A Patterns Framework for Incorporating Structure in Reinforcement Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Reinforcement Learning (RL), empowered by Deep Neural Networks (DNNs) for function approximation, has achieved notable success in diverse applications. However, its applicability to real-world scenarios with complex dynamics, noisy signals, and large state and action spaces remains limited due to challenges in data efficiency, generalization, safety guarantees, and interpretability, among other factors. To overcome these challenges, one promising avenue is to incorporate additional structural information about the problem into the RL learning process. Various sub-fields of RL have proposed methods for incorporating such inductive biases. We amalgamate these diverse methodologies under a unified framework, shedding light on the role of structure in the learning problem, and classify these methods into distinct patterns of incorporating structure, that address different auxiliary objectives. By leveraging this comprehensive framework, we not only provide valuable insights into the challenges associated with integrating structure into RL but also lay the groundwork for a design pattern perspective on RL research. This novel perspective paves the way for future advancements and aids in the development of more effective and efficient RL algorithms that can better handle real-world scenarios.

1 Introduction

Most of the traditional research in Reinforcement Learning (RL) focuses on designing agents that learn to solve a sequential decision problem induced by the inherent dynamics of the task they aim to solve, e.g. the differential equations governing the cart pole task in the classic control suite [Brockman et al., 2016]. However, their performance significantly degrades when even small aspects of the environment change [Meng and Khushi, 2019, Lu et al., 2020]. Moreover, deploying RL agents for real-world learning-based optimization comes with additional problems like noisy dynamics, intractable and computationally expensive state and action spaces, and noisy reward signals.

Thus, research in RL has started to address these issues through methods that can generally be categorized on a spectrum of two dogmas [Mannor and Tamar, 2023]: (i) **Generalization:** RL pipelines developed to solve a broader class of problems where the agent is trained on a variety of different tasks and environments [Kirk et al., 2023, Benjamins et al., 2023]. (ii) **Deployability:** RL pipelines that are specifically engineered towards concrete real-world problems by incorporating additional aspects such as feature engineering, or computational budget optimization [Dulac-Arnold et al., 2020]. The intersection of generalization and deployability presents a class of problems where we need RL pipelines that can handle sufficient diversity in the task while incorporating issues of deployability. To foster research in this area, Mannor and Tamar [2023] argue for a design-pattern oriented approach, where RL pipelines can be abstracted into patterns that are specialized to specific kinds of problems, while robust to a certain level of changes to these problems.

Submitted to 16th European Workshop on Reinforcement Learning (EWRL 2023). Do not distribute.

The path to RL design patterns, however, is hindered by gaps in our understanding of the relationship between the design decisions for RL methods and the properties of environments that they might be suited for. While decisions like using state abstractions for high-dimensional spaces seem to be obvious, decisions like the use of relational neural architectures for problems are not so obvious to a designer. One way to add principle to this process is to understand the different ways of incorporating domain knowledge into the learning pipeline. One strong source of such knowledge is the structure present in the learning problem itself, including priors about the state and action spaces, the nature of the reward function, the dynamics of the environment, assumptions on policy representation, and the sequence of learning tasks.

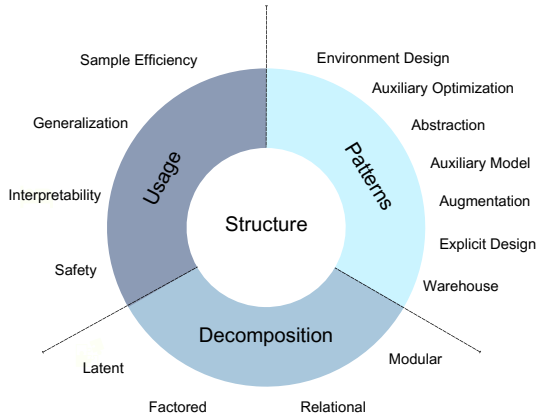


Figure 1: **Three major elements of understanding structure.** A learning problem can be decomposed using four archetypical styles (Section 3.2). Incorporating information about such decomposability can help a pipeline achieve auxiliary learning objectives, four of which are Sample Efficiency, Generalization, Interpretability, and Safety. The way such information is incorporated as an inductive bias follows one, or a combination of, seven different patterns (Section 4). Thus, understanding the interplay of decomposability, usage, and patterns provides a new perspective on RL literature that can help us take steps toward design patterns for RL.

Contributions and Structure of the Paper. Figure 1 shows a general overview of three elements that form our framework for understanding the role of structure in RL. In Section 2, we provide the background needed to formally define the problem presented in this paper. We then introduce side information and formulate structure as a particular kind of side information about decomposability in problem in Section 3. We additionally categorize decompositions in the literature into four major archetypes. In Section 4 we fomulate seven patterns of incorporating structure into the RL learning process and provide an overview of each pattern by connecting it to the relevant surveyed literature, further presented in Appendix A in detail. The framework developed in this work opens new avenues for research while providing a common reference point for understanding what kind of design decisions work under which situations. We discuss these aspects further in Section 5 for more concrete takeaways for both, researchers and practitioners.

2 Preliminaries

In the following sections, we summarize the main background necessary for our approach to studying structural decompositions and related patterns.

2.1 Markov Decision Processes

One way to formalize Sequential decision-making problems through a Markov Decision Process (MDP) [Bellman, 1954, Puterman, 2014] $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, R, P, \rho \rangle$. At any timestep, the environment exists in a state $s \in \mathcal{S}$, with ρ being the initial state distribution. The agent takes an action $a \in \mathcal{A}$ which *transitions* the environment to a new state $s' \in \mathcal{S}$. The transition function $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ governs the dynamics, taking the state s and action a as input and outputting a probability distribution over the next states $\Delta(.)$ from which the next state s' can be sampled. For each transition, the agent receives a reward $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, with $R \in \mathcal{R}$. The sequence (s, a, r, s') is called an experience.

A policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$, in a space of policies Π, generates these experiences, and a sequence of such experiences is also called a *trajectory* (τ). The rewards in τ can be accumulated into an expected sum called the return G, which can be calculated for any starting state s as $G(\pi, s) = \mathbb{E}_{(s_0=s, a_1, r_1, \dots) \sim \pi} \left[\sum_{t=0}^{\infty} r_t \right]$. To make the return sum tractable, we either assume the horizon of the problem to be of a fixed length T (finite-horizon return) i.e. the trajectory to terminate after T-steps, or we discount the future rewards by a discount factor γ (infinite horizon return). Solving an MDP amounts to determining the policy $\pi^* \in \Pi$ that maximizes the expectation over the returns of its

trajectory. This expectation can be captured by the notion of the (state-action) value function $Q \in \mathcal{Q}$. Given a policy π, the expectation can be written recursively:

$$Q^{\pi}(s,a) = \mathbb{E}_{s\sim\rho}\big[G_t|s,a\big] = \mathbb{E}_{s'\sim\mathcal{M}}\big[R(s,a) + \gamma\mathbb{E}_{a'\sim\pi(.|s')}[Q^{\pi}(s',a')]\big]. \tag{1}$$

Thus, the goal can now be formulated as the task of finding an optimal policy that can maximize the $Q^{\pi}(s,a)$:

$$\pi^* \in \arg\max_{\pi\in\Pi} Q^{\pi}(s,a). \tag{2}$$

2.2 Reinforcement Learning

The task of an RL algorithm is to interact with the MDP by simulating its transition dynamics $P(s'|s,a)$ and reward function $R(s,a)$ and learn the optimal policy mentioned in Equation (2). In Deep RL [Franccois-Lavet et al., 2018], the policy is a Deep Neural Network [Goodfellow et al., 2016] that is used to generate τ. The optimization of such a policy is done by minimizing an appropriate objective $J \in \mathcal{J}$.

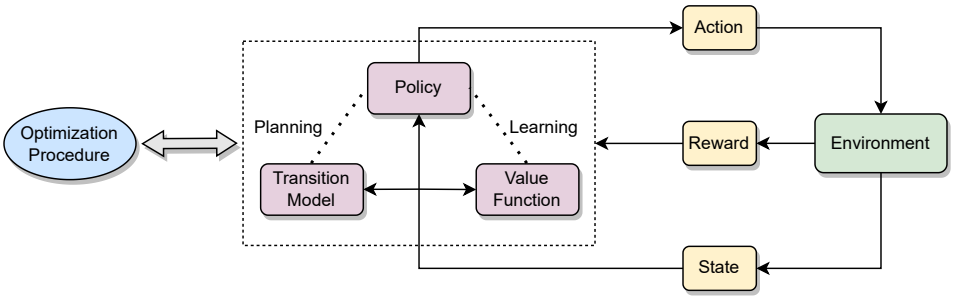


Figure 2: The anatomy of an RL pipeline.

We use the notion of a pipeline to talk about different RL methods, which can be defined as a mathematical tuple $\Omega = \langle \mathcal{S}, \mathcal{A}, R, P, Q, \pi, \hat{\mathcal{M}}, J\rangle$, where all definitions remain the same as before. Figure 2 shows the anatomy of such a pipeline. The pipeline operates on given states and action spaces, $\mathcal{S}, \mathcal{A}$ with dynamics P and a reward function R.

The optimization procedure encompasses the interplay between the current policy π, its value function Q, the reward R, and the learning objective J. With a slight abuse of notation, we refer to any of the components of a pipeline as X and assume the space in which it exists as $\mathcal{X}$s

The pipeline might generate experiences by directly interacting with an environment i.e. *learning* from experiences (*Model-Free RL*), or *plan* a trajectory by simulating a learned model $\hat{\mathcal{M}}$ of the environment to generate experiences (*Model-Based RL*). Learning can either utilize value functions, and correspondingly the TD error for J (*Value-based RL*), or parameterize the policy directly and use the Policy Gradient [Williams, 1992a, Sutton et al., 1999a] to create J (*Policy-Based RL*).

3 Structure as Side Information

In this section, we discuss the relationship between structure and decomposability. We first introduce side information in RL in Section 3.1, and consider structure to be a particular form of side information. We then discuss the major archetypes of decompositions through the spectrum of decomposability in Section 3.2.

3.1 Side Information

In addition to the characterization of the problem by an MDP, there can still be information on the table that could be potentially helpful. We call this *Side Information*. Jonschkowski et al. [2015] have previously defined side information for (semi-)supervised and unsupervised paradigms as any

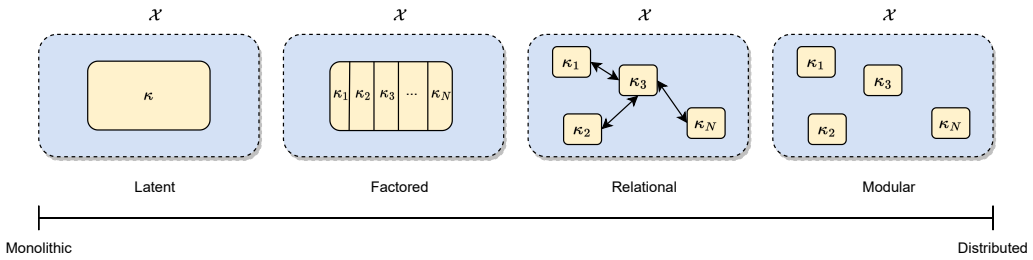


Figure 3: **Spectrum of Decomposability and Structural Archetypes.** On the left end of the spectrum exist monolithic structural decompositions where knowledge about a *latent* subspace of $\mathcal{X}$ can be learned and incorporated as an inductive bias. As we move towards the right, we see the possibility of learning multiple independent subspaces, albeit in a monolithic solution. These are *factored* decompositions. Further ahead, we see the emergence of interactionally complex decompositions, where knowledge about factorization and how they relate to each other can be incorporated into the learning process. We call these *relational decompositions*. Finally, we see fully distributed subsystems that can be incorporated and learned using individual policies. We call these *modular decompositions*.

additional information $z \in \mathcal{Z}$ that can contribute to the learning process but is not captured in the input of output spaces. Translated to the RL setting, side information can be understood as any additional information z not provided in the original MDP definition $\mathcal{M}$ but potentially helpful with additional objectives such as *Sample Efficiency*, *Generalization*, *Interpretability*, or *Safety*. This information can be incorporated into the learning process by biasing one, or more, of the components of Ω.

