

Structural Similarity for Improved Transfer in Reinforcement Learning

C. Chace Ashcraft

Research and Exploratory Development
Johns Hopkins University Applied Physics Lab
Laurel MD, USA
chace.ashcraft@jhuapl.edu

Benjamin Stoler

Research and Exploratory Development
Johns Hopkins University Applied Physics Lab
Laurel MD, USA
benjamin.stoler@jhuapl.edu

Chigozie Ewulum

Research and Exploratory Development
Johns Hopkins University Applied Physics Lab
Laurel MD, USA
chigozie.ewulum@jhuapl.edu

Susama Agarwala

Research and Exploratory Development
Johns Hopkins University Applied Physics Lab
Laurel MD, USA
susama.agarwala@jhuapl.edu

Abstract—Transfer learning is an increasingly common approach for developing performant RL agents. However, it is not well understood how to define the relationship between the source and target tasks, and how this relationship contributes to successful transfer. We present an algorithm called *Structural Similarity for Two MDPS*, or *SS2*, that calculates a state similarity measure for states in two finite MDPs based on previously developed bisimulation metrics, and show that the measure satisfies properties of a distance metric. Then, through empirical results with GridWorld navigation tasks, we provide evidence that the distance measure can be used to improve transfer performance for Q-Learning agents over previous implementations.

Index Terms—task similarity, bisimulation, reinforcement learning, transfer learning

I. INTRODUCTION

The last decade has seen impressive results in reinforcement learning (RL), particularly for deep reinforcement learning (DRL) [12, 14, 24]. However, many of these algorithms tend to be computationally expensive and produce policies that are brittle to shifts in their deployment distributions. One potential solution to both of these problems is to leverage transfer learning [5, 15, 29], i.e., using previously trained policies to jumpstart learning in a new task, or by developing curricula [3, 13] of tasks to facilitate the learning of advanced skills or behaviors.

To best leverage transfer, the task being transferred from should be *similar* to the task being transferred

to. Yet what makes one task similar to another in RL, aside from good transfer [20, 25], is currently not well understood, nor how to exploit such similarity to facilitate transfer learning.

In this work, we propose a new approach, based on [26] and [18], to estimate state and action similarities between two tasks (where each task is considered to be a Markov Decision Process, or MDP), and then leverage those similarities for knowledge transfer in Q-learning. We show that our approach, referred to as *Structural Similarity for Two MDPs* or *SS2*, satisfies the properties of a similarity metric, compare SS2-based knowledge transfer to prior methods [18], and propose ways in which the prior methods can be augmented by SS2. We assess task similarity and knowledge transfer on a set of gridworld RL tasks, and show that the SS2-augmented methods outperform all other tested methods.

We also discuss some observations regarding task distances such as those proposed by [18], limitations of our proposed algorithms, and possible future work. Notably, reducing a set of state similarities into a single value for task similarity, as done in [18] using the Hausdorff and Kantorovich distances, did not correlate well with transfer in our experiments.

II. RELATED WORK

Task similarity measures tend to fall into two categories: model-based and performance-based [25].

Model-based metrics tend to focus on similarities in the structures of the MDPs, e.g. bisimulation metrics [9]. Song et al. [18] introduced a metric d' that extends

This research was funded by the DARPA Lifelong Learning Machines (L2M) program.

this bisimulation metric to compare similarity between separate MDPs [18], but their metric requires that two MDPs be homogeneous, i.e., having equivalent state representations, a one-to-one correspondence between action spaces, and by satisfying a condition they refer to as *reward-linked*. Their metric iteratively builds a matrix of distances between states by adding the magnitude of the difference of expected rewards of actions between states and the Kantorovich distance, or earth mover’s distance (EMD), between the probability distributions of the corresponding actions between the two states. They then propose algorithms that leverage d' to transfer knowledge from a Q function learned from one task to initialize a Q table for another task to accelerate learning. The Q table of the new task is initialized either with Q values from similar states in the trained task (what the authors refer to as the “state method”), or with a sum of Q values from similar states weighted by their similarity as computed using d' (the “weight method”). For simplicity, we instead refer to these algorithms as T-STATE and T-AVG, respectfully. We discuss these algorithms in more detail in Section VI.

Wang et. al. [26] proposed a similar solution to quantify the similarities between states and actions within a single MDP by utilizing the Hausdorff distance between out-neighbors of states and the EMD between out-neighbors of actions in a bipartite MDP graph. Their solution generates two matrices of similarities, one for state similarities and one for action similarities, but for states and actions within a single MDP.

In [28], Zhang et al. also utilized a bisimulation metric in order to disassociate task-irrelevant information between observations within MuJoCo [22] tasks and high-fidelity highway driving tasks using CARLA, an open urban driving simulator [8].

Performance-based metrics, on the other hand, aim to find similarities based on transfer performance between tasks. Carrol et. al. [5] proposes a task library system as a part of the *lifelong learning* [21] paradigm that allows an agent to improve its learning ability as it is exposed to successive tasks through task localization, similarity discovery, and task transfer. They propose multiple task similarity measures: similarity based on rewards, similarity based on number of states with identical policy, mean square error between Q-values, and mean squared error between rewards (R values). Castro et. al. [6] expanded upon the preexisting bisimulation metrics by creating two metrics that are less computationally intensive, with one having guaranteed convergence and the other being able to learn an approximation for continuous

state MDPs. However, these require an actively learning policy to operate on the MDP in order to compute their measures, therefore we categorize it here. Other examples in this vein include [11] and [7].

Some methods additionally include the learning of the metric itself. Works such as [1] and [17] propose metrics that require training models to extract useful information on source and target tasks. Serrano et. al.’s method performs operations on Q tables of source and target tasks, while Ammar et. al.’s method utilizes restricted Boltzmann machines on the source and target tasks, allowing a modelling of the behavioural dynamics of the two MDPs to be used as a similarity measure on tasks within the domain the measure was trained on.

III. PRELIMINARIES

A. MDPs and MDP Graphs

In this work we primarily follow the definitions and notation from Wang et. al. [26], so we define an MDP as $M = (S, A, P, R)$, where S is a finite set of states, A is a finite set of actions, $P : S \times A \times S \rightarrow [0, 1]$ is the transition function providing the probabilities of an agent going to a state $s' \in S$ given it is currently in state $s \in S$ and took action $a \in A$, and finally, $R : S \times A \times S \rightarrow [0, 1]$ is the reward function for taking action a while in state s and transitioning to s' . We note that, while common definitions of MDPs include a discount factor, usually denoted γ , as with the Song and Wang approaches, we do not make explicit use of this value. The same can be said for the initial agent state $s_0 \in S$.

We define the MDP graph of an MDP $M = (S, A, P, R)$ as $G_M = (V, \Lambda, E, \Psi, p, r)$. G_M is a heterogeneous directed bipartite graph with two types of nodes, *state* nodes and *action* nodes. V denotes the set of state nodes, and Λ denotes the set of action nodes. E is the set of decision edges from state nodes to action nodes, and Ψ is the set of transition nodes from action nodes to state nodes. Transition edges are weighted by the transition probabilities p and by rewards r . Note that there is a one-to-one correspondence between an MDP and its constructed graph as described above. For more details and facts about constructing an MDP graph from an MDP, see [26], as well as our experiment methodology in Section VI.

B. Reinforcement Learning and Transfer

The aim of reinforcement learning is to find an optimal policy π^* for a given MDP M , utilizing the information gained by interacting with the MDP, particularly by discovering rewards in the MDP and maximizing them. Again following notation from [18], given an MDP $M =$

(S, A, P, R) , we define a policy $\pi : S \times A \rightarrow [0, 1]$ as a mapping from state, action pairs to reward values in the interval $[0, 1]$. The optimal policy can be characterized as

$$\begin{aligned} \pi^* &= \operatorname{argmax}_{\pi} E_{\pi} \left[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s \right] \\ &= V^{\pi}(s) \end{aligned}$$

where γ is a discount factor, r_t is the reward at time t , and s_0 is the starting state of M (γ and s_0 are often included in the definition of the MDP, but in this case we omit them). $V^{\pi} : S \rightarrow \mathcal{R}$ is called the *value function* of the policy π , and is the expected cumulative reward of following π from state s onward. Deviating slightly in notation, we say $P(s, a, s')$ is the probability of transitioning from state s to state s' given that agent takes action a .

