Robot Learning Benchmark & Learning Environment.. arXiv:1909.12271 [cs]., Comment: Videos and code: https://sites.google.com/view/rlbench.

James, S., Wohlhart, P., Kalakrishnan, M., Kalashnikov, D., Irpan, A., Ibarz, J., Levine, S., Hadsell, R., & Bousmalis, K. (2019b). 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*, pp. 12627–12637. Computer Vision Foundation / IEEE.

Jiang, M., Dennis, M., Parker-Holder, J., Foerster, J., Grefenstette, E., & Rocktäschel, T. (2021a). Replay-Guided Adversarial Environment Design.. arXiv:2110.02439 [cs]., Comment: NeurIPS 2021.

Jiang, M., Grefenstette, E., & Rocktäschel, T. (2021b). Prioritized Level Replay.. arXiv:2010.03934 [cs]..

Jiang, M., Luketina, J., Nardelli, N., Minervini, P., Torr, P. H., Whiteson, S., & Rocktäschel, T. (2020). WordCraft: An environment for benchmarking commonsense agents. In *Workshop on Language in Reinforcement Learning (LaRel)*.

Johnson, M., Hofmann, K., Hutton, T., & Bignell, D. (2016). The malmo platform for artificial intelligence experimentation. In Kambhampati, S. (Ed.), *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pp. 4246–4247. IJCAI/AAAI Press.

Juliani, A., Khalifa, A., Berges, V., Harper, J., Teng, E., Henry, H., Crespi, A., Togelius, J., & Lange, D. (2019). Obstacle tower: A generalization challenge in vision, control, and planning. In Kraus, S. (Ed.), *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pp. 2684–2691. ijcai.org.

Kanagawa, Y., & Kaneko, T. (2019). Rogue-Gym: A New Challenge for Generalization in Reinforcement Learning.. arXiv:1904.08129 [cs, stat]., Comment: 8 pages, 14 figures, 4 tables, accepted to IEEE COG 2019.

Kansky, K., Silver, T., Mély, D. A., Eldawy, M., Lázaro-Gredilla, M., Lou, X., Dorfman, N., Sidor, S., Phoenix, D. S., & George, D. (2017). Schema networks: Zero-shot transfer with a generative causal model of intuitive physics. In Precup, D., & Teh, Y. W. (Eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, Vol. 70 of *Proceedings of Machine Learning Research*, pp. 1809–1818. PMLR.

Katz, G., Huang, D. A., Ibeling, D., Julian, K., Lazarus, C., Lim, R., Shah, P., Thakoor, S., Wu, H., Zeljić, A., Dill, D. L., Kochenderfer, M. J., & Barrett, C. (2019). The Marabou Framework for Verification and Analysis of Deep Neural Networks. In Dillig, I., & Tasiran, S. (Eds.), *Computer Aided Verification*, Lecture Notes in Computer Science, pp. 443–452, Cham. Springer International Publishing.

Ke, N. R., Dasgupta, I., Chiappa, S., Goyal, A., Weber, T., Mitrovic, J., Hill, F., Chan, S. C. Y., Mozer, M. C., Rezende, D. J., & Kohli, P. (2021). Parametric Generalization for Benchmarking Reinforcement Learning Algorithms..

Kemertas, M., & Aumentado-Armstrong, T. (2021). Towards Robust Bisimulation Metric Learning.. arXiv:2110.14096 [cs]., Comment: Accepted to NeurIPS 2021.

Kempka, M., Wydmuch, M., Runc, G., Toczek, J., & Jaśkowski, W. (2016). ViZDoom: A Doom-based AI research platform for visual reinforcement learning. In *IEEE Conference on Computational Intelligence and Games*, pp. 341–348, Santorini, Greece. IEEE. The best paper award.

Keysers, D., Schärli, N., Scales, N., Buisman, H., Furrer, D., Kashubin, S., Momchev, N., Sinopalnikov, D., Stafiniak, L., Tihon, T., Tsarkov, D., Wang, X., van Zee, M., & Bousquet, O. (2020). 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.

