Machine Learning (2022) 111:4217–4247
https://doi.org/10.1007/s10994-022-06242-4

A taxonomy for similarity metrics between Markov decision processes

Javier García[1] · **Álvaro Visús**[2] · **Fernando Fernández**[2]

Received: 14 July 2021 / Revised: 13 May 2022 / Accepted: 8 September 2022 /
Published online: 14 October 2022

Abstract
Although the notion of task similarity is potentially interesting in a wide range of areas such as curriculum learning or automated planning, it has mostly been tied to transfer learning. Transfer is based on the idea of reusing the knowledge acquired in the learning of a set of source tasks to a new learning process in a target task, assuming that the target and source tasks are *close enough*. In recent years, transfer learning has succeeded in making reinforcement learning (RL) algorithms more efficient (e.g., by reducing the number of samples needed to achieve (near-)optimal performance). Transfer in RL is based on the core concept of *similarity*: whenever the tasks are *similar*, the transferred knowledge can be reused to solve the target task and significantly improve the learning performance. Therefore, the selection of good metrics to measure these similarities is a critical aspect when building transfer RL algorithms, especially when this knowledge is transferred from simulation to the real world. In the literature, there are many metrics to measure the similarity between MDPs, hence, many definitions of *similarity* or its complement *distance* have been considered. In this paper, we propose a categorization of these metrics and analyze the definitions of *similarity* proposed so far, taking into account such categorization. We also follow this taxonomy to survey the existing literature, as well as suggesting future directions for the construction of new metrics.

Keywords Markov decision processes · Similarity metrics · Transfer learning

Editor: Kurt Driessens.

✉ Javier García
franciscojavier.garcia.polo@usc.es

Álvaro Visús
avisus@pa.uc3m.es

Fernando Fernández
ffernand@inf.uc3m.es

[1] Departamento de Electrónica y Computación, Universidad de Santiago de Compostela, Santiago de Compostela, Spain

[2] Departamento de Informática, Universidad Carlos III de Madrid, Leganés, Spain

1 Introduction

Markov decision processes (MDPs) are a common way of encoding decision-making problems in Reinforcement Learning (RL) tasks (Sutton and Barto, 2011). In RL, an MDP is considered to be solved when a policy (i.e., a way of behaving for each state) has been discovered which maximizes the long-term expected return. However, although RL is known as an effective machine learning technique, it might perform poorly in complex problems, leading to a slow rate of convergence. This issue magnifies when facing realistic continuous problems where the curse of dimensionality is inevitable. Transfer learning in RL is a successful technique to remedy such a problem. Specifically, rather than learning a new policy for every MDP, a policy could be learned on one MDP, then transferred to another, similar MDP, and either used as is, or treated as a starting point from which to learn the new policy. Clearly, this transfer cannot be done successfully between any two MDPs, but only in the case they are *similar*.

Therefore, in this context, one question arises: when are two MDPs *similar*? In this paper, we consider the concept of *similar* to be related to the notion of "positive transfer" (Taylor and Stone, 2009). Formally, positive transfer happens when the knowledge in the source task contributes to the improved performance of learning in the target task, and it is considered a negative transfer otherwise, i.e., when the transfer hurts the learning performance when compared with learning from scratch. Additionally, the greater the improvement in the target task, i.e., the greater the positive transfer, the more similar the tasks have to be considered. It is important to be aware of the fact that, based on this description, the concept of *similarity* might not be related to the *structural* similarities between the MDPs. So the correct selection of metrics that allows us to measure the similarity between MDPs is a critical issue in transfer learning, precisely to avoid the negative transfer. Obviously, the use of the positive transfer to measure the similarity between two tasks has a major drawback: the similarity measure between MDPs is obtained *after* the transfer has been run, when really the ideal would be to compute this similarity *before* it, particularly if the point is to use the task similarity measure to choose a task to use in transfer. This issue magnifies when the transfer happens between simulation and the real world, where it is imperative for the efficient and safe deployment of previously learned knowledge.

The literature in transfer learning has proposed different metrics to measure the level of similarity between MDPs, hence, different definitions of the concept of *similarity* have been considered so far. This paper surveys the existing task similarity metrics and contributes a taxonomy that, in its roots, classifies them into two clearly distinct categories: *model-based*, and *performance-based* metrics. We consider such a distinction as a core contribution, allowing us to categorize metrics in a novel and useful way. Model-based metrics are based on the *structural* similarities between the MDP models. Such model-based metrics can be computed in different ways depending on what elements of the MDP models come into play to compute the similarity (Ammar et al., 2014; Taylor et al., 2008c; Milner, 1982; Castro and Precup, 2011; Svetlik et al., 2017). The major strength of these approaches lies in that they can be computed a priori, i.e., before the transfer happens. Therefore, they are independent of the transfer algorithm that is later used to transfer knowledge from one task to another. However, most of them require to know in advance the exact MDP models or accurate approximations of them. Instead, performance-based metrics are computed by comparing the *performance* of the learning agents in the source task and the target task. Such a performance

comparison can be done in two different ways: by comparing the resulting policies from learning in the source task and the target task (Carroll and Seppi, 2005; Karimpanal and Bouffanais, 2018) or, from a transfer point of view, by measuring the transfer gain, i.e., the positive transfer (Mahmud et al., 2013; Sinapov et al., 2015; Fernández and Veloso, 2013). Many metrics can be used to measure such a transfer gain (e.g., jumpstart, asymptotic performance, total reward) (Taylor and Stone, 2009). In some ways, this transfer gain could be the best method for measuring similarity between two tasks (Carroll and Seppi, 2005). Unfortunately, it is often difficult to compute all of these performance-based measures before actually solving the target task, since most of them require to be computed a posteriori, i.e., after the learning processes. However, there are a few exceptions to this rule, within which, for example, the similarity is computed on-line, i.e., *during* solving the target task (Fernández and Veloso, 2013).

Therefore, it is important to bear in mind that, despite their different nature, both model-based and performance-based metrics allow us to measure the similarity between tasks. This survey aims to categorize and discuss the main lines of current research within the computation of similarity metrics between MDPs. Our main purpose is to highlight the advantages and disadvantages of the surveyed metrics, making it easier to identify crossing points and open problems for which research communities could merge their expertise to work on, bridging the gap between the literature and the application of all of these metrics in real-world complex problems. To the best of our knowledge, this is the first survey focused on similarity metrics between MDPs. Previously, Lan et al. (2021) published an extraordinary paper surveying and proposing new similarity metrics, but it is focused on metrics *between states* and not *between MDPs*, which is the real purpose of this paper. In other words, Lan et al. (2021) focus on *state similarity*, whilst this paper focuses on *task similarity*. Nevertheless, the metrics presented by Lan et al. (2021) are discussed here as a good starting point to compose similarity metrics between MDPs. We hope our taxonomy applies to a wide range of researchers, not just those interested in transfer learning. For example, the proposed metrics could also be a critical step forward to sort the samples and tasks in *Curriculum Learning* (Narvekar et al., 2020), or could be used to measure the distance between simulation and the real world in a *Sim-to-Real* context (Zhao et al., 2020). They could also be used to understand the similarities within a set of tasks in *Multi-task Learning* (Shui et al., 2019), or *Automated Planning* (Fernández et al, 2011).

Since different audiences are expected to read this survey, the following guide provides forward references to key insights and sections for target groups with different needs and motivations:

- If you are familiarized with the main concepts of RL and transfer learning, you can skip Sect. 2 and go directly to Sect. 3.
- If you are interested in just an overview of the similarity metrics and how they are organized, go to Sect. 3.
- If you are interested in a deep understanding of the different approaches, you will need to read Sects. 4 and 5 .
- If you want a comparative analysis about what is the best method to use for a specific task, you will find your answer in Sect. 6.
- If you are interested in this area and willing to go forward, see Sect. 7 to see the future directions.

2 Background

This section introduces key concepts required to better understand the rest of the paper. First, some background in RL is introduced (Sect. 2.1), then the main concepts of RL transfer are visited (Sect. 2.2), and finally the concepts of similarity and distance (Sect. 2.3).

2.1 Reinforcement learning

Typically, RL tasks are described as Markov Decision Processes (MDPs) represented by tuples in the form $\mathcal{M} = \langle S, A, T, R \rangle$, where S is the state space, A is the action space, $T : S \times A \to S$ is the transition function between states, and $R : S \times A \to \mathbb{R}$ is the reward function (Sutton and Barto, 2011). At each step, the agent is able to observe the current state, and choose an action according to its policy $\pi : S \to A$. The goal of the RL agent is to learn an optimal policy π^* that maximizes the return $J(\pi)$:

$$J(\pi) = \sum_{k=0}^{K} \gamma^k r_k \tag{1}$$

where r_k is the immediate reward obtained by the agent on step k, γ is the discount factor, which determines how relevant the future is (with $0 \leq \gamma \leq 1$), and K is a final time step for *finite-horizon* models (including the possibility of $K = \infty$ and $0 \leq \gamma < 1$ for *infinite-horizon* models). On the one hand, if the task is an *episodic* task, the interaction between the agent and the environment tends to be divided into episodes. In *finite-* and *infinite-horizon episodic* tasks, an episode always ends when reaching a terminal state but, for *finite-horizon* tasks, it also ends when a fixed number of steps K has passed. On the other hand, a task can be *infinity-horizon continuing*, which means the task will never end. With the goal of learning the policy π, Temporal Differences methods (Sutton and Barto, 2011) estimate the sum of rewards represented in Eq. (1). The function that estimates the sum of rewards, i.e., the return for each state s given the policy π is called the value-function $V^\pi(s) = E[J(\pi)|s_0 = s]$. Similarly, the action-value function $Q^\pi(s, a) = E[J(\pi)|s_0 = s, a_0 = a]$ is the estimation of the value of performing a given action a at a state s being π the policy followed. The corresponding value function and action-value function for the optimal policy π^* are denoted respectively V^* and Q^*.

The Q-learning algorithm (Watkins, 1989) is one of the most widely used for computing the action-value function. In small domains with a small number of states and actions, the Q^π function and π can be fully represented with a lookup table. However, as the state and action spaces grow, a different approach is required. One way to extend the Q^π function to continuous state-action space, is to discretize the environment in order to reduce such space (García et al., 2010), thus the use of a tabular representation of Q^π is still possible. However, in such continuous scenarios, both V^π and Q^π functions are typically estimated using a universal function approximation such as an artificial neural network (Wiering and van Otterlo, 2014). In this case, the value function is expressed as a linear $V^\pi(s) = \theta^T \phi(s)$ or non-linear $V^\pi(s) = V^\pi(\phi(s), \theta)$ combination of a parameter vector θ and a feature vector $\phi(s)$. Equivalently, the Q^π function can also be expressed in terms of θ and ϕ (Van Hasselt, 2012).

2.2 Transfer learning for reinforcement learning

In the transfer learning scenario we assume there is an agent who previously has addressed a set of source tasks represented as a sequence of MDPs, $\mathcal{M}_1, \ldots, \mathcal{M}_n$. If these tasks are somehow "similar" to a new task $\mathcal{M}_{n+1}$, then it seems reasonable the agent uses the acquired knowledge solving $\mathcal{M}_1, \ldots, \mathcal{M}_n$ to solve the new task $\mathcal{M}_{n+1}$ faster than it would be able to from scratch. Transfer learning is the problem of how to obtain, represent and, ultimately, use the previous knowledge of an agent (Torrey and Shavlik, 2010; Taylor and Stone, 2009).

However, transferring knowledge is not an easy endeavour. On the one hand, we can distinguish different transfer settings depending on whether the source and the target tasks share or not the state and action spaces, the transition probabilities and the reward functions. It is common to assume that the tasks share the state space and the action set, but differing the transition probabilities and/or reward functions. However, in case the tasks do not share the state and/or the action spaces, it is required to build mapping functions, $\mathcal{X}_S(s_t) = s_s$, $\mathcal{X}_A(a_t) = a_s$, able to map a state s_t or action a_t in the target task to a state s_s or action a_s in the source task. Such mapping functions require not only knowing if two tasks are related, but *how* they are related, which means an added difficulty. On the other hand, it is required to select what type of information is going to be transferred. Different types of information have been transferred so far ranging from instance transfer (a set of samples collected in the source task) to policy transfer (i.e., the policy π learned in the source task). Nor is this a simple task, because depending on how much and how the source and the target tasks are related, it could be transferred one type of information or another.

Finally, the most "similar" task among $\mathcal{M}_1, \ldots, \mathcal{M}_n$ to solve $\mathcal{M}_{n+1}$ should be selected in the hope that it produces the most positive transfer. For this purpose, similarity metrics could be used, which translate into a measurable quantity of how related two tasks are.