IV. STRUCTURAL SIMILARITY FOR TWO MDPs (SS2)

We define our distance between two given MDPs mostly following the recursion given in [26]’s implementation, which we will call SS1 for the remainder of this section, but with some notable differences. We construct a similarity matrix from an MDP graph composed of two disconnected components, each component being the MDP graph of the respective MDP. We also impose some additional restrictions on the initial conditions of SS1’s recursion, and then use only a submatrix of the final similarity matrix for the final distance measure; doing so allows us to establish properties of that submatrix that are desirable, as shown in [26]. In Section V we provide a simplification of this algorithm wherein we can compute the sub-matrix directly. For the remainder of this work, we will refer to this metric as Structural Similarity for two MDPs¹, or SS2, for short.

A. Foundation

Let $M = (S_M, A_M, P_M, R_M)$ and $N = (S_N, A_N, P_N, R_N)$ be two MDPs. We represent both as MDP graphs $G_M = (V^M, \Lambda^M, E^M, \Psi^M, p^M, r^M)$ and $G_N = (V^N, \Lambda^N, E^N, \Psi^N, p^N, r^N)$ as established for SS1². Then, let $G = G_M \sqcup G_N = (V, \Lambda, E, \Psi, p, r)$ be the disjoint union of the two graph MDPs. This

¹In theory, this could be extended to more than two MDPs, but we only consider two in this work.

²It is worth noting here that while the action space is a finite size, say σ , each action available to the agent at each state is its own unique node. So if all σ actions are available to the agent at each state, there would be $\sigma \cdot |S|$ total action nodes.

represents a situation where there are two sets of states, V^M and V^N , such that if one starts in one set, there are no sets of actions such that one can land in a state in the other set. We desire to establish the state and action distance measures:

$$\delta_{\hat{S}}(u, v) := 1 - \sigma_{\hat{S}}(u, v) \quad \forall u, v \in V \quad (1)$$

$$\delta_{\hat{A}}(\alpha, \beta) := 1 - \sigma_{\hat{A}}(\alpha, \beta) \quad \forall \alpha, \beta \in \Lambda \quad (2)$$

where $\sigma_{\hat{S}}$ and $\sigma_{\hat{A}}$ are the state and action similarities for state space $\hat{S} = S_M \cup S_N$ and action space $\hat{A} = A_M \cup A_N$, respectively. In practice, these functions will be defined as matrices, i.e. $\sigma_S(u, v) = \mathbf{S}[u_{idx}, v_{idx}]$, where \mathbf{S} is a matrix, u_{idx} and v_{idx} denote the row and column of \mathbf{S} representing u and v , respectively, and $\mathbf{S}[i, j]$ is the $(i, j)^{th}$ entry of \mathbf{S} . For G as defined above, we see that $\sigma_{\hat{S}} = \mathbf{S}^G$ takes on a useful form.

Let \mathbf{S}^M and \mathbf{S}^N be similarity matrices for the states of M and N respectively. Let $\mathbf{S}^{M,N}$ (respectively, $\mathbf{S}^{N,M}$) be the similarity matrix between the two MDPs with rows corresponding to the states of M and the columns the states of N (respectively, rows corresponding to states of N and columns corresponding to states of M). Similarly we define matrices \mathbf{A}^M , \mathbf{A}^N , $\mathbf{A}^{M,N}$ and $\mathbf{A}^{N,M}$ with rows and columns corresponding to states in M and states in N as appropriate. Then one may write the comparison matrix of G as follows:

$$\mathbf{S}^G = \begin{bmatrix} \mathbf{S}^M & \mathbf{S}^{M,N} \\ \mathbf{S}^{N,M} & \mathbf{S}^N \end{bmatrix} \quad ; \quad \mathbf{A}^G = \begin{bmatrix} \mathbf{A}^M & \mathbf{A}^{M,N} \\ \mathbf{A}^{N,M} & \mathbf{A}^N \end{bmatrix}$$

It is worth noting here that the sub-matrix $\mathbf{S}^{M,N}$ represents the distances between the state nodes of the MDP graph of M and MDP graph of N , and similarly with $\mathbf{A}^{M,N}$ and the action nodes in each MDP graph. Thus, the real final product we are truly interested in is $\mathbf{S}^{M,N}$, but for now we consider the computation of \mathbf{S}^G in order to leverage results from SS1 and make the argument that \mathbf{S}^G is, in fact, a distance metric for MDP states.

B. Base Cases

For $u, v \in \mathbf{S}^G$, we set the similarity of a node to itself as 1. Similarly, two absorbing nodes are maximally similar, while absorbing and non-absorbing nodes are as dissimilar as possible. In other words, let $\mathbf{S}_{u,v}^{G,t}$ denote

the $(u, v)^{th}$ entry of the similarity matrix for graph G at time step t , then

$$\mathbf{S}_{u,v}^{G,0} = \begin{cases} 1 & u = v \\ \omega & N_u = N_v = \emptyset \\ 0 & N_u = \emptyset \text{ xor } N_v = \emptyset \\ d_{uv} \in [0, 1] & u \neq v \end{cases}$$

where $\omega \in (0, 1]$ denotes the maximal similarity value assigned to any two *different* states. For example, if two states are exactly the same but are in two different connected components of the graph, their similarity would be ω .

As with SS1, we allow freedom in the initialization of similarity between non-absorbing pairs; however, we also impose one additional requirement. We insist that \mathbf{S}^G be initialized *such that the entries in the matrix* $1 - \mathbf{S}^{G,0}$ *define a distance function on the nodes*. Namely, we insist that $d_{uv} = d_{vu}$ and that for all triples $u, v, w \in \mathbf{S}_G$, $(1 - \mathbf{S}_{u,v}^{G,0}) + (1 - \mathbf{S}_{v,w}^{G,0}) \geq (1 - \mathbf{S}_{u,w}^{G,0})$. We impose these constraints in order to ensure convergence of the recursion and to ensure \mathbf{S} and \mathbf{A} represent distances (more details are provided in the proof of Theorem IV.1). We follow the same procedure for \mathbf{A}^G .

C. Distance equations

In this section we restate the distance functions on which SS1 and our recursion algorithms depend. Namely, a reward distance δ_{rwd} , the Hausdorff distance δ_{Haus} , and the earth mover's distance δ_{EMD} . Let N_x , for $x \in V^M \cup \Lambda^M \cup V^N \cup \Lambda^M$, denote the out neighbors of x , then, for action node α and another set of action nodes N , define $\delta_A(\alpha, N) := \min_{\beta \in N} \delta_A(\alpha, \beta)$. The previously listed distance equations are then defined as follows:

$$\delta_{rwd} = |\mathbb{E}[r_\alpha] - \mathbb{E}[r_\beta]| \quad (3)$$

$$\delta_{Haus}(u, v; \delta_A) = \max_{\substack{\alpha \in N_u \\ \beta \in N_v}} \{\delta_A(\alpha, N_v), \delta_A(\beta, N_u)\} \quad (4)$$

$$\begin{aligned} \delta_{EMD}(p_\alpha, p_\beta; \delta_S) &= \min_{\mathbf{F}} \sum_{u \in N_\alpha} \sum_{v \in N_\beta} f_{u,v} \delta_S(\alpha, \beta) \\ \text{s.t. } &\forall u, v \in V : f_{u,v} \geq 0, \\ &\forall u \in V : \sum_{v \in V} f_{u,v} = p(\alpha, u), \\ &\forall v \in V : \sum_{u \in V} f_{u,v} = p(\beta, v). \end{aligned} \quad (5)$$

With this distance functions defined, we can restate the equations used in SS1's recursion. Choose values

$C_S, C_A \in (0, 1)$, where C_S can be thought of as an upper bound on state similarities given their action similarities are equal, and C_A can be thought of as a weighting factor between magnitudes of contribution of reward differences and state transition differences to final distance between actions. The state updates are given by the following equations:

$$\mathbf{S}_{u,v}^{G,i} = C_S(1 - \delta_{Haus}(u, v; 1 - \mathbf{A}^{G,i-1})) \quad (6)$$

and

$$\begin{aligned} \mathbf{A}_{\alpha,\beta}^{G,i} &= 1 - (1 - C_A)\delta_{rwd}(\alpha, \beta) \\ &\quad - C_A\delta_{EMD}(\alpha, \beta; 1 - \mathbf{S}^{G,i-1}). \end{aligned} \quad (7)$$

D. Algorithm

Combining definitions from previous sections, we are able to state our algorithm for determining inter-state distances as Algorithm 1. While we do not specify any specific convergence criteria, it is worth noting that the exact results produced Algorithm 1 will vary based on the criteria chosen. However, iterating the name number of times produces consistent results.