Khetarpal, K., Riemer, M., Rish, I., & Precup, D. (2020). Towards Continual Reinforcement Learning: A Review and Perspectives.. arXiv:2012.13490 [cs]., Comment: Preprint, 52 pages, 8 figures.

Ko, B., & Ok, J. (2021). Time Matters in Using Data Augmentation for Vision-based Deep Reinforcement Learning.. arXiv:2102.08581 [cs]..

Kostrikov, I., Yarats, D., & Fergus, R. (2021). Image Augmentation Is All You Need: Regularizing Deep Reinforcement Learning from Pixels.. arXiv:2004.13649 [cs, eess, stat]..

Koutras, D. I., Kapoutsis, A. C., Amanatiadis, A. A., & Kosmatopoulos, E. B. (2021). MarsExplorer: Exploration of Unknown Terrains via Deep Reinforcement Learning and Procedurally Generated Environments.. arXiv:2107.09996 [cs]..

Kumar, A., Fu, Z., Pathak, D., & Malik, J. (2021). RMA: Rapid Motor Adaptation for Legged Robots.. arXiv:2107.04034 [cs]., Comment: RSS 2021. Webpage at https://ashish-kmr.github.io/rma-legged-robots/.

Kumar, A., Zhou, A., Tucker, G., & Levine, S. (2020). Conservative q-learning for offline reinforcement learning. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., & Lin, H. (Eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Küttler, H., Nardelli, N., Miller, A. H., Raileanu, R., Selvatici, M., Grefenstette, E., & Rocktäschel, T. (2020). The nethack learning environment. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., & Lin, H. (Eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Lee, K., Lee, K., Shin, J., & Lee, H. (2020). 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.

Levine, S., Kumar, A., Tucker, G., & Fu, J. (2020). Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems.. arXiv:2005.01643 [cs, stat]..

Li, B., François-Lavet, V., Doan, T., & Pineau, J. (2021a). Domain Adversarial Reinforcement Learning.. arXiv:2102.07097 [cs]..

Li, Q., Peng, Z., Xue, Z., Zhang, Q., & Zhou, B. (2021b). MetaDrive: Composing Diverse Driving Scenarios for Generalizable Reinforcement Learning..

Liu, G. T., Cheng, P.-J., & Lin, G. (2020). Cross-State Self-Constraint for Feature Generalization in Deep Reinforcement Learning..

Lomonaco, V., Desai, K., Culurciello, E., & Maltoni, D. (2020). Continual Reinforcement Learning in 3D Non-stationary Environments.. arXiv:1905.10112 [cs, stat]., Comment: Accepted in the CLVision Workshop at CVPR2020: 13 pages, 4 figures, 5 tables.

Lu, X., Lee, K., Abbeel, P., & Tiomkin, S. (2020). Dynamics Generalization via Information Bottleneck in Deep Reinforcement Learning.. arXiv:2008.00614 [cs, stat]., Comment: 16 pages.

Luketina, J., Nardelli, N., Farquhar, G., Foerster, J. N., Andreas, J., Grefenstette, E., Whiteson, S., & Rocktäschel, T. (2019). A survey of reinforcement learning informed by natural language. In Kraus, S. (Ed.), *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pp. 6309–6317. ijcai.org.

Lynch, C., & Sermanet, P. (2021). Language Conditioned Imitation Learning over Unstructured Data.. arXiv:2005.07648 [cs]., Comment: Published at RSS 2021.

Machado, M. C., Bellemare, M. G., Talvitie, E., Veness, J., Hausknecht, M., & Bowling, M. (2017). Revisiting the Arcade Learning Environment: Evaluation Protocols and Open Problems for General Agents.. arXiv:1709.06009 [cs]..

Malik, D., Li, Y., & Ravikumar, P. (2021). When Is Generalizable Reinforcement Learning Tractable?.. arXiv:2101.00300 [cs, stat]., Comment: v2 extends results to function approximation setting.

Mankowitz, D. J., Levine, N., Jeong, R., Abdolmaleki, A., Springenberg, J. T., Shi, Y., Kay, J., Hester, T., Mann, T. A., & Riedmiller, M. A. (2020). 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.