Structure is a particular kind of side information that captures knowledge about decomposability. Consider the task of managing a large factory with many production cells (example taken from Guestrin et al. [2003b]). If a cell positioned early in the production line generates faulty parts, the whole factory may be affected. However, the quality of the parts a cell generates depends directly only on the state of this cell and the quality of the parts it receives from neighboring cells. Additionally, the cost of running the factory depends, among other things, on the sum of the costs of maintaining each local cell. Finally, while a cell responsible for anodization may receive parts directly from any other cell in the factory, a work order for a cylindrical part may restrict this dependency to cells with a lathe. Thus, by incorporating information about the additive nature of production, costs, and the context of the part that needs to be produced, the learning pipeline can show better performance across the aforementioned objectives.

3.2 Decomposability and Structural Archetypes

Decomposability allows breaking a system down into smaller components or subsystems that can be independently analyzed and potentially learned more efficiently. [Hofer, 2017]. For the RL pipeline in Figure 2, decomposability can be seen along three axes: (i) *Problem Decomposition* i.e., the environment parameterization, states, actions, transitions, and rewards; (ii) *Solution Decomposition* i.e. the learned policies, value functions, and models; (iii) *Training Regime Decomposition* i.e. decomposition of a task into subtasks and their sequence. The spectrum of decomposability [Hofer, 2017] provides an intuitive way to understand where a system lies in this regard. On one end of the spectrum, there are problems that are non-decomposable, while on the other end, problems can be decomposed into weakly interacting sub-problems. Consequently, solutions on the former end are monolithic, and on the latter end, distributed. We capture this problem-solution interplay by marking four different archetypes of decomposability, as shown in Figure 3. The following sections will dive into further details regarding each of these archetypes.

Latent Decompositions are monolithic and can be useful in complex environments where the underlying structure is either not clear or non-stationary. Under this view, X can be approximated by a subspace κ, which can then be integrated into the learning process, leading to the learning process being re-conditioned on κ.

Latent states classically feature in the Latent MDP literature [Kwon et al., 2021], where the aim is to discover a latent representation of the state space that is sufficient to learn an optimal policy. Extensions such as Block MDPs [Du et al., 2019] and Contextual MDPs [Hallak et al., 2015] have shown success in generalization problems. Latent decompositions of transitions have been studied in

Linear MDPs [Papini et al., 2021] and corresponding applications in Model-based RL [Woo et al., 2022, van Rossum et al., 2021], where transition matrices are directly decomposed into an inner product of low-rank approximations. Latent rewards have been used in noisy reward settings, where the reward signal is assumed to be generated from a latent function [Wang et al., 2020].

Factored Decompositions decompose X into (latent) factors $\kappa_1, \ldots, \kappa_n$. Thus, the spaces become inner products of the individual factor spaces. A crucial aspect that differentiates factorization is that the factors can potentially impose conditional independence in their effects on the learning dynamics.

Factored states have been explored in the Factored MDPs Kearns and Koller [1999], Boutilier et al. [2000], Guestrin et al. [2003b], where the next state distribution is captured using a Dynamic Bayesian Network [Mihajlovic and Petkovic, 2001]. Factorization actions have helped tackle high-dimensional action spaces [Mahajan et al., 2021] by either factorizing subsets of high-dimensional action sets [Kim and Dean, 2002] or through factored Q-values used to produce actions [Tang et al., 2022a]. Factored rewards in conjunction with factored states induce factorization in Q-values [Koller and Parr, 1999, Sodhani et al., 2022a], and have additionally be used in multi-objective settings [Mambelli et al., 2022].

Relational Decompositions add more separability by capturing immutable relations between factors [Dzeroski et al., 2001]. The relational assumption posits that there exists a space of predicates that can ground these entities, and it can be modeled as a set of rules (such as inductive logic) that define how κ_i, κ_j interact with each other.

Classically, relational representations have been used to model state spaces in Relational MDPs [Dzeroski et al., 2001] and Object-Oriented MDPs [Guestrin et al., 2003a, Diuk et al., 2008], which use first-order representations of factored state spaces by representing states using objects, predicates, and functions to describe a set of ground MDPs. Traditional work in Relational MDPs has additionally used first-order representations of value functions and/or policies to generalize to new instances. These include Regression Trees [Mausam and Weld, 2003], Decision Lists [Fern et al., 2006], Algebraic Decision Diagrams [Joshi and Khardon, 2011], and Linear Basis Functions [Guestrin et al., 2003a, Sanner and Boutilier, 2012]. Recent approaches have started looking into DNN representations [Zambaldi et al., 2019, Garg et al., 2020], graph-based representations [Janisch et al., 2020, Sharma et al., 2022], or utilizing symbolic inductive biases [Garnelo et al., 2016]. Action relations help tackle large action sets through attention mechanisms [Jain et al., 2021b, Biza et al., 2022b] or action graphs [Wang et al., 2019]. Task perturbations have also been modeled as relational goals [Illanes et al., 2020, Kumar et al., 2022] or rewards [Sohn et al., 2018].

Modular Decompositions exist at the other end of the spectrum of decomposability, where individual value functions and/or policies can be learned for each decomposed entity. Specifically, a task can be broken down into individual subsystems $\kappa_1, \ldots, \kappa_N$ for which models, value functions, and policies can be subsequently learned. Such modularity can exist along two axes: (i) *Spatial Modularity* allows learning quantities specific to parts of the state space, thus, effectively reducing the dimensionality of the states, and (ii) *Temporal Modularity* allows breaking down tasks into sequences over a learning horizon and thus, reusing knowledge in a more continual fashion. A natural consequence of such breakdown is the emergence of a hierarchy, and when learning problems exploit this hierarchical relationship, then these problems come under the purview of Hierarchical RL (HRL) [Pateria et al., 2022]. However, hierarchy is not a necessity for Modularity.

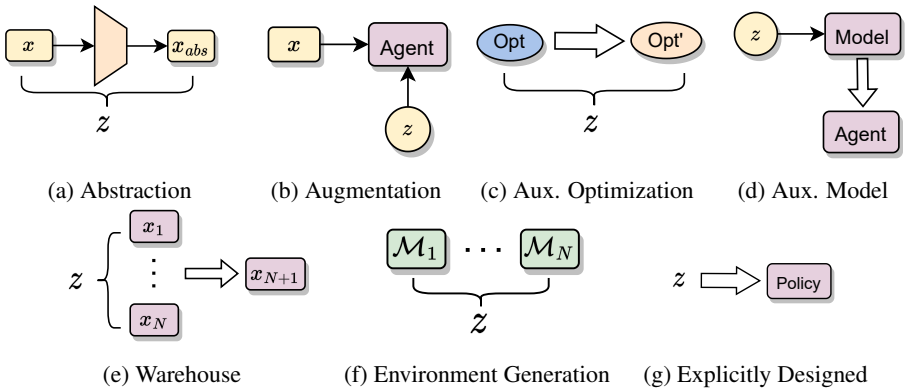
Modular decomposition of states is primarily studied at high-level planning and state abstractions for HRL methods [Kokel et al., 2021]. Additionally, work on skills has looked into the direction of training policies for individual parts of the state-space [Goyal et al., 2020]. Goals have been specifically considered in methods that either use goals as an interface between levels of hierarchy [Kulkarni et al., 2016, Nachum et al., 2018, Gehring et al., 2021], or as outputs of task specification methods [Jiang et al., 2019, Illanes et al., 2020].

Modularity in action spaces refers to conditioning policies on learned action abstraction. The classic example of such methods belongs to the realm of the Options framework [Sutton et al., 1999b]. In HRL methods, learning and planning of the higher levels are based on the lower-level policies and termination conditions of their execution.

Continual settings utilize policies compositionally by treating already learned policies as primitives [Eysenbach et al., 2019]. Such methods either feed these primitives to the discrete optimization problems for selection mechanisms or to continuous optimization settings involving ensembling [Goyal et al., 2020]. Modularity in such settings manifests itself by construction and serves as a central factor in building solutions. Even though the final policy in such paradigms can be monolithic, the method of obtaining such policies is a distributed regime.

4 Patterns of Incorporating Structure

To understand how information about decomposability can be incorporated into the RL pipeline, we survey the literature with a very specific question in mind: Do existing methods demonstrate repeatability in the way they use structure? The answer to this question, inspired by the categorization of Jonschkowski et al. [2015], brings us to *patterns of incorporating structure*. Please refer Appendix A for a detailed discussion on how individual works apply patterns for different objectives of our literature review.



(a) Abstraction (b) Augmentation (c) Aux. Optimization (d) Aux. Model

(e) Warehouse (f) Environment Generation (g) Explicitly Designed

Figure 4: **Patterns of incorporating structural information.** We categorize the methods of incorporating structure as inductive biases into the learning pipeline into patterns that can be applied for different kinds of usages.

A pattern is a principled change in the RL pipeline Ω that allows the pipeline to achieve one, or a combination of, the auxiliary objectives: *Sample Efficiency*, *Generalization*, *Safety*, and *Interpretability*. We categorize the literature into seven patterns, an overview of which has been shown in Figure 4, and explain them further in the following sections. To develop intuition about this categorization, let's consider the running example of a taxi service, where the task of the RL agent (the taxi) is to pick up passengers from various locations and drop them at their desired destinations within a city grid. The agent receives a positive reward when a passenger is successfully dropped off at their destination, and incurs a small penalty for each time step to encourage efficiency.

Abstraction Pattern utilizes structural information to create abstract entities in the RL pipeline. For any entity, X, an abstraction utilizes the structural information to create X_{abs}, which takes over the role of X in the optimization procedure. In the taxi example, the state space can be abstracted to the current grid cell of the taxi, the destination grid cell of the current passenger, and whether the taxi is currently carrying a passenger. This significantly simplifies the state space compared to representing the full details of the city grid. The action space could also be abstracted to moving in the four cardinal directions plus picking up and dropping off a passenger. Finding appropriate abstractions can be a challenging task in itself. Too much abstraction can lead to loss of critical information, while too little might not significantly reduce complexity. Consequently, learning-based methods that jointly learn abstractions factor this granularity into the learning process.

Abstraction have have been thoroughly explored in the literature with early work addressing a formal theory on state abstractions [Li et al., 2006]. Recent works have primarily used abstractions for tackling generalization, which also peaks in Figure 5.

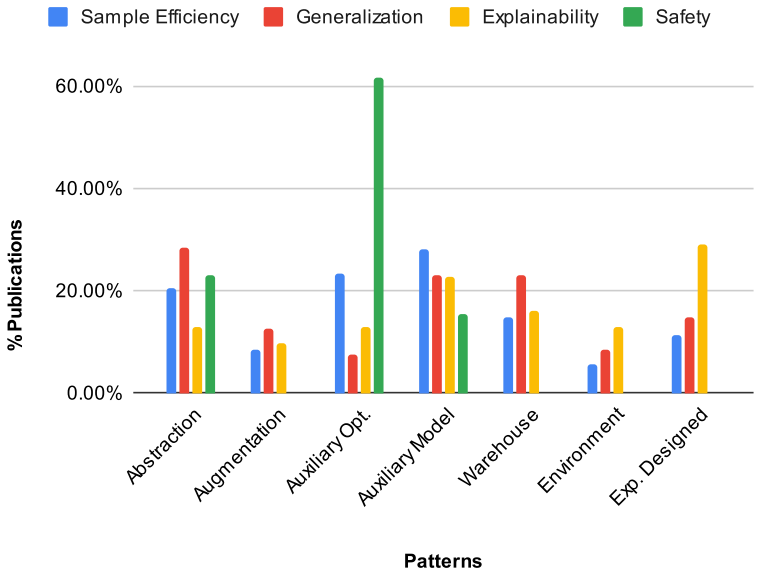


Figure 5: A meta-analysis of the proclivities of each pattern to auxiliary objectives.

Augmentation Pattern treats X and z as separate input entities, the combination of which can range from the simple concatenation of additional information to more involved methods of conditioning policy and/or value functions on additional information. Crucially, the structural information neither directly influences the optimization procedure nor changes the nature of X. For the taxi example, one way to achieve this would be conditioning the policy on additional information like the time of day or day of the week. This information could be useful because traffic conditions and passenger demands can vary depending on these factors. However, augmentations can increase the complexity of the policy, and care needs to be taken to ensure that the policy does not overfit to the additional information. Due to this, this pattern is generally not explored to its fullest extent. While the usage of augmentations is equitable for most use cases, the number of methods utilizing this pattern still falls short in comparison to more established techniques, such as abstraction.