The algorithm requires as input the specification of the graph MDP $(V, \Lambda, E, \Psi, p, r)$, values for C_S, C_A , and the initialized values of S^G, A^G as previously described. For the remainder of this document, we assume the value of ω for each $S^{G,0}, A^{G,0}$ is set to C_S . Initializing S^G and A^G this way ensures convergence to a unique solution as shown in the proof of Lemma IV.1 and the following Corollary.

This choice of ω is further justified by considering the output of the Hausdorff distance when the nodes in the input have no out-neighbors. This results in the distance between nodes in two empty sets, which in theory, is trivially 0. Therefore, the outcome of Equation 6 will always be C_S . A similar argument can be made for the case where only one set is non-empty, in which case the Hausdorff distance is theoretically infinity, or in our case, the max distance of 1, which results in Equation 6 being 0.

Finally, we emphasize that the range of the reward function must be constrained to be within $[0, 1]$. For tasks where this is not the default reward scheme, the reward function can simply be normalized to fit into this interval for the purpose of this similarity (or distance) computation.

E. Comparison with Original Algorithm

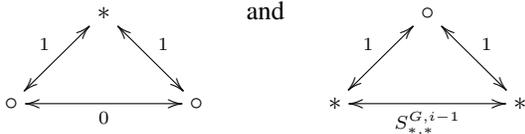
Before proceeding, we make a few remarks on Algorithm 1, in particular on how it differs from SS1.

Algorithm 1 Structural Similarity for two MDPs (SS2)

Require: Graph MDP = $(V, \Lambda, E, \Psi, p, r)$ **Require:** $C_s, C_a \in (0, 1), S^{G,0}, A^{G,0}$ **Ensure:** $(1 - \mathbf{S}_{u,v}^{G,0}) + (1 - \mathbf{S}_{v,w}^{G,0}) \geq (1 - \mathbf{S}_{u,w}^{G,0});$
 $\forall u, v, w \in V$ **Ensure:** $(1 - \mathbf{A}_{u,v}^{G,0}) + (1 - \mathbf{A}_{v,w}^{G,0}) \geq (1 - \mathbf{A}_{u,w}^{G,0});$
 $\forall u, v, w \in V$

- 1: **repeat**
 - 2: **for all** $\alpha \in N_u, \beta \in N_v: u, v \in V; u \neq v$ **do**
 - 3: $\mathbf{A}_{\alpha,\beta}^{G,i} \leftarrow 1 - (1 - C_A)\delta_{rwd}(\alpha, \beta)$
 $\quad - C_A\delta_{EMD}(\alpha, \beta; 1 - \mathbf{S}^{G,i-1})$
 - 4: **end for**
 - 5: **for all** $u, v \in V; N_u \neq \emptyset; N_v \neq \emptyset$ **do**
 - 6: $\mathbf{S}_{u,v}^{G,i} \leftarrow C_S(1 - \delta_{Haus}(u, v; 1 - \mathbf{A}^G))$
 - 7: **end for**
 - 8: **until** \mathbf{S}^G and \mathbf{A}^G converge
 - 9: **return** $(\mathbf{S}^{G,*}, \mathbf{A}^{G,*})$
 $\{\mathbf{S}^{G,*} = \text{state similarity matrix}, \mathbf{A}^{G,*} = \text{action}$
 $\text{similarity matrix}\}$
-

In Algorithm 1 we insist that the matrices $\mathbf{A}^{G,0}$ and $\mathbf{S}^{G,0}$ are initialized such that $1 - \mathbf{A}^{G,0}$ and $1 - \mathbf{S}^{G,0}$ are both distances. This is because, in the iterative step, the entries of $\mathbf{A}^{G,i}$ and $\mathbf{S}^{G,i}$ are defined by distance functions that take as an input the distance matrix $1 - \mathbf{A}^{G,i-1}$ or $1 - \mathbf{S}^{G,i-1}$. In other words, if $1 - \mathbf{A}^{G,i-1}$ and $1 - \mathbf{S}^{G,i-1}$ distances, then both $1 - \mathbf{A}^{G,i}$ and $1 - \mathbf{S}^{G,i}$ are guaranteed to be distances. This is obvious for $1 - \mathbf{A}^{G,i}$. It remains to check that if $1 - \mathbf{S}^{G,i-1}$ defines a distance on the non-absorbing nodes, then this extends to a distance on all nodes in \mathbf{S}_G . For this, we need only check the triangle inequality. Let \circ indicate an absorbing node and $*$ a non-absorbing node. The triangle inequality can be seen from the following diagrams



along with the fact that $0 \leq \mathbf{S}_{*,*}^{G,i-1} \leq 1$. This boundedness comes from the fact that δ_{rwd} , δ_{EMD} , and δ_{Haus} are bounded between 0 and 1, as are the constants C_A and C_S , due to the fact that rewards are constrained to values between 0 and 1. Thus we may conclude that $1 - \mathbf{A}^{G,i}$ and $1 - \mathbf{S}^{G,i}$ are distances.

We are unsure why these conditions were not specified for SS1. The examples the authors provided in their work satisfy these conditions, but in general they do not enforce it.

SS1 also does not specify the value of d_{uv} in the case where neither u or v is an absorbing node. It claims, however, that the entries of $\mathbf{A}^{G,i}$ and $\mathbf{S}^{G,i}$ are monotonically increasing. This is not true for all values of $d_{u,v}$. In our algorithm, we universally insist that $d_{u,v} = 0$ in the case where neither u or v is an absorbing node. The rationale for this is given in the proof of Lemma IV.1.

F. Properties of \mathbf{S}^G and \mathbf{A}^G

In general we claim that the matrices \mathbf{S}^G and \mathbf{A}^G have the same properties as σ_{S^*} and σ_{A^*} in SS1, and substantiate those claims here. First we clarify the claim in [26] that the sequences $\{\mathbf{A}_{a,b}^{G,i}\}_{i \in \mathbb{N}}$ and $\{\mathbf{S}_{u,v}^{G,i}\}_{i \in \mathbb{N}}$ are monotonically increasing for all $a, b \in \Lambda$ and $u, v \in V$.

Lemma IV.1. There exists an initial matrix $\mathbf{S}^{G,0}$ such that the matrices $\mathbf{A}^{G,i}$ and $\mathbf{S}^{G,i}$ are monotonically increasing.

