Using Bisimulation Metrics to Analyze and Evaluate Latent State Representations

Albers, N.; Suau de Castro, M.; Oliehoek, F.A.

Publication date
2021

Document Version
Final published version

Published in
BNAIC/BeneLearn 2021

Citation (APA)
Albers, N., Suau de Castro, M., & Oliehoek, F. A. (2021). Using Bisimulation Metrics to Analyze and Evaluate Latent State Representations. In E. L. A. Leiva, C. Pruski, R. Markovich, A. Najjar, & C. Schommer (Eds.), *BNAIC/BeneLearn 2021: 33rd Benelux Conference on Artificial Intelligence and 30th Belgian-Dutch Conference on Machine Learning* (pp. 320-334)

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Using Bisimulation Metrics to Analyze and Evaluate Latent State Representations

Nele Albers[1 [0000−0002−0502−6176]], Miguel Suau[1], and Frans A. Oliehoek[1]

Intelligent Systems, Delft University of Technology, Delft, The Netherlands
{n.albers, m.suaudecastro, f.a.oliehoek}@tudelft.nl

Abstract. Deep Reinforcement Learning (RL) is a promising technique towards constructing intelligent agents, but it is not always easy to understand the learning process and the factors that impact it. To shed some light on this, we analyze the Latent State Representations (LSRs) that deep RL agents learn, and compare them to what such agents should ideally learn. We propose a crisp definition of 'ideal LSR' based on a bisimulation metric, which measures how behaviorally similar states are. The ideal LSR is that in which the distance between two states is proportional to this bisimulation metric. Intuitively, forming such an ideal representation is highly favorable due to its compactness and generalization properties. Here we investigate if this type of representation is also desirable in practice. Our experiments suggest that learning representations that are close to this ideal LSR may improve upon generalization to new irrelevant feature values and modified dynamics. Yet, we show empirically that the extent to which such representations are learned depends on both the network capacity and the state encoding, and that with the current techniques the exact ideal LSR is never formed.

Keywords: Deep Reinforcement Learning · Bisimulation Metrics.

1 Introduction

Recent years have seen a surge of algorithms and architectures for deep Reinforcement Learning (RL), many of which have shown remarkable success for various problems. Yet, little work has attempted to relate the performance of these algorithms and architectures to what the resulting deep RL agents actually learn, and whether this corresponds to what we suppose they should ideally learn. Such a comparison may allow for both an improved understanding of why certain algorithms or network architectures perform better than others and the development of methods that specifically address discrepancies between what is and what should be learned. We thus explore empirically the Latent State Representations (LSRs) a deep RL agent forms of its environment to see whether these match our theoretical expectations.

When we speak of what a deep RL agent learns, we mean the internal representation that a neural network forms of the environment. That is, the activation patterns that arise in each hidden network layer as the result of feeding

(histories of) observations to the network. As the observation space is potentially very large and the capacity of an RL agent is limited, an agent has to learn what to attend to when creating this internal representation. A robot that is trained to fight fires in a residential area, for instance, might learn that certain features such as the house colors do not matter. If so, it will map two observations that differ only in this feature to the same activation pattern. The house color will then no longer influence the action choices, as the agent has learned to ignore it.

Among the desirable properties of such an LSR are that it should make only necessary distinctions between (histories) of observations, allow the agent to learn to act optimally, and enable generalization to new irrelevant feature values and modified dynamics. An LSR that has these properties is one in which the Euclidean distances between states are proportional to a bisimulation metric [6], which measures how "behaviorally different" [7] states are. As such an LSR makes only those distinctions that are needed for the prediction of the next reward and state [12], we call it the *Coarsest Markov State Representation* (CMSR). It is this CMSR that we suppose a deep RL agent should ideally learn. Our main contribution is that we propose a way to measure the degree to which the CMSR is learned, and use this measure to gain insights into the learning process of deep RL agents using Deep Q-Networks (DQNs) [22] as example. Moreover, we show empirically that learning closer to the CMSR may lead to better generalization to new irrelevant feature values and modified dynamics. These evaluations are based on differences in the Markovianity of LSRs that either occur naturally or are obtained via a novel auxiliary loss that pushes a DQN to learn the CMSR.