Auxiliary Optimization Pattern uses structural biases to modify the optimization procedure. This includes methods involving contrastive losses, reward shaping, concurrent optimization, masking strategies, regularization, baselining, etc. Given that the changes in the optimization can go hand-in-hand with modifications of other components, many methods utilize this pattern in conjunction with other patterns discussed in this section (E.g. contrastive losses to learn state abstractions). In the case of the taxi example, reward shaping could help the policy to be reused for slight perturbances in the city grid, where the shaped reward encourages the taxi to stay near areas where passengers are frequently found when it doesn't have a passenger. It is crucial to ensure that the modified optimization process remains aligned with the original objective i.e. there needs to exist some form of regularization that controls how the modification of the optimization procedure respects the original objective. For reward shaping techniques, this amounts to the invariance of the optimal policy under the shaped reward [Ng et al., 1999]. For auxiliary objectives, this manifests in some form of entropy [Fox et al., 2016, Haarnoja et al., 2018c] or divergence regularization [Eysenbach et al., 2019]. Constraints ensure this through recursion [Lee et al., 2022], while baselines control the variance of updates [Wu et al., 2018]. The strongest usage of constraints is in the safety literature, where constraints either help control the updates using some form of safety criterion or for constraining the exploration. Consequently, the auxiliary optimization pattern peaks in its proclivity towards addressing safety.

Auxiliary Model Pattern captures structural decomposition in learned Model(s), that can subsequently be used to generate experiences, either fully or partially. Our taxi agent could learn a latent model of city traffic based on past experiences. This model could be used to plan routes that avoid traffic and hence reach destinations faster. Alternatively, the agent could learn an ensembling technique to combine multiple models each of which model-specific components of the traffic dynamics. With models, there is usually a trade-off between model complexity and accuracy, and it's essential to

manage this carefully to avoid overfitting and maintain robustness. To this end, incorporating structure helps in making the model-learning phase more efficient, while allowing reuse for generalization. Hence, the Auxiliary Model pattern shows a strong proclivity to utilize structural biases for sample efficiency.

Warehouse Pattern uses structural decomposition to create a database of entities in the solution space, such as value functions, policies, or models. The inherent modularity in such methods leads them to focus on knowledge reuse as a central theme, and their online nature often overlaps with continual settings. The taxi from our running example could maintain a database of value functions or policies for different parts of the city or at different times of the day. These could be reused as the taxi navigates through the city, making the learning process more efficient. While warehousing can generally improve efficiency, it has primarily been explored through the skills and options framework for targeting generalization. An important consideration in warehousing is to manage the size and diversity of the warehouse so as to avoid biasing the learning process too much toward past experiences.

So far, the warehousing pattern seems to be applied to sample efficiency and generalization. However, warehousing additionally overlaps with interpretability since the stored data can be easily utilized to analyze the agent's behavior and understand the policy for novel scenarios. Consequently, these objectives are equitably distributed for warehousing techniques.

Environment Generation Pattern uses structure to create task, goal, or dynamics distributions from which MDPs can be sampled. These settings can also reflect decomposition along the training regime by addressing curriculum learning methods Narvekar et al. [2020]. In the taxi example, a curriculum of tasks could be generated starting with simple tasks (like navigating an empty grid) and gradually introducing complexity (like adding traffic and passengers with different destinations). It is crucial, though, to ensure that the generated MDPs provide good coverage of the problem space to avoid overfitting to a specific subset of tasks. This necessitates additional diversity constraints that need to be incorporated into the environment generation process. Structure, crucially, provides additional interpretability and controllability in the environment generation process, thus, making benchmarking easier than methods that use unsupervised techniques [Laskin et al., 2021].

Explicitly Designed Pattern encompasses all methods where the inductive biases manifest in specific architectures or setups that reflect the decomposability of the problem that they aim to utilize. Naturally, this includes highly specific neural architectures, but it also easily extends to other methods like sequential architectures to capture hierarchies or relations. Crucially, the usage of structural information is limited to the specificity of the architecture and not any other part of the pipeline. In the case of the taxi, a neural architecture could be designed to process the city grid as an image and output a policy. Techniques like convolutional layers could be used to capture the spatial structure of the city grid. Different parts of the network could be specialized for different subtasks, like identifying passenger locations and planning routes. However, this pattern involves a considerable amount of manual tuning and experimentation, and the generalization of these designs across different tasks is not trivial.

5 Open Problems in Structured Reinforcement Learning

We explore various open areas of RL research and discuss ways that the structural patterns we have introduced can be applied to these settings.

Offline RL Offline RL methods need to tackle distributional shifts as they extract the most out of available passive data. Modular task decompositions combined with warehousing can help learn individual policies or value functions for different subtasks. They can additionally maximize the utility of available offline datasets when combined with abstractions. Factored decompositions combined with attention mechanisms can help agents direct more focus on factors that are less prone to distributional shifts while learning and help create more robust RL methods. Relational decompositions could help define auxiliary tasks that involve predicting the relationships between different entities, which could then help in learning useful relational representations of data that are more interpretable.

Partial observability and non-Markovian models Non-Markovian models can be appropriate in situations where the environment has either temporal dependencies in state or reward, whereas POMDPs help when the agent cannot fully observe the state. Temporal abstractions such as options or state-irrelevance abstractions can create an auxiliary MDP in the domain of options or abstract states. Structural decompositions can make such methods more sample efficient, such as by simplifying the observation space, or reducing the complexity of the belief update process, for POMDPs. Any additional information such as belief states and memory of past observations can be used for augmentations. Decompositions can additionally support learning transition models for planning more efficiently, while warehousing can improve methods in HRL that operate at different levels of abstraction. Curricula that start with simpler MDPs and gradually introduce partial observability or other non-Markovian features can create structured training regimes.

Big worlds can benefit from modular decompositions where the agent can explore different modules independently, which would be more efficient than exhaustive or random exploration of the entire environment. Knowledge of structure in these environments can be utilized to learn to generalize better across different parts of the environment. For instance, in a relational decomposition, the agent could learn relationships between different entities, which could help it generalize to unseen parts of the environment. Auxiliary optimization could help the agent learn faster by optimizing auxiliary tasks that are easier to learn or that provide useful information about the environment's structure.

AutoRL methods can utilize decompositions for their search processes. Algorithm selection methods can utilize the suitability of RL methods for specific decomposability offered by environments for the selection and ranking of algorithms for a given task. This can also help prefilter algorithms for further processes, such as hyperparameter optimization. Parameters related to structural decomposition (e.g., the number of subtasks in a modular decomposition) could be part of the hyperparameter optimization process in AutoRL. Investigating the effects of various structural decomposition-related parameters on the learning process could lead to novel insights and methods for more effective hyperparameter optimization in AutoRL.

Meta-RL Rather than only leveraging structure that exists in the dynamics or reward of a task, structure can also exist across tasks. Meta-RL methods can benefit from knowledge about meta-task decomposition by guiding the design of the meta-learning process catered to specific decompositions. This can additionally help make methods more generalizable. The adaptation strategy can itself be catered to the kind of decomposition an environment offers. For example, highly decomposable tasks can benefit from a modular adaptation strategy.

Foundation Models in RL can utilize decompositions in the fine-tuning phase, where methods could leverage this information for designing methods specific to individual use cases. Decompositions can additionally improve our understanding of the performance of the pre-trained model by benchmarking its performance against specific aspects of the environment. Consequently, by understanding the spectrum of decomposability and how various methods incorporate structure, we can create better benchmarks and evaluation protocols for foundational models. For example, warehousing and fine-tuning can help benchmark interpretable versions of specific aspects of foundation models.

6 Conclusion and Future Work

We introduce a novel framework of different patterns for incorporating the structure of a learning problem as an inductive bias into RL algorithms. We first ground structure as side information about decomposability in a learning problem and potential solutions. By categorizing decomposability into four archetypes along a spectrum, we establish connections with existing literature, shedding light on the diverse ways in which structure influences RL. Through a meticulous analysis of the RL landscape, we identify seven patterns that serve as robust pathways for integrating structural knowledge. Our research concludes with a pattern-centric lens, revealing the vital role of structural decompositions in present and future RL paradigms. We aim to inspire researchers and practitioners to embrace this perspective, fostering advancements and innovation in the field of RL. By presenting this comprehensive framework, we provide a valuable resource for researchers, facilitating further exploration and investigation into the incorporation of structure in RL.

References

M. Abdulhai, D. Kim, M. Riemer, M. Liu, G. Tesauro, and J. How. Context-specific representation abstraction for deep option learning. In Sycara et al. [2022].

D. Adjodah, T. Klinger, and J. Joseph. Symbolic relation networks for reinforcement learning. In *Proceedings of the Workshop on Relational Representation Learning in Conference on Neural Information Processing Systems (NeurIPS)*, 2018.

A. Alabdulkarim and M. Riedl. Experiential explanations for reinforcement learning. *CoRR*, abs/2210.04723, 2022.

C. Allen, N. Parikh, O. Gottesman, and G. Konidaris. Learning markov state abstractions for deep reinforcement learning. In Ranzato et al. [2021].

S. Amin, M. Gomrokchi, H. Aboutalebi, H. Satija, and D. Precup. Locally persistent exploration in continuous control tasks with sparse rewards. In Meila and Zhang [2021].

G. Andersen and G. Konidaris. Active exploration for learning symbolic representations. In Guyon et al. [2017].

J. Andreas, D. Klein, and S. Levine. Learning with latent language. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,*, 2018.

K. Azizzadenesheli, A. Lazaric, and A. Anandkumar. Reinforcement learning in rich-observation mdps using spectral methods. *CoRR*, abs/1611.03907, 2016.

P. Bacon, J. Harb, and D. Precup. The option-critic architecture. In S.Singh and S. Markovitch, editors, *Proceedings of the Thirty-First Conference on Artificial Intelligence (AAAI'17)*. AAAI Press, 2017.

B. Balaji, P. Christodoulou, B. Jeon, and J. Bell-Masterson. Factoredrl: Leveraging factored graphs for deep reinforcement learning. In *NeurIPS Deep Reinforcement Learning Workshop*, 2020.

V. Bapst, A. Sanchez-Gonzalez, C. Doersch, K. Stachenfeld, P. Kohli, P. Battaglia, and J. Hamrick. Structured agents for physical construction. In Chaudhuri and Salakhutdinov [2019].

A. Barreto, W. Dabney, R. Munos, J. Hunt, T. Schaul, H. van Hasselt, and D. Silver. Successor features for transfer in reinforcement learning. In Guyon et al. [2017].

A. Barreto, D. Borsa, J. Quan, T. Schaul, D. Silver, M. Hessel, D. Mankowitz, A. Zidek, and R. Munos. Transfer in deep reinforcement learning using successor features and generalised policy improvement. In Dy and Krause [2018].

R. Bellman. Some applications of the theory of dynamic programming - A review. *Oper. Res.*, 2(3): 275–288, 1954.

S. Belogolovsky, P. Korsunsky, S. Mannor, C. Tessler, and T. Zahavy. Inverse reinforcement learning in contextual mdps. *Mach. Learn.*, 110(9):2295–2334, 2021.

S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors. *Proceedings of the 31st International Conference on Advances in Neural Information Processing Systems (NeurIPS'18)*, 2018. Curran Associates.

C. Benjamins, T. Eimer, F. Schubert, A. Mohan, S. Döhler, A. Biedenkapp, B. Rosenhahn, F. Hutter, and M. Lindauer. Contextualize me - the case for context in reinforcement learning. *Transactions on Machine Learning Research*, 2835-8856, 2023.

T. Bewley and F. Lecune. Interpretable preference-based reinforcement learning with tree-structured reward functions. In *21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2022, Auckland, New Zealand, May 9-13, 2022*. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2022.

O. Biza, T. Kipf, D. Klee, R. Platt, J. van de Meent, and L. Wong. Factored world models for zero-shot generalization in robotic manipulation. In *arXiv preprint arXiv:2202.05333*, 2022a.

O. Biza, R. Platt, J. van de Meent, L. Wong, and T. Kipf. Binding actions to objects in world models. In *arXiv preprint arXiv:2204.13022*, 2022b.