Proof. Note that $1 - \mathbf{A}_{\alpha,\beta}^{G,i}$ is a weighted average of a fixed reward value and the earth mover's distance of the transition probabilities of α and β . Then $1 - \mathbf{S}_{u,v}^{G,i}$ is just the lowest highest value for some set of these weighted sums. If the entries of $1 - \mathbf{A}_{\alpha,\beta}^{G,i}$ are less than the entries of $1 - \mathbf{A}_{\alpha,\beta}^{G,i-1}$, then the same can be said for the entries of $1 - \mathbf{S}_{u,v}^{G,i}$. Furthermore, as the transition probabilities are fixed, the entries of $1 - \mathbf{A}_{\alpha,\beta}^{G,i}$ are less than the entries of $1 - \mathbf{A}_{\alpha,\beta}^{G,i-1}$ if and only if the entries of $1 - \mathbf{S}_{u,v}^{G,i}$ are smaller than those of $1 - \mathbf{S}_{u,v}^{G,i-1}$. Therefore, it is imperative all entries of $1 - \mathbf{S}_{u,v}^{G,0}$ exceed all entries of $1 - \mathbf{S}_{u,v}^{G,1}$. While there are non-trivial assignments that will satisfy this for any given graph G , this can be universally assured by setting $d_{u,v} = 0$. \square

Corollary IV.2. With the correct initial conditions, the entries of these matrices are guaranteed to converge.

Proof. This is a comes directly from monotonicity and the boundedness of these values \square

Theorem IV.3. Algorithm 1 defines a distance function on the state nodes of two graph MDPs, and a pseudo-metric on the corresponding action nodes.

Proof. As Theorem 2 of SS1 does not specify that the graph must be connected. Therefore, the proof holds for this disconnected case at hand. \square

The only case that needs to be understood here is what happens if the graph $G = G_M \sqcup G_N$ is the disjoint union of two identical graphs ($G_M \simeq G_N$). In this case, one initializes $S^{G,0}$ to be the identity matrix. Then for the corresponding isomorphic actions α_M and α_N in the

components G_M and G_N respectively, $\mathbf{A}_{\alpha_M, \alpha_N}^{G,1} = 1$. This is because the rewards and the transition probabilities are identical for these two functions. Therefore, if v_M and v_N are the source nodes for these actions, respectively, $\mathbf{S}_{v_M, v_N}^{G,1} = C_S$. By similar argument, we see that these values do not change for any i . Therefore, the value C_S denotes the discount one puts on two different states being isomorphic but distinct.

V. ALGORITHM SIMPLIFICATION

In the previous section we show that the matrix $1 - \mathbf{S}^G$ is a distance metric for two disconnected components $G_M \simeq G_N = G$, so the submatrix $\mathbf{S}^{M,N}$ is a distance metric between the states in MDP M and MDP N , of which we are chiefly interested. In fact, we need not compute all of \mathbf{S}^G in order to obtain $\mathbf{S}^{M,N}$, we can simply compute it directly by only considering $u \in V_M$ and $v \in V_N$ in both `for` loops. In this case, we initialize $\mathbf{S}^{M,N}$ and $\mathbf{A}^{M,N}$ (\mathbf{S}^G 's and \mathbf{A}^G 's respective replacements in algorithm 1) as $C_S \cdot \mathbf{I}$, where \mathbf{I} is the identity matrix.

This is justified by examining the data dependencies of each Hausdorff (as used to compute entries in \mathbf{S}^G) and Kantorovich (as used to compute entries in \mathbf{A}^G) operation. In the former, note that while the entirety of the \mathbf{A}^G matrix is passed in as the distance function, only the subset of that matrix where the rows are the actions of state u and columns are the actions of state v is actually used in computing the entry at $\mathbf{S}_{u,v}^G$. That is to say, computing the state similarity within each sub-matrix only relies on the action similarities of actions belonging to states within that sub-matrix.

Furthermore, these entries in \mathbf{A}^G rely on the Kantorovich distances between the action α and β from u and v respectively. When expanding α into a full probability distribution, note that the probability of transitioning from MDP M to N (or vice versa) via any action is 0. Thus, those entries in the distribution have 0 weight associated with them, and are disregarded when computing the overall distance. Thus, only the weights in \mathbf{A}^G corresponding to the sub-matrix in which u and v belong to have any bearing on the computation, and each sub-matrix can therefore be computed independently.

Regarding run-time, the big-O complexity of SS1 still holds for this simplification, as it still scales with the number of nodes in the set of state nodes and action nodes in the graph MDP; however, the memory and computation reduction from reducing the \mathbf{S}^G matrix ($|S_M + S_N|^2$ entries) to $\mathbf{S}_{M,N}$ ($|S_M||S_N|$ entries) can be significant in practice. In essence the complexity has changed from $O(N_{iter} \cdot |S|^2 |A|^2 K_{max}^2 / \epsilon^2)$ to

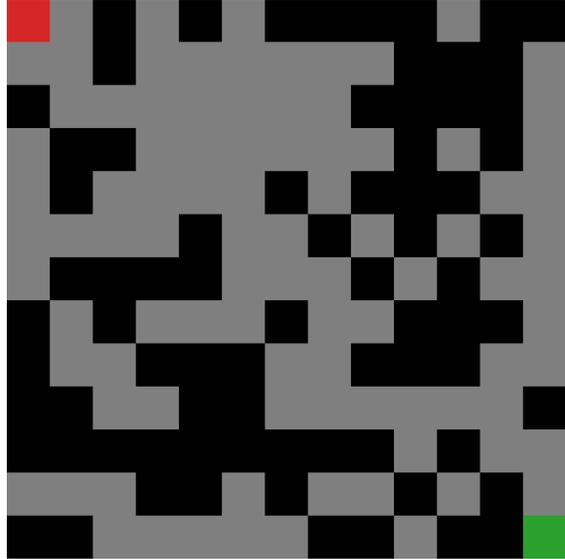


Fig. 1. An example maze used in the transfer experiments. The agent begins in the upper-left red state and must navigate to the lower-right green state.

$O(N_{iter} \cdot |S_M||S_N||A|^2 K_{max}^2 / \epsilon^2)$, where $|S_M| < |S|$ and $|S_N| < |S|$, where N_{iter} is the number of iterations the algorithms runs before converging and $K_{max} \leq |V|$ is the maximum out-degree of action nodes in G .

VI. EXPERIMENT METHODOLOGY

We validate SS2 by conducting several experiments wherein we assess the relationship between its similarity measurements and transfer performance. In general, our setting consists of finite MDPs representing various gridworld environments (Figure 1). In each experiment, we consider a given target environment and a corpus of source environments, each environment consisting of a single start state and single goal state. We then utilize table-based Q Learning to train an agent to 90% optimal performance, as measured by the moving average of number of steps required to complete an episode compared against the optimal path length [27]. Finally, we use the transfer methods proposed and assessed in [18], as well as a novel expansion of those methods that utilizes the action similarities ($\mathbf{A}^{G,*}$) captured by SS2, and assess the performance of each source environment to the fixed target environment.

A. Environment Generation

We define our tasks by randomly generating a target environment and 100 source environments. The goal of the task is to navigate the environment from the

top leftmost square to the bottom rightmost square, at which point the agent is given a positive reward. Each environment generated is populated with randomly distributed obstacles, but ensures that there exists at least one path to the goal. Additionally, we repeat this generation process while modifying three independent variables to better understand how different conditions affect performance. These variables, and their associated settings, are as follows:

- 1) Grid size: either “Small” (9x9) or “Large” (13x13)
- 2) Rotations: whether or not to include random rotations when generating the source tasks. If included, then for each source task the goal state is chosen to be one of four random corners, with the start state set to the opposite corner along the diagonal.
- 3) Reward: either 1 or 100, associated with the goal state. All other states have no rewards or costs associated.

This results in a total of eight experimental conditions, each with their own target task and set of 100 source tasks. Additionally, note that in these transfer experiments, the MDPs generated are deterministic, unless otherwise specified.

B. Agent Training

We use a simple version of Q-Learning [27, 23] when training the initial source agents. We first set an optimality criterion by performing a breadth-first search from the environment’s start state to goal state, where there is unit cost in moving to an adjacent state. We then use a learning rate $\alpha = 0.1$, discount factor $\gamma = 0.9$, and an ϵ -greedy strategy starting at $\epsilon = 0.9$ to train each agent until the moving average of its last 20 episodes has reached 90% optimal. We utilize a simple form of simulated annealing by decreasing the ϵ value by a constant decay amount of 10^{-6} , with a minimum value of 0.1. The agents tend to converge after approximately 5×10^5 steps.