2 Related Work

Exploring the Learning of Deep RL Agents. Our main goal is to contribute to a better understanding of the learning process of deep RL agents. To this end, we propose using measures based on bisimulation metrics that quantitatively denote how Markov an LSR is. Other research has used saliency maps [13] or t-SNE plots [22][25], the latter of which we also use as supporting evidence. These approaches result in figures that are easy to understand, but they do not produce quantitative measures to effectively summarize the characteristics of an LSR. Instead, to compare state representations, one has to look at multiple images and deduce based on domain knowledge what an agent has learned. An alternative is to plot the test performance [16] or state-action values for certain states [22] during training. Yet, in contrast to our approach, these approaches do not say anything about whether an agent has actually learned or simply memorized [14], the latter of which may hinder generalization. Although offering some improvement, this also holds for measuring out-of-distribution generalization [4][26]. The reason is that such out-of-distribution generalization may be good even if the agent has largely memorized. Lastly, to the best of our knowledge, no prior work has analyzed the learning process by computing how similar to the CMSR an LSR is.

Representation Learning Based on Bisimulation Metrics. To investigate the properties of LSRs that are more similar to the CMSR, we design

an auxiliary loss based on bisimulation metrics. Related work in this regard is presented by [25], who also propose learning LSRs based on bisimulation metrics. Yet, while [25] create an LSR in which distances between states correspond to how behaviorally different they are *under a varying policy*, we take *all actions* into consideration. Thus, an LSR learned by means of the approach of [25] potentially makes fewer distinctions than are needed to predict the reward and next state for all actions. Such an LSR hence generalizes to only a subset of the changes made to the dynamics that still allow for generalization based on the LSR that we propose to learn. In a similar vein to [25], [1] also base their approach on π-bisimulation metrics. Another related work is the one by [11]. Yet, whereas the Euclidean distances in our proposed LSR are *proportional* to the distances assigned by a bisimulation metric, the Euclidean distances between states in the LSR learned by means of the auxiliary loss of [11] provide an *upper bound* to bisimulation metric-based distances. Lastly, [23] employ the more general notion of *MDP homomorphism metrics* for representation learning. MDP homomorphism metrics differ from bisimulation metrics in that actions are also abstracted.

Representation Learning Based on Other Notions. The auxiliary loss we design introduces a bias to the learning. Several other approaches to bias the representation learning of deep RL agents have been proposed. For example, [17] and [8] put forward auxiliary losses based on predicting the next reward or the discount factor. Such methods tend to be successful in practice, but do not have strong theoretical foundations. Other work such as [19] is based on forming a model of the environment as auxiliary task. Yet, this tends to not work well for high-dimensional observations with large amounts of irrelevant information. Furthermore, rather than biasing the learning of deep neural networks by means of auxiliary losses, other work has proposed different models to learn more useful representations such as by incorporating ideas from symbolic reasoning [10]. For instance, [24] constrain neural networks to capture typical characteristics of relational reasoning. Another approach to learning more useful representations is to specifically focus on factors that may hurt generalization. [16], for example, improve generalization by reducing the non-stationarity an agent encounters during training. Moreover, [15] adapt to RL several regularization techniques from the context of classification that are based on injecting noise during training.

3 Background

Markov Decision Process. An infinite-horizon Markov Decision Process (MDP) is a tuple $\langle S, A, P, R, \gamma \rangle$ where S and A describe the space of Markov states and possible actions, respectively, $P : S \times A \to \Pi(S)$ is the transition function such that $P(s'|s, a) \in [0, 1]$ is the probability of arriving in state s' after taking action a in state s, $R : S \times A \to \mathbb{R}$ is the reward function such that $R(s, a)$ is the instant reward for taking action a in state s, and $0 \leq \gamma \leq 1$ is a discount factor. The goal of an agent in an MDP is to learn an optimal policy $\pi^* : S \to \Pi(A)$ that maximizes the expected cumulative (discounted) reward $\mathbb{E}\big[\sum_t^\infty \gamma^t r_t\big]$ for acting in the given environment. The Q-value function $Q^\pi : S \times A \to \mathbb{R}$ describes

the expected cumulative reward for taking action a in state s and executing π thereafter. The expected cumulative reward for taking an action a in a state s and following an optimal policy afterwards is given by $Q^*(s,a)$, where $Q^* = \max_\pi Q^\pi$.