D. Borsa, T. Graepel, and J. Shawe-Taylor. Learning shared representations in multi-task reinforcement learning. *CoRR*, abs/1603.02041, 2016.

D. Borsa, A. Barreto, J. Quan, D. Mankowitz, H. van Hasselt, R. Munos, D. Silver, and T. Schaul. Universal successor features approximators. In *Proceedings of the 7th International Conference on Learning Representations (ICLR'19)*, 2019.

C. Boutilier, R. Dearden, and M. Goldszmidt. Stochastic dynamic programming with factored representations. *Artificial Intelligence*, 121(1-2):49–107, 2000.

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

P. Buchholz and D. Scheftelowitsch. Computation of weighted sums of rewards for concurrent mdps. *Math. Methods Oper. Res.*, 89(1):1–42, 2019.

C. Burgess, L. Matthey, N. Watters, R. Kabra, I. Higgins, M. Botvinick, and A. Lerchner. Monet: Unsupervised scene decomposition and representation. *CoRR*, abs/1901.11390, 2019. URL `http://arxiv.org/abs/1901.11390`.

Y. Chandak, G. Theocharous, J. Kostas, S. Jordan, and P. Thomas. Learning action representations for reinforcement learning. In Chaudhuri and Salakhutdinov [2019].

K. Chaudhuri and R. Salakhutdinov, editors. *Proceedings of the 36th International Conference on Machine Learning (ICML'19)*, volume 97, 2019. Proceedings of Machine Learning Research.

K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvári, G. Niu, and S. Sabato, editors. *Proceedings of the 39th International Conference on Machine Learning (ICML'22)*, volume 162 of *Proceedings of Machine Learning Research*, 2022. PMLR.

C. Chen, S. Hu, P. Nikdel, G. Mori, and M. Savva. Relational graph learning for crowd navigation. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020.

Z. Chu and H. Wang. Meta-reinforcement learning via exploratory task clustering. In *arXiv preprint arXiv:2302.07958*, 2023.

P. Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Comput.*, 5(4):613–624, 1993.

E. Van der Pol, D. Worrall, H. van Hoof, F. Oliehoek, and M. Welling. Mdp homomorphic networks: Group symmetries in reinforcement learning. In Larochelle et al. [2020].

C. D'Eramo, D. Tateo, A. Bonarini, M. Restelli, and Jan J. Peters. Sharing knowledge in multi-task deep reinforcement learning. In *Proceedings of the 8th International Conference on Learning Representations (ICLR'20)*, 2020.

C. Devin, A. Gupta, T. Darrell, P. Abbeel, and S. Levine. Learning modular neural network policies for multi-task and multi-robot transfer. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

C. Diuk, A. Cohen, and M. Littman. An object-oriented representation for efficient reinforcement learning. In W. Cohen, A. McCallum, and S. Roweis, editors, *Proceedings of the 25th International Conference on Machine Learning (ICML'08)*. Omnipress, 2008.

S. Du, A. Krishnamurthy, N. Jiang, A. Agarwal, M. Dudík, and J. Langford. Provably efficient RL with rich observations via latent state decoding. In Chaudhuri and Salakhutdinov [2019].

G. Dulac-Arnold, N. Levine, D. Mankowitz, J. Li, C. Paduraru, S. Gowal, and T. Hester. An empirical investigation of the challenges of real-world reinforcement learning. *CoRR*, abs/2003.11881, 2020. URL `https://arxiv.org/abs/2003.11881`.

J. Dy and A. Krause, editors. *Proceedings of the 35th International Conference on Machine Learning (ICML'18)*, volume 80, 2018. Proceedings of Machine Learning Research.

S. Dzeroski, L. De Raedt, and K. Driessens. Relational reinforcement learning. *Machine Learning Journal*, 43(1/2):7–52, 2001.

B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine. Diversity is all you need: Learning skills without a reward function. In *Proceedings of the 7th International Conference on Learning Representations (ICLR'19)*, 2019.

A. Fern, S. Yoon, and R. Givan. Approximate policy iteration with a policy language bias: Solving relational markov decision processes. *Journal of Artificial Intelligence Research*, 25:75–118, 2006.

C. Florensa, Y. Duan, and P. Abbeel. Stochastic neural networks for hierarchical reinforcement learning. In *Proceedings of the International Conference on Learning Representations (ICLR'17)*, 2017.

R. Fox, A. Pakman, and N. Tishby. Taming the noise in reinforcement learning via soft updates. In A. Ihler and D. Janzing, editors, *Proceedings of the 32nd conference on Uncertainty in Artificial Intelligence (UAI'16)*. AUAI Press, 2016.

V. Franccois-Lavet, P. Henderson, R. Islam, M. Bellemare, and J. Pineau. An introduction to deep reinforcement learning. *Found. Trends Mach. Learn.*, 11(3-4):219–354, 2018.

X. Fu, G. Yang, P. Agrawal, and T. Jaakkola. Learning task informed abstractions. In Meila and Zhang [2021].

S. Garg, A. Bajpai, and Mausam. Symbolic network: Generalized neural policies for relational mdps. In III and Singh [2020].

M. Garnelo, K. Arulkumaran, and M. Shanahan. Towards deep symbolic reinforcement learning. In *arXiv preprint arXiv:1609.05518*, 2016.

M. Gasse, D. Grasset, G. Gaudron, and P. Oudeyer. Causal reinforcement learning using observational and interventional data. In *arXiv preprint arXiv:2106.14421*, 2021.

J. Gaya, T. Doan, L. Caccia, L. Soulier, L. Denoyer, and R. Raileanu. Building a subspace of policies for scalable continual learning. In *arXiv preprint arXiv:2211.10445*, 2022a.

J. Gaya, L. Soulier, and L. Denoyer. Learning a subspace of policies for online adaptation in reinforcement learning. In *Proceedings of the International Conference on Learning Representations (ICLR'22)*, 2022b.

J. Gehring, G. Synnaeve, A. Krause, and N. Usunier. Hierarchical skills for efficient exploration. In Ranzato et al. [2021].

F. Geißer, D. Speck, and T. Keller. Trial-based heuristic tree search for mdps with factored action spaces. In *Proceedings of the International Symposium on Combinatorial Search*, 2020.

C. Gelada, S. Kumar, J. Buckman, O. Nachum, and M. Bellemare. Deepmdp: Learning continuous latent space models for representation learning. In Chaudhuri and Salakhutdinov [2019].

M. Ghorbani, R. Hosseini, S. Shariatpanahi, and M. Ahmadabadi. Reinforcement learning with subspaces using free energy paradigm. In *arXiv preprint arXiv:2012.07091*, 2020.

S. Gillen and K. Byl. Explicitly encouraging low fractional dimensional trajectories via reinforcement learning. In *Conference on Robot Learning*, pages 2137–2147. PMLR, 2021.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

A. Goyal, S. Sodhani, J. Binas, X. Peng, S. Levine, and Y. Bengio. Reinforcement learning with competitive ensembles of information-constrained primitives. In *Proceedings of the 8th International Conference on Learning Representations (ICLR'20)*, 2020.

A. Goyal, A. Lamb, J. Hoffmann, S. Sodhani, S. Levine, Y. Bengio, and B. Schölkopf. Recurrent independent mechanisms. In *Proceedings of the 9th International Conference on Learning Representations (ICLR'21)*, 2021.

C. Guestrin, D. Koller, C. Gearhart, and N. Kanodia. Generalizing plans to new environments in relational mdps. In G. Gottlob and T. Walsh, editors, *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI'03)*, 2003a.

C. Guestrin, D. Koller, R. Parr, and S. Venkataraman. Efficient solution algorithms for factored mdps. *Journal of Artificial Intelligence Research*, 19:399–468, 2003b.

A. Gupta, C. Devin, Y. Liu, P. Abbeel, and S. Levine. Learning invariant feature spaces to transfer skills with reinforcement learning. In *Proceedings of the International Conference on Learning Representations (ICLR'17)*, 2017.

A. Gupta, R. Mendonca, Y. Liu, P. Abbeel, and S. Levine. Meta-reinforcement learning of structured exploration strategies. In Bengio et al. [2018].

I. Gur, N. Jaques, Y. Miao, J. Choi, M. Tiwari, H. Lee, and A. Faust. Environment generation for zero-shot compositional reinforcement learning. In Ranzato et al. [2021].

I. Guyon, U. von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors. *Proceedings of the 30th International Conference on Advances in Neural Information Processing Systems (NeurIPS'17)*, 2017. Curran Associates.

T. Haarnoja, K. Hartikainen, P. Abbeel, and S. Levine. Latent space policies for hierarchical reinforcement learning. In Dy and Krause [2018].

T. Haarnoja, V. Pong, A. Zhou, M. Dalal, P. Abbeel, and S. Levine. Composable deep reinforcement learning for robotic manipulation. In *2018 IEEE international conference on robotics and automation (ICRA)*, 2018b.

T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Dy and Krause [2018].

D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi. Dream to control: Learning behaviors by latent imagination. In III and Singh [2020].

D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap. Mastering diverse domains through world models. In *arXiv preprint arXiv:2301.04104*, 2023.

A. Hallak, D. Di Castro, and S. Mannor. Contextual markov decision processes. *CoRR*, abs/1502.02259, 2015. URL `http://arxiv.org/abs/1502.02259`.

P. Hansen-Estruch, A. Zhang, A. Nair, P. Yin, and Sergey. Levine. Bisimulation makes analogies in goal-conditioned reinforcement learning. In Chaudhuri et al. [2022].

K. Hausman, J. Springenberg, Z. Wang, N. Heess, and M. Riedmiller. Learning an embedding space for transferable robot skills. In *Proceedings of the 6th International Conference on Learning Representations (ICLR'18)*, 2018.

N. Heess, G. Wayne, Y. Tassa, T. Lillicrap, M. Riedmiller, and D. Silver. Learning and transfer of modulated locomotor controllers. In *arXiv preprint arXiv:1610.05182*, 2016.

I. Higgins, A. Pal, A. Rusu, L. Matthey, C. Burgess, A. Pritzel, M. Botvinick, C. Blundell, and A. Lerchner. Darla: Improving zero-shot transfer in reinforcement learning. In Precup and Teh [2017].

S. Hofer. *On Decomposability in Robot Reinforcement Learning*. Technische University of Berlin (Germany), 2017.

Z. Hong, G. Yang, and P. Agrawal. Bilinear value networks. *CoRR*, abs/2204.13695, 2022.

Y. Hu and G. Montana. Skill transfer in deep reinforcement learning under morphological heterogeneity. In *arXiv preprint arXiv:1908.05265*, 2019.

W. Huang, I. Mordatch, and D. Pathak. One policy to control them all: Shared modular policies for agent-agnostic control. In III and Singh [2020].

H. Daume III and A. Singh, editors. *Proceedings of the 37th International Conference on Machine Learning (ICML'20)*, volume 98, 2020. Proceedings of Machine Learning Research.

L. Illanes, X. Yan, R. Icarte, and S. McIlraith. Symbolic plans as high-level instructions for reinforcement learning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, 2020.

C. Innes and A. Lascarides. Learning factored markov decision processes with unawareness. In J. Peters and D. Sontag, editors, *Proceedings of The 36th Uncertainty in Artificial Intelligence Conference (UAI'20)*. PMLR, 2020.

R. Islam, H. Zang, A. Goyal, A. Lamb, K. Kawaguchi, X. Li, R. Laroche, Y. Bengio, and R. Combes. Discrete factorial representations as an abstraction for goal conditioned reinforcement learning. In *arXiv preprint arXiv:2211.00247*, 2022.

A. Jain, A. Szot, and J. Lim. Generalization to new actions in reinforcement learning. In III and Singh [2020].

A. Jain, K. Khetarpal, and D. Precup. Safe option-critic: learning safety in the option-critic architecture. *The Knowledge Engineering Review*, 36:e4, 2021a.

A. Jain, N. Kosaka, K. Kim, and J. Lim. Know your action set: Learning action relations for reinforcement learning. In Meila and Zhang [2021].

J. Janisch, T. Pevnỳ, and V. Lisỳ. Symbolic relational deep reinforcement learning based on graph neural networks. In *arXiv preprint arXiv:2009.12462*, 2020.

Y. Jiang, S. Gu Shane, K. Murphy, and C. Finn. Language as an abstraction for hierarchical deep reinforcement learning. In Wallach et al. [2019].