2.3 Similarity and distance metrics

Similarity metrics are a very important part of transfer learning, as they provide a measure of distance between tasks. A similarity function $s(\cdot, \cdot)$, or its complementary distance function $d(\cdot, \cdot)$, is a mathematical function that assigns a numerical value to each pair of concepts or objects in a given domain. This value measures how similar these two concepts or objects are: if they are very similar, it is assigned a very low distance, and if they are very dissimilar, it is assigned a larger distance (Ontañón, 2020). Intuitively, for each distance function $d(\cdot, \cdot)$ we can define its associated similarity function $s(\cdot, \cdot) = u/(1 + d(\cdot, \cdot))$, where u is the maximum similarity value, usually $u = 1$. For simplicity, in this survey we use the distance function $d(\cdot, \cdot)$ to formulate the distance between MDPs, knowing that this distance also captures the similarity between tasks.

3 Taxonomy of similarity metrics for MDPs

We consider there are two tasks, $\mathcal{M}_i$ and $\mathcal{M}_j$, described formally by the tuples $\mathcal{M}_i = \langle S_i, A_i, T_i, R_i \rangle$ and $\mathcal{M}_j = \langle S_j, A_j, T_j, R_j \rangle$, where they could share (or not) the state space, the action space, or the transition and reward dynamics.

Definition 1 Given two tasks $\mathcal{M}_i$ and $\mathcal{M}_j$, we define a task distance metric as a heuristic function $d(\mathcal{M}_i, \mathcal{M}_j) \rightarrow [0, \infty)$, such that if $d(\mathcal{M}_i, \mathcal{M}_j) < d(\mathcal{M}_k, \mathcal{M}_j)$, then $\mathcal{M}_i$ is considered more similar to $\mathcal{M}_j$ than $\mathcal{M}_k$.[1]

Definition 1 allows us to use the function $d(\cdot, \cdot)$ to obtain a partial order between tasks in such a way that we can select the more *similar* one. Ideally, the concept of similarity should be related to the concept of positive transfer: the smaller the distance $d(\mathcal{M}_i, \mathcal{M}_j)$, the greater the positive transfer. However, in most of the cases, similarity metrics do not provide guarantees for this ideal behavior. Additionally, $d(\mathcal{M}_i, \mathcal{M}_j)$ should be computed *before* or, at least, *during* the transfer experiment, in order to select an adequate task to use in transfer. However, the literature proposes different ways to compute $d(\mathcal{M}_i, \mathcal{M}_j)$.

In this paper, we consider two main trends for the computation of the distance metric $d(\mathcal{M}_i, \mathcal{M}_j)$. Such trends are depicted in Table 1. The first one measures the structural or model similarities between the given MDPs. The second measures the similarities by using the performance of the learning agent in both the source and the target tasks.

Model-Based Metrics As regards to the first, they measure the degree of similarity between a source and a target task by using their corresponding MDP models (i.e., states, actions, transition and rewards dynamics). There are several alternatives to these model-based metrics depending on what components of the MDPs are taken into account. In this survey, we categorize these metrics in four groups: (i) transition and reward dynamics, (ii) transitions, (iii) state and actions and (iv) states:

- *Transition and reward dynamics* They require complete knowledge of the MDP models both of the source task and the target task. We distinguish three ways of computing similarity metrics using such a complete knowledge: (i) by a sort of metrics based on state abstraction (or state aggregation) techniques (Li et al., 2006; Ferns et al., 2004, 2012; Castro, 2020), (ii) by compliance metrics (Lazaric et al., 2008; Lazaric, 2008; Fachantidis et al., 2015; Fachantidis, 2016) and (iii) by metrics based on the construction of *MDP graphs* (Kuhlmann and Stone, 2007; Wang et al., 2019).
  - *State abstraction* In RL, it is a common practice to aggregate states in order to obtain an *abstract* description of the problem, i.e., a more compact and easier representation of the task to work with (Giunchiglia and Walsh, 1992; Li et al., 2006). These approaches are based on the same common principle: if a number of states are considered to be *similar*, they can be aggregated as a single one. This same principle can be also used to compute the *similarity* between two states belonging to different MDPs. In fact, the cumulative *similarity* between each pair of these states could be used to compute a sort of similarity metric between the MDPs. In this paper, we survey two of these methods which actually have been used for transfer in RL: bisimulation (Ferns et al., 2004, 2012; Castro and Precup, 2011; Song et al., 2016), and homomorphism (Ravindran and Barto, 2002; Sorg and Singh, 2009), where both of them require complete knowledge or accurate approximations of the MDP models to compute the similarity between states. Although only bisimulation

[1] It should be noted that in the context of task similarity such a measure of distance need does not need to be a "metric" in the mathematical sense of this term, but it needs to allow us to define a partial order between tasks.

Table 1 Taxonomy and summary of the similarity metrics considered in this survey

- **Similarity Metrics**
  - **Model Based**
    - **Transition & reward dynamics**
      - **Abstraction**
        - **Bisimulation**
        - *Li et al (2006)*
        - *Ferns et al (2004, 2012)*
        - *Castro and Precup (2011)*
        - *Song et al (2016)*
        - *Castro et al (2021)*
        - **Homomorphism**
        - *Ravindran and Barto (2002)*
        - *Sorg and Singh (2009)*
        - *Taylor et al (2008a)*
      - **Compliance**
      - *Lazaric et al (2008)*
      - *Lazaric (2008)*
      - *Fachantidis et al (2011, 2015)*
      - **MDP graphs**
      - *Kuhlmann and Stone (2007)*
      - *Liu and Stone (2006)*
      - *Wang et al (2019)*
    - **Transitions**
    - *Taylor et al (2008c)*
    - *Ammar et al (2014)*
    - *Castro (2020)*
    - **Rewards**
    - *Carroll and Seppi (2005)*
    - *Tao et al (2021)*
    - *Gleave et al (2020)*
    - **State & Actions**
    - *Taylor et al (2008b)*
    - *Narayan and Leong (2019)*
    - **States**
    - *Svetlik et al (2017)*
    - *Celiberto Jr et al (2011)*
  - **Performance Based**
    - **Policy similarity**
      - **Policy values**
      - *Carroll and Seppi (2005)*
      - *Zhou and Yang (2020)*
      - **Policy parameters**
      - *Karimpanal and Bouffanais (2018)*
      - *Ferrante et al (2008)*
    - **Transfer gain**
      - **Off-line transfer gain**
      - *Mahmud et al (2013)*
      - *Sinapov et al (2015)*
      - *Carroll and Seppi (2005)*
      - *Zhan et al (2016)*
      - **On-line transfer gain**
      - *Azar et al (2013)*
      - *Fernández and Veloso (2013)*
      - *Li and Zhang (2017)*

and homomorphism are discussed in detail in this paper, we consider other state abstraction techniques might be also used as similarity metrics between MDPs.
  - *Compliance* The *compliance* measure is defined as the probability of a sample $\langle s, a, s', r\rangle$ in the target task of being generated in the source task (Lazaric et al., 2008; Fachantidis, 2016; Fachantidis et al., 2015). Therefore, it is easy to deduce that the compliance between the entire target task and the entire source task allows us to measure the similarity between the two tasks.
  - *MDP graphs*. They are based on the construction of graphs that represent the transition and the reward functions both of the source task and the target task (Kuhlmann and Stone, 2007; Liu and Stone, 2006; Wang et al., 2019). Then, they find structural similarities between tasks based on graph-similarity or graph-matching algorithms. Such approaches are based on an interesting idea for similarity computation: the alternative representation of tasks through descriptive structures that allow an *easy* comparison between them.

- *Transitions* The metrics considered in this category use tuples in the form $\langle s, a, s'\rangle$ to measure the similarity between MDPs (Taylor et al., 2008c; Ammar et al., 2014). By using such tuples, they model the behavioral dynamics of the two MDPs to be compared, and then they try to find differences between them.
- *Rewards* The metrics considered in this category use tuples in the form $\langle s, a, r\rangle$ to measure the similarity between MDPs. This set includes different techniques (Carroll and Seppi, 2005; Tao et al, 2021; Gleave et al, 2020).
- *State & actions* These metrics use pairs in the form $\langle s, a\rangle$ both in the source task and the target task to compute the similarity between MDPs. Such pairs can be used in different ways resulting in different similarity metrics (Carroll and Seppi, 2005; Taylor et al, 2008b; Narayan and Leong, 2019).
- *States* Finally, metrics in this category use the state space both in the source and the target task to compute the similarity between them (Svetlik et al., 2017). Another area of research that is relevant in this category is that of case-based reasoning (CBR) (Aamodt and Plaza, 1994). RL approaches based on CBR use a similarity function between the states in the target task and the states stored in a case base corresponding to a previous source task (Celiberto Jr et al., 2011). Such similarity function could be used to measure the similarity between the state spaces, hence, the similarity between the two tasks.

Performance Based. As regards the second major category in the proposed taxonomy, performance-based metrics are based on the *performance* of the agents in the source task and the target task, where this *performance* can be related to the policies themselves learned by the agents in these tasks, or to the transfer gain an agent obtains reusing the knowledge of a source task in a target task. So, we distinguish two different approaches to overcoming the problem of computing such performance-based similarities: (i) by the policy similarity and (ii) by the transfer gain obtained transferring the knowledge from a source task to the target task.

- *Policy similarity* They are based on the use of the learned value function V^{π} or the action-value function Q^{π}, or equivalently, on the behavioral policies π obtained in the source task and the target task. Therefore, these metrics require the full (or partial) learning of these policies before the computation of the similarity between tasks. Such a comparison can be conducted in two different ways depending on what is being compared: (i) the policy values, or (ii) the policy parameters.

  - *Policy values* In this case, the comparison is conducted by observing the specific values of the Q^{π}-function or the V^{π}-function corresponding to the source and the target tasks (Carroll and Seppi, 2005; Zhou and Yang, 2020). Therefore, in this case, it is really being measured the degree of similarity of the policies obtained in both tasks.
  - *Policy parameters*. In RL, the value function V^{π} or the action-value function Q^{π} usually are represented as a parameter vector θ. Intuitively, the metrics within this category compare the particular weights of the parameter vectors corresponding to the value functions of the source task and the target task in order to measure the similarity between them (Karimpanal and Bouffanais, 2018). Therefore, these metrics are only applicable with parametric representations of policies, and not with tabular representations. Such an approach opens the door to the comparison of other policy representations (Ferrante et al., 2008).

- *Transfer gain* In these techniques, the level of similarity is an approximation to the *advantage* gained by using the knowledge in one source task to speed the learning of another target task (Carroll and Seppi, 2005; Carroll, 2005; Taylor and Stone, 2009). So, it is important to bear in mind that these metrics actually require that the transfer experiment be entirely or partially run before measuring the degree of similarity between tasks. Many metrics to measure such a transfer gain are possible, including jumpstart, asymptotic performance, total reward (see (Taylor and Stone, 2009) for a complete listing of metrics for transfer gain). However, regardless of the particular technique used to compute such a transfer gain, the higher the transfer gain, the greater the similarity between the tasks. In this paper, we distinguish two approaches within this category depending on whether the transfer gain is computed *after* or *during* the transfer process: (i) off-line transfer gain, and (ii) on-line transfer gain.

  - *Off-line transfer gain* The transfer gain is estimated as the difference in performance between the learning process with and without transfer (Carroll, 2005; Mahmud et al., 2013; Sinapov et al., 2015). Therefore, it is important to be aware of the fact that, in these approaches, the gain is computed once the learning processes are considered to be finished.
  - *On-line transfer gain* On the contrary, in these approaches, the gain is estimated on-line at the same time that the policy in the target task is computed (Azar et al., 2013; Fernández and Veloso, 2013; Li and Zhang, 2017). In this way, it is possible to decide *on-line* which is the closest task within a library composed of past tasks, so that the knowledge of the selected closest task can have a greater influence on learning about the policy in the new task.

4 Model-based metrics

This section presents in detail the model-based metrics considered in this paper, i.e., the metrics that evaluate the similarity between tasks using the components of their respective MDPs. As presented in Sect. 2.1, the structure of an MDP is formally described by a tuple $\langle S, A, T, R \rangle$. Different model-based metrics result depending on what components of the MDPs take part in the computation of the similarity. Therefore, this survey categorizes the model-based metrics in five groups: (i) transition and reward dynamics, (ii) transitions, (iii) rewards, (iv) state and actions and (v) states.