Bisimulation Metrics. Bisimulation metrics [6] are based on the notion of *stochastic bisimulation* [12], which considers states as equivalent if and only if they have the same expected reward and the same transition distribution over all other abstract states for all actions. Such states that are equivalent under the notion of stochastic bisimulation are called *bisimilar*. Bisimulation metrics can be regarded as a quantitative version of stochastic bisimulation in that they assign a distance of zero only to bisimilar states, and that if the parameters of two bisimilar states are altered on a small scale, the metric distance between the two states will stay small. Thus, bisimulation metrics can be seen as a measure of behavioral similarity [7]. Theorem 4.5 in [6] defines one bisimulation metric d_{fix} that considers states as equivalent *if and only if* they are bisimilar. Given $F : M \to M$, where M is the set of all semimetrics on S that assign distances of at most 1, this d_{fix} is defined as the least fixed point of the following equation:

$$F(d)(s,s') = \max_{a \in A} \Big(c_R |R(s,a) - R(s',a)| + c_T T_K(d)\big(P(s,a), P(s',a)\big)\Big). \quad (1)$$

c_R and c_T are two positive one-bounded constants and $T_K(d)$ is the Kantorovich distance. It is d_{fix} that Euclidean distances in the CMSR are proportional to.

4 Markovianity of LSRs During Learning

Here we analyze the LSRs deep RL agents naturally form of their environments and how they compare to what such agents should ideally learn.

4.1 Methodology

Measuring Characteristics of LSRs. We propose using Pearson correlation coefficients[1] to gain insights into the learning process. These correlation coefficients are based on (components of) bisimulation metrics one the one hand, and the Euclidean distances between the activations states are mapped to in a network layer on the other hand. Let z_i, z_j be the activations s_i, s_j are mapped to in a network layer, $d_E(z_i, z_j)$ the Euclidean distance of z_i and z_j, $d_B(s_i, s_j)$ the distance of s_i and s_j for some bisimulation-based measure, and $\overline{d_E}$ and $\overline{d_B}$ averages. Then the Pearson correlation coefficient r_{d_B} is:

$$r_{d_B} = \frac{\sum_{i=0}^{|S|-2} \sum_{j=i+1}^{|S|-1} (d_E(z_i,z_j) - \overline{d_E})(d_B(s_i,s_j) - \overline{d_B})}{\sqrt{\sum_{i=0}^{|S|-2} \sum_{j=i+1}^{|S|-1} (d_E(z_i,z_j) - \overline{d_E})^2} \sqrt{\sum_{i=0}^{|S|-2} \sum_{j=i+1}^{|S|-1} (d_B(s_i,s_j) - \overline{d_B})^2}}. \quad (2)$$

Using measures based on or inspired by bisimulation metrics for d_B leads to the Pearson correlation coefficients that are defined in Table 1. These correlation

[1] The Pearson correlation coefficient measures the linear correlation of two variables.

Table 1. Correlation coefficients (CCs) based on Equation 2 and their interpretations. $r_{d_{fix}}$ and r_{Rew} are based on (components of) bisimulation metrics, and r_{Q^*} replaces the immediate reward in r_{Rew} by Q^*.

CC	d_B	Interpretation
$r_{d_{fix}}$	$d_{fix}(s_i, s_j)$	Similarity of representation to CMSR.
r_{Rew}	$\max_{a \in A} \lvert R(s_i, a) - R(s_j, a) \rvert$	Degree of clustering based on rewards.
r_{Q^*}	$\max_{a \in A} \lvert Q^*(s_i, a) - Q^*(s_j, a) \rvert$	Similarity to Q^*-irrelevance abstraction.

coefficients allow us to analyze the degrees to which the CMSR is learned, states are grouped based on instant rewards, and states are clustered based on Q-values in an LSR. Moreover, we can formally define the CMSR based on $r_{d_{fix}}$, which is obtained by letting d_B in Equation 2 be the bisimulation metric d_{fix}[2].

Definition 1 (Coarsest Markov State Representation (CMSR)). *The CMSR is a representation for which the following holds:*

$$r_{d_{fix}} = 1. \tag{3}$$

Theoretical Properties of the CMSR. We suppose that a deep RL agent should ideally learn the CMSR. This is due to several desirable theoretical properties of this representation. These theoretical properties arise because 1) the CMSR makes the lowest number of distinctions that still enables the prediction of the reward and next state [12], and 2) Euclidean distances between states in the CMSR are proportional to how behaviorally different states are. This leads to the following advantageous characteristics of the CMSR:

- *Feasibility of Learning π^*.* If an agent can predict the next reward and state for each action, an LSR is said to be *Markov* and the agent may find an optimal policy based on (histories of) observations[3] [21]. If, however, the reward and next state cannot be predicted based on the LSR, the agent in the most general case cannot learn an optimal policy.
- *Indifference to Irrelevant Features.* The CMSR does not distinguish observations that refer to the same state in the abstract MDP. That is, the CMSR treats as equivalent two observations that differ only in features that are irrelevant for predicting next states and rewards. This is especially important for domains with high-dimensional observations such as images.
- *Generalization to Modified Dynamics.* If a subset of the features required for predicting the reward and next internal state for an original domain is sufficient for predicting the reward and next internal state after modifying the dynamics, the distinctions the CMSR makes for the original domain

[2] Computed via the MCFZIB solver [9].