Rico Jonschkowski, Sebastian Höfer, and Oliver Brock. Patterns for learning with side information. In *arXiv preprint arXiv:1511.06429*, 2015.

S. Joshi and R. Khardon. Probabilistic relational planning with first order decision diagrams. *Journal of Artificial Intelligence Research*, 41:231–266, 2011.

R. Karia and S. Srivastava. Relational abstractions for generalized reinforcement learning on symbolic problems. In *arXiv preprint arXiv:2204.12665*, 2022.

M. Kearns and D. Koller. Efficient reinforcement learning in factored mdps. In T. Dean, editor, *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI'99)*. Morgan Kaufmann Publishers, 1999.

M. Khamassi and C. Tzafestas. Active exploration in parameterized reinforcement learning. *CoRR*, abs/1610.01986, 2016. URL `http://arxiv.org/abs/1610.01986`.

K. Kim and T. Dean. Solving factored mdps with large action space using algebraic decision diagrams. In *Trends in Artificial Intelligence*, 2002.

T. Kipf, E. van der Pol, and M. Welling. Contrastive learning of structured world models. In *Proceedings of the 8th International Conference on Learning Representations (ICLR'20)*, 2020.

R. Kirk, A. Zhang, E. Grefenstette, and T. Rocktäschel. A survey of zero-shot generalisation in deep reinforcement learning. *Journal of Artificial Intelligence Research*, 76:201–264, 2023.

H. Kokel, A. Manoharan, S. Natarajan, B. Ravindran, and P. Tadepalli. Reprel: Integrating relational planning and reinforcement learning for effective abstraction. In *Proceedings of the International Conference on Automated Planning and Scheduling*, 2021.

D. Koller and R. Parr. Computing factored value functions for policies in structured mdps. In *IJCAI*, 1999.

T. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, and R. Garnett, editors, *Proceedings of the 29th International Conference on Advances in Neural Information Processing Systems (NeurIPS'16)*. Curran Associates, 2016.

S. Kumar, I. Dasgupta, J. Cohen, N. Daw, and T. Griffiths. Meta-learning of structured task distributions in humans and machines. In *Proceedings of the 9th International Conference on Learning Representations (ICLR'21)*, 2021.

S. Kumar, C. Correa, I. Dasgupta, R. Marjieh, M. Hu, R. Hawkins, N. Daw, J. Cohen, K. Narasimhan, and T. Griffiths. Using natural language and program abstractions to instill human inductive biases in machines. In *Proceedings of the 35th International Conference on Advances in Neural Information Processing Systems (NeurIPS'22)*, 2022.

J. Kwon, Y. Efroni, C. Caramanis, and S. Mannor. Rl for latent mdps: Regret guarantees and a lower bound. In Ranzato et al. [2021].

A. Lampinen, N. Roy, I. Dasgupta, S. Chan, A. Tam, J. Mcclelland, C. Yan, A. Santoro, N. Rabinowitz, Jane J. Wang, and F. Hill. Tell me why! explanations support learning relational and causal structure. In Chaudhuri et al. [2022].

H. Larochelle, M. Ranzato, R. Hadsell, M.-F. Balcan, and H. Lin, editors. *Proceedings of the 33rd International Conference on Advances in Neural Information Processing Systems (NeurIPS'20)*, 2020. Curran Associates.

M. Laskin, D. Yarats, H. Liu, K. Lee, A. Zhan, K. Lu, C. Cang, L. Pinto, and P. Abbeel. URLB: unsupervised reinforcement learning benchmark. In Joaquin Vanschoren and Sai-Kit Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021.

A. Lee, A. Nagabandi, P. Abbeel, and S. Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. In Larochelle et al. [2020].

J. Lee, S. Sedwards, and K. Czarnecki. Recursive constraints to prevent instability in constrained reinforcement learning. In *arXiv preprint arXiv:2201.07958*, 2022.

S. Lee and S. Chung. Improving generalization in meta-rl with imaginary tasks from latent dynamics mixture. In Ranzato et al. [2021].

L. Li, T. Walsh, and M. Littman. Towards a unified theory of state abstraction for mdps. In *AI&M*, 2006.

T. Li, J. Pan, D. Zhu, and M. Meng. Learning to interrupt: A hierarchical deep reinforcement learning framework for efficient exploration. In *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 648–653. IEEE, 2018.

Y. Li, Y. Wu, H. Xu, X. Wang, and Y. Wu. Solving compositional reinforcement learning problems via task reduction. In *Proceedings of the 9th International Conference on Learning Representations (ICLR'21)*, 2021.

L. Liao, Z. Fu, Z. Yang, Y. Wang, M. Kolar, and Z. Wang. Instrumental variable value iteration for causal offline reinforcement learning. In *arXiv preprint arXiv:2102.09907*, 2021.

K. Lu, S. Zhang, P. Stone, and X. Chen. Robot representation and reasoning with knowledge from reinforcement learning. *CoRR*, abs/1809.11074, 2018.

M. Lu, Z. Shahn, D. Sow, F. Doshi-Velez, and L. H. Lehman. Is deep reinforcement learning ready for practical applications in healthcare? A sensitivity analysis of duel-ddqn for hemodynamic management in sepsis patients. In *AMIA 2020, American Medical Informatics Association Annual Symposium, Virtual Event, USA, November 14-18, 2020*. AMIA, 2020.

D. Lyu, F. Yang, B. Liu, and S. Gustafson. SDRL: interpretable and data-efficient deep reinforcement learning leveraging symbolic planning. In P. Van Hentenryck and Z. Zhou, editors, *Proceedings of the Thirty-Third Conference on Artificial Intelligence (AAAI'19)*. AAAI Press, 2019.

A: Mahajan and T. Tulabandhula. Symmetry learning for function approximation in reinforcement learning. In *arXiv preprint arXiv:1706.02999*, 2017.

A. Mahajan, M. Samvelyan, L. Mao, V. Makoviychuk, A. Garg, J. Kossaifi, S. Whiteson, Y. Zhu, and A. Anandkumar. Reinforcement learning in factored action spaces using tensor decompositions. In *arXiv preprint arXiv:2110.14538*, 2021.

D. Mambelli, F. Träuble, S. Bauer, B. Schölkopf, and F. Locatello. Compositional multi-object reinforcement learning with linear relation networks. In *arXiv preprint arXiv:2201.13388*, 2022.

D. Mankowitz, T. Mann, and S. Mannor. Bootstrapping skills. In *arXiv preprint arXiv:1506.03624*, 2015.

S. Mannor and A. Tamar. Towards deployable rl–what's broken with rl research and a potential fix. In *arXiv preprint arXiv:2301.01320*, 2023.

D. Martinez, G. Alenya, and C. Torras. Relational reinforcement learning with guided demonstrations. *Artificial Intelligence*, 247:295–312, 2017.

D. Mausam and D. Weld. Solving relational mdps with first-order machine learning. In *Proceedings of the ICAPS workshop on planning under uncertainty and incomplete information*, 2003.

M. Meila and T. Zhang, editors. *Proceedings of the 38th International Conference on Machine Learning (ICML'21)*, volume 139 of *Proceedings of Machine Learning Research*, 2021. PMLR.

J. Mendez, H. van Seijen, and E. Eaton. Modular lifelong reinforcement learning via neural composition. In *Proceedings of the International Conference on Learning Representations (ICLR'22)*, 2022.

T. Meng and M. Khushi. Reinforcement learning in financial markets. *Data*, 4(3):110, 2019.

L. Metz, J. Ibarz, N. Jaitly, and J. Davidson. Discrete sequential prediction of continuous actions for deep rl. In *arXiv preprint arXiv:1705.05035*, 2017.

V. Mihajlovic and M. Petkovic. Dynamic bayesian networks: A state of the art. In *University of Twente Document Repository*, 2001.

R. Mirsky, S. Shperberg, Y. Zhang, Z. Xu, Y. Jiang, J. Cui, and P. Stone. Task factorization in curriculum learning. In *Decision Awareness in Reinforcement Learning Workshop at ICML 2022*, 2022. URL `https://openreview.net/forum?id=QgeozWoz64Q`.

J. Mu, V. Zhong, R. Raileanu, M. Jiang, N. Goodman, T. Rocktäschel, and E. Grefenstette. Improving intrinsic exploration with language abstractions. In *Proceedings of the 35th International Conference on Advances in Neural Information Processing Systems (NeurIPS'22)*, 2022a.

T. Mu, K. Lin, F. Niu, and G. Thattai. Learning two-step hybrid policy for graph-based interpretable reinforcement learning. *Trans. Mach. Learn. Res.*, 2022, 2022b.

O. Nachum, S. Gu Shane, H. Lee, and S. Levine. Data-efficient hierarchical reinforcement learning. In Bengio et al. [2018].

S. Narvekar, B. Peng, M. Leonetti, J. Sinapov, M. Taylor, and P. Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research*, 2020.

A. Ng, D. Harada, and S. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In I. Bratko, editor, *Proceedings of the Sixteenth International Conference on Machine Learning (ICML'99)*. Morgan Kaufmann Publishers, 1999.

J. Ok, A. Proutière, and D. Tranos. Exploration in structured reinforcement learning. In Bengio et al. [2018].

M. Oliva, S. Banik, J. Josifovski, and A. Knoll. Graph neural networks for relational inductive bias in vision-based deep reinforcement learning of robot control. In *International Joint Conference on Neural Networks, IJCNN 2022, Padua, Italy, July 18-23, 2022*, pages 1–9. IEEE, 2022.

M. Papini, A. Tirinzoni, A. Pacchiano, M. Restelli, A. Lazaric, and M. Pirotta. Reinforcement learning in linear mdps: Constant regret and representation selection. In Ranzato et al. [2021].

S. Pateria, B. Subagdja, A. Tan, and C. Quek. Hierarchical reinforcement learning: A comprehensive survey. *ACM Computing Surveys*, 54(5):109:1–109:35, 2022.

D. Pathak, C. Lu, T. Darrell, P. Isola, and A. Efros. Learning to control self-assembling morphologies: a study of generalization via modularity. In Wallach et al. [2019].

A. Payani and F. Fekri. Incorporating relational background knowledge into reinforcement learning via differentiable inductive logic programming. *CoRR*, abs/2003.10386, 2020.

C. Perez, F. Such, and T. Karaletsos. Generalized hidden parameter mdps transferable model-based rl in a handful of trials. In Rossi et al. [2020].

J. Peters, P. Buhlmann, and N. Meinshausen. Causal inference by using invariant prediction: identification and confidence intervals. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 78(5):947–1012, 2016.

B. Prakash, N. Waytowich, A. Ganesan, T. Oates, and T. Mohsenin. Guiding safe reinforcement learning policies using structured language constraints. In Huáscar Espinoza, José Hernández-Orallo, Xin Cynthia Chen, Seán S. ÓhÉigeartaigh, Xiaowei Huang, Mauricio Castillo-Effen, Richard Mallah, and John A. McDermid, editors, *Proceedings of the Workshop on Artificial Intelligence Safety, co-located with 34th AAAI Conference on Artificial Intelligence, SafeAI@AAAI 2020, New York City, NY, USA, February 7, 2020*, volume 2560 of *CEUR Workshop Proceedings*, pages 153–161. CEUR-WS.org, 2020.

B. Prakash, N. Waytowich, T. Oates, and T. Mohsenin. Towards an interpretable hierarchical agent framework using semantic goals. *CoRR*, abs/2210.08412, 2022.

D. Precup and Y. Teh, editors. *Proceedings of the 34th International Conference on Machine Learning (ICML'17)*, volume 70, 2017. Proceedings of Machine Learning Research.

M. Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

M. Ranzato, A. Beygelzimer, K. Nguyen, P. Liang, J. Vaughan, and Y. Dauphin, editors. *Proceedings of the 34th International Conference on Advances in Neural Information Processing Systems (NeurIPS'21)*, 2021. Curran Associates.

S. Raza and M. Lin. Policy reuse in reinforcement learning for modular agents. In *IEEE 2nd International Conference on Information and Computer Technologies (ICICT)*. IEEE, 2019.

S. Ross and J. Pineau. Model-based bayesian reinforcement learning in large structured domains. In David A. McAllester and Petri Myllymäki, editors, *UAI 2008, Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence, Helsinki, Finland, July 9-12, 2008*, pages 476–483. AUAI Press, 2008.