C. Transfer Methodology

We consider metric-method pairs in a factorial design. In these experiments, we utilize SS2, as well as the similarity metric, d' , defined by Song et al. in [18] as a point of comparison, as it is similar to ours. Song’s similarity metric is also computed recursively, following Equation 8.

$$d'(s, s') = \max_{a \in A} |\mathbb{E}[r_s^a] - \mathbb{E}[r_{s'}^a]| - C\delta_{EMD}(P_s^a, P_{s'}^a; d') \quad (8)$$

Algorithm 2 Weight Transfer with Action (T-AVG-ACT)

Require: $\delta_{\hat{S}}, \delta_{\hat{A}}$
Require: $Q_{in} \in \mathbb{R}^{|S_{in}| \times |A_{in}|}$
Require: $N_u = N_v, \forall u, v \in S_{in} \times S_{out}$

- 1: $Q_{out} \leftarrow \mathbf{0}^{|S_{out}| \times |A_{out}|}$
- 2: $K \leftarrow EMD(U_{|S_{in}|}, U_{|S_{out}|}, \delta_{\hat{S}})$
- 3: **for all** $s_i \in S_{in}, s_o \in S_{out}$ **do**
- 4: $w \leftarrow K[s_i, s_o] / (\sum_{s'_o \in S_{out}} K[s_i, s'_o])$
- 5: $A_{sub} \leftarrow \delta_{\hat{A}}[N_{s_i}, N_{s_o}]$
- 6: **for all** $a_o \in N_{s_o}$ **do**
- 7: $a_m \leftarrow \text{argmin } A_{sub}[:, a_o]$
- 8: **if** $A_{sub}[a_o, a_o] = A_{sub}[a_m, a_o]$ **then**
- 9: $a \leftarrow a_o$
- 10: **else**
- 11: $a \leftarrow a_m$
- 12: **end if**
- 13: $Q_{out}[s_o, a_o] \leftarrow w \times Q_{in}[s_i, a]$
- 14: **end for**
- 15: **end for**
- 16: **return** Q_{out}

TABLE I
ANOVA RESULTS. REPORTED P-VALUES ARE BONFERRONI CORRECTED; REWARD IS OMITTED AS ITS P-VALUE WAS 1.0000 IN ALL CASES.

Algorithm	Dimension P-Value	Rotate P-Value
SS2, T-STATE-ACT	4.9812e-02	1.0000e+00
SS2, T-AVG-ACT	3.9507e-30	1.0000e+00
SS2, T-STATE	1.1683e-30	1.2733e-22
SS2, T-AVG	1.2963e-48	6.3088e-01
Song, T-STATE	3.8663e-23	3.1010e-44
Song, T-AVG	2.1109e-63	3.6490e-01
Uniform, T-STATE	8.9806e-07	3.7355e-45
Uniform, T-AVG	1.0000e+00	5.6624e-73

In Equation 8, $\mathbb{E}[r_s^a]$ is the expected reward for taking action a at state s , P_s^a is the transition probability distribution for taking action a at state s , and C is a user defined constant. We also include a baseline `Uniform` metric which returns a constant distance from each source state to target state. For fairness of comparison, we ensure that the convergence criteria for each metric is identical, as both our metric and Song’s utilize fixed point iterations to determine when the matrices have stopped changing. This criterion is satisfied when all pairwise comparisons between the previous iteration’s distance values d and current iteration’s distance values d' meet the following condition: $|d' - d| \leq 10^{-8} + 10^{-5} \times |d|$.

Similarly, we ensure that constants that are common to both algorithms (i.e. Song’s analogue of C_A) have

Algorithm 3 State Transfer with Action (T-STATE-ACT)

Require: $\delta_{\hat{S}}, \delta_{\hat{A}}$ **Require:** $Q_{in} \in \mathbb{R}^{|S_{in}| \times |A_{in}|}$ **Require:** $N_u = N_v, \forall u, v \in S_{in} \times S_{out}$

```
1:  $Q_{out} \leftarrow \mathbf{0}^{|S_{out}| \times |A_{out}|}$ 
2: for all  $s_i \in S_{in}, s_o \in S_{out}$  do
3:   if  $\delta_{\hat{S}}[s_i, s_o] = \min \delta_{\hat{S}}[:, s_o]$  then
4:      $w \leftarrow 1$ 
5:   else
6:      $w \leftarrow 0$ 
7:   end if
8:    $A_{sub} \leftarrow \delta_{\hat{A}}[N_{s_i}, N_{s_o}]$ 
9:   for all  $a_o \in N_{s_o}$  do
10:     $a_m \leftarrow \operatorname{argmin} A_{sub}[:, a_o]$ 
11:    if  $A_{sub}[a_o, a_o] = A_{sub}[a_m, a_o]$  then
12:       $a \leftarrow a_o$ 
13:    else
14:       $a \leftarrow a_m$ 
15:    end if
16:     $Q_{out}[s_o, a_o] \leftarrow w \times Q_{in}[s_i, a]$ 
17:  end for
18: end for
19: return  $Q_{out}$ 
```

the same value. In this case, we set $C_A = 0.5$. We also initialize $C_S = 0.9995$, but note that d' has no corresponding constant.

We use methods described in Song’s work, namely Weight and State transfer [18] to transfer agent knowledge from one task for use in another. The Weight transfer method (not to be confused with neural network weights) initializes the Q function, or value function, of the target task by using the values of the Q-function (or value function) in the source task(s). Each contribution is weighed by the corresponding entry of the optimal transport matrix computed by the Kantorovich metric upon the State-State distance matrix. As stated in I, we refer to this method as T-AVG. The State transfer, which we will henceforth refer to as T-STATE is similar, but simply sets the Q-function (or value function) for each state by selecting the entry of the source task corresponding to the state that is least distant to the target state. That is, only the most similar source state to each target state will contribute to its initialization.

We then extend these methods by using the action similarity matrix, (\mathbf{A}^G) to better inform mappings between actions in each source-target state pair, as described in Algorithm 2 (T-AVG-ACT). In performing each transfer

experiment, we utilize the specified metric to obtain a distance matrix $\delta_{\hat{S}}$ between states in each source environment and the fixed target environment, as well as a corresponding distance matrix $\delta_{\hat{A}}$ between actions.

Intuitively then, Algorithm 2 can be seen as assigning the minimally distant source action to each target action, breaking ties by the original action ordering alignment. Note that this action-space correspondence precondition is required by Song’s metric, but we are only using it as a heuristic to break ties. Algorithm 3 (T-STATE-ACT) is quite similar, except it weighs only the least distant State-State pair, similar to T-STATE.

Finally, we then use each of the transfer methods to initialize the Q table for the target environment. Note that while all four methods can support transferring information from multiple source tasks to a single target task, we only consider the one-to-one case in this experiment. We train the resulting agents for 10^3 steps, with a fixed ϵ value of 0.1 (other hyper-parameters are held the same as when training the source tasks). We repeat this procedure for 50 trials, to reduce variance that results from ϵ -greedy exploration, and store each resulting performance curve.

Note that the transfer variants that use the action similarity matrix are only applicable to our algorithm, since d' only computes state similarity. This results in a total of eight metric-method combinations.

VII. RESULTS

A. Transfer Performance

We begin by calculating the Average Episodes Completed across the 50 trials for each (source, metric-method, experiment) tuple, which is simply the mean number of episodes the agents were able to complete in a chosen measurement period (in our case, the first 2500 steps of training). This enables us to compute an Episode Performance which is the average number of steps per episode observed (i.e. 2500 / Average Episodes Completed). We then calculate a Relative Performance by comparing this Episode Performance to the optimal step count in the target environment, via Optimal Length / Episode Performance, and converting to a percentage by multiplying by 100.