[3] While *representing* an optimal policy may require solely a coarser abstraction of the state space, such a representation may not suffice for *learning* an optimal policy [21].

suffice to learn the Q-values of such a modified domain. Moreover, since the Euclidean distance between two states in the CMSR varies smoothly as their parameters are changed, the CMSR is likely to still be useful if small such changes are made. This is important, as dynamics are commonly estimated and domain shifts may arise in problems such as robotics [15].

Q^*-irrelevance Abstraction. We suppose that LSRs should ideally be similar to the CMSR. Yet, the output layer of a DQN is pushed to represent Q-values, which may also cause LSRs to do so. We call an LSR in which the Euclidean distances between activations are proportional to the Euclidean distances between the corresponding Q-values a *Q^*-irrelevance abstraction*. This definition is based on generalizing the levels of state abstraction by [18] to the Euclidean space in which the activations in network layers fall. As non-bisimilar states may have the same Q-values, such an LSR may make fewer distinctions than the CMSR and hence no longer preserve the one-step model. Thus, a Q^*-irrelevance abstraction may not have the theoretical properties of the CMSR. We measure the extent to which a Q^*-irrelevance abstraction is formed via the correlation coefficient r_{Q^*}.

Domain. Our results presented here are based on a modified version of the fully observable Gridworld 3x3 domain [5], but supporting results from Gridworld 5x5, FrozenLake 8x8 from OpenAI Gym and the partially observable Hallway domain are described in [2]. In Gridworld 3x3, the state is a combination of the agent's position on a 3x3 grid and its orientation. Apart from the ground state, the agent's observations in our domain version contain a superfluous feature f_S, which can take 5 possible values sampled uniformly at random. This creates 5 behaviorally identical or bisimilar states out of each ground state. The agent can choose from the deterministic actions $\{forward,\ rotate\}$. The reward is 1 for reaching the goal location in the center of the grid and 0 otherwise.

State Encoding. We one-hot encode the ground states, and use 3 different ways of encoding f_S (Table 2). The encodings vary in the degree to which bisimilar states are encoded similarly, as mirrored by the encoding-based value for $r_{d_{fix}}$ in Table 2. Thus, the encodings have different effects on the initial LSR, which may impact the final LSR and its similarity to the CMSR.

4.2 Analysis of the Learning Process

In the following, we now use our proposed correlation coefficients and t-SNE [20] plots to shed light on the natural learning process of deep RL agents. Fig. 1 shows that the learning process consists of three overlapping learning phases:

1) States are grouped based on multi-step rewards. Since the target network provides the estimates of the Q-values of next states during training, it is not surprising that the activations of states with the same $n+1$-step rewards tend to be grouped together, where n is the number of times the target network has been updated. Fig. 1-1 shows the hidden activation patterns right after the DQN has been initialized[4]. At this point, any clustering is incidental in that it

[4] Since the encoding of f_S is lower-dimensional than the one of the ground state, the t-SNE plot shows one cluster for each value for f_S rather than for each ground state.

Table 2. State encodings and their definition of the superfluous feature f_S. We also show the value for $r_{d_{fix}}$ based on the encoded states.

Encoding	f_S	$r_{d_{fix}}$
Norm (N)	$f_S \in \{0, 0.25, 0.5, 0.75, 1\}$	0.251
One-hot (OH)	$f_S \in \{[1,0,0,0,0], [0,1,0,0,0], [0,0,1,0,0], [0,0,0,1,0], [0,0,0,0,1]\}$	0.087
Original (O)	$f_S \in \{0, 1, 2, 3, 4\}$	0.015