Mausam, D. S. W. (2003). Solving relational MDPs with first-order machine learning. In *IN PROC. ICAPS WORKSHOP ON PLANNING UNDER UNCERTAINTY AND INCOMPLETE INFORMATION*. Citeseer.

Mazoure, B., Ahmed, A. M., MacAlpine, P., Hjelm, R. D., & Kolobov, A. (2021). Cross-Trajectory Representation Learning for Zero-Shot Generalization in RL.. arXiv:2106.02193 [cs]..

Milani, S., Topin, N., Veloso, M., & Fang, F. (2022). A Survey of Explainable Reinforcement Learning..

Mishra, N., Rohaninejad, M., Chen, X., & Abbeel, P. (2018). A simple neural attentive meta-learner. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Morimoto, J., & Doya, K. (2000). Robust reinforcement learning. In Leen, T. K., Dietterich, T. G., & Tresp, V. (Eds.), *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*, pp. 1061–1067. MIT Press.

Müller-Brockhausen, M., Preuss, M., & Plaat, A. (2021). Procedural Content Generation: Better Benchmarks for Transfer Reinforcement Learning.. arXiv:2105.14780 [cs]..

Nagabandi, A., Clavera, I., Liu, S., Fearing, R. S., Abbeel, P., Levine, S., & Finn, C. (2019a). Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Nagabandi, A., Finn, C., & Levine, S. (2019b). Deep online learning via meta-learning: Continual adaptation for model-based RL. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Nair, A., Gupta, A., Dalal, M., & Levine, S. (2021). AWAC: Accelerating Online Reinforcement Learning with Offline Datasets.. arXiv:2006.09359 [cs, stat]., Comment: 17 pages. Website: https://awacrl.github.io/.

Narvekar, S., Peng, B., Leonetti, M., Sinapov, J., Taylor, M. E., & Stone, P. (2020). Curriculum Learning for Reinforcement Learning Domains: A Framework and Survey.. arXiv:2003.04960 [cs, stat]..

Ng, A. Y., & Russell, S. J. (2000). Algorithms for inverse reinforcement learning. In Langley, P. (Ed.), *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000), Stanford University, Stanford, CA, USA, June 29 - July 2, 2000*, pp. 663–670. Morgan Kaufmann.

Ni, T., Eysenbach, B., & Salakhutdinov, R. (2021). Recurrent Model-Free RL is a Strong Baseline for Many POMDPs.. arXiv:2110.05038 [cs]..

OpenAI, Akkaya, I., Andrychowicz, M., Chociej, M., Litwin, M., McGrew, B., Petron, A., Paino, A., Plappert, M., Powell, G., Ribas, R., Schneider, J., Tezak, N., Tworek, J., Welinder, P., Weng, L., Yuan, Q., Zaremba, W., & Zhang, L. (2019a). Solving Rubik's Cube with a Robot Hand.. arXiv:1910.07113 [cs, stat]..

OpenAI, Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., Józefowicz, R., Gray, S., Olsson, C., Pachocki, J., Petrov, M., Pinto, H. P. d. O., Raiman, J., Salimans, T., Schlatter, J., Schneider, J., Sidor, S., Sutskever, I., Tang, J., Wolski, F., & Zhang, S. (2019b). Dota 2 with Large Scale Deep Reinforcement Learning.. arXiv:1912.06680 [cs, stat]..

OpenAI, O. (2016). OpenAI Gym: The DoomTakeCover-v0 environment. https://gym.openai.com/envs/DoomTakeCover-v0.

Osband, I., Doron, Y., Hessel, M., Aslanides, J., Sezener, E., Saraiva, A., McKinney, K., Lattimore, T., Szepesvári, C., Singh, S., Roy, B. V., Sutton, R. S., Silver, D., & van Hasselt, H. (2020). Behaviour suite for reinforcement learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Osband, I., & Roy, B. V. (2014). Near-optimal reinforcement learning in factored mdps. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., & Weinberger, K. Q. (Eds.), *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 604–612.