F. Rossi, V. Conitzer, and F. Sha, editors. *Proceedings of the Thirty-Fourth Conference on Artificial Intelligence (AAAI'20)*, 2020. Association for the Advancement of Artificial Intelligence, AAAI Press.

S. Sanner and C. Boutilier. Approximate linear programming for first-order mdps. In *arXiv preprint arXiv:1207.1415*, 2012.

A. Saxe, A. Earle, and B. Rosman. Hierarchy through composition with multitask lmdps. In Precup and Teh [2017].

T. Schaul, D. Horgan, K. Gregor, and D. Silver. Universal value function approximators. In F. Bach and D. Blei, editors, *International conference on machine learning*, volume 37. Omnipress, 2015.

M. Seitzer, B. Schölkopf, and G. Martius. Causal influence detection for improving efficiency in reinforcement learning. In Ranzato et al. [2021].

M. Shanahan, K. Nikiforou, A. Creswell, C. Kaplanis, D. Barrett, and M. Garnelo. An explicitly relational neural network architecture. In *icml20*, 2020.

V. Sharma, D. Arora, F. Geisser, A. Mausam, and P. Singla. Symnet 2.0: Effectively handling non-fluents and actions in generalized neural policies for rddl relational mdps. In *Uncertainty in Artificial Intelligence*, pages 1771–1781. PMLR, 2022.

T. Shu, ng C. Xio, and R. Socher. Hierarchical and interpretable skill acquisition in multi-task reinforcement learning. In *Proceedings of the 6th International Conference on Learning Representations (ICLR'18)*, 2018.

P. Shyam, W. Jaskowski, and F. Gomez. Model-based active exploration. In Chaudhuri and Salakhutdinov [2019].

T. Simao, N. Jansen, and M. Spaan. Alwayssafe: Reinforcement learning without safety constraint violations during training. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, 2021.

S. Sodhani, A. Zhang, and J. Pineau. Multi-task reinforcement learning with context-based representations. In Meila and Zhang [2021].

S. Sodhani, S. Levine, and A. Zhang. Improving generalization with approximate factored value functions. In *ICLR2022 Workshop on the Elements of Reasoning: Objects, Structure and Causality*, 2022a.

S. Sodhani, F. Meier, J. Pineau, and A. Zhang. Block contextual mdps for continual learning. In *Learning for Dynamics and Control Conference*, 2022b.

S. Sohn, J. Oh, and H. Lee. Hierarchical reinforcement learning for zero-shot generalization with subtask dependencies. In Bengio et al. [2018].

A. Solway, C. Diuk, N. Córdova, D. Yee, A. Barto, Y. Niv, and M. Botvinick. Optimal behavioral hierarchy. *PLoS Comput. Biol.*, 10(8), 2014.

T. Spooner, N. Vadori, and S. Ganesh. Factored policy gradients: Leveraging structure for efficient learning in momdps. In Ranzato et al. [2021].

M. Srouji, J. Zhang, and R. Salakhutdinov. Structured control nets for deep reinforcement learning. In Dy and Krause [2018].

L. Steccanella, S. Totaro, and A. Jonsson. Hierarchical representation learning for markov decision processes. In *arXiv preprint arXiv:2106.01655*, 2021.

R. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In S. Solla, T. Leen, and K. Müller, editors, *Proceedings of the 12th International Conference on Advances in Neural Information Processing Systems (NeurIPS'99)*. The MIT Press, 1999a.

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

K. Sycara, V. Honavar, and M. Spaan, editors. *Proceedings of the Thirty-Sixth Conference on Artificial Intelligence (AAAI'22)*, 2022. Association for the Advancement of Artificial Intelligence, AAAI Press.

N. Talele and K. Byl. Mesh-based tools to analyze deep reinforcement learning policies for underactuated biped locomotion. In *arXiv preprint arXiv:1903.12311*, 2019.

S. Tang, M. Makar, M. Sjoding, F. Doshi-Velez, and J. Wiens. Leveraging factored action spaces for efficient offline reinforcement learning in healthcare. In *Decision Awareness in Reinforcement Learning Workshop at ICML 2022*, 2022a.

S. Tang, M. Makar, M. Sjoding, F. Doshi-Velez, and J. Wiens. Leveraging factored action spaces for efficient offline reinforcement learning in healthcare. In *Decision Awareness in Reinforcement Learning Workshop at ICML 2022*, 2022b.

M. Tavakol and U. Brefeld. Factored mdps for detecting topics of user sessions. In *Proceedings of the 8th ACM Conference on Recommender Systems*, 2014.

A. Tavakoli, F. Pardo, and P. Kormushev. Action branching architectures for deep reinforcement learning. In S. McIlraith and K. Weinberger, editors, *Proceedings of the Thirty-Second Conference on Artificial Intelligence (AAAI'18)*. AAAI Press, 2018.

G. Tennenholtz and S. Mannor. The natural language of actions. In Chaudhuri and Salakhutdinov [2019].

G: Trimponias and T. Dietterich. Reinforcement learning with exogenous states and rewards. In *arXiv preprint arXiv:2303.12957*, 2023.

P. Tsividis, J. Loula, J. Burga, N. Foss, A. Campero, T. Pouncy, S. Gershman, and J. Tenenbaum. Human-level reinforcement learning through theory-based modeling, exploration, and planning. *CoRR*, abs/2107.12544, 2021. URL `https://arxiv.org/abs/2107.12544`.

E. van der Pol, T. Kipf, F.. Oliehoek, and M. Welling. Plannable approximations to MDP homomorphisms: Equivariance under actions. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems*, 2020.

C. van Rossum, C. Feinberg, A. Abu Shumays, K. Baxter, and B. Bartha. A novel approach to curiosity and explainable reinforcement learning via interpretable sub-goals. *CoRR*, abs/2104.06630, 2021. URL `https://arxiv.org/abs/2104.06630`.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In Guyon et al. [2017].

R. Veerapaneni, J. Co-Reyes, M. Chang, M. Janner, C. Finn, J. Wu, J. Tenenbaum, and S. Levine. Entity abstraction in visual model-based reinforcement learning. In *Conference on Robot Learning*. PMLR, 2020.

A. Verma, V. Murali, R. Singh, P. Kohli, and S. Chaudhuri. Programmatically interpretable reinforcement learning. In *icml18*, 2018.

H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alche Buc, E. Fox, and R. Garnett, editors. *Proceedings of the 32nd International Conference on Advances in Neural Information Processing Systems (NeurIPS'19)*, 2019. Curran Associates.

G. Wang, Z. Fang, B. Li, and P. Li. Integrating symmetry of environment by designing special basis functions for value function approximation in reinforcement learning. In *Fourteenth International Conference on Control, Automation, Robotics and Vision*, 2016.

H. Wang, S. Dong, and L. Shao. Measuring structural similarities in finite mdps. In S. Kraus, editor, *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI'19)*, 2019.

J. Wang, Y. Liu, and B. Li. Reinforcement learning with perturbed rewards. In Rossi et al. [2020].

J. Wang, M. King, N. Porcel, Z. Kurth-Nelson, T. Zhu, C. Deck, P. Choy, M. Cassin, M. Reynolds, H. Song, G. Buttimore, D. Reichert, N. Rabinowitz, L. Matthey, D. Hassabis, A. Lerchner, and M. Botvinick. Alchemy: A benchmark and analysis toolkit for meta-reinforcement learning agents. In Ranzato et al. [2021].

T. Wang, R. Liao, J. Ba, and S. Fidler. Nervenet: Learning structured policy with graph neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR'18)*, 2018. Published online: `iclr.cc`.

T. Wang, S. Du, A. Torralba, P. Isola, A. Zhang, and Y. Tian. Denoised mdps: Learning world models better than the world itself. In *arXiv preprint arXiv:2206.15477*, 2022.

R. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8:229–256, 1992a.

R. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8:229–256, 1992b.

H. Woo, G. Yoo, and M. Yoo. Structure learning-based task decomposition for reinforcement learning in non-stationary environments. In Sycara et al. [2022].

C. Wu, A. Rajeswaran, Y. Duan, V. Kumar, A. Bayen, S. Kakade, I. Mordatch, and P. Abbeel. Variance reduction for policy gradient with action-dependent factorized baselines. In *Proceedings of the 6th International Conference on Learning Representations (ICLR'18)*, 2018.

D. Xu and F. Fekri. Interpretable model-based hierarchical reinforcement learning using inductive logic programming. *CoRR*, abs/2106.11417, 2021. URL `https://arxiv.org/abs/2106.11417`.

C. Yang, I. Hung, Y. Ouyang, and P. Chen. Training a resilient q-network against observational interference. In Sycara et al. [2022].

R: Yang, H. Xu, Y. Wu, and X. Wang. Multi-task reinforcement learning with soft modularization. In Larochelle et al. [2020].

K. Young, A. Ramesh, L. Kirsch, and J. Schmidhuber. The benefits of model-based generalization in reinforcement learning. In *arXiv preprint arXiv:2211.02222*, 2022.

D. Yu, H. Ma, S. Li, and J. Chen. Reachability constrained reinforcement learning. In Chaudhuri et al. [2022].

V. Zambaldi, D. Raposo, A. Santoro, V. Bapst, Y. Li, I. Babuschkin, K. Tuyls, D. Reichert, T. Lillicrap, E. Lockhart, M. Shanahan, V. Langston, R. Pascanu, M. Botvinick, O. Vinyals, and P. Battaglia. Deep reinforcement learning with relational inductive biases. In *Proceedings of the Seventh International Conference on Learning Representations, ICLR 2019*. OpenReview.net, 2019.

A. Zhang, C. Lyle, S. Sodhani, A. Filos, M. Kwiatkowska, J. Pineau, Y. Gal, and D. Precup. Invariant causal prediction for block mdps. In III and Singh [2020].

A. Zhang, S. Sodhani, K. Khetarpal, and J. Pineau. Multi-task reinforcement learning as a hidden-parameter block mdp. In *arXiv preprint arXiv:2007.07206*, 2020b.

A. Zhang, R. McAllister, R. Calandra, Y. Gal, and S. Levine. Learning invariant representations for reinforcement learning without reconstruction. In *Proceedings of the 9th International Conference on Learning Representations (ICLR'21)*, 2021a.

A. Zhang, S. Sodhani, K. Khetarpal, and J. Pineau. Learning robust state abstractions for hidden-parameter block mdps. In *Proceedings of the 9th International Conference on Learning Representations (ICLR'21)*, 2021b.

D. Zhang, A. Courville, Y. Bengio, Q. Zheng, A. Zhang, and R. Chen. Latent state marginalization as a low-cost approach for improving exploration. *CoRR*, abs/2210.00999, 2022.

H. Zhang, Z. Gao, Y. Zhou, H. Zhang, K. Wu, and F. Lin. Faster and safer training by embedding high-level knowledge into deep reinforcement learning. *CoRR*, abs/1910.09986, 2019a. URL `http://arxiv.org/abs/1910.09986`.

H. Zhang, H. Chen, C. Xiao, B. Li, M. Liu, D. Boning, and C. Hsieh. Robust deep reinforcement learning against adversarial perturbations on state observations. In Larochelle et al. [2020].

S. Zhang, H. Tong, J. Xu, and R. Maciejewski. Graph convolutional networks: a comprehensive review. *Computational Social Networks*, 6(1):1–23, 2019b.

T. Zhao, K. Xie, and M. Eskénazi. Rethinking action spaces for reinforcement learning in end-to-end dialog agents with latent variable models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.

A. Zhou, V. Kumar, C. Finn, and A. Rajeswaran. Policy architectures for compositional generalization in control. In *arXiv preprint arXiv:2203.05960*, 2022.

J. Zhu, T. Park, P. Isola, and A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

A Patterns in Existing Literature

In the following subsections, we delve deeper into each pattern, explaining different lines of literature that apply each pattern for different uses. To further provide intuition about this categorization, we will consider the running example of a taxi service, where the task of the RL agent (the taxi) is to pick up passengers from various locations and drop them at their desired destinations within a city grid. The agent receives a positive reward when a passenger is successfully dropped off at their destination, and incurs a small penalty for each time step to encourage efficiency.