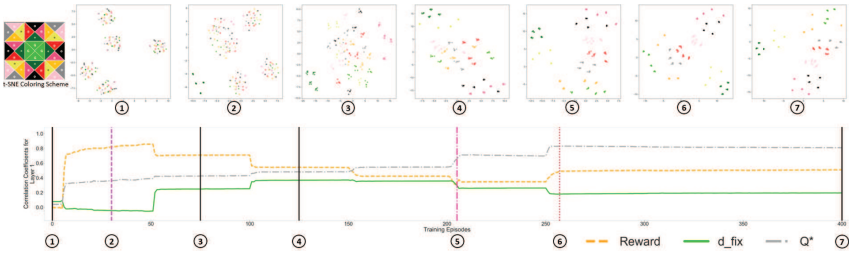
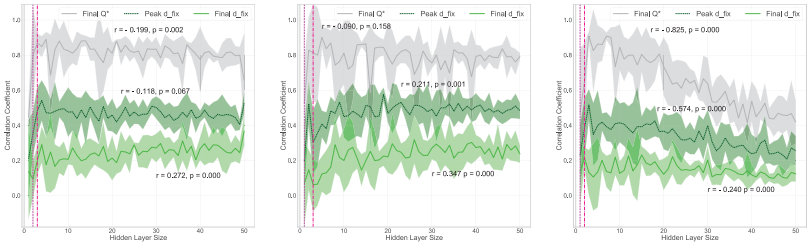


Fig. 1. r_{Rew}, $r_{d_{fix}}$, r_{Q^*} and t-SNE plots of the activations observations are mapped to during training for the LSR of a 2-layer DQN for the OH-encoding. The hidden layer size is 50 and the target network is updated every 50 episodes. All observations differing solely in f_S are drawn in the same color in the t-SNE plots and the coloring scheme for the ground states is shown on the left. Bisimilar ground states are shown in the same color. The vertical lines mark the episodes for which we show t-SNE plots. The 3 non-black lines thereby indicate 1) the first time the agent reaches the goal in each of 100 test episodes, 2) the first time the agent has learned π^* and 3) convergence to π^*.

depends on the state encoding[5] and network initialization. In Fig. 1-2, we see that the DQN has formed a separate cluster for those states that have an immediate reward of 1 (dark green). The target network has not yet been updated, so all other states, which have an immediate reward of 0, should not yet fall into separate clusters. Also note that the yellow curve (r_{Rew}) is now at its maximum. This is expected, because r_{Rew} measures the degree of similarity between the current LSR and a representation that clusters states together if and only if they have the same immediate reward. After the target network has been updated once, a new separate cluster is formed for those states that have a non-zero two-step reward (Fig. 1-3, dark pink). This is accompanied by a drop in r_{Rew}, as states are now no longer distinguished solely based on their immediate rewards.

2) The LSR becomes more similar to the CMSR. This pattern is mirrored by the increase in the green curve ($r_{d_{fix}}$) at the beginning of training. However, the exact CMSR is not learned, as $r_{d_{fix}}$ is never equal to 1.

[5] The impact of the state encoding is discussed in the next section.



(a) OH-encoding. (b) N-encoding. (c) O-encoding.

Fig. 2. Mean peak and final $r_{d_{fix}}$ and final r_{Q^*} with 95%-confidence intervals for the LSRs of 2-layer DQNs for different state encodings and hidden layer sizes. The vertical lines indicate the smallest hidden layer sizes for which 1) the agent always arrives at the goal in 100 test episodes and 2) the DQN converges to π^* at least 1 out of 5 times. Each curve is labeled with the Pearson correlation of the respective correlation coefficient and the hidden layer sizes that are large enough for the DQN to learn π^* at least 1 out of 5 times.

3) States are increasingly clustered based on Q-values, as visualized by the step-wise increase in the gray curve (r_{Q^*}), after an initial plateau. Ultimately, r_{Q^*} reaches a value near 1 when the DQN converges to π^*. At the same time, $r_{d_{fix}}$ decreases for this domain as the inter-cluster distances become more and more different from those of the CMSR[6]. This is shown near episode 200, where $r_{d_{fix}}$ begins to decrease when r_{Q^*} strongly increases again. The final LSR is thus less similar to the CSMR for this domain than during the second phase.

This analysis suggests that while a DQN does naturally form the CMSR to some degree, the exact CMSR is not learned. Instead, states are at some point clustered based on Q-values rather than bisimilarity, which may cause the LSR to become less similar to the CMSR. Given the useful theoretical properties of the CMSR, the latter might have negative consequences for a network's generalization ability. We examine this impact on the generalization performance in Section 5.