We conduct an Analysis of Variance (ANOVA) test for each of the three independent factors used in creating the environments [19]. In these tests, the dependent variable was the Relative Performance. The Bonferroni corrected p-values can be seen in Table I [4]. As shown, for most metric-method pairs, both grid size (Dimension) and the presence of rotations (Rotate)

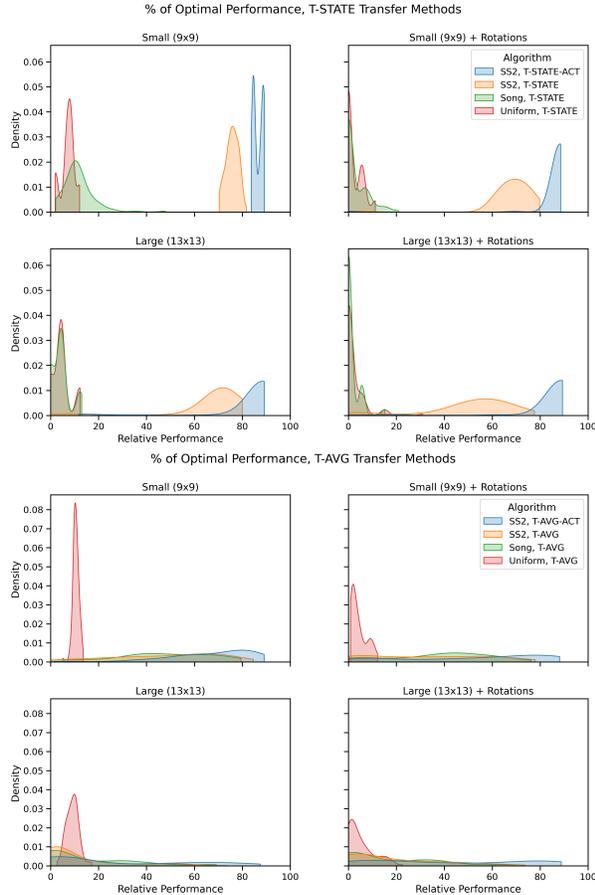


Fig. 2. Distributions of agent performance over all source tasks, averaged over 50 trials. The x-axis represents the average number of episodes completed by an agent in the first 2500 steps of training divided by the number of episodes completed by an optimal agent in the same number of steps, which we call *Relative Performance*. In essence, farther right indicates closer to optimal performance.

were statistically significant in affecting transfer performance, while the reward amount (Reward) was not significant for any pair. Thus, in the remainder of our analyses and visualizations, we focus only on grid size and presence of rotations, using the results in the reward = 100 experiments to most directly compare with T-AVG and T-STATE.

The full results consisting of the mean and standard deviations for each metric-method in each experiment can be seen in Table II. Interestingly, across all conditions for SS2, T-STATE-ACT dominated T-AVG-ACT and T-STATE dominated T-AVG. However, for Uniform and d' similarities, T-AVG transfer dominated T-STATE transfer.

We thus choose to highlight these four metric-method

pairs, as they are the best methods for each metric:

- 1) SS2 Metric, T-STATE-ACT Transfer
- 2) SS2 Metric, T-STATE Transfer
- 3) d' Metric, T-AVG Transfer
- 4) Uniform Metric, T-AVG Transfer

These results are highlighted in Figures 2 and 3. Figure 2 highlights the percent of optimal performance able to be achieved in the first 2500 iterations, as a density plot over the 100 source tasks. Figure 3 displays the cumulative number of episodes completed over time, averaged over the 100 source tasks. Overall we observe that SS2, with both the T-STATE-ACT and T-STATE transfer methods, significantly outperforms d' , both in terms of consistency across the source tasks as well as transfer performance attained. We suspect that this is due to the action distances effectively allowing our algorithm to leverage states which are similar despite their out-actions being misaligned (e.g. states that are one move away from the goal should be similar even if the action to perform that move is different).

VIII. DISCUSSION

A. Structural Similarity; MDP Distance

Ideally, we would want a single value to characterize the overall similarity between two MDPs, a straightforward way to do this is as follows:

Using Algorithm 1 or a simplified version, assume we may now compute the function $\delta_{\hat{S}} = 1 - \sigma_{\hat{S}} = 1 - \sigma_{S_M \cup S_N} = 1 - \mathbf{S}^{G,*}$, for MDPs $M = (S_M, A, P_M, R_M)$ and $N = (S_N, A, P_N, R_N)$. We analogously define $\delta_{\hat{A}} = 1 - \mathbf{A}^{G,*}$. We may now apply the measures used in [18] to compute a final similarity measure between the MDPs, i.e. the Hausdorff and Kantorovich metrics. Assuming M, N and S defined as stated previously, we state the following definitions:

Definition VIII.1. The *Hausdorff structural similarity distance*, Φ , of M and N is defined as

$$\Phi(M, N) = \delta_{Haus}(V_M, V_N, \delta_{\hat{S}}),$$

where V_M and V_N are the state nodes of the MDP graphs for M and N , respectively.

Definition VIII.2. The *Kantorovich structural similarity distance*, Ψ , is defined as

$$\Psi(M, N) = \delta_{EMD}(U_{|S_M|}, U_{|S_N|}, \delta_{\hat{S}}),$$

where U_a is a discrete Uniform distribution with $p = \frac{1}{a}$.

We tested these measures using the Q-learning framework we used for VI, but we did not find correlation

Average Number of Episodes Completed, Best Transfer Methods for Each Metric

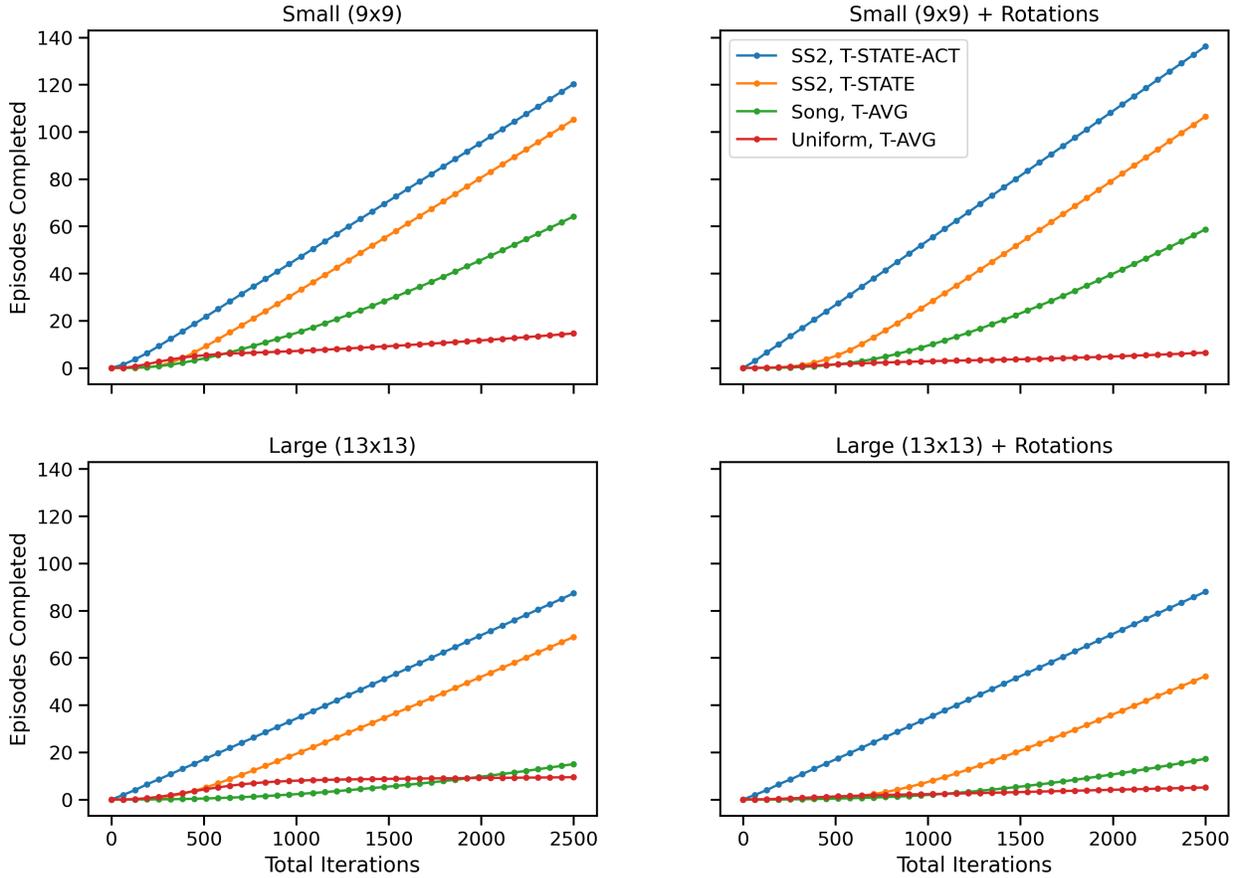


Fig. 3. Performance curves of best transfer methods for each metric over time. Data points are averaged over 50 trials and 100 source tasks.