Packer, C., Gao, K., Kos, J., Krähenbühl, P., Koltun, V., & Song, D. (2019). Assessing Generalization in Deep Reinforcement Learning.. arXiv:1810.12282 [cs, stat]., Comment: 17 pages, 6 figures.

Peng, X. B., Andrychowicz, M., Zaremba, W., & Abbeel, P. (2018). Sim-to-Real Transfer of Robotic Control with Dynamics Randomization.. 2018 IEEE International Conference on Robotics and Automation (ICRA)..

Perez, C., Such, F., & Karaletsos, T. (2020). Generalized Hidden Parameter MDPs:Transferable Model-Based RL in a Handful of Trials.. Proceedings of the AAAI Conference on Artificial Intelligence..

Perez-Liebana, D., Liu, J., Khalifa, A., Gaina, R. D., Togelius, J., & Lucas, S. M. (2019). General Video Game AI: A Multi-Track Framework for Evaluating Agents, Games and Content Generation Algorithms.. arXiv:1802.10363 [cs]., Comment: 20 pages, 1 figure, accepted by IEEE ToG.

Pinto, L., Davidson, J., Sukthankar, R., & Gupta, A. (2017). Robust adversarial reinforcement learning. In Precup, D., & Teh, Y. W. (Eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, Vol. 70 of *Proceedings of Machine Learning Research*, pp. 2817–2826. PMLR.

Portelas, R., Colas, C., Weng, L., Hofmann, K., & Oudeyer, P. (2020). Automatic curriculum learning for deep RL: A short survey. In Bessiere, C. (Ed.), *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pp. 4819–4825. ijcai.org.

Powers, S., Xing, E., Kolve, E., Mottaghi, R., & Gupta, A. (2021). CORA: Benchmarks, Baselines, and Metrics as a Platform for Continual Reinforcement Learning Agents.. arXiv:2110.10067 [cs]., Comment: Repository available at https://github.com/AGI-Labs/continual_rl.

Racanière, S., Weber, T., Reichert, D. P., Buesing, L., Guez, A., Rezende, D. J., Badia, A. P., Vinyals, O., Heess, N., Li, Y., Pascanu, R., Battaglia, P. W., Hassabis, D., Silver, D., & Wierstra, D. (2017). Imagination-augmented agents for deep reinforcement learning. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., & Garnett, R. (Eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 5690–5701.

Raileanu, R., & Fergus, R. (2021). Decoupling Value and Policy for Generalization in Reinforcement Learning.. arXiv:2102.10330 [cs]..

Raileanu, R., Goldstein, M., Yarats, D., Kostrikov, I., & Fergus, R. (2021). Automatic Data Augmentation for Generalization in Deep Reinforcement Learning.. arXiv:2006.12862 [cs]..

Rajan, R., Diaz, J. L. B., Guttikonda, S., Ferreira, F., Biedenkapp, A., von Hartz, J. O., & Hutter, F. (2021). MDP Playground: A Design and Debug Testbed for Reinforcement Learning.. arXiv:1909.07750 [cs, stat]., Comment: NeurIPS 2021 Data and Benchmark Track submission (with slight formatting differences, most notably citation style).

Ren, Y., Duan, J., Li, S. E., Guan, Y., & Sun, Q. (2020). Improving Generalization of Reinforcement Learning with Minimax Distributional Soft Actor-Critic.. arXiv:2002.05502 [cs, stat]..

Risi, S., & Togelius, J. (2020). Increasing generality in machine learning through procedural content generation.. Nat Mach Intell..

Sadeghi, F., & Levine, S. (2017). CAD2RL: Real Single-Image Flight without a Single Real Image.. arXiv:1611.04201 [cs]., Comment: To appear at Robotics: Science and Systems Conference (R:SS), 2017. Supplementary video: https://www.youtube.com/watch?v=nXBWmzFrj5s.

Samvelyan, M., Kirk, R., Kurin, V., Parker-Holder, J., Jiang, M., Hambro, E., Petroni, F., Kuttler, H., Grefenstette, E., & Rocktäschel, T. (2021). 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)*.