For each of the following sections, we present a table of the surveyed methods that categorizes the work in the following manner: (i) The structured space, information about which is incorporated as side information; (ii) The type of decomposition exhibited for that structured space. We specifically categorize works that use structured task distributions through goals and/or rewards; (iii) The auxiliary objectives for which the decomposition is utilized. Our rationale behind the format of the tables, in addition to demonstrating our categorization, is to highlight the areas where further research might be lucrative. These are the spots in the tables where we could not yet find literature, and/or we believe additional work can be important.

A.1 Abstraction Pattern

Abstraction pattern utilizes structural information to create abstract entities in the RL pipeline. For any entity, X, an abstraction utilizes the structural information to create X_{abs}, which takes over the role of X in the optimization procedure.

Space	Type	Efficiency	Generalization	Interpretabiltiy	Safety
Goals	Latent		Hansen-Estruch et al. [2022]		
	Relational			Prakash et al. [2022]	
	Modular			Prakash et al. [2022]	
States	Latent	Zhang et al. [2022], Ghorbani et al. [2020], Allen et al. [2021], Zhang et al. [2021a], Gelada et al. [2019], Lee et al. [2020], Aziz-zadenesheli et al. [2016]	Lee et al. [2020], Zhang et al. [2021b], Gelada et al. [2019], Zhang et al. [2020a]	Gillen and Byl [2021]	Yang et al. [2022], Gillen and Byl [2021]
	Factored	Sodhani et al. [2022a]	Higgins et al. [2017], Sodhani et al. [2021], Perez et al. [2020], Sodhani et al. [2022a]	Sodhani et al. [2021], Bewley and Lecune [2022]	
	Relational	Martinez et al. [2017], Garnelo et al. [2016], Kipf et al. [2020], Kokel et al. [2021]	Janisch et al. [2020], Kokel et al. [2021], Bapst et al. [2019], Adjodah et al. [2018], Garnelo et al. [2016], Kipf et al. [2020]	Adjodah et al. [2018], Garnelo et al. [2016]	
	Modular	Kokel et al. [2021]	Kokel et al. [2021], Steccanella et al. [2021]		
Actions	Latent	Zhao et al. [2019], Chandak et al. [2019]			
	Factored		Perez et al. [2020]	Bewley and Lecune [2022]	
	Relational		Bapst et al. [2019]		
	Modular		Steccanella et al. [2021]		
Rewards	Latent		Zhang et al. [2021b], Barreto et al. [2017], Barreto et al. [2018], Borsa et al. [2016]		

	Factored	Sodhani et al. [2022a]	Perez et al. [2020], Sodhani et al. [2022a], Sodhani et al. [2021],	Sodhani et al. [2021]	Wang et al. [2020]
Dynamics	Latent	Zhang et al. [2020b]	Zhang et al. [2020b], Borsa et al. [2019]		
	Factored		Fu et al. [2021]		
	Modular				

Generalization State abstractions are a general choice for improving generalization performance using methods such as Invariant Causal Prediction [Zhang et al., 2020a, Peters et al., 2016], bisimulation [Hansen-Estruch et al., 2022, Zhang et al., 2021b], Free Energy Minimization Ghorbani et al. [2020], etc. Disentangled representations [Higgins et al., 2017] impose factored dynamics-aware decomposability onto the state-space to tackle zero-shot transfer, which has been further extended to the using VAEs [Burgess et al., 2019].

Value functions have served as abstractions for shared dynamics in Multi-task Settings. Successor Features (SF) [Dayan, 1993, Barreto et al., 2017] exploit latent reward and dynamic decompositions by using value functions as an abstraction. Subsequent works have combined them with Generalized Policy Iteration [Barreto et al., 2018] and Universal Value Function Approximators [Borsa et al., 2019, Schaul et al., 2015]. In parallel, works such as Sodhani et al. [2021, 2022a] have exploited attention mechanisms over factored states for better performance across changing dynamics. Additionally, Latent variable models [Perez et al., 2020] utilize factorization as abstractions to impose independence conditions in the transition dynamics.

Relational abstractions help incorporate symbolic spaces into the RL pipeline. Karia and Srivastava [2022] learn generalizable Q values over abstract states and actions that can be transferred to new tasks, while Kokel et al. [2021] use a symbolic planner to generate state abstractions to facilitate faster learning across new tasks.

Sample Efficiency Latent state-space models can improve sample efficiency in Model-based RL [Gelada et al., 2019]. In model-free tasks, these can also be learned as inverse models visual features Allen et al. [2021], or for control in a latent space [Lee et al., 2020]. Latent transition models demonstrate efficiency gains by capturing task-relevant information in noisy settings [Fu et al., 2021], or by preserving bisimulation distances between original states [Zhang et al., 2021a].

Zhao et al. [2019] use latent action models for actions in a dialog generation process. This allows them to shorten the learning horizon, and thus, converge using REINFORCE [Williams, 1992b] faster. Chandak et al. [2019] learn an embedding space of actions using Supervised Learning, and then train a policy on this latent space instead of the full action space for Model-based RL.

Safety and Interpretability Relational abstractions are a very good choice for interpretability since they capture interactionally complex decompositions. Adjodah et al. [2018] combine designed object representations and learned abstractions to add transparency, thus, attaining better interpretability. Garnelo et al. [2016] build a relational state by tracking objects across frames using heuristics and using relational measures between identified objects

Abstraction for safety is generally limited to state spaces and rewards. Yang et al. [2022] use a given set of inference labels to train an RL agent to learn a causal inference model by embedding the confounders into a latent state. During testing, the agent uses the learned model to estimate the confounding latent state and the interference label. Meshes [Talele and Byl, 2019], on the other hand, help with benchmarking the robustness of the learned policy. Gillen and Byl [2021] tackle safety through a latent representation of the state by learning a lower dimensional number of bins that can discretize a mesh.

A.2 Augmentation Pattern

Structural biases can be additionally incorporated as an augmentation to X to achieve the aforementioned objectives. Intuitively, this pattern treats X and z as separate input entities, the combination of which can range from the simple concatenation of additional information to more involved methods

of conditioning policy and/or value functions on additional information. Crucially, the structural information neither directly influences the optimization procedure nor changes the nature of X.

Space	Type	Efficiency	Generalization	Interpretabiltiy	Safety
Goals	Latent		Andreas et al. [2018], Schaul et al. [2015]		
	Factored	Islam et al. [2022]	Jiang et al. [2019]		
	Relational	Andreas et al. [2018]	Andreas et al. [2018], Jiang et al. [2019]		
	Modular	Gehring et al. [2021]	Jiang et al. [2019], Gehring et al. [2021]		Gehring et al. [2021]
States	Latent	Islam et al. [2022], Andreas et al. [2018], Gupta et al. [2018]	Andreas et al. [2018], Sodhani et al. [2022b], Gupta et al. [2018]		
	Factored	Islam et al. [2022]			
	Relational	Andreas et al. [2018]	Andreas et al. [2018]		
	Modular				
Actions	Latent	Tennenholtz and Mannor [2019]	Jain et al. [2021b], Jain et al. [2020]		
	Relational		Jain et al. [2021b]		
Rewards	Modular		Pathak et al. [2019]		
Dynamics	Latent		Sodhani et al. [2022b]		
	Factored		Goyal et al. [2021]		
Policies	Modular	Raza and Lin [2019], Haarnoja et al. [2018a]	Haarnoja et al. [2018a]		

Context-based Augmentations Contextual representations of dynamics in block MDPs [Sodhani et al., 2022b] and discretized goal-abstractions [Islam et al., 2022] augmented to the state improve generalization and sample efficiency. Augmentation of Action history vectors to the state help with sample efficiency [Tennenholtz and Mannor, 2019], and action relations [Jain et al., 2020, 2021b] contribute to generalization over large action sets.

Language Augmentations Language augmentation can capture relational metadata in the world. Andreas et al. [2018] condition policy search by assigning a probability to actions that are proportional to the environment state and a bilinear function of the output of a latent language interpretation model. This augmentation allows them to achieve exploration and generalization gains using language descriptions. Jiang et al. [2019] encode goal-conditioned modular decomposition using language in a two-level hierarchical framework where the higher level policy learns to generate language instructions, subsequently encoded as goals for the lower level policy using a GRU.

Control Augmentations Augmentations can additionally help with primitive control, such as multi-level control seen in the HRL literature. Haarnoja et al. [2018a] tackle hierarchical DNN policies by augmenting internal latent variables to policies of each layer. Gehring et al. [2021] present an HRL method with three levels: a policy that specifies a goal space (a set of features to operate on), a policy to specify the goal configuration conditioned on the goal space, and a low-level policy to reach the desired goal configuration. They additionally learn the low-level policy through unsupervised learning and then optimize the high-level options offline by optimizing the value function. A parallel line of work uses the augmentation pattern for morphological control [Huang et al., 2020]. Pathak et al. [2019] model the different limbs as individual agents that need to learn to join together into a morphology to solve a task.

A.3 Auxiliary Optimization Pattern

In this pattern, structural decompositions are used to modify the optimization procedure. Given that the changes in the optimization can go hand-in-hand with modifications of other components, many methods in this utilize other patterns in conjunction (E.g. contrastive losses to learn state abstractions).

Space	Type	Efficiency	Generalization	Interpretabiltiy	Safety

Goals	Relational		Kumar et al. [2022]		
	Modular	Nachum et al. [2018], Illanes et al. [2020], Li et al. [2021], Gehring et al. [2021]			
States	Latent	Mahajan and Tulabandhula [2017], Li et al. [2021], Azizzadenesheli et al. [2016], Ok et al. [2018], Amin et al. [2021]			Zhang et al. [2020c], Yu et al. [2022], Lee et al. [2022]
	Factored	Tavakol and Brefeld [2014], Trimponias and Dietterich [2023], Ross and Pineau [2008]		Liao et al. [2021]	
	Relational	Li et al. [2021]			
	Modular			Lyu et al. [2019]	
Actions	Latent	Ok et al. [2018], Amin et al. [2021]			Zhang et al. [2019a]
	Factored	Balaji et al. [2020], Wu et al. [2018], Tang et al. [2022a], Metz et al. [2017], Spooner et al. [2021], Tang et al. [2022b], Khamassi and Tzafestas [2016], Tavakol and Brefeld [2014]			
	Modular	Metz et al. [2017]		Lyu et al. [2019]	Jain et al. [2021a]
Rewards	Factored	Trimponias and Dietterich [2023], Saxe et al. [2017]	Belogolovsky et al. [2021], Saxe et al. [2017]		Prakash et al. [2020]
	Modular	Li et al. [2021]			
Dynamics	Latent	Mu et al. [2022a]			
	Factored	Buchholz and Scheftelowitsch [2019]	Belogolovsky et al. [2021], Buchholz and Scheftelowitsch [2019]		
	Relational	Mu et al. [2022a]			
Policy Space	Latent	Hausman et al. [2018]	Hausman et al. [2018]		

Reward Modification Reward shaping is a very common way to incorporate additional information into the optimization procedure. Methods can gain sample efficiency by exploiting modular and relational decompositions through task descriptions [Illanes et al., 2020], or goal information from a higher level policy with off-policy modification to the lower level transitions [Nachum et al., 2018]. Mahajan and Tulabandhula [2017] leverage a symmetric and interpretable latent decomposition through a tree for reward histories, which they leverage to select symmetric states in a minibatch. Trimponias and Dietterich [2023] achieve safety and sample efficiency by factoring the state and rewards into endogenous and exogenous factors and using a reward correction for the endogenous MDP.

Auxiliary Learning objectives Skill-based methods transfer skills between morphologically different agents by learning n invariant subspace and using that to create a transfer auxiliary objective (through a reward signal) Gupta et al. [2017], or an entropic term for policy regularization Hausman et al. [2018]. Li et al. [2021] tackles sample efficiency for subtask discovery [Solway et al., 2014] in HRL by composing values of the sub-trajectories under the current policy, which they subsequently use for behavior cloning. Zhang et al. [2020c] use latent decomposition as policy regularization to study adversarially perturbed MDPs for robustness and, potentially, safety. Kumar et al. [2022] tackles generalization through an auxiliary loss based on the MSE between a prediction of the board state and actual state to regularize the agent towards human-like inductive biases.