4.3 Factors Impacting the Learning Process

When training a DQN, one has to make a plethora of choices such as for the network architecture and the state encoding. Commonly, we make such choices primarily based on average returns. However, the decisions we make might also impact the LSRs that are formed. We therefore analyzed how different factors impact the learning process described above. We find that the extent to which LSRs become similar to the CMSR *during* and still are *at the end of* training depends on the network capacity and state encoding. This is discussed below.

Network Capacity. The dark green curve (*peak* $r_{d_{fix}}$) in Fig. 2(a) shows that the LSR becomes most similar to the CMSR *during* training for hidden layer sizes just to the right of the second vertical line. These hidden layer sizes

[6] The decrease in $r_{d_{fix}}$ is related to the network capacity, discussed in the next section.

are necessary for the DQN to be able to converge to π^*. For larger hidden layers, the LSR becomes progressively less similar to the CMSR during training. This is captured by the value of -0.118 for the Pearson correlation coefficient between *peak* $r_{d_{fix}}$ and sufficiently large hidden layer sizes (Fig. 2(a)). The reason for this pattern is that larger hidden layers make a network more flexible, and thus allow the network to converge to the true Q-values even if less similar to the CMSR is learned in the hidden layer during training. Such large networks hence learn the Q-values without grouping behaviorally equivalent observations together.

The LSR *at the end of* training, however, is more similar to the CMSR for larger hidden layers. This is indicated by the bright green curve (*final* $r_{d_{fix}}$) and the corresponding Pearson correlation coefficient of 0.272 with respect to sufficiently large hidden layer sizes in Fig. 2(a). The reason is that DQNs with smaller hidden layers eventually need to largely cluster states based on Q-values in their hidden layers due to their lower flexibility. Otherwise, their output layers cannot represent the true Q-values. Thus, while DQNs with smaller hidden layers *initially* learn closer to the CMSR, their LSR is *ultimately* further abstracted towards a Q^*-irrelevance abstraction. The latter is supported by the observation that the final values for r_{Q^*} (gray curve) are higher for smaller hidden layers, which is captured by the Pearson correlation coefficient of -0.199 between the final values for r_{Q^*} and sufficiently large hidden layer sizes in Fig. 2(a).

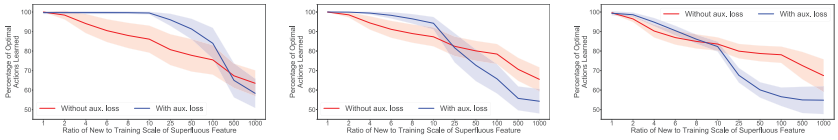
State Encoding. The CMSR is formed to a lesser degree during learning if it is more difficult and less necessary to be learned. Based on the three dark green curves (*peak* $r_{d_{fix}}$) in Fig. 2, we can see that the LSRs become most similar to the CMSR *during* learning for large hidden layers for the N-encoding and least similar for the O-encoding. The reason for this pattern is that bisimilar states have the most similar encodings in the N- and the least similar ones in the O-encoding (see $r_{d_{fix}}$ in Table 2). Hence, for the latter encoding it is most difficult to group bisimilar states together in the LSR. Thus, as the network capacity increases and it therefore becomes less necessary to learn the CMSR, the CMSR is progressively less formed *during* learning for state encodings that make it more difficult to do so. This also impacts the LSRs present *at the end of* training, as mirrored by the three bright green curves (*final* $r_{d_{fix}}$) in Fig. 2.

Given that both the network capacity and the state encoding impact the degree to which the CMSR is formed, it is important to make a considerate choice of the network architecture and state encoding if learning the CMSR is desired.

5 Practical Usefulness of the CMSR

While the *theoretical* advantages are apparent, we will now investigate whether striving to learn the CMSR is also useful *in practice*. To obtain LSRs that are very similar to the CMSR, we introduce a bisimulation-based auxiliary loss that pushes a network to form the CMSR as LSR.

Bisimulation-based Auxiliary Loss. We calculate $d_{fix}(s_i, s_j)$ for all $s_i, s_j \in S, i \neq j$. During training, we then compute an auxiliary loss based on the premise that we want the Euclidean distances between the activations of states to be



(a) Hidden layer size of 10. (b) Hidden layer size of 20. (c) Hidden layer size of 65.