Schmeckpeper, K., Rybkin, O., Daniilidis, K., Levine, S., & Finn, C. (2020). Reinforcement Learning with Videos: Combining Offline Observations with Interaction.. arXiv:2011.06507 [cs]..

Schölkopf, B. (2019). Causality for Machine Learning.. arXiv:1911.10500 [cs, stat]..

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., Lillicrap, T., & Silver, D. (2020). Mastering Atari, Go, chess and shogi by planning with a learned model.. Nature..

Schrittwieser, J., Hubert, T., Mandhane, A., Barekatain, M., Antonoglou, I., & Silver, D. (2021). Online and Offline Reinforcement Learning by Planning with a Learned Model.. arXiv:2104.06294 [cs]..

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal Policy Optimization Algorithms.. arXiv:1707.06347 [cs]..

Seo, Y., Lee, K., Clavera, I., Kurutach, T., Shin, J., & Abbeel, P. (2020). Trajectory-wise Multiple Choice Learning for Dynamics Generalization in Reinforcement Learning.. arXiv:2010.13303 [cs]., Comment: Accepted in NeurIPS2020. First two authors contributed equally, website: https://sites.google.com/view/trajectory-mcl code: https://github.com/younggyoseo/trajectory_mcl.

Sermanet, P., Lynch, C., Chebotar, Y., Hsu, J., Jang, E., Schaal, S., & Levine, S. (2018). Time-Contrastive Networks: Self-Supervised Learning from Video.. arXiv:1704.06888 [cs]..

Sestini, A., Kuhnle, A., & Bagdanov, A. D. (2020). Demonstration-efficient Inverse Reinforcement Learning in Procedurally Generated Environments.. arXiv:2012.02527 [cs]., Comment: Presented at the AAAI-21 Workshop on Reinforcement Learning in Games.

Shapley, L. S. (1953). Stochastic Games*.. Proceedings of the National Academy of Sciences..

Shen, B., Xia, F., Li, C., Martín-Martín, R., Fan, L., Wang, G., Pérez-D'Arpino, C., Buch, S., Srivastava, S., Tchapmi, L. P., Tchapmi, M. E., Vainio, K., Wong, J., Fei-Fei, L., & Savarese, S. (2021). iGibson 1.0: A Simulation Environment for Interactive Tasks in Large Realistic Scenes.. arXiv:2012.02924 [cs]..

Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on Image Data Augmentation for Deep Learning.. J Big Data..

Singh, J., & Zheng, L. (2021). Sparse Attention Guided Dynamic Value Estimation for Single-Task Multi-Scene Reinforcement Learning.. arXiv:2102.07266 [cs, stat]..

Sodhani, S., Meier, F., Pineau, J., & Zhang, A. (2022). Block Contextual MDPs for Continual Learning. In *Proceedings of The 4th Annual Learning for Dynamics and Control Conference*, pp. 608–623. PMLR.

Sonar, A., Pacelli, V., & Majumdar, A. (2020). Invariant Policy Optimization: Towards Stronger Generalization in Reinforcement Learning.. arXiv:2006.01096 [cs, stat]., Comment: 16 pages, 4 figures.

Song, X., Jiang, Y., Tu, S., Du, Y., & Neyshabur, B. (2020). Observational overfitting in reinforcement learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Stone, A., Ramirez, O., Konolige, K., & Jonschkowski, R. (2021). The Distracting Control Suite – A Challenging Benchmark for Reinforcement Learning from Pixels.. arXiv:2101.02722 [cs]., Comment: Code available at https://github.com/google-research/google-research/tree/master/distracting_control.

Strehl, A. L., Diuk, C., & Littman, M. L. (2007). Efficient structure learning in factored-state MDPs. In *Proceedings of the 22nd National Conference on Artificial Intelligence - Volume 1*, AAAI'07, pp. 645–650, Vancouver, British Columbia, Canada. AAAI Press.

Tachet, R., Bachman, P., & van Seijen, H. (2020). Learning Invariances for Policy Generalization.. arXiv:1809.02591 [cs, stat]., Comment: 7 pages, 1 figure.