Constraints and Baselines Constrained optimization is commonplace in Safe RL. Yu et al. [2022] use a factored space of safe and unsafe states to constrain the value function, thus, allowing persistent safety conditions. Lee et al. [2022] recursively learn a latent subset of safe actions using factored states to implicitly influences the optimization procedure. Jain et al. [2021a] apply safety to HRL by modifying the option critic to restrict exploration to non-risky states

Works such as Wu et al. [2018] apply action factorization to as a baseline to reduce the variance of policy gradients, thus, improving sample efficiency.

Concurrent Optimization Parallelizing optimization using structural decompositions can help with sample efficiency. Tavakol and Brefeld [2014] model factors that influence the content presented to users as an FMDP, and use them to ensemble factored value function in a parallel regime. In a similar vein, Saxe et al. [2017] use a factored reward decomposition in a hierarchical setting to decompose the Multi-task problem into a linear combination of individual task MDPs. Metz et al. [2017] tackle multi-dimensional action spaces by discretizing continuous sub-action, extending the MDP for each sub-action to an undiscounted lower-level MDP, and modifying the backup for each Q value. Balaji et al. [2020] capture the utilize relational decompositions to mask the inputs in a Factored Neural Network.

A.4 Auxiliary Model Pattern

This pattern captured structural decomposition in learned Model(s), that can subsequently be used to generate experiences, either fully or partially.

Space	Type	Efficiency	Generalization	Interpretabiltiy	Safety
States	Latent	Gasse et al. [2021], Wang et al. [2022], Hafner et al. [2023], van der Pol et al. [2020], Ghorbani et al. [2020], Tsividis et al. [2021]	van der Pol et al. [2020], Wang et al. [2022], Hafner et al. [2023], Hafner et al. [2020], Zhang et al. [2021b], Tsividis et al. [2021]		Simao et al. [2021]
	Factored	Innes and Lascarides [2020], Seitzer et al. [2021], Andersen and Konidaris [2017]	Young et al. [2022]		
	Relational	Chen et al. [2020], Biza et al. [2022b], Biza et al. [2022a], Kipf et al. [2020], Tsividis et al. [2021]	Biza et al. [2022b], Biza et al. [2022a], Veerapaneni et al. [2020], Kipf et al. [2020], Tsividis et al. [2021]	Xu and Fekri [2021]	
	Modular	Abdulhai et al. [2022], Andersen and Konidaris [2017]			
Actions	Latent	van der Pol et al. [2020]	van der Pol et al. [2020]		
	Factored	Spooner et al. [2021], Geißer et al. [2020], Innes and Lascarides [2020]			
	Relational	Biza et al. [2022b]	Biza et al. [2022b]		
Rewards	Latent	van der Pol et al. [2020]	Zhang et al. [2021b], van der Pol et al. [2020], Lee and Chung [2021], Sohn et al. [2018]		
	Factored				Wang et al. [2020]
	Relational		Sohn et al. [2018]		
	Modular		Sohn et al. [2018]		
Dynamics	Latent	Woo et al. [2022]	Zhang et al. [2021b], Woo et al. [2022], van der Pol et al. [2020]	van Rossum et al. [2021]	
	Modular	Abdulhai et al. [2022]			

Models with structured representations Young et al. [2022] utilize factored decomposition for state space to demonstrate the benefits of model-based methods in combinatorially complex environments. In a similar vein, the dreamer models [Hafner et al., 2020, 2023] utilize latent representations of pixel-based environments.

Object-oriented representation for states can help bypass the need to learn latent factors using CNNs in MBRL [Biza et al., 2022a], or as random variables whose posterior can be refined using NNs [Veerapaneni et al., 2020]. Graph (Convolutional) Networks Zhang et al. [2019b] can capture rich higher-order interaction data, such as crowd navigation Chen et al. [2020], or invariances [Kipf et al., 2020] Action equivalences can help learn latent models (Abstract MDPs) van der Pol et al. [2020] for planning and Value Iterations.

Models for task-specific decompositions Another way to utilize decompositions in models is to capture task-specific decompositions. Models that capture some form of relevance, such as observational and interventional data in Causal RL [Gasse et al., 2021], or task-relevant vs irrelevant data [Fu et al., 2021] can help with Generalization and Sample Efficiency gains. Latent representations help models capture control-relevant information Wang et al. [2022] or subtask dependencies [Sohn et al., 2018].

Models for safety usually incorporate some measure of cost to abstract safe states [Simao et al., 2021], or unawareness to factor states and actions [Innes and Lascarides, 2020].

Models can directly guide exploration mechanisms through latent causal decompositions [Seitzer et al., 2021] and state subspaces Ghorbani et al. [2020] to gain sample efficiency. Generative methods such as CycleGAN [Zhu et al., 2017] are also very good ways to use Latent models of different components of an MDP to generate counterfactual trajectories Woo et al. [2022]

A.5 Warehouse Pattern

Warehousing refers to using structural decomposition to create a database of entities in the solution space, such as value functions, policies, or models. The inherent modularity in such methods leads them to focus on knowledge reuse as a central theme, and their online nature often overlaps with continual settings.

Space	Type	Efficiency	Generalization	Interpretabiltiy	Safety
Goals	Factored		Mendez et al. [2022], Devin et al. [2017]		
	Relational			Prakash et al. [2022]	
	Modular	Gehring et al. [2021]	Mendez et al. [2022]	Prakash et al. [2022]	
States	Latent		Hu and Montana [2019]		
	Factored	Mankowitz et al. [2015]	Mendez et al. [2022], Goyal et al. [2020]		
	Modular		Mendez et al. [2022], Goyal et al. [2020]		
Actions	Latent		Gupta et al. [2017]		
	Modular	Li et al. [2018]			
Rewards	Factored		Haarnoja et al. [2018b], Mendez et al. [2022]		
	Modular		Haarnoja et al. [2018b], Gaya et al. [2022a], Gaya et al. [2022b], Mendez et al. [2022]		
Dynamics	Factored	Shyam et al. [2019]	Devin et al. [2017], Mendez et al. [2022]		
	Modular		Gaya et al. [2022a], Gaya et al. [2022b], Mendez et al. [2022]		
Policies	Latent		Gupta et al. [2017]	Verma et al. [2018]	
	Modular	Florensa et al. [2017], Heess et al. [2016], Eysenbach et al. [2019], Raza and Lin [2019], Mankowitz et al. [2015]	Florensa et al. [2017], Heess et al. [2016]	Verma et al. [2018]	

Policy Warehousing Policy subspaces [Gaya et al., 2022b] is a relatively new concept that utilizes shared latent parameters in policies to learn a subspace that can be subsequently combined linearly to create new policies. Extending these subspaces by warehousing additional policies naturally extends them to continual settings [Gaya et al., 2022a]

Task factorization using goals and rewards endows warehousing policies and Q values in multi-task lifelong settings. Mendez et al. [2022] treat a multi-task lifelong problem as a relationship graph between existing tasks, generated from a latent space. Devin et al. [2017] factor the MDP into agent-specific and task-specific degrees of variation, for which individual modules can be trained. Hu and Montana [2019] use a paired variational encoder-decoder model to disentangle the control of morphologically different agents into shared and agent-specific factors. Raza and Lin [2019] partition the agent's problem into interconnected sub-agents that learn local control policies.

Methods in URL and HRL that apply this pattern typically focus on the skills framework, where the warehousing is in the form of learned primitives. These can subsequently be used for maximizing mutual information in lower layers [Florensa et al., 2017], sketching together a policy Heess et al. [2016], diversity-seeking priors in continual settings Eysenbach et al. [2019], or for partitioned states spaces Mankowitz et al. [2015]. In a similar vein, Gupta et al. [2017] apply the warehouse pattern on a latent embedding space, learned using auxiliary optimization.

Decomposed Models Decompositions that inherently exist in models lead to approaches that often ensemble multiple models that individually reflect different aspects of the problem. Goyal et al. [2021] capture the dynamics in individual modules that sparsely interact and use attention mechanisms [Vaswani et al., 2017] for ensembling them. Biza et al. [2022b] bind actions to object-centric representations using factored world models. Lee and Chung [2021] ensemble dynamics into a model for better few-shot adaptation to unseen MDPs. Abdulhai et al. [2022] mitigate the sample inefficiency of Deep Option Critic [Bacon et al., 2017] using subsets of state-space.

A.6 Environment Generation Pattern

Space	Type	Efficiency	Generalization	Interpretabiltiy	Safety
Goals	Relational	Illanes et al. [2020]	Kumar et al. [2022]		
	Modular	Kulkarni et al. [2016], Illanes et al. [2020]			
State	Latent		Wang et al. [2021]		
	Factored	Lu et al. [2018], Mirsky et al. [2022]	Mirsky et al. [2022]	Lu et al. [2018], Mirsky et al. [2022]	
	Relational	Lu et al. [2018]		Lu et al. [2018]	
Rewards	Latent		Wang et al. [2021], Lee and Chung [2021]		
	Factored	Chu and Wang [2023]			
	Relational	Gur et al. [2021]		Gur et al. [2021]	
	Modular	Illanes et al. [2020]			
Dynamics	Latent		Kumar et al. [2021]		
	Factored	Chu and Wang [2023], Mirsky et al. [2022]	Mirsky et al. [2022]	Mirsky et al. [2022]	
	Relational	Illanes et al. [2020]			
	Modular	Illanes et al. [2020], Mirsky et al. [2022]	Mirsky et al. [2022]	Mirsky et al. [2022]	

In this pattern, structural information is used to create task, goal, or dynamics distributions from which MDPs can be sampled. This subsumes the idea of procedurally generated environments, while additionally incorporating methods that use auxiliary models inducing structure in the environment generation process. The decomposition is reflected in the aspects of the environment generation that are impacted by the generative process, such as dynamics, reward structure, state space, etc. Given the online nature of this pattern, methods in this pattern end up addressing curriculum learning, in one way or another.

Kumar et al. [2021] generate environments with compositional structure using rule-based grammars, where the decompositions particularly impact the transition dynamics. This allows them to train agents with an implicit compositional curriculum. This is further used by Kumar et al. [2022] in their auxiliary optimization procedure. Wang et al. [2021] use a latent graphical model to generate the state-space, reward functions, and transition dynamics.

Lee and Chung [2021], even though using an auxiliary model pattern, further apply their latent dynamics model to generate imagine task distributions that are used to generalize to out-of-distribution tasks. Chu and Wang [2023] explore task similarities by meta-learning a clustering method through an exploration policy. In a way, they recover a factored decomposition on the task space where individual clusters can be further used for policy adaptation.

A.7 Explicitly Designed

This pattern encompasses all methods where the inductive biases manifest in specific architectures or setups that reflect the decomposability of the problem that they aim to utilize. Naturally, this includes highly specific Neural architectures, but it also easily extends to other methods like sequential

architectures to capture hierarchies, relations, etc. Crucially, the usage of structural information is limited to the specificity of the architecture and not any other part of the pipeline.

Space	Type	Efficiency	Generalization	Interpretabiltiy	Safety
Goals	Factored	Zhou et al. [2022]	Zhou et al. [2022]	Alabdulkarim and Riedl [2022]	
	Relational	Zhou et al. [2022]	Zhou et al. [2022]		
States	Latent	Wang et al. [2016]	Yang et al. [2020]		
	Factored	Zhou et al. [2022]	Zhou et al. [2022]		
	Relational	Zhou et al. [2022],Mambelli et al. [2022],Shanahan et al. [2020],Zambaldi et al. [2019]	Zhou et al. [2022],Mambelli et al. [2022],Shanahan et al. [2020],Zambaldi et al. [2019],Lampinen et al. [2022]	Zambaldi et al. [2019], Payani and Fekri [2020]	
	Modular				
Actions	Latent	Wang et al. [2016]			
	Factored	Tavakoli et al. [2018]		Tavakoli et al. [2018]	
	Relational	Garg et al. [2020]		Garg et al. [2020]	
Rewards	Latent		Yang et al. [2020]		
	Modular				
Dynamics	Latent	der Pol et al. [2020]	D'Eramo et al. [2020]		
	Factored	Srouji et al. [2018],Hong et al. [2022]			
	Relational		Lampinen et al. [2022]		
Policies	Relational	Oliva et al. [2022], Garg et al. [2020]	Wang et al. [2018]	Garg et al. [2020]	
	Modular		Shu et al. [2018]	Shu et al. [2018],Mu et al. [2022b]	