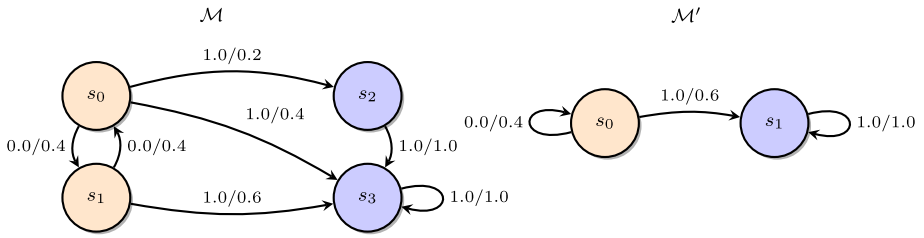

Fig. 1 A bisimulation example where an MDP $\mathcal{M}$ of four state (left) is reduced to an MDP $\mathcal{M}'$ with two abstract states (right). In both $\mathcal{M}$ and $\mathcal{M}'$ there is a single action a that allows a transition from a state s_i to a state s_j. The arcs between the nodes are labeled with $R(s_i, a)/T(s_i, a, s_j)$

4.1 Transition and reward dynamics

This first group of techniques make use of all the components of the MDPs (i.e., the state and action spaces, and the transition and reward dynamics) to compute the similarity metrics. Depending on the use of this information, we can distinguish three different groups of metrics: (i) by a sort of metrics based on state abstraction (or state aggregation) techniques (Li et al., 2006; Ferns et al., 2004, 2012; Castro, 2020), (ii) by compliance metrics (Lazaric et al., 2008; Lazaric, 2008; Fachantidis et al., 2015; Fachantidis, 2016) and (iii) by metrics based on the construction of *MDP graphs* (Kuhlmann and Stone, 2007; Wang et al., 2019).

4.1.1 State abstraction

Learning in a high-dimensional state space not only increases the time and memory requirements of learning algorithms, but also degenerates performance due to the curse of dimensionality (Kaelbling et al., 1996). This motivates the need for state abstraction, the process of grouping states into abstract representations, more compact and easier to work with, while preserving dynamics of the original system. In this paper, we survey two methods for state abstraction: bisimulation (Ferns et al., 2004, 2012; Castro and Precup, 2011; Song et al., 2016), and homomorphism (Ravindran and Barto, 2002; Sorg and Singh, 2009).

Bisimulation Bisimulation considers two states are equivalent (hence, they can be grouped) when for every action, they achieve the same immediate reward and have the same probability of transitioning to classes of equivalent states (Givan et al., 2003; Phillips, 2006). Figure 1 shows an example of bisimulation reduction by transforming an MDP $\mathcal{M}$ of four states to an abstract MDP $\mathcal{M}'$ with two states. Based on the equivalence relation of bisimulation between each pair of states in $\mathcal{M}$, $s_0, s_1 \in \mathcal{M}$ can be grouped as $s_0 \in \mathcal{M}'$, and $s_{2,}, s_3 \in \mathcal{M}'$ can be grouped as $s_1 \in \mathcal{M}$.

However, we are interested in metrics and not in equivalence relations. Such a metric could assign a distance of 1 to states that are not bisimilar, and 0 otherwise, not possessing more distinguishing power than that of bisimulation itself. Therefore, as a desirable property of this metric, it should vary smoothly and proportionally with differences in rewards and transition probabilities. Such a bisimulation metric was first proposed by Ferns et al. (2004) and, although it was originally defined as the distance between states belonging to

the same MDP, the definition can be easily extended as the distance between states belonging to different MDPs.

Definition 2 Given two MDPs, $\mathcal{M}_i$ and $\mathcal{M}_j$, the distance between two states $s_i \in S_i$ and $s_j \in S_j$ is defined as:

$$d(s_i, s_j) = \max_{a \in A} \{(1-c) \cdot |(R_i)^a_{s_i} - (R_j)^a_{s_j}| + c \cdot W((T_i)^a_{s_i}, (T_j)^a_{s_j}; d)\} \tag{2}$$

where $c \in [0, 1)$, $(R_i)^a_{s_i}$ and $(T_i)^a_{s_i}$ are the immediate reward and the probabilistic transition when action $a \in A$ is taken at state s_i in $\mathcal{M}_i$ (resp. $(R_j)^a_{s_j}$ and $(T_j)^a_{s_j}$ in $\mathcal{M}_j$), and $W((T_i)^a_{s_i}, (T_j)^a_{s_j}; d)$ is the Kantorovich [2] distance between the two probabilistic transitions.

The bisimulation metric $d(s_i, s_j)$ is constructed by comparing the transition and reward dynamics of $s_i \in S_i$ and $s_j \in S_j$: the more similar the reward and transition structures of s_i and s_j are, the smaller $d(s_i, s_j)$. $d(s_i, s_j)$ is a (unique) fixed-point metric and it can be calculated iteratively by starting with a metric that is zero everywhere, and iterating until the difference in metric distances between iterations drops below a certain threshold (we refer the reader to Ferns et al. (2004, 2006)). However, bisimilation metrics are difficult to use at scale and compute online, which is why other bisimulation-inspired metrics have recently appeared such as π-bisimulation (Castro, 2020), deep bisimulation for control (Zhang et al., 2021), policy similarity metric (Agarwal et al., 2021), or the MICo distance (Castro et al., 2021). Finally, it would be worth noting that the *model*-irrelevance metric (Li et al., 2006) and its approximate version (Abel et al., 2016) share the same principles of bisimulation: two states are considered similar if they have similar transition and reward functions.

Homomorphism One of the shortcomings of the bisimulation metric presented in Definition 2 is that it requires both MDPs, $\mathcal{M}_i$ and $\mathcal{M}_j$, to have the same action sets, i.e., it requires that the behavior matches for exactly the same actions, which is not always the case. In many practical problems, actions with the exact same label may not match, so it should be allowed correspondences between states by matching their behavior with different actions. This idea is formalized as MDP homomorphism (Ravindran and Barto, 2002; Sorg and Singh, 2009). The aim of abstraction in MDP homomorphism is to group similar state-action pairs instead of just states (Castro and Precup, 2010). Therefore, MDP homomorphisms do not require behavioral equivalence under the same action labels, and this idea was elegantly extended with the lax bisimulation metric proposed by Taylor et al. (2008a).

Definition 3 Given two MDPs $\mathcal{M}_i$ and $\mathcal{M}_j$ the distance between the state-action pairs (s_i, a_i) and (s_j, a_j) is defined as:

$$d_L((s_i, a_i), (s_j, a_j)) = \{(1-c) \cdot |(R_i)^{a_i}_{s_i} - (R_j)^{a_j}_{s_j}| + c \cdot W((T_i)^{a_i}_{s_i}, (T_j)^{a_j}_{s_j}; d)\} \tag{3}$$

where $c \in [0, 1)$, $(R_i)^a_{s_i}$ and $(T_i)^a_{s_i}$ are the immediate reward and the probabilistic transition when action a_i is taken at state s_i in $\mathcal{M}_i$ (resp. $(R_j)^{a_j}_{s_j}$ and $(T_j)^{a_j}_{s_j}$ in $\mathcal{M}_j$), and $W((T_i)^{a_i}_{s_i}, (T_j)^{a_j}_{s_j}; d)$ is the Kantorovich distance between the two probabilistic transitions. From the distance between state-action pairs in Eq. (3) we can then define a state metric as:

[2] Also known as Monge-Kantorovich, Kantorovich-Rubinstein, Hutchinson, Mallows, Wasserstein, Vasserstein, Earth Mover's Distance, Fortet-Mourier, and Dudley.

$$d(s_i, s_j) = \max(\max_{a_i \in A_i} \min_{a_j \in A_j} d_L((s_i, a_i), (s_j, a_j)), \max_{a_j \in A_j} \min_{a_i \in A_i} d_L((s_i, a_i), (s_j, a_j))) \tag{4}$$

Taylor et al. (2008a) demonstrate that the lax bisimulation metric relates more states allowing for more compression than bisimulation metrics (Ferns et al., 2004), as it allows capturing different regularities and other types of special structures in the environment. Bisimulation-based metrics have been successfully used as measures of similarity between states with applications including state aggregation (Li et al., 2006) or representation learning (Comanici et al., 2015). It has also been used to discover regions of state space that can be transferred from one task to another (Castro and Precup, 2010). However, few works have been proposed on how the individual distances between states can be used to determine how similar two tasks are *in toto* (Song et al., 2016). Using the metrics in Definitions 2 and 3 we can compute the distance between all state pairs in $\mathcal{M}_i$ and $\mathcal{M}_j$. Once the distance between all state pairs in $\mathcal{M}_i$ and $\mathcal{M}_j$ is computed, it is required to composite them to compute the distance $d(\mathcal{M}_i, \mathcal{M}_j)$ between the two MDPs.

Definition 4 Given two MDPs $\mathcal{M}_i$ and $\mathcal{M}_j$, we can define the distance between them as:

$$d(\mathcal{M}_i, \mathcal{M}_j) = \boldsymbol{\Phi}(S_i, S_j) \tag{5}$$

where $\boldsymbol{\Phi}$ measures the distance between the state spaces S_i and S_j by using the individual distances between the state pairs, $d(s_i, s_j)$.

It could not be appropriate to simply accumulate or average the distances between all different state pairs. For this reason, Song et al. (2016) define $\boldsymbol{\Phi}(\cdot, \cdot)$ as a function that measures the distance between the sets corresponding to the state spaces S_i and S_j, by using the Hausdorff and the Kantorovich metrics. However, in the transfer experiments conducted, the Kantorovich metric can avoid negative transfer by filtering the dissimilar tasks, while the Hausdorff one does not have such property. Additionally, Song et al. (2016) are only focused on finite MDPs. Much work needs to be done to determine if $\boldsymbol{\Phi}(\cdot, \cdot)$ is also computable for continuous tasks, or if it is possible the use of other metrics between sets (Conci and Kubrusly, 2018).

4.1.2 Compliance

Task compliance was first introduced by Lazaric et al. (Lazaric et al., 2008; Lazaric, 2008) with the goal of transferring samples from a source task to a target task. For this transfer to result in a *positive transfer*, it is required to select source tasks whose samples are similar to those produced in the target task. Such a problem could be stated as a *model identification problem* in which the goal is to identify a particular task from a distribution of tasks, by determining its transition dynamics and reward function (Mendonca et al., 2020). *Compliance* can assist to measure the similarity between tasks by calculating the average probability of the source task generating target's samples (Lazaric et al., 2008; Fachantidis, 2016; Fachantidis et al., 2015).

Definition 5 Given two MDPs, $\mathcal{M}_i$ and $\mathcal{M}_j$, and a set of experience tuples generated in $\mathcal{M}_i$, $D_{\mathcal{M}_i} = \{\tau_0, \ldots, \tau_m\}$ with $\tau_k = \langle s_k, a_k, s'_k, r_k \rangle$, the probability of an experience tuple $\sigma = \langle s, a, s', r \rangle$ in the task $\mathcal{M}_j$ of being generated by $\mathcal{M}_i$ is defined as:

$$P(\sigma|D_{\mathcal{M}_i}) = (T_i)^a_{ss'}(\mathcal{R}_i)^a_{sr} \tag{6}$$

where $(T_i)^a_{ss'}$ is the probability of transiting to s', and $(\mathcal{R}_i)^a_{sr}$ is the probability of generating the reward r after executing the action a in state s in $\mathcal{M}_i$.

Definition 5 provides a formal description of the probability of a sample σ generated in an MDP $\mathcal{M}_j$ of having been generated in an MDP $\mathcal{M}_i$. If instead of having a single σ tuple of the target task M_j, we have a set of tuples, $D_{\mathcal{M}_j}$, Definition 5 could be used to compute the compliance between the entire target task $\mathcal{M}_j$ and the entire source task $\mathcal{M}_i$ by repeating this operation for all samples in $D_{\mathcal{M}_j}$.

Definition 6 Given two MDPs, $\mathcal{M}_i$ and $\mathcal{M}_j$, and two sets of experience tuples generated $D_{\mathcal{M}_i}$ and $D_{\mathcal{M}_j}$ gathered from $\mathcal{M}_i$ and $\mathcal{M}_j$, the compliance between $\mathcal{M}_i$ and $\mathcal{M}_j$ is computed as:

$$\Lambda = \frac{1}{n}\sum_{t=0}^{n} P(\sigma_t|D_{\mathcal{M}_i}) \tag{7}$$

where n is the number of samples in $D_{\mathcal{M}_j}$, and σ_t is the t-th tuple in $D_{\mathcal{M}_j}$.