Tang, Y., & Ha, D. (2021). The Sensory Neuron as a Transformer: Permutation-Invariant Neural Networks for Reinforcement Learning.. arXiv:2109.02869 [cs]..

Tang, Y., Nguyen, D., & Ha, D. (2020). Neuroevolution of Self-Interpretable Agents.. Proceedings of the 2020 Genetic and Evolutionary Computation Conference., Comment: To appear at the Genetic and Evolutionary Computation Conference (GECCO 2020) as a full paper.

Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., Casas, D. d. L., Budden, D., Abdolmaleki, A., Merel, J., Lefrancq, A., Lillicrap, T., & Riedmiller, M. (2018). DeepMind Control Suite.. arXiv:1801.00690 [cs]., Comment: 24 pages, 7 figures, 2 tables.

Team, O. E. L., Stooke, A., Mahajan, A., Barros, C., Deck, C., Bauer, J., Sygnowski, J., Trebacz, M., Jaderberg, M., Mathieu, M., McAleese, N., Bradley-Schmieg, N., Wong, N., Porcel, N., Raileanu, R., Hughes-Fitt, S., Dalibard, V., & Czarnecki, W. M. (2021). Open-Ended Learning Leads to Generally Capable Agents.. arXiv:2107.12808 [cs]..

Tishby, N., & Zaslavsky, N. (2015). Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pp. 1–5.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., & Abbeel, P. (2017). Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World.. arXiv:1703.06907 [cs]., Comment: 8 pages, 7 figures. Submitted to 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2017).

Tosch, E., Clary, K., Foley, J., & Jensen, D. (2019). Toybox: A Suite of Environments for Experimental Evaluation of Deep Reinforcement Learning.. arXiv:1905.02825 [cs, stat]..

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., Oh, J., Horgan, D., Kroiss, M., Danihelka, I., Huang, A., Sifre, L., Cai, T., Agapiou, J. P., Jaderberg, M., Vezhnevets, A. S., Leblond, R., Pohlen, T., Dalibard, V., Budden, D., Sulsky, Y., Molloy, J., Paine, T. L., Gulcehre, C., Wang, Z., Pfaff, T., Wu, Y., Ring, R., Yogatama, D., Wünsch, D., McKinney, K., Smith, O., Schaul, T., Lillicrap, T., Kavukcuoglu, K., Hassabis, D., Apps, C., & Silver, D. (2019). Grandmaster level in StarCraft II using multi-agent reinforcement learning.. Nature..

Vithayathil Varghese, N., & Mahmoud, Q. H. (2020). A survey of multi-task deep reinforcement learning.. Electronics..

Vlastelica, M., Rolínek, M., & Martius, G. (2021). Neuro-algorithmic Policies enable Fast Combinatorial Generalization.. arXiv:2102.07456 [cs]., Comment: 15 pages.

Wang, J. X., King, M., Porcel, N., Kurth-Nelson, Z., Zhu, T., Deck, C., Choy, P., Cassin, M., Reynolds, M., Song, F., Buttimore, G., Reichert, D. P., Rabinowitz, N., Matthey, L., Hassabis, D., Lerchner, A., & Botvinick, M. (2021). Alchemy: A structured task distribution for meta-reinforcement learning.. arXiv:2102.02926 [cs]., Comment: 16 pages, 9 figures.

Wang, J. X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J. Z., Munos, R., Blundell, C., Kumaran, D., & Botvinick, M. (2017). Learning to reinforcement learn.. arXiv:1611.05763 [cs, stat]., Comment: 17 pages, 7 figures, 1 table.

Wang, K., Kang, B., Shao, J., & Feng, J. (2020). Improving Generalization in Reinforcement Learning with Mixture Regularization.. arXiv:2010.10814 [cs, stat]., Comment: NeurIPS 2020.

Wang, R., Lehman, J., Clune, J., & Stanley, K. O. (2019). Paired Open-Ended Trailblazer (POET): Endlessly Generating Increasingly Complex and Diverse Learning Environments and Their Solutions.. arXiv:1901.01753 [cs]., Comment: 28 pages, 9 figures.