Fig. 3. Average percentage of optimal actions learned by 2-layer DQNs with different hidden layer sizes for the O-encoding, trained with and without the auxiliary loss. Optimal actions returned by the DQN for each non-terminal ground state are measured for $1,000$ values for f_S sampled uniformly at random from an interval that is i times as large as the one used during training. The value i is shown on the x-axis. 95%-confidence intervals based on 10 repetitions are shown.

proportional to their distances assigned by the bisimulation metric d_{fix}. In other words, we want that $d_E(z_i, z_j)$ is equal to $d^*_E(z_i, z_j) = d^{max}_E \times d_{fix}(s_i, s_j)$, where d^{max}_E is a hyperparameter for how far apart the activations of non-bisimilar states should be. We thus compute a target activation z^*_i for all $s_i \in S$:

$$z^*_i = z_i + \frac{1}{2} \times \sum_{j \neq i} \big(d^*_E(z_i, z_j) - d_E(z_i, z_j)\big) \frac{z_i - z_j}{||z_i - z_j||}, \tag{4}$$

where $||z_i - z_j||$ is the length of the vector $z_i - z_j$. Note that the unit-length vector $\frac{z_i - z_j}{||z_i - z_j||}$ between z_i and z_j is multiplied by half of the amount by which $d_E(z_i, z_j)$ should change. The idea behind this is that if z_i and z_j should be pulled apart or closer together, both are moved by half the total amount in the respective direction. Based on this, we minimize the MSE between z_i and z^*_i for all $s_i \in S$. We found this approach to work better than directly minimizing the MSE between d_E and d^*_E.

5.1 Generalization to New Irrelevant Feature Values

The first type of generalization we consider is the one to new values of irrelevant[7] features. We train 2-layer DQNs for Gridworld 3x3 with and without the auxiliary loss. At test time, we sample $1,000$ values for the superfluous feature f_S randomly from an interval that is i times as large as the one used during training, where $i \in \{1, 2, 4, 6, 8, 10, 25, 50, 100, 500, 1000\}$. For each sampled value for f_S, we compute the optimal action returned by the trained DQN and compare it to π^*.

Fig. 3 reveals that if the auxiliary loss is used, the generalization to new values for f_S tends to be better than if no auxiliary loss is used. This makes sense, as using the auxiliary loss causes the LSR to ignore f_S to a larger extent (Fig. 4). However, Fig. 3 shows that there are two exceptions to the observation that introducing the auxiliary loss improves upon the generalization. These are 1) the generalization to very large intervals and 2) DQNs with large hidden layers:

[7] Irrelevant features are not required for predicting the next reward and internal state.

Very Large Intervals. Generalization to values for f_S sampled from very large intervals tends to be better if LSRs that are not entirely indifferent to f_S are closer to a Q^*-irrelevance abstraction. Notice that while introducing the auxiliary loss leads to improved generalization for small and moderately sized intervals, it deteriorates the generalization for very large intervals. This can be explained by the fact that even though the LSRs learn to ignore f_S to a larger extent when we apply the auxiliary loss, they do not do so entirely. At the same time, the Euclidean distances between the activations of states with *different* optimal actions are on average more similar to those between the activations of states with the *same* optimal actions in the CMSR than in a Q^*-irrelevance abstraction for this domain[8]. Hence, that for very different values for f_S an observation is mapped to a latent representation that causes the DQN to return a sub-optimal action is less likely if the DQN learns closer to a Q^*-irrelevance abstraction. Yet, this only holds because the DQNs do not learn the *precise* CMSR.

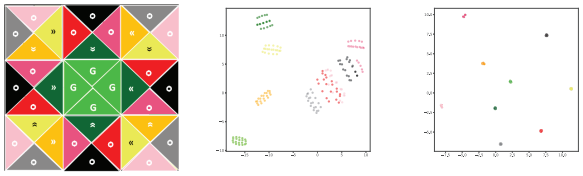
DQNs with Large Hidden Layers. One would expect DQNs with varying hidden layer sizes to generalize similarly well if the LSRs are very close to the CMSR. Yet, using the auxiliary loss tends to lead to worse generalization to large intervals for large hidden layers (Fig. 3(c)) than for smaller ones (Fig. 3(a) and 3(b)). The reason is that the LSRs of DQNs with large hidden layers become less similar to the CMSR again towards the end of training for our settings for the auxiliary loss. More precisely, we decay the weight of the auxiliary loss during training and continue to train even after the weight has become 0. This continued training after the auxiliary loss is no longer applied causes the LSRs of larger DQNs to increasingly distinguish observations based on f_S again and hence to generalize worse to large intervals. Thus, for large DQNs to have an LSR that is very similar to the CMSR by the end of training, it is not sufficient to apply the auxiliary loss only until close to the CMSR is formed. Instead, the auxiliary loss needs to be applied longer, if not during the entire training.