Λ is not strictly a distance metric but a probability: the more likely the samples of the target task are generated in the source task, the closer Λ to 1. Therefore, *compliance* could be used to obtain a distance metric between MDPs like the ones this survey is looking for, e.g., $d(\mathcal{M}_i, \mathcal{M}_j) = 1 - \Lambda$.

4.1.3 MDP graphs

The methods in this section are based on an interesting principle: the translation of the MDPs into alternative representations that allow to more easily measure the similarities between them. In the particular case of the approaches in this section, such an alternative representation is based on a graph-theoretical perspective, i.e., the states, actions, transition and reward functions can be represented as a graph with nodes and edges (Wang et al., 2019; Kuhlmann and Stone, 2007). Therefore, these approaches are in a way based on the concepts of bisimulation and homomorphism described in Sect. 4.1.1, with the difference that they use specific techniques of structural similarity between graphs to measure the similarity between MDPs.

Definition 7 Given two MDPs $\mathcal{M}_i$ and $\mathcal{M}_j$ and their corresponding alternative representation as graphs, $G_{\mathcal{M}_i}$ and $G_{\mathcal{M}_j}$, we define $d(\mathcal{M}_i, \mathcal{M}_j)$ as inversely related to $\Phi(G_{\mathcal{M}_i}, G_{\mathcal{M}_j})$, where $\Phi(\cdot,\cdot) \rightarrow [0, \infty)$ is a function that measures the structural similarity between $G_{\mathcal{M}_i}$ and $G_{\mathcal{M}_j}$.

Therefore, the representation of MDPs as graphs opens the door to using graph-theoretic similarity metrics to measure the similarity between MDPs. One of these metrics is SimRank (Jeh and Widom, 2002) which is based on the intuition that *two nodes are similar iff their neighbors are similar*. Formally, given a graph $G = \{V, E\}$, where V is the set of nodes and E is the set of directed links between any two nodes in G, the SimRank metric between any two nodes $i, j \in V$ with $i \neq j$ is defined as in Eq. (8), where

$I(i) = \{j|(j,i) \in E, j \in V\}$ denotes the set of neighbours of i, and c is called a decay factor. If $i = j$, then $\phi_{ij} = 1$. In addition, if $i \neq j$ and $I(i) =$ or $I(j) =$, then $\phi_{ij} = 0$.

$$\phi_{ij} = \frac{c}{|I(i)||I(j)|} \sum_{a \in I(i)} \sum_{b \in I(j)} \phi_{ab} \tag{8}$$

Besides the SimRank metric, other node-to-node proximities in graphs have been proposed such as RoleSim (Jin et al., 2014) or MatchSim (Lin et al., 2012). Based on the principles of the graph-theoretic similarity metrics, Wang et al. (2019) propose to first represent an MDP as a bipartite directed graph $G_{\mathcal{M}} = \{V, \Lambda, E, \Psi, p, r\}$, where V are state nodes, Λ action nodes, E is a set of decision edges from state nodes to action nodes, and Ψ is the set of transition edges with each edge $(\alpha, v) \in \Psi$ weighted with a transition probability $p(\alpha, v)$ and a reward $r(\alpha, v)$. Then, Wang et al. (2019) compute the similarities between the state nodes in $G_{\mathcal{M}}$ using a SimRank-inspired metric. In the experimental results, Wang et al. (2019) demonstrate the proposed metric is able to detect certain structural similarities which are undetectable by bisimulation metrics. Although the paper by Wang et al. (2019) only focuses on the computation of the similarity between states in the same MDP, such a computation could be easily extended to states belonging to different MDPs. Thus, the similarities ϕ_{ij} between all state pairs belonging to $\mathcal{M}_i$ and $\mathcal{M}_j$ could be composite by a function $\Phi(\cdot, \cdot)$ in order to compute the similarity between two graphs $G_{\mathcal{M}_i}$ and $G_{\mathcal{M}_j}$, in a similar way as in Sect. 4.1.1.

Other graph similarity metrics studied in the context of task similarity are based on graph isomorphism (McKay and Piperno, 2014). The graph isomorphism problem asks whether two graphs $G_i = \{V_i, E_i\}$ and $G_j = \{V_j, E_j\}$ have the same structure, i.e., if there exist a mapping function $f : V_i \rightarrow V_j$, with $|V_i| = |V_j|$ such that $(u, v) \in E_i$ iff $(f(u), f(v)) \in E_j$. Kuhlmann and Stone (2007) investigate the use of graph isomorphism in the context of MDPs. Specifically, they represent the MDPs as rule graphs instead of bipartite graphs for the particular problem of General Game Playing (Genesereth et al., 2005). Such a rule graph is an accurate abstraction of the MDP problem, and can be properly compared to other rule graphs. In this case, the similarity function $\Phi(\cdot, \cdot)$ is a binary function with value of 1 if $G_{\mathcal{M}_i}$ and $G_{\mathcal{M}_j}$ are isomorphic, and 0 otherwise. However, it would be desirable that $\Phi(\cdot, \cdot)$ vary smoothly with the difference in the MDPs. Therefore, it may be worth investigating the *graph edit distance* which denotes the number of edit steps (insertions, deletions, or updates to nodes or edges) required to transform G_i to G_j (Gao et al., 2010). The function $\Phi(\cdot, \cdot)$ could be inversely related to the number of steps required to transform one graph into the other. Additionally, Kuhlmann and Stone (2007) assumes full knowledge of the transition function. A more general approach is presented by Liu and Stone (2006) where the agent has only a qualitative understanding of the transition function. Liu and Stone (2006) models the problem as a Qualitative Dynamic Bayes Network (QDBN). This assigns types to the nodes and edges, providing additional characteristics to compare.

4.2 Transitions

The metrics in this category use tuples in the form $\langle s, a, s' \rangle$ to measure the similarity between MDPs.

Definition 8 Given two tasks $\mathcal{M}_i$ and $\mathcal{M}_j$, and two sets $\mathcal{D}_{\mathcal{M}_i}$ and $\mathcal{D}_{\mathcal{M}_j}$ of experience tuples in the form $\tau = \langle s, a, s' \rangle$ gathered from $\mathcal{M}_i$ and $\mathcal{M}_j$, we can define the distance between them as $d(\mathcal{M}_i, \mathcal{M}_j) = d(\mathcal{D}_{\mathcal{M}_i}, \mathcal{D}_{\mathcal{M}_j})$.

Ammar et al. (2014) use $\mathcal{D}_{\mathcal{M}_j}$ to build a Restricted Boltzmann Machine (RBM) model which describes the transitions in $\mathcal{M}_j$ in a richer feature space. RBMs are stochastic two-layered energy-based models with generative capabilities for unsupervised learning (Ghojogh et al., 2021). The first layer is the visible layer that represents input data, whereas the hidden layer is used to discover more informative spaces to describe the input data. RBMs have the capability of regenerating visible layer values given a hidden layer configuration. Therefore, the learning process consists of several forward and backward passes, where the RBM tries to reconstruct the input data. Such a reconstruction capability is particularly interesting to discover informative hidden features in unlabeled data (Hinton and Salakhutdinov, 2006). In the specific problem of task similarity, Ammar et al. (2014) first train a RBM model using the tuples in $D_{\mathcal{M}_j}$. Then, they propose feeding the tuples $\tau_k \in \mathcal{D}_{\mathcal{M}_i}$ into this RBM model to obtain a reconstruction τ'_k. Afterward, they compute the Euclidean distance e_k between τ_k and τ'_k. The distance $d(\mathcal{D}_{\mathcal{M}_i}, \mathcal{D}_{\mathcal{M}_j})$ between $\mathcal{D}_{\mathcal{M}_i}$ and $\mathcal{D}_{\mathcal{M}_j}$ is computed as the mean of all errors $d(\mathcal{D}_{\mathcal{M}_i}, \mathcal{D}_{\mathcal{M}_j}) = \frac{1}{n} \sum_{k=1}^{n} e_k$, where n is the number of tuples in $D_{\mathcal{M}_i}$. If $\mathcal{M}_i$ and $\mathcal{M}_j$ are closely related, the reconstruction will be very accurate, and the distance $d(\mathcal{D}_{\mathcal{M}_i}, \mathcal{D}_{\mathcal{M}_j})$, will be close to zero. Similarly, Taylor et al. (2008c) use $\mathcal{D}_{\mathcal{M}_j}$ to learn a one-step transition model $M(s, a) \rightarrow s'$. Afterward, for each tuple $\tau_k = \langle s, a, s' \rangle$ in $\mathcal{D}_{\mathcal{M}_i}$, they compute the Euclidean distance between s' and the predicted state by the model, $M(s, a)$. In this case, the distance $d(\mathcal{D}_{\mathcal{M}_i}, \mathcal{D}_{\mathcal{M}_j})$ is also computed as the average of the Euclidean distances obtained for each $\tau_k \in D_{\mathcal{M}_i}$. Finally, Castro (2020) presents an interesting approach for computing bisimulation metrics (Sect. 4.1.1) via access to transition tuples, and in this way to circumvent the problems for computing them in large or continuous state spaces. However, the results are limited to deterministic MDPs.

4.3 Rewards

The metrics can also measure the similarity between MDPs according to the distance between their reward dynamics.

Definition 9 Given two tasks $\mathcal{M}_i$ and $\mathcal{M}_j$, we define the distance between them as $d(\mathcal{M}_i, \mathcal{M}_j) = d(R_i, R_j)$.

For instance, Carroll and Seppi (2005) computes $d(R_i, R_j)$ as in Eq. (9), where n is the total number of state-action pairs in the source and the target task.

$$d(R_i, R_j) = \frac{1}{n} \sum_{s \in S} \sum_{a \in A} (R_i(s, a) - R_j(s, a))^2 \tag{9}$$

Instead, Tao et al. (2021) assumes the reward functions are a linear combination of some common *features* $\phi(\cdot, \cdot)$, $R_i(s, a) = \phi(s, a)^T w_i$ and $R_j(s, a) = \phi(s, a)^T w_j$, and then use the cosine distance function between w_i and w_j to compute $d(R_i, R_j)$. Finally, Gleave et al. (2020) introduce the Equivalent-Policy Invariant Comparison (EPIC) pseudometric which

is composed of two steps. First, transitions from an offline dataset are used to convert reward functions to a canonical form. This canonical form is invariant to reward transformations that do not affect the optimal policy. Second, the correlation between reward values on transition samples is computed, yielding a metric capturing reward function similarity. A significant drawback of this approach is that it evaluates the rewards on transitions between all state-state combinations, regardless of whether such state-state combinations are possible in a transition or not, which in practice leads to unreliable reward values as these are outside the distribution of the transitions. For this reason, some recent works have focused on improving the EPIC metric by making it consider only feasible state transitions (Wulfe et al., 2022). In any case, although the metrics in this category may function correctly in some particular cases, in others it is not a good approximation of the similarity between tasks (Carroll and Seppi, 2005). As a way of example, imagine two mazes, with the goal in different locations, but with everything else left the same. In both tasks, the agent receives a reward of 1 if it reaches the goal, -1 if it hits an obstacle, and 0 elsewhere. These metrics will consider as equally similar to two mazes where the goals are in close positions or in very different positions. These metrics cannot capture the distance that a goal is moved. However, it can easily capture the fact that new obstacles or goals have been added to the tasks. Similarity metrics based on the reward functions can be computed before the policy is learned but in general they are less sensitive to policy trends than those based on policy values (Carroll and Seppi, 2005).

4.4 State and action spaces

In this case, the distance between MDPs is computed as the distance between the state-action pairs $\langle s, a \rangle$ both in the source and the target tasks.

Definition 10 Given two MDPs $\mathcal{M}_i$ and $\mathcal{M}_j$, and their corresponding state-action spaces, $S_i \times A_i$ and $S_j \times A_j$, we define the distance as $d(\mathcal{M}_i, \mathcal{M}_j) = d(S_i \times A_i, S_j \times A_j)$.

For instance, Narayan and Leong (2019) compute this distance as the difference between the corresponding state-action transition distributions between the two tasks. In particular, the proposed metric is based on the Jensen-Shannon Distance (JSD) which measures the difference between two probability distributions (Nielsen, 2019). The distance $d(\mathcal{M}_i, \mathcal{M}_j)$ is computed as the averaged JSD over all the state-action pairs in order to composite a distance between both tasks. Instead, Taylor et al (2008b) compute the Euclidean distance between state-action pairs in the source and the target tasks. This work is not focused on the construction of a distance measure between MDPs, but such Euclidean distances between all the state-action pairs could be composited to obtain a sort of similarity metric.