Wang, R., Lehman, J., Rawal, A., Zhi, J., Li, Y., Clune, J., & Stanley, K. O. (2020). 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*, Vol. 119 of *Proceedings of Machine Learning Research*, pp. 9940–9951. PMLR.

Wang, X., Lian, L., & Yu, S. X. (2021). Unsupervised Visual Attention and Invariance for Reinforcement Learning.. arXiv:2104.02921 [cs]., Comment: Accepted at CVPR 2021.

Wellmer, Z., & Kwok, J. T. (2021). Dropout's Dream Land: Generalization from Learned Simulators to Reality. In Oliver, N., Pérez-Cruz, F., Kramer, S., Read, J., & Lozano, J. A. (Eds.), *Machine Learning and Knowledge Discovery in Databases. Research Track*, Lecture Notes in Computer Science, pp. 255–270, Cham. Springer International Publishing.

Wenke, S., Saunders, D., Qiu, M., & Fleming, J. (2019). Reasoning and Generalization in RL: A Tool Use Perspective.. arXiv:1907.02050 [cs]., Comment: 13 pages, 5 figures.

Whiteson, S., Tanner, B., Taylor, M. E., & Stone, P. (2011). Protecting against evaluation overfitting in empirical reinforcement learning. In *2011 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pp. 120–127.

Wolpert, D., & Macready, W. (1997). No free lunch theorems for optimization.. IEEE Transactions on Evolutionary Computation..

Xie, S., Ma, X., Yu, P., Zhu, Y., Wu, Y. N., & Zhu, S.-C. (2021). HALMA: Humanlike Abstraction Learning Meets Affordance in Rapid Problem Solving.. arXiv:2102.11344 [cs]..

Xing, E., Gupta, A., Powers, S., & Dean, V. (2021a). Evaluating Generalization of Policy Learning Under Domain Shifts..

Xing, E., Gupta, A., Powers, S., & Dean, V. (2021b). KitchenShift: Evaluating Zero-Shot Generalization of Imitation-Based Policy Learning Under Domain Shifts. In *NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications*.

Xue, C., Pinto, V., Gamage, C., Nikonova, E., Zhang, P., & Renz, J. (2021). Phy-Q: A Benchmark for Physical Reasoning.. arXiv:2108.13696 [cs]., Comment: For the associated website, see https://github.com/phy-q/benchmark.

Yang, R., Xu, H., Wu, Y., & Wang, X. (2020a). Multi-Task Reinforcement Learning with Soft Modularization.. arXiv:2003.13661 [cs, stat]., Comment: Our project page: https://rchalyang.github.io/SoftModule.

Yang, Y., Cer, D., Ahmad, A., Guo, M., Law, J., Constant, N., Hernandez Abrego, G., Yuan, S., Tar, C., Sung, Y.-h., Strope, B., & Kurzweil, R. (2020b). Multilingual universal sentence encoder for semantic retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 87–94, Online. Association for Computational Linguistics.

Yen-Chen, L., Bauza, M., & Isola, P. (2019). Experience-Embedded Visual Foresight.. arXiv:1911.05071 [cs]., Comment: CoRL 2019. Project website: http://yenchenlin.me/evf/.

Yu, T., Quillen, D., He, Z., Julian, R., Hausman, K., Finn, C., & Levine, S. (2019). Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning.. arXiv preprint arXiv:1910.10897..

Yu, W., Tan, J., Liu, C. K., & Turk, G. (2017). Preparing for the Unknown: Learning a Universal Policy with Online System Identification.. arXiv:1702.02453 [cs]., Comment: Accepted as a conference paper at RSS 2017.

Zambaldi, V., Raposo, D., Santoro, A., Bapst, V., Li, Y., Babuschkin, I., Tuyls, K., Reichert, D., Lillicrap, T., Lockhart, E., Shanahan, M., Langston, V., Pascanu, R., Botvinick, M., Vinyals, O., & Battaglia, P. (2018). Relational Deep Reinforcement Learning.. arXiv:1806.01830 [cs, stat]..