Worse generalization hence only arises when the exact CMSR is not formed. Moreover, even then it only occurs when either extremely different values for f_S are sampled or the auxiliary loss is stopped too soon for very large DQNs.

5.2 Generalization to Modified Dynamics

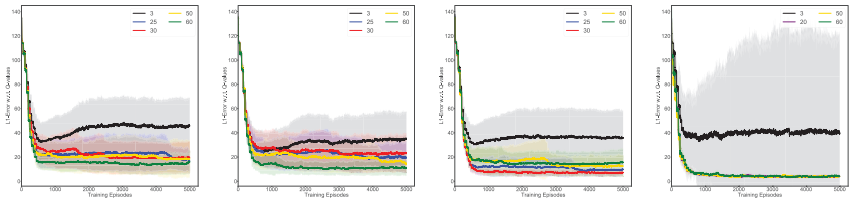
Here we now explore a second type of generalization, namely the one to modifications of the dynamics that do not make formerly irrelevant features relevant. 2-layer DQNs with hidden layer sizes between 3 and 60 are trained each 10 times on Gridworld 3x3, and subsequently retrained after modifying the transition function. We reset the output-layer representation before and hold the LSR fixed during retraining. Based on Fig. 5, we find that the following three factors impact the generalization to the modified domain:

[8] Non-terminal ground states have mean Euclidean distances of 0.175 and 0.333 to other non-terminal ground states with the same and different optimal actions, respectively, in a Q^*-irrelevance abstraction for Gridworld 3x3. In the CMSR, however, the mean Euclidean distances to non-terminal ground states with the same and different optimal actions are 0.141 and 0.144, respectively, if $d_E^{max} = 1$.



(a) Color scheme. (b) $r_{d_{fix}} = 0.11$. (c) $r_{d_{fix}} = 0.98$.

Fig. 4. t-SNE plots and $r_{d_{fix}}$ of the LSRs at the end of training b) without and c) with the auxiliary loss for a 2-layer DQN with a hidden layer size of 10 for the O-encoding. Activation patterns of bisimilar observations have the same color.



(a) N - Without. (b) OH - Without. (c) O - Without. (d) O - With.

Fig. 5. Mean L_1-error with respect to the Q-values during retraining of 2-layer DQNs with varying hidden layer sizes on the modified domain for the Norm (N), One-Hot (OH) and Original (O) state encodings. The hidden-layer weights are initialized to those of DQNs trained on Gridworld 3x3 either *with* or *without* the auxiliary loss and are not updated during retraining. The output-layer weights are newly initialized before retraining. Values are based on 10 repetitions and 95%-confidence intervals are shown.

Similarity to Q^*-irrelevance Abstraction. The generalization is better when the LSR is less similar to a Q^*-irrelevance abstraction for the original domain. Recall that DQNs with larger hidden layers learn LSRs that are less similar to a Q^*-irrelevance abstraction (gray curves in Fig. 2). This explains why DQNs with larger hidden layers tend to generalize best for the Norm (N) and One-Hot (OH) encodings. Moreover, the created LSRs for large hidden layers are closer to a Q^*-irrelevance abstraction for the N- and OH- than for the Original (O)-encoding (gray curves in Fig. 2), which is why the former lead to higher L_1-errors on the modified domain.

Similarity to CMSR. Lower L_1-errors are achieved when the LSR is closer to the CMSR. Moderately sized hidden layers are more similar to the CMSR for the O-encoding than even larger hidden layers (bright green curve in Fig. 2(c)), which is why the former lead to better generalization. Note that this occurs despite the higher flexibility of larger networks. For the OH- and N-encodings, the largest tested hidden layer sizes do not yet cause the final LSR to be less similar to the CMSR (Fig. 2(a) and 2(b)). Thus, DQNs with moderately sized hidden layers do not outperform DQNs with larger ones for those two encodings.

Network Capacity. DQNs with larger hidden layers are less dependent on the LSR when it comes to learning the new Q-values due to their higher capacity. This adds to the fact that larger DQNs generalize best for the N- and OH-encodings. Note also that due to their lower flexibility, DQNs with very small hidden layer sizes need to learn an LSR that is more similar to a Q^*-irrelevance abstraction for the new domain to be able to learn the new