4.5 States

When comparing closely related tasks, it may be sufficient to use only the state space. The metrics in this category use precisely the state space in both the source and the target tasks to compute the similarity between them.

Definition 11 Given two MDPs $\mathcal{M}_i$ and $\mathcal{M}_j$, we can define the distance between them as $d(\mathcal{M}_i, \mathcal{M}_j) = d(S_i, S_j)$, where $d(S_i, S_j)$ measures the distance between S_i and S_j.

Svetlik et al. (2017) propose to compute $d(S_i, S_j)$ as described in Eq. (10):

$$d(S_i, S_j) = \frac{|S_i| \cap |S_j|}{1 + |S_j| - |S_i|} \tag{10}$$

Equation 10 compute $d(S_i, S_j)$ as the relation between the applicability of the value function (measured as the number of states in $\mathcal{M}_i$ that also appears in $\mathcal{M}_j$), and the experience required to learn in $\mathcal{M}_j$ (measured as the difference of size between S_j and S_i). Another area of research that is relevant in this category is that of case-based reasoning (CBR) (Aamodt and Plaza, 1994). CBR uses the knowledge of previous cases to solve new problems, by comparing the actual state to the previous ones, and finding the closest. This results in an action that was already used to solve a very similar case, and therefore it must be useful. RL approaches based on CBR use a similarity function between the states in the target task and the states stored in a case base corresponding to a previous source task (Celiberto Jr et al., 2011; Bianchi et al., 2009). Such similarity function could be used to measure the similarity between the state spaces, hence, the similarity between the two tasks.

5 Performance-based metrics

The metrics proposed in the second major category of this survey evaluate the similarity between tasks by either comparing policies, or by comparing the performance of the agent in the source and the target tasks. So, we distinguish two different approaches to overcoming the problem of computing such performance-based similarities: (i) by the policy similarity and (ii) by the transfer gain obtained transferring the knowledge from a source task to the target task.

5.1 Policy similarity

These approaches measure the similarity between MDPs by comparing the learned value function V^π or the action-value function Q^π, hence, the behavioral policies π obtained in the source and the target tasks. Therefore, these metrics require the full (or partial) learning of these policies before the computation of the similarity between tasks. Such a comparison can be conducted in two different ways depending on what is being compared: (i) the policy values, or (ii) the policy parameters.

5.1.1 Policy values

One approach to compare the similarity between MDPs is by comparing the specific q-values or v-values of their respective value or action-value functions. The approaches in this category aim to capture the difference between the policy trends in two MDPs. The more the policies of the two tasks *overlap*, the more *similar* the two tasks become.

Definition 12 Given two tasks $\mathcal{M}_i$ and $\mathcal{M}_j$ and the q-values of Q^{π_i} and Q^{π_j} learned in these tasks, we define $d(\mathcal{M}_i, \mathcal{M}_j) = d(Q^{\pi_i}, Q^{\pi_j})$. This metric is transformed into $d(\mathcal{M}_i, \mathcal{M}_j) = d(V^{\pi_i}, V^{\pi_j})$ if V^{π_i} and V^{π_j} are computed instead.

Carroll and Seppi (2005) propose to compute $d(V^{\pi_i}, V^{\pi_j})$ as the number of states in $\mathcal{M}_i$ and $\mathcal{M}_j$ with identical maximum v-value. The most obvious problem with this distance is that it is overly restrictive: it requires two states with exactly the same v-value to consider an *overlap*, which is not often realistic. Carroll and Seppi (2005) also propose to compute $d(Q^{\pi_i}, Q^{\pi_j})$ as described in Eq. (11):

$$d(Q^{\pi_i}, Q^{\pi_j}) = \frac{1}{n} \sum_{s \in S} \sum_{a \in A} (Q^{\pi_i}(s, a) - Q^{\pi_j}(s, a))^2 \tag{11}$$

where n is the total number of state-action pairs in the source and the target task, and which is less restrictive than $d(V^{\pi_i}, V^{\pi_j})$. Carroll and Seppi (2005) demonstrate that these similarity metrics are only accurate after the q-values or v-values have been learned. Additionally, similarity metrics based on the number of states with maximum v-value requires the task to be more thoroughly learned than those based on the mean squared error of the q-values. Instead, (Zhou and Yang, 2020) compute $d(Q^{\pi_i}, Q^{\pi_j})$ deriving latent structures of tasks and finding matches between Q^{π_i} and Q^{π_j}. Finally, Serrano et al. (2021) compute the similarity between tasks with different but discrete state-action spaces by analyzing the differences between Q^{π_i} and Q^{π_j}. In particular, they identify pairs of state-action pairs that perform similar roles in their respective task, based on their Q-values.

Other works also make use of the action-value function Q^{π} to compute whether two states are *similar*, but they focus on state abstractions (i.e., on aggregating similar states belonging to the same MDP) and not on the definition of similarity metrics between different MDPs. For instance, Li et al. (2006) present some abstractions which consider that two states $s_1, s_2 \in S$ in an MDP $\mathcal{M}$, can be aggregated if the condition in Eq. (12) is fulfilled.

$$\forall a \; |f(s_1, a) - f(s_2, a)| \leq \eta \tag{12}$$

In Eq. (12), if $\eta = 0$ we can obtain different forms of exact abstractions depending on the choice of f: Q^{π} (Q^{π}-irrelevance), Q^* (Q^*-irrelevance), or $\max_A Q^*$ (a^*-irrelevance). However, similarly to the distance $d(V^{\pi_i}, V^{\pi_j})$ proposed by Carroll and Seppi (2005), exact abstractions fail to find opportunities for abstraction in tasks where no two situations are exactly alike. For this reason, Abel et al. (2016) investigate approximate state abstractions, which treat nearly-identical situations as equivalent, and where $\eta \geq 0$ and f is Q^*. From these abstractions, we can construct discrete metrics from any state aggregation as $d(s_1, s_2) = 0$ if Eq. (12) is fulfilled, and $d(s_1, s_2) = 1$ otherwise. In a similar line of research, Lan et al. (2021) present different value-based metrics for computing the similarity between states. The metric $d(s_1, s_2) = \max_{a \in A} |Q^*(s_1, a) - Q^*(s_2, a)|$ is a representative of such value-based metrics. It shares the same fundamentals of the *policy irrelevance* abstraction proposed by Jong and Stone (2005) which consider that two states can be aggregated if they have the same optimal action. Although all of these metrics have been proposed as distances between two states, they could also be used to calculate the distance between two MDPs as described in Sect. 4.1.1.

5.1.2 Policy approximation parameters

This section considers the value function is approximated by $V^{\pi}(\phi(s), \theta) \approx V^{\pi}(s)$, where θ denotes an adaptable parameter vector, $\phi(s)$ is the feature vector of state s, and V^{π} may be a linear (e.g., linear combination of features) or a non-linear function (e.g., a neural network) (Van Hasselt, 2012). The action-value function Q^{π} can be expressed in similar parameterized terms, $Q^{\pi}(\phi(s, a), \theta) \approx Q^{\pi}(s, a)$. The metrics within this category compare the particular weights of the parameter vectors θ corresponding to the value functions V^{π} or Q^{π} of the source task and the target task in order to measure the similarity between them. Such metrics reflect the authors intuition that *two tasks are more likely to be similar to each other if they have similar parameter vectors.*

Definition 13 Given two tasks $\mathcal{M}_i$ and $\mathcal{M}_j$ and the parameterize functions $Q^{\theta_i} \approx Q^{\pi_i}$ and $Q^{\theta_j} \approx Q^{\pi_j}$ learned in these tasks, we consider $d(\mathcal{M}_i, \mathcal{M}_j) = d(\theta_i, \theta_j)$.[3]

The same definition applies if it is used the value functions V^{π_i} and V^{π_j} instead of the action-value functions Q^{π_i} and Q^{π_j}. For instance, Karimpanal and Bouffanais (2018) compute the distance $d(\theta_i, \theta_j)$ as described in Eq. (13):

$$c_{\theta_i, \theta_j} = \frac{\boldsymbol{\theta}_i \cdot \boldsymbol{\theta}_j}{|\theta_i||\theta_j|} \tag{13}$$

i.e., by using the cosine similarity between two non-zero vectors. Karimpanal and Bouffanais (2018) demonstrate the better the estimate of the agent's parameter vector, the more accurate the distance $d(\mathcal{M}_i, \mathcal{M}_j)$. The cosine similarity has some advantages, such as boundedness and the ability to handle parameter vectors θ with largely different magnitudes. However, Karimpanal and Bouffanais (2018) focused on linear function approximation. Computing the similarity between parameter vectors presents particular challenges for non-linear function approximations, such as neural networks, where neurons could learn the *same* information at different positions in different runs, and still have identical behaviors. Therefore, the design of $d(\theta_i, \theta_j)$ should consider that two parameter vectors are *similar* if they lead to *similar* behaviors, regardless of the magnitude of the weights, or possible *shuffled* vectors. Some metrics have been proposed to measure the distance between neural networks (Ashmore, 2015), but they require further investigation in the context of *task similarity*. Anyway, we consider that this alternative representation of policies as parameter vectors opens the door to the comparison of other policy representations (Ferrante et al., 2008).

5.2 Transfer gain

One possible metric to measure the similarity between two MDPs is an approximation to the *advantage* gained by using one source task to speed up the learning of another target task, which is commonly known as transfer gain (Carroll and Seppi, 2005; Carroll, 2005; Taylor and Stone, 2009).

[3] For simplicity, we write Q^{θ} as an abbreviation for $Q^{\pi}(\phi(s, a), \theta)$.

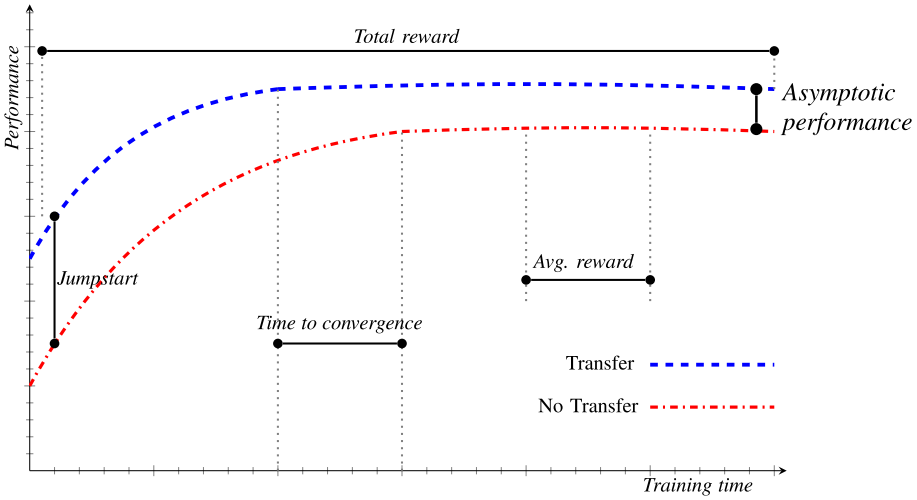

Fig. 2 Different metrics for measuring the transfer gain, $g(\mathcal{M}_i, \mathcal{M}_j)$. This graph show benefits to the jumpstart, time to convergence, total reward, average reward received within some window of time, and asymptotic performance

Definition 14 Given two tasks $\mathcal{M}_i$ and $\mathcal{M}_j$, and $g(\mathcal{M}_i, \mathcal{M}_j)$ which denotes the gain of transferring the knowledge learned in $\mathcal{M}_i$ to $\mathcal{M}_j$, we can consider $d(\mathcal{M}_i, \mathcal{M}_j)$ as inversely related to $g(\mathcal{M}_i, \mathcal{M}_j)$.

Many forms to measure the *advantage* $g(\mathcal{M}_i, \mathcal{M}_j)$ are possible. Some of them are graphically represented in Fig. 2:

- *Jumpstart* The performance at the initial steps of the learning process of an agent learning $\mathcal{M}_j$ may be improved by transfer from $\mathcal{M}_i$.
- *Asymptotic performance* The final learned performance of an agent learning $\mathcal{M}_j$ may also be improved by transfer from $\mathcal{M}_i$.
- *Total reward* The total reward accumulated by an agent learning $\mathcal{M}_j$ with transfer from $\mathcal{M}_i$, may be improved if it uses transfer, compared to learning without transfer.
- *Average reward* The average reward received within some window of time by an agent learning $\mathcal{M}_j$ with transfer from $\mathcal{M}_i$ may also be improved.
- *Time to convergence* It is expected for a transfer learner to reach the asymptotic performance earlier than another non-transfer learner.