Zambaldi, V. F., Raposo, D., Santoro, A., Bapst, V., Li, Y., Babuschkin, I., Tuyls, K., Reichert, D. P., Lillicrap, T. P., Lockhart, E., Shanahan, M., Langston, V., Pascanu, R., Botvinick, M., Vinyals, O., & Battaglia, P. W. (2019). 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.

Zhang, A., Ballas, N., & Pineau, J. (2018). A Dissection of Overfitting and Generalization in Continuous Reinforcement Learning.. arXiv:1806.07937 [cs, stat]., Comment: 20 pages, 16 figures.

Zhang, A., Lyle, C., Sodhani, S., Filos, A., Kwiatkowska, M., Pineau, J., Gal, Y., & Precup, D. (2020). Invariant causal prediction for block mdps. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, Vol. 119 of *Proceedings of Machine Learning Research*, pp. 11214–11224. PMLR.

Zhang, A., McAllister, R., Calandra, R., Gal, Y., & Levine, S. (2021). Learning Invariant Representations for Reinforcement Learning without Reconstruction.. arXiv:2006.10742 [cs, stat]., Comment: Accepted as an oral at ICLR 2021.

Zhang, A., Wu, Y., & Pineau, J. (2018a). Natural Environment Benchmarks for Reinforcement Learning.. arXiv:1811.06032 [cs, stat]., Comment: 12 figures.

Zhang, C., Vinyals, O., Munos, R., & Bengio, S. (2018b). A Study on Overfitting in Deep Reinforcement Learning.. arXiv:1804.06893 [cs, stat]..

Zhang, H., & Guo, Y. (2021). Generalization of Reinforcement Learning with Policy-Aware Adversarial Data Augmentation.. arXiv:2106.15587 [cs]..

Zhang, H., Cissé, M., Dauphin, Y. N., & Lopez-Paz, D. (2018). 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.

Zhao, C., & Hospedales, T. (2020). Robust Domain Randomised Reinforcement Learning through Peer-to-Peer Distillation.. arXiv:2012.04839 [cs]..

Zhao, C., Sigaud, O., Stulp, F., & Hospedales, T. M. (2019). Investigating Generalisation in Continuous Deep Reinforcement Learning.. arXiv:1902.07015 [cs, stat]..

Zhao, W., Queralta, J. P., & Westerlund, T. (2020). Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: A Survey. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 737–744.

Zhong, V., Rocktäschel, T., & Grefenstette, E. (2021). RTFM: Generalising to Novel Environment Dynamics via Reading.. arXiv:1910.08210 [cs]., Comment: ICLR 2020; 17 pages, 13 figures.

Zhou, K., Liu, Z., Qiao, Y., Xiang, T., & Loy, C. C. (2022). Domain Generalization: A Survey.. IEEE Trans. Pattern Anal. Mach. Intell.., Comment: IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2022.

Zhou, K., Yang, Y., Qiao, Y., & Xiang, T. (2021). Domain Generalization with MixStyle.. arXiv:2104.02008 [cs]., Comment: ICLR 2021; Code is available at https://github.com/KaiyangZhou/mixstyle-release.

Zhu, Y., Wong, J., Mandlekar, A., & Martín-Martín, R. (2020). Robosuite: A Modular Simulation Framework and Benchmark for Robot Learning.. arXiv:2009.12293 [cs]., Comment: For more information, please visit https://robosuite.ai.

Zhu, Z., Lin, K., & Zhou, J. (2021). Transfer Learning in Deep Reinforcement Learning: A Survey.. arXiv:2009.07888 [cs, stat]..

Zintgraf, L., Feng, L., Lu, C., Igl, M., Hartikainen, K., Hofmann, K., & Whiteson, S. (2021). Exploration in Approximate Hyper-State Space for Meta Reinforcement Learning.. arXiv:2010.01062 [cs, stat]., Comment: Published at the International Conference on Machine Learning (ICML) 2021.

Zintgraf, L. M., Shiarlis, K., Igl, M., Schulze, S., Gal, Y., Hofmann, K., & Whiteson, S. (2020). 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.