between transfer performance and MDP distances (for more details, see Appendix A. However, because the state distances computed using SS2 seem to be useful in terms of improving transfer from one task to another, we believe the issue is in the reduction technique, i.e. the use of the Kantorovich and Hausdorff distances. Discovering other techniques to reduce the information from $\mathbf{S}^{G,*}$ into a single value may be a possible avenue of future research, but another potentially more fruitful avenue might be how to better leverage the similarity information contained in $\mathbf{S}^{G,*}$ and $\mathbf{A}^{G,*}$ to improve transfer in other RL tasks and algorithms.

B. Limitations

Our results here suggest that using $\mathbf{S}^{G,*}$ and $\mathbf{A}^{G,*}$ as similarity characterizations to improve transfer between RL tasks has potential, but we acknowledge it currently

has various limitations. The most obvious is that SS2 is limited to finite MDPs that can be fully specified (i.e. complete knowledge of the transition and reward functions), and even in this case, computing $\mathbf{S}^{G,*}$ and $\mathbf{A}^{G,*}$ can be very expensive as the size of the state and action spaces increase.

Another limitation we found in our experiments is that results are strongly dependent on how one defines the MDP for a given task. For example, DRL often assumes that every action is available to the agent at every step, which is not always the case. Choosing to include action nodes for every action at every state (where unavailable actions would be modeled by say, returning to the current state with probability 1.0) will produce different similarity values than omitting actions nodes where the actions are not available. This is a modeling choice for how to deal with obstacles in the

gridworld, but has a significant effect on the similarity values returned by SS2.

C. Future Work

Future work of interest on this subject could include performing additional experiments to better characterize the relationship between the state similarities returned by SS2 and transfer performance between tasks. In particular, we would find experiments that include MDP environments other than the traditional gridworld of interest. Another avenue would be assessing transfer in the continual learning setting [16, 10], even with simple gridworld tasks. How might we leverage similarity measures such as those from SS2 to continuously modify an RL agent over a lifetime of performing a set of tasks? Finally, an area of particular interest would be to extend SS2 to operate on larger MDPs through sampling techniques, such those used by [6, 7].

IX. CONCLUSION

We proposed the Structural Similarity for two MDPs, or SS2, algorithm for computing state similarities between two MDPs. We proved that the result of SS2 can be used to define a distance metric using results from [26], upon which SS2 is based. We then showed that by leveraging the state-to-state similarity results from SS2 with RL transfer algorithms from [18], we could improve transfer between tasks in a gridworld navigation environment over those shown in [18] for the same tasks. Furthermore, by incorporating action similarity information from SS2 into the transfer algorithm, we further improved transfer performance.

We also tested task-to-task similarity measures comprising the SS2 outputs and the Hausdorff and Kantorovich distance metrics, and found that transfer performance did not correlate with those similarity scores. Therefore, we conclude that, while the state-to-state similarity values produced by SS2 between two MDPs is useful for transfer, reducing those values into a single scalar one, using the Hausdorff and Kantorovich distances in particular, do not produce useful measures of task similarity for our environment and RL algorithm. Future work should include additional verification of these results with additional tasks and algorithms, and investigate additional uses of information in SS2 for RL tasks. Other natural extensions could include attempting to apply SS2 to larger MDPs via sampling and graph reduction as well as extending SS2 to return state-to-state similarities for any finite number of tasks represented as finite MDPs.

ACKNOWLEDGEMENTS

This work was funded by the DARPA Lifelong Learning Machines (L2M) Program. The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

REFERENCES

- [1] Haitham Bou Ammar et al. “An automated measure of mdp similarity for transfer in reinforcement learning”. In: *Workshops at the Twenty-Eighth AAAI Conference on Artificial Intelligence*. 2014.
- [2] Jacob Benesty et al. “Pearson correlation coefficient”. In: *Noise reduction in speech processing*. Springer, 2009, pp. 37–40.
- [3] Yoshua Bengio et al. “Curriculum learning”. In: *Proceedings of the 26th annual international conference on machine learning*. 2009, pp. 41–48.
- [4] Carlo Bonferroni. “Teoria statistica delle classi e calcolo delle probabilita”. In: *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commerciali di Firenze* 8 (1936), pp. 3–62.
- [5] James Lamond Carroll. “Task Localization, Similarity, and Transfer; Towards a Reinforcement Learning Task Library System”. In: (2005).
- [6] Pablo Samuel Castro. *Scalable methods for computing state similarity in deterministic Markov Decision Processes*. 2019. arXiv: 1911.09291 [cs.LG].
- [7] Pablo Samuel Castro et al. “MICo: Improved representations via sampling-based state similarity for Markov decision processes”. In: *Advances in Neural Information Processing Systems* 34 (2021).
- [8] Alexey Dosovitskiy et al. “CARLA: An open urban driving simulator”. In: *Conference on robot learning*. PMLR. 2017, pp. 1–16.
- [9] Norm Ferns, Prakash Panangaden, and Doina Precup. “Metrics for Finite Markov Decision Processes.” In: *UAI*. Vol. 4. 2004, pp. 162–169.
- [10] Khimya Khetarpal et al. *Towards Continual Reinforcement Learning: A Review and Perspectives*. 2020. arXiv: 2012.13490 [cs.LG].
- [11] MM Mahmud et al. “Clustering markov decision processes for continual transfer”. In: *arXiv preprint arXiv:1311.3959* (2013).