Regardless of the particular technique used to compute $g(\mathcal{M}_i, \mathcal{M}_j)$, the higher the transfer gain, the greater the similarity between the tasks. While there is a general consensus that $g(\mathcal{M}_i, \mathcal{M}_j)$ is the best similarity metric between tasks (Carroll, 2005; Carroll and Seppi, 2005), since it allows to accurately measure *positive transfer*, it actually also requires that the transfer experiment be entirely or partially run before measuring the degree of similarity between tasks. This fact precludes its use in tasks where it is mandatory to know the similarity between tasks *before* the transfer experiment takes place.

In this paper, we distinguish two approaches within this category depending on whether the transfer gain is computed *after* or *during* the transfer process: (i) off-line transfer gain, and (ii) on-line transfer gain.

5.2.1 Off-line transfer gain

In this case, the transfer gain $g(\mathcal{M}_i, \mathcal{M}_j)$ is estimated as the difference in performance between the learning process with and without transfer, and once the learning processes are considered to be finished (Carroll, 2005; Mahmud et al., 2013; Sinapov et al., 2015; Zhan et al., 2016). Such gain $g(\mathcal{M}_i, \mathcal{M}_j)$ can be computed as the *jumpstart* (Sinapov et al., 2015; Carroll, 2005), the time to convergence (Carroll, 2005), the asymptotic performance (Mahmud et al., 2013), although other metrics such as the total reward or the transfer ratio could be also used (Taylor and Stone, 2009).

5.2.2 On-line transfer gain

On the contrary, in these approaches, the transfer gain is estimated *on-line* while the policy in the target task is computed (Fernández and Veloso, 2013; Azar et al., 2013; Li and Zhang, 2017). It is important to bear in mind that the *on-line* computation of this gain only makes sense if during the learning process we have several transfer sources to choose from. At the beginning of the learning process, these approaches have at their disposal the knowledge learned in solving a set of previous tasks $\{\mathcal{M}_1, \ldots, \mathcal{M}_n\}$ to learn the new task $\mathcal{M}_j$. During learning, they compute $g(\mathcal{M}_i, \mathcal{M}_j)$ of each past task $\mathcal{M}_i \in \{\mathcal{M}_1, \ldots, \mathcal{M}_n\}$. To do that, they transfer the knowledge acquired solving $\mathcal{M}_i$ to $\mathcal{M}_j$ during a limited number of episodes m. Then, $g(\mathcal{M}_i, \mathcal{M}_j)$ is computed as the average reward obtained during those m episodes (Fernández and Veloso, 2013; Azar et al., 2013). Once all gains are computed, it is possible to decide *on-line* which is the closest task to $\mathcal{M}_j$ within $\{\mathcal{M}_1, \ldots, \mathcal{M}_n\}$, so that the knowledge of the selected closest task can have a greater influence on learning about the policy in $\mathcal{M}_j$.

6 Discussion

From a transfer point of view, the ultimate goal of all similarity metrics is in some way to *predict* the relative *advantage* that would be gained by using a source task in a target task. The more similar the source and the target tasks are, the greater the *positive transfer*. However, there is probably no one best universal metric that works with all transfer techniques and problems. Since each metric can capture different types of similarity and each transfer technique induces different bias in the learning process, the question of selecting the best metric turns into finding the correct metric for a transfer technique to be applied to a particular problem. For this reason, in order to facilitate the selection of the best metric for a particular task, this section analyzes the distance metrics surveyed in this paper across five dimensions (Table 2): (i) nature of the state-action space (denoted by *Spaces* in Table 2), (ii) the required knowledge to compute the distance metric (denoted by *Knowl.*), (iii) allowed differences between the tasks (*Differ.*), (iv) the type of information that is transferred from the source tasks and the target tasks (*Transfer*), and (v) when the computation take places (*Comp.*). Table 3 provides a key for the abbreviations in Table 2. Furthermore, Table 2 and the proposed discussion will serve to analyze the pros and cons of the categories surveyed.

Table 2 This table lists most of the metrics discussed in this survey and classifies each in terms of five dimensions. Use Table 3 as key

Citation	Spaces	Knowl.	Differ.	Transfer	Comp.	Category
Section 4: model-based						
Ferns et al. (2004)	*s/a*	t,r	-	-	*b*	Bisimulation
Castro and Precup (2010)	*s/a*	t,r	s,t,r	π	*b*	
Song et al. (2016)	*s/a*	t,r	s,t,r	π	*b*	
Ferns et al. (2012)	*S/a*	t,r	-	-	*b*	
Castro et al. (2021)	*S/a*	I	-	-	*b*	
Ravindran and Barto (2002)	*s/a*	t,r	-	-	*b*	Homomorphism
Ravindran and Barto (2003)	*s/a*	t,r	s,a,t,r	π_p	*b*	
Sorg and Singh (2009)	*S/a*	t,r	s,a,t,r	Q	*b*	
Taylor et al. (2008a)	*s/a*	t,r	-	-	*b*	
Lazaric et al. (2008)	*S/A*	t,r	t,r	I	*b*	Compliance
Fachantidis (2016)	*S/A*	t,r	s,a,t,r	I	*b*	
Fachantidis et al. (2015)	*S/A*	t,r	s,a,t,r	I	*b*	
Wang et al. (2019)	*s/a*	t,r	t,r	-	*b*	MDP graphs
Liu and Stone (2006)	*s/a*	t,r	t,r	V	*b*	
Kuhlmann and Stone (2007)	*s/a*	t,r	t,r	V	*b*	
Ammar et al. (2014)	*S/A*	I	s,a,t,r	Q, θ	*b*	Transitions
Taylor et al. (2008c)	*S/a*	t	s,a,t,r	Q	*b*	
Castro (2020)	*S/a*	I	-	-	*b*	
Carroll and Seppi (2005)	*s/a*	r	t,r	Q, π	*b*	Rewards
Tao et al. (2021)	*S/A*	r	t,r	π	*b*	
Gleave et al. (2020)	*S/A*	r	t,r	-	*b*	
Narayan and Leong (2019)	*S/a*	$s \times a$	s,a,t,r	π	*b*	*S* & *A* Spaces
Taylor et al. (2008b)	*S/a*	$s \times a$	s,a,t,r	π	*b*	
Svetlik et al. (2017)	*s/a*	$\|s\|$	s	π	*b*	States
Celiberto Jr et al. (2011)	*S/A*	s	s,a,t,r	π	*b*	
Section 5: Performance-based						
Carroll (2005)	*s/a*	Q	t,r	Q, π	*b*	Policy values
Zhou and Yang (2020)	*S/a*	Q	r	Q	*b*	
Serrano et al. (2021)	s/a	Q	s,a	Q	*b*	
Karimpanal and Bouffanais (2018)	*S/a*	Q	r	θ	*b*	Policy param.
Carroll and Seppi (2005)	*s/a*	$\sum r, c$	t,r	Q, π	*a*	Offline transfer
Mahmud et al. (2013)	*s/a*	V	t,r	π	*a*	
Sinapov et al. (2015)	*S/a*	j	s	Q	*a*	
Fernández and Veloso (2013)	*S/a*	$\sum r$	t,r	π	*d*	Online transfer
Fernández et al. (2010)	*S/a*	$\sum r$	s,a,t,r	π	*d*	
Li and Zhang (2017)	*S/a*	$\sum r$	t,r	π	*d*	
Azar et al. (2013)	*S/a*	$\sum r$	t,r	π	*d*	

6.1 State-action space

Obviously, the selection of the metric depends on the nature of the state-action space. Some approaches require a finite set of states and actions. This is particularly true in the

Table 3 Key that provides a reference to the abbreviations in Table 2

Spaces		Transfer	
s/S	Finite/large state space	π	Policies
a/A	Finite/large action space	π_p	Partial policies (e.g., options)
		Q	Action-value function
Knowl.		V	Value function
t	Transition function	θ	Policy parameters
r	Reward function	I	Experience instances
I	Experiences instances	–	For a purpose other than transfer (e.g.,
$s \times a$	State-action pairs		state aggregation)
s	States		
$\|s\|$	States and size of the state spaces	Comp.	
Q/V	Action-value or value function	b	*Before* transfer
$\sum r$	Average reward within some window of time	a	*After* transfer
c	Time to convergence	d	*During* transfer
j	Jumpstart		
Differ.			
s/a	State/action spaces		
t/r	Transition/reward functions		
-	For a purpose other than similarity between tasks (e.g., state aggregation)		

case of *bisimulation* and *homomorphishm* metrics. These metrics are expensive to compute and typically require enumerating all the states pairs even when using on-the-fly approximations (Comanici et al., 2012), sampling-based approximations (Ferns et al., 2012; Castro and Precup, 2010) or approximations using the structure in the state space (Bacci et al., 2013). Such full state enumeration is impractical for large state spaces, and impossible for continuous state spaces. Additionally, these metrics can be overly pessimistic in the sense that they consider worst-case differences between states, which is overly restrictive for many problems (Castro and Precup, 2010). Although there have been positive steps to circumvent these drawbacks (Castro, 2020; Zhang et al., 2021; Castro et al., 2021), bisimulation-based approaches compute distances between states (belonging to the same MDP or not), and few works have been proposed to compose these distances into a single distance between MDPs (Song et al., 2016). For this reason, significant work needs to be done to determine if such approaches are feasible in practice for this task. Metrics based on MDP graphs have similar limitations on the size of the state and action spaces (Wang and Liang, 2019; Kuhlmann and Stone, 2007; Liu and Stone, 2006). Although they are based on an interesting principle, at the moment all of these graph-based approaches have limited applications. On the one hand, they require MDPs with a finite number of states and actions that can be adequately represented as a graph. On the other hand, graphs are required to be small since the computation of graph similarity metrics is computationally demanding. Regarding the latter, the computation of these measures can be accelerated via parallelism, but such approaches need to be investigated in the context of similarity between MDPs.

6.2 Required knowledge

Another issue that needs to be addressed is what and how much information is required to compute the similarity between tasks. As can be seen in Table 2, most model-based approaches require *prior* full information (or accurate approximations) about the transition and reward dynamics (Castro and Precup, 2010; Lazaric et al., 2008; Wang and Liang, 2019), or about the size of the state space (Svetlik et al., 2017). In this sense, performance-based metrics have a clear advantage over model-based ones: in general, performance-based metrics require less *a priori* information about the task to be solved, although as a counterpoint they need to fully or partially run the transfer experiment to obtain accurate approximations of the Q-function (Carroll, 2005; Carroll and Seppi, 2005; Zhou and Yang, 2020; Karimpanal and Bouffanais, 2018) or the V-function (Mahmud et al., 2013). In other cases, only the average cumulative reward obtained during a predetermined time window by the ongoing policy is required (Sinapov et al., 2015; Li and Zhang, 2017; Azar et al., 2013; Fernández and Veloso, 2013), which allows the fast computation of the similarity between tasks without the need to fully run the transfer experiment.

In this dimension, the model-based approaches based on the *structural* comparison of the instances in the form $\langle s, a, s' \rangle$ gathered from the two tasks are highly attractive (Ammar et al., 2014). On the one hand, they are not computationally expensive because they do not need to approximate the dynamics of the environment, nor do they need to learn a behavior policy, just to gather instances in both tasks. On the other hand, the collection of these instances can be done with a random policy, but also with a *safe* suboptimal one. The latter is particularly interesting if we are dealing with tasks where random behaviors are not allowed or where a single bad action can lead to catastrophic consequences. In this same line of research, the approaches that only use partial information from the instances to carry out similar comparisons, such as state-action pairs $\langle s, a \rangle$ (Narayan and Leong, 2019; Taylor et al., 2008b), or simply the states $\langle s \rangle$ (Svetlik et al., 2017; Celiberto Jr et al., 2011), are also interesting. However, in these cases, it is required a strong relationship between the tasks to be compared, which is either learned (Narayan and Leong, 2019; Taylor et al., 2008b) or taken for granted (Svetlik et al., 2017). As an example of the latter, Svetlik et al. (2017) assume that the only difference allowed between tasks is between the state spaces.

6.3 Allowed tasks differences

Regarding the allowed tasks differences, distance metrics can be computed between tasks that have different transition and/or reward functions (Ferns et al., 2004; Castro and Precup, 2010; Song et al., 2016; Lazaric et al., 2008), and/or state-action spaces (Fachantidis et al., 2015; Fernández and Veloso, 2013). Most of the methods in Table 2 require that both tasks have the same state-action space (Lazaric et al., 2008; Azar et al., 2013). However, the latter can be partially alleviated by the construction of inter-task mapping functions between state and/or action variables that allows translating a state $s_i \in S_i$ in one task to its equivalent state $s_j \in S_j$ in another tasks, $\mathcal{X}_S(s_i) = s_j$, and an action $a_i \in A_i$ to its equivalent $a_j \in A_j$, $\mathcal{X}_A(a_i) = a_j$. In some cases, the mappings functions $\mathcal{X}_S$ and $\mathcal{X}_A$ are explicitly provided (Fernández et al., 2010), but in other cases they are learned (Fachantidis et al., 2015; Taylor et al., 2008c). However, current

approaches for autonomously learning such mapping functions require domain knowledge or are inefficient. In this context, *bisimulation* and *homomorphism* metrics offer an interesting alternative to perform such a mapping between tasks. The mapping between the states of two MDPs is defined implicitly by the distance metric $d(\cdot,\cdot)$ as described in Definition 2: for each state in one MDP, finding the most similar state in the other MDP is equivalent to mapping the states from one MDP to the other. Instead, *homomorphisms* allow correspondences to be defined between state-action pairs, rather than just states (Definition 3). Therefore, one possibility is to allow the mapping to be determined automatically from *bisimulation* or *homomorphism* (Castro and Precup, 2010; Sorg and Singh, 2009).

6.4 Transferred knowledge

Different kinds of knowledge may transfer better or worse depending on the similarity between the tasks. The surveyed papers in Table 2 primarily transfer two types of knowledge: policies (Castro and Precup, 2010; Tao et al., 2021), and value functions (Sorg and Singh, 2009; Carroll, 2005; Ammar et al., 2014), and the type of knowledge transferred does not seem to depend on the way of computing the similarity between tasks. It should be noted that some approaches in Table 2 are not explicitly used for measuring the similarity between tasks. This is the case of some *bisimulation* and *homomorphism* approaches which are focused on state aggregation, but, as discussed in Sect. 4.1.1, they can be also used to measure the similarity between MDPs.

6.5 Computation

This leads us to the following issue: the computation moment. Ideally, the computation of the similarity metric should be *before* or, at least, *during* the transfer. Off-line transfer gain approaches are undoubtedly the best method for measuring similarity between two tasks: they produce such a measure *after* the transfer experiment has been run, in such a way that we can compute the real gain. However, if the point is to use the task similarity measure to choose a task to use in transfer, these metrics are useless. In this case, model-based metrics have an advantage over performance-based metrics: they allow to compute the metrics *before* the transfer process. These metrics can be used to choose the most similar MDP before the transfer, but as far as we know there are no theoretical guarantees that the most similar MDP is *similar* enough to produce a positive transfer. By contrast, the metrics based on the on-line transfer gain are at the point halfway between both. They allow computing the similarity metric *during* transfer, so that depending on the similarity of the source task, it will introduce a greater or lesser *exploration bias* in the learning process of the new task.

7 Future directions

The previous discussion points out several future directions. On the one hand, since there is no best metric, it would be useful to use several of them. For instance, model-based metrics can be used to return a useful approximation of task similarity before the tasks are learned, although this measure can be adapted *on-line* during the learning

process so that the bias that the source task induces in the exploration process is adjusted dynamically. On the other hand, given that different metrics compute different types of similarity (i.e., model-based metrics measure *structural* similarities, whilst performance-based metrics measure *performance* similarities), the agent can be equipped with the ability not only to determine which source task to use, but also which transfer technique to use given the type of similarity between the source and the target tasks. The proposed taxonomy also suggests other holes that should be considered in future work. Sect. 6.2 considers model-based approaches based on the *structural* comparison of tuples in the form $\langle s, a, s' \rangle$ as highly attractive (Ammar et al., 2014). Surprisingly, these approximations leave the reward out of the tuples. We suggest that the comparison of tuples in the form $\langle s, a, s', r \rangle$ would lead to better estimates of the similarity between two MDPs. Such metrics would not only be able to capture *structural* differences between MDPs, but also changes on the reward dynamics. The taxonomy also shows metrics that measures the similarity between two tasks as the similarity between their state spaces (Sect. 4.5). We suggest that this idea could also be applied to the action space: a novel metric not proposed in the taxonomy would be one that measure the similarity between two tasks by only considering the similarity between their action spaces. Such a metric can be particularly interesting in robotic tasks, where the robot *skills* could change from one task to another, but with everything else left the same.

Another interesting line of research is that based on building semantic representations of the tasks through domain-dependent features. For instance, we can define a particular Pac-Man task from features like the number of ghosts, behavior of the ghosts, or the type of the maze, and use these features to build a similarity metric between different Pac-Man tasks. In fact, one may heuristically combine structural, performance, but also semantic similarity aspects into the same metric. Thus, it could be obtained a metric more aligned with the way in which humans decides what is similar, since humans analyze the similarity between concepts or objects from different perspectives (Kemmerer, 2017).

There is also future work in the context of *Sim-to-Real*. Transferring learned models from simulation to the real world remains one of the hardest problems in control theory (Zhao et al., 2020). In this case, similarity metrics can help to answer how similar simulations and the actual world are. They could be used to provide theoretical guarantees that ensure the learned policies transferred from simulation to the actual world will perform as required, or to define mechanisms to tune/modify the simulated environments, so the gap between the simulated world and the actual one decreases. In another scenario, recent successes achieved by Deep Learning for learning feature representations have significantly impacted RL, and the combination of both methods (known as Deep RL) has achieved impressive results in recent years (Silver et al., 2016). However, using Deep RL introduces additional challenges, especially, their sample complexity. Initial investigations show that Deep RL agents also benefit from reuse of knowledge, but the effects of negative transfer could be catastrophic if uninformed transfer is performed (Pan et al., 2018; Rusu et al., 2016). Therefore, similarity metrics should play an important role also in the context of Deep RL. Such metrics will have to deal with the particular characteristics of the MDPs in Deep RL, i.e., possibly huge (and continuous) state and action spaces. This complexity makes most of the metrics discussed in this paper are not directly usable. However, some of these metrics could take advantage of advances in Deep Learning itself, e.g., using *deep* architectures instead of multi-layer perceptrons (Taylor et al., 2008c), or Deep RBMs instead of RBMs (Ammar et al., 2014). Therefore, the adaptation or the creation of new similarity metrics for Deep RL is a promising area of research.

Finally, comparing similarity metrics from different publications and recreating results from the literature is problematic because there are few available standard implementations. Researchers who wish to compare their new similarity metrics to existing results must often re-implement everything from scratch using the (sometimes incomplete and/or unclear) parameter setting of the publications. In this scenario, comparisons can be inaccurate and often not empirically verifiable. It would be required to establish ways to fairly compare results yielded by similarity metrics that leverage very different methodologies to understand their advantages and disadvantages. We need a benchmark suite consisting of implemented tasks and similarity metrics in order to better gauge the progress and applicability of new metrics. It is worth noting that such tool can open the door to the standardization of the comparison between transfer learning algorithms: since existing metrics measure the similarity from different perspectives, they can determine the performance of a transfer learning algorithm depending on *how* and *how much* similar the source and the target tasks are.

8 Concluding remarks

This paper contributes a compact and useful taxonomy of similarity metrics for Markov Decision Processes. The leaves of the taxonomy have been used to provide a literature review that surveys the existing work. We differentiated between model-based and performance-based metrics, depending on whether a *structural* or *performance* criterion has been used in its creation. The proposed taxonomy permits to organize clearly the different similarity metrics, or find commonalities between them. This can help the reader to choose similarity metrics for their tasks, or even define their own. We also discussed different selection criteria and some promising future research directions.

Acknowledgements We offer our deep gratitude and special thanks to Matthew E. Taylor for his generous and invaluable comments during the revision of this paper. We would also like to thank to Svitlana Vyetrenko, Sumitra Ganesh and Nelson Vadori from JPMorgan Chase & Co.

Author Contributions Fernando Fernández and Javier García contributed to the study conception and design. Material preparation and data collection were performed by Javier García and Álvaro Visús. The first draft of the manuscript was written by Javier García, Álvaro Visús and Fernando Fernández and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature. This research was funded in part by JPMorgan Chase & Co. Any views or opinions expressed herein are solely those of the authors listed, and may differ from the views and opinions expressed by JPMorgan Chase & Co. or its affiliates. This material is not a product of the Research Department of J.P. Morgan Securities LLC. This material should not be construed as an individual recommendation for any particular client and is not intended as a recommendation of particular securities, financial instruments or strategies for a particular client. This material does not constitute a solicitation or offer in any jurisdiction. This work has also been supported by the Madrid Government (Comunidad de Madrid-Spain) under the Multiannual Agreement with UC3M in the line of Excellence of University Professors (EPUC3M17), and in the context of the V PRICIT (Regional Programme of Research and Technological Innovation).

Availability of data and materials Not applicable.

Declarations

Conflict of interest The authors have no competing interests to declare that are relevant to the content of this

article.

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

References

Aamodt, A., & Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications, 7*(1), 39–59.

Abel, D., Hershkowitz, D. E., & Littman, M. L. (2016). Near Optimal Behavior via Approximate State Abstraction. In *Proceedings of the 33rd international conference on machine learning*, JMLR.org (pp. 2915–2923).

Agarwal, R., Machado, M. C., Castro, P. S., & Bellemare, M. G. (2021). Contrastive behavioral similarity embeddings for generalization in reinforcement learning. In *9th International conference on learning representations, ICLR 2021, Virtual Event*, Austria, May 3-7, 2021, OpenReview.net, URL https://openreview.net/forum?id=qda7-sVg84.

Ammar, H., Eaton, E., Taylor, M., Decebal, C., Mocanu, D., Driessens, K., Weiss, G., & Tuyls, K. (2014). An automated measure of MDP similarity for transfer in reinforcement learning. In *Workshops at the twenty-eighth AAAI conference on artificial intelligence*.

Ashmore, S. C. (2015). Evaluating the intrinsic similarity between neural networks. University of Arkansas.

Azar, M. G., Lazaric, A., & Brunskill, E. (2013). Regret bounds for reinforcement learning with policy advice. In *Joint european conference on machine learning and knowledge discovery in databases* (pp. 97–112). Springer.

Bacci, G., Bacci, G., Larsen, K. G., & Mardare, R. (2013). On-the-fly exact computation of bisimilarity distances. In *International conference on tools and algorithms for the construction and analysis of systems* (pp. 1–15). Springer.

Bianchi, R. A. C., Ros, R., & Lopez de Mantaras, R. (2009). Improving reinforcement learning by using case based heuristics. In L. McGinty & D. C. Wilson (Eds.), *Case-based reasoning research and development* (pp. 75–89). Berlin: Springer.

Carroll, J. L. (2005). Task localization, similarity, and transfer; towards a reinforcement learning task library system. PhD thesis.

Carroll, J. L., & Seppi, K. (2005). Task similarity measures for transfer in reinforcement learning task libraries. In *Proceedings of the 2005 IEEE international joint conference on neural networks, 2005*, Vol. 2 (Vol. 2, pp. 803–808). https://doi.org/10.1109/IJCNN.2005.1555955.

Castro, P. S. (2020). Scalable methods for computing state similarity in deterministic markov decision processes. In *Proceedings of the Thirty-Fourth AAAI conference on artificial intelligence (AAAI-20)*.

Castro, P., & Precup, D. (2010). Using bisimulation for policy transfer in MDPs. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 24).

Castro, P. S., & Precup, D. (2011). Automatic construction of temporally extended actions for MDPs using bisimulation metrics. In *European workshop on reinforcement learning* (pp. 140–152). Springer.