- [12] Volodymyr Mnih et al. “Human-level control through deep reinforcement learning”. In: *Nature* 518.7540 (Feb. 2015), pp. 529–533. ISSN: 00280836. URL: <http://dx.doi.org/10.1038/nature14236>.
- [13] Sanmit Narvekar et al. “Curriculum learning for reinforcement learning domains: A framework and survey”. In: *Journal of Machine Learning Research* 21.181 (2020), pp. 1–50.
- [14] OpenAI et al. “Dota 2 with Large Scale Deep Reinforcement Learning”. In: (2019). arXiv: 1912.06680. URL: <https://arxiv.org/abs/1912.06680>.
- [15] Sinno Jialin Pan and Qiang Yang. “A survey on transfer learning”. In: *IEEE Transactions on knowledge and data engineering* 22.10 (2009), pp. 1345–1359.
- [16] Mark Bishop Ring et al. “Continual learning in reinforcement environments”. In: (1994).
- [17] Sergio Arredondo Serrano et al. “Inter-Task Similarity for Lifelong Reinforcement Learning in Heterogeneous Tasks”. In: (2021).
- [18] Jinhua Song et al. “Measuring the Distance Between Finite Markov Decision Processes”. In: *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*. AAMAS ’16. Singapore, Singapore: International Foundation for Autonomous Agents and Multiagent Systems, 2016, pp. 468–476. ISBN: 9781450342391.
- [19] Lars St, Svante Wold, et al. “Analysis of variance (ANOVA)”. In: *Chemometrics and intelligent laboratory systems* 6.4 (1989), pp. 259–272.
- [20] Matthew E. Taylor and Peter Stone. “Transfer Learning for Reinforcement Learning Domains: A Survey”. In: *Journal of Machine Learning Research* 10.56 (2009), pp. 1633–1685. URL: <http://jmlr.org/papers/v10/taylor09a.html>.
- [21] Sebastian Thrun. “Lifelong learning algorithms”. In: *Learning to learn*. Springer, 1998, pp. 181–209.
- [22] Emanuel Todorov, Tom Erez, and Yuval Tassa. “Mujoco: A physics engine for model-based control”. In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5026–5033.
- [23] Gelana Tostaeva. *Introduction to Q-learning with OpenAI Gym*. 2020. URL: <https://medium.com/swlh/introduction-to-q-learning-with-openai-gym-2d794da10f3d> (visited on 02/12/2020).
- [24] Oriol Vinyals et al. “Grandmaster level in StarCraft II using multi-agent reinforcement learning”. In: *Nature* 575.7782 (2019), pp. 350–354.
- [25] Álvaro Visús, Javier García, and Fernando Fernández. “A Taxonomy of Similarity Metrics for Markov Decision Processes”. In: *arXiv preprint arXiv:2103.04706* (2021).
- [26] Hao Wang, Shaokang Dong, and Ling Shao. “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, July 2019, pp. 3684–3690. DOI: 10.24963/ijcai.2019/511. URL: <https://doi.org/10.24963/ijcai.2019/511>.
- [27] Christopher JCH Watkins and Peter Dayan. “Q-learning”. In: *Machine learning* 8.3-4 (1992), pp. 279–292.
- [28] Amy Zhang et al. “Learning invariant representations for reinforcement learning without reconstruction”. In: *arXiv preprint arXiv:2006.10742* (2020).
- [29] Zhuangdi Zhu, Kaixiang Lin, and Jiayu Zhou. “Transfer learning in deep reinforcement learning: A survey”. In: *arXiv preprint arXiv:2009.07888* (2020).

APPENDIX A
MDP DISTANCE AND TRANSFER CORRELATION

To assess the effectiveness of the MDP Distance, as discussed in Section VIII-A, we computed the Pearson’s correlation coefficient, for all metric-method combinations in their individual conditions, as well as across all conditions combined [2]. These results can be seen in full in Tables III and IV, regarding the final Kantorovich and Hausdorff distances respectively. We desired a strong negative correlation (i.e. greater distance implying worse performance). There were two key takeaways from performing this analysis:

- 1) No metric-method resulted in having the desired negative correlation across all conditions combined over all sizes, rewards, and rotations, using either of the two final distance scores. The implication here is that while the distance matrices can effectively be used to perform transfer, they do not seem to be predictive of transfer in the sense of simply “higher distance” \implies “worse transfer”. This is likely because the distances observed in each experiment tended to group together (e.g. having a larger reward in Song’s metric would lead to a larger distance, scaled linearly, but not necessarily any difference in transfer performance).
- 2) In individual experiments, some metric-method combinations demonstrated correlation, although note again that no combination had a consistent correlation across the conditions. One reason why our metric with its best method (i.e. state and action transfer) did not exhibit correlation could be the fact that regardless of the distance, the transfer performance was quite good. This highlights the need to further experiment in more complex scenarios or perhaps different environments altogether, which we hope to explore in the near future.

TABLE II
FULL STATISTICAL RESULTS. ENTRIES ARE "MEAN (STD)" FOR THE GIVEN ALGORITHM AND CONDITION.

Condition	SS2, AVG	SS2, AVG-ACT	SS2, STATE	SS2, STATE-ACT	Song, AVG	Song, STATE	Uniform, AVG	Uniform, STATE
Lg, R1	11.3 (16.6)	26.1 (29.9)	66.0 (17.4)	84.2 (14.9)	14.6 (18.2)	4.3 (3.9)	9.0 (2.5)	4.6 (3.8)
Lg, R1, Rot	20.8 (21.2)	42.2 (33.0)	51.0 (19.0)	85.4 (14.8)	16.0 (18.8)	1.5 (3.2)	5.0 (5.5)	1.6 (3.4)
Lg, R100	11.1 (17.0)	25.9 (29.9)	66.1 (17.5)	83.8 (16.4)	14.4 (18.5)	4.3 (3.9)	9.1 (2.6)	4.6 (3.8)
Lg, R100, Rot	18.9 (19.8)	42.4 (32.7)	50.1 (19.5)	84.5 (16.1)	16.6 (18.9)	1.4 (3.1)	4.9 (5.4)	1.9 (4.5)
Sm, R1	47.9 (22.3)	67.1 (18.4)	75.7 (2.5)	86.6 (2.0)	45.9 (21.0)	11.6 (6.6)	10.5 (1.2)	7.2 (2.8)
Sm, R1, Rot	33.1 (24.5)	52.0 (30.8)	68.0 (10.7)	86.9 (9.6)	37.4 (20.7)	2.7 (4.6)	4.1 (3.2)	2.2 (3.2)
Sm, R100	47.9 (22.3)	67.1 (18.5)	75.7 (2.5)	86.6 (2.0)	46.2 (20.7)	11.6 (6.6)	10.5 (1.3)	7.2 (2.9)
Sm, R100, Rot	32.9 (24.5)	51.9 (30.6)	68.1 (10.6)	87.2 (9.0)	37.5 (20.9)	2.8 (4.6)	4.1 (3.2)	2.3 (3.3)

TABLE III
KANTAROVICH PEARSON CORRELATION RESULTS: DESIRED RELATION IS NEGATIVE.

Condition	SS2, STATE-ACT	SS2, AVG-ACT	SS2, STATE	SS2, AVG	Song, STATE	Song, AVG	Uniform, STATE	Uniform, AVG
All	0.099	0.138	0.198	0.120	-0.011	0.117	0.211	0.095
Lg, R1	0.224	-0.083	0.168	-0.109	-0.081	0.025	-0.059	0.205
Lg, R1, Rot	-0.159	0.084	-0.148	-0.088	0.174	-0.227	0.024	-0.015
Lg, R100	0.148	-0.072	0.140	-0.123	-0.084	-0.059	-0.060	0.190
Lg, R100, Rot	-0.142	0.102	-0.135	-0.072	0.169	-0.264	0.032	-0.010
Sm, R1	0.156	-0.263	0.099	-0.343	-0.408	-0.371	-0.115	0.230
Sm, R1, Rot	0.208	-0.113	0.149	-0.161	-0.488	-0.382	-0.183	-0.021
Sm, R100	0.145	-0.261	0.113	-0.344	-0.408	-0.378	-0.119	0.216
Sm, R100, Rot	0.237	-0.114	0.156	-0.163	-0.474	-0.392	-0.172	-0.023

TABLE IV
HAUSDORFF PEARSON CORRELATION RESULTS: DESIRED RELATION IS NEGATIVE.

Condition	SS2, STATE-ACT	SS2, AVG-ACT	SS2, STATE	SS2, AVG	Song, STATE	Song, AVG	Uniform, STATE	Uniform, AVG
All	0.014	-0.046	-0.056	-0.083	-0.042	-0.020	0.211	0.095
Lg, R1	0.017	0.015	-0.001	-0.039	-0.013	0.030	-0.059	0.205
Lg, R1, Rot	-0.101	0.090	-0.160	0.044	0.225	-0.186	0.024	-0.015
Lg, R100	0.076	0.016	0.086	-0.007	-0.016	0.051	-0.060	0.190
Lg, R100, Rot	-0.141	0.074	-0.146	-0.031	0.216	-0.228	0.032	-0.010
Sm, R1	0.260	-0.323	0.158	-0.222	-0.233	-0.275	-0.115	0.230
Sm, R1, Rot	0.150	-0.230	0.201	-0.188	-0.431	-0.244	-0.183	-0.021
Sm, R100	0.235	-0.320	0.171	-0.223	-0.233	-0.266	-0.119	0.216
Sm, R100, Rot	0.162	-0.229	0.200	-0.186	-0.425	-0.263	-0.172	-0.023