Castro, P. S., Kastner, T., Panangaden, P., & Rowland, M. (2021). Mico: Improved representations via sampling-based state similarity for markov decision processes. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, & J. W. Vaughan (Eds.), *Advances in neural information processing systems*,

Curran Associates, Inc., vol 34 (pp. 30113–30126), URL https://proceedings.neurips.cc/paper/2021/file/fd06b8ea02fe5b1c2496fe1700e9d16c-Paper.pdf.
Celiberto Jr, L. A., Matsuura, J. P., De Mantaras, R. L., & Bianchi, R. A. (2011). Using cases as heuristics in reinforcement learning: a transfer learning application. In *Twenty-Second international joint conference on artificial intelligence*.
Comanici, G., Panangaden, P., & Precup, D. (2012). On-the-fly algorithms for bisimulation metrics. In *2012 ninth international conference on quantitative evaluation of systems* (pp. 94–103). IEEE.
Comanici, G., Precup, D., & Panangaden, P. (2015). Basis refinement strategies for linear value function approximation in MDPs. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 28). Curran Associates, Inc.
Conci, A., & Kubrusly, C. (2018). Distance between sets-a survey. arXiv:1808.02574.
Fachantidis, A. (2016). Knowledge transfer in reinforcement learning. PhD thesis.
Fachantidis, A., Partalas, I., Taylor, M., & Vlahavas, I. (2011). Transfer learning via multiple inter-task mappings (pp. 225–236), https://doi.org/10.1007/978-3-642-29946-9_23.
Fachantidis, A., Partalas, I., Taylor, M. E., & Vlahavas, I. (2015). Transfer learning with probabilistic mapping selection. *Adaptive Behavior, 23*(1), 3–19.
Fernández, S., Aler, R., & Borrajo, D. (2011). Knowledge transfer between automated planners. *AI Magazine, 32*(2), 79–94.
Fernández, F., García, J., & Veloso, M. (2010). Probabilistic policy reuse for inter-task transfer learning. *Robotics and Autonomous Systems, 58*(7), 866–871.
Fernández, F., & Veloso, M. (2013). Learning domain structure through probabilistic policy reuse in reinforcement learning. *Progress in Artificial Intelligence, 2*(1), 13–27.
Ferns, N., Castro, P. S., Precup, D., & Panangaden, P. (2006). Methods for computing state similarity in markov decision processes. In *UAI '06, Proceedings of the 22nd conference in uncertainty in artificial intelligence*, Cambridge, MA, USA, July 13-16, 2006, AUAI Press.
Ferns, N., Panangaden, P., & Precup, D. (2004). Metrics for finite Markov decision processes. In *UAI*, Vol. 4 (pp. 162–169).
Ferns, N., Panangaden, P., & Precup, D. (2012). Metrics for Markov decision processes with infinite state spaces. arXiv:1207.1386.
Ferrante, E., Lazaric, A., & Restelli, M. (2008). Transfer of task representation in reinforcement learning using policy-based proto-value functions. In *AAMAS (3)* (pp. 1329–1332).
Gao, X., Xiao, B., Tao, D., & Li, X. (2010). A survey of graph edit distance. *Pattern Analysis and Applications, 13*(1), 113–129.
García J, López-Bueno, I., Fernández, F., & Borrajo, D. (2010). *A comparative study of discretization approaches for state space generalization in the Keepaway Soccer task*. New York: Nova Science Publishers.
Genesereth, M., Love, N., & Pell, B. (2005). General game playing: Overview of the AAAI competition. *AI Magazine, 26*(2), 62–62.
Ghojogh, B., Ghodsi, A., Karray, F., & Crowley, M. (2021). Restricted boltzmann machine and deep belief network: Tutorial and survey. arXiv:2107.12521.
Giunchiglia, F., & Walsh, T. (1992). A theory of abstraction. *Artificial Intelligence, 57*(2–3), 323–389.
Givan, R., Dean, T., & Greig, M. (2003). Equivalence notions and model minimization in Markov decision processes. *Artificial Intelligence, 147*(1–2), 163–223.
Gleave, A., Dennis, M., Legg, S., Russell, S., & Leike, J. (2020). Quantifying differences in reward functions. arXiv:2006.13900.
Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science, 313*(5786), 504–507.
Jeh, G., & Widom, J. (2002). Simrank: A measure of structural-context similarity. In *Proceedings of the Eighth ACM SIGKDD international conference on knowledge discovery and data mining, association for computing machinery*, New York, NY, USA, KDD '02 (pp. 538-543), https://doi.org/10.1145/775047.775126.
Jin, R., Lee, V. E., & Li, L. (2014). Scalable and axiomatic ranking of network role similarity. *ACM Transactions on Knowledge Discovery from Data (TKDD), 8*(1), 1–37.
Jong, N. K., & Stone, P. (2005). State abstraction discovery from irrelevant state variables. In *IJCAI*, Citeseer, Vol. 8 (pp. 752–757).
Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research, 4,* 237–285.
Karimpanal, T. G., & Bouffanais, R. (2018). Self-organizing maps as a storage and transfer mechanism in reinforcement learning. arXiv:1807.07530.

Kemmerer, D. (2017). Categories of object concepts across languages and brains: the relevance of nominal classification systems to cognitive neuroscience. *Language, Cognition and Neuroscience, 32*(4), 401–424.

Kuhlmann, G., & Stone, P. (2007). Graph-based domain mapping for transfer learning in general games. In *Proceedings of the 18th European conference on machine learning*, URL http://www.cs.utexas.edu/users/ai-lab?kuhlmann:ecml07.

Lan, C. L., Bellemare, M. G., & Castro, P. S. (2021). Metrics and continuity in reinforcement learning. In *Thirty-Fifth AAAI conference on artificial intelligence, AAAI 2021, thirty-third conference on innovative applications of artificial intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021* (pp. 8261–8269). AAAI Press, URL https://ojs.aaai.org/index.php/AAAI/article/view/17005.

Lazaric, A. (2008). Knowledge transfer in reinforcement learning. PhD thesis.

Lazaric, A., Restelli, M., & Bonarini, A. (2008). Transfer of samples in batch reinforcement learning. In *ICML* (pp. 544–551), https://doi.org/10.1145/1390156.1390225.

Li, S., & Zhang, C. (2017). An optimal online method of selecting source policies for reinforcement learning. arXiv:1709.08201.

Li, L., Walsh, T. J., & Littman, M. L. (2006). Towards a unified theory of state abstraction for MDPs. In *ISAIM*.

Lin, Z., Lyu, M. R., & King, I. (2012). Matchsim: a novel similarity measure based on maximum neighborhood matching. *Knowledge and Information Systems, 32*(1), 141–166.

Liu, Y., & Stone, P. (2006). Value-function-based transfer for reinforcement learning using structure mapping. In *Proceedings of the twenty-first national conference on artificial intelligence* (pp. 415–420).

Mahmud, M., Hawasly, M., Rosman, B., & Ramamoorthy, S. (2013). Clustering markov decision processes for continual transfer. arXiv:1311.3959.

McKay, B. D., & Piperno, A. (2014). Practical graph isomorphism, ii. *Journal of Symbolic Computation, 60,* 94–112.

Mendonca, R., Geng, X., Finn, C., & Levine, S. (2020). Meta-reinforcement learning robust to distributional shift via model identification and experience relabeling. arXiv:2006.07178.

Milner, R. (1982). *A Calculus of Communicating Systems*. Berlin: Springer.

Narayan, A., & Leong, T. Y. (2019). Effects of task similarity on policy transfer with selective exploration in reinforcement learning. In *Proceedings of the 18th international conference on autonomous agents and multiagent systems* (pp. 2132–2134).

Narvekar, S., Peng, B., Leonetti, M., Sinapov, J., Taylor, M. E., & Stone, P. (2020). Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research, 21*(181), 1–50.

Nielsen, F. (2019). On the Jensen-Shannon symmetrization of distances relying on abstract means. *Entropy, 21,* 485. https://doi.org/10.3390/e21050485.

Ontañón, S. (2020). An overview of distance and similarity functions for structured data. *Artificial Intelligence Review, 53*(7), 5309–5351.

Pan, J., Wang, X., Cheng, Y., & Yu, Q. (2018). Multisource transfer double DQN based on actor learning. *IEEE Transactions on Neural Networks and Learning Systems, 29*(6), 2227–2238.

Phillips, C. (2006). Knowledge transfer in markov decision processes. Tech. rep., Technical report, McGill University, School of Computer Science, 2006. URL ...

Ravindran, B., & Barto, A. G. (2002). Model minimization in hierarchical reinforcement learning. In *International symposium on abstraction, reformulation, and approximation* (pp. 196–211). Springer.

Ravindran, B., & Barto, A. G. (2003). Relativized options: Choosing the right transformation. In *Proceedings of the 20th international conference on machine learning (ICML-03)* (pp. 608–615).

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., & Hadsell, R. (2016). Progressive neural networks. arXiv:1606.04671.

Serrano, S. A., Martinez-Carranza, J., & Sucar, L. E. (2021). Inter-task similarity measure for heterogeneous tasks. In *RoboCup symposium. Lecture notes in computer science*, Springer.

Shui, C., Abbasi, M., Robitaille, L., Wang, B., & Gagné, C. (2019). A principled approach for learning task similarity in multitask learning. In *Proceedings of the twenty-eighth international joint conference on artificial intelligence, IJCAI 2019*.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., et al. (2016). Mastering the game of go with deep neural networks and tree search. *Nature, 529*(7587), 484.

Sinapov, J., Narvekar, S., Leonetti, M., & Stone, P. (2015). Learning inter-task transferability in the absence of target task samples. Vol. 2.

Song, J., Gao, Y., Wang, H., & An, B. (2016). Measuring the distance between finite markov decision processes. In *Proceedings of the 15th international conference on autonomous agents and multiagent systems (AAMAS 2016)*.

Sorg, J., & Singh, S. (2009). Transfer via soft homomorphisms. In: *Proceedings of The 8th international conference on autonomous agents and multiagent systems-volume 2* (pp. 741–748).

Sutton, R. S., & Barto, A. G. (2011). *Reinforcement learning: An introduction*. Cambridge: MIT Press.

Svetlik, M., Leonetti, M., Sinapov, J., Shah, R., Walker, N., & Stone, P. (2017). Automatic curriculum graph generation for reinforcement learning agents. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 2590-–2596).

Tao, Y., Genc, S., Chung, J., Sun, T., & Mallya, S. (2021). Repaint: Knowledge transfer in deep reinforcement learning. arXiv:2011.11827

Taylor, M. E., & Stone, P. (2009). Transfer learning for reinforcement learning domains: A survey. Journal of Machine Learning Research 10(7).

Taylor, M. E., Kuhlmann, G., & Stone, P. (2008c). Autonomous transfer for reinforcement learning. In *AAMAS (1)*, Citeseer (pp. 283–290).

Taylor, M. E., Jong, N. K., & Stone, P. (2008). Transferring instances for model-based reinforcement learning. In W. Daelemans, B. Goethals, & K. Morik (Eds.), *Machine learning and knowledge discovery in databases* (pp. 488–505). Berlin: Springer.

Taylor, J., Precup, D., & Panagaden, P. (2008). Bounding performance loss in approximate MDP homomorphisms. *Advances in Neural Information Processing Systems, 21,* 1649–1656.

Torrey, L., & Shavlik, J. (2010). Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques, IGI global* (pp. 242–264).

Van Hasselt, H. (2012). Reinforcement learning in continuous state and action spaces. In *Reinforcement learning* (pp. 207–251). Springer.

Wang, H., Dong, S., & Shao, L. (2019). Measuring structural similarities in finite mdps. In *Proceedings of the twenty-eighth international joint conference on artificial intelligence, IJCAI-19, international joint conferences on artificial intelligence organization* (pp. 3684–3690), https://doi.org/10.24963/ijcai.2019/511.

Wang, D. z., & Liang, J. y. (2019). Research and design of theme image crawler based on difference hash algorithm. In *IOP conference series: Materials science and engineering*, IOP Publishing, Vol. 563 (p. 042080).

Watkins, C. (1989). Learning from delayed rewards. PhD thesis, Cambridge, UK: King's College.

Wiering, M., & van Otterlo, M. (2014). *Reinforcement Learning: State-of-the-Art*. Springer Publishing Company, Incorporated.

Wulfe, B., Balakrishna, A., Logan, E., Mercat, J., McAllister, R., & Gaidon, A. (2022). Dynamics-aware comparison of learned reward functions. In *International conference on learning representations (ICLR), ICLR*.

Zhan, Y., Ammar, H. B., & taylor, M. E. (2016). Theoretically-grounded policy advice from multiple teachers in reinforcement learning settings with applications to negative transfer. arXiv:1604.03986.

Zhang, A., McAllister, R. T., Calandra, R., Gal, Y., & Levine, S. (2021). Learning invariant representations for reinforcement learning without reconstruction. In *9th international conference on learning representations, ICLR 2021, Virtual Event*, Austria, May 3-7, 2021, OpenReview.net.

Zhao, W., Queralta, J. P., & Westerlund, T. (2020). Sim-to-real transfer in deep reinforcement learning for robotics: A survey. In *2020 IEEE symposium series on computational intelligence (SSCI)* (pp. 737–744). IEEE.

Zhou, Y., & Yang, F. (2020). Latent structure matching for knowledge transfer in reinforcement learning. *Future Internet*. https://doi.org/10.3390/fi12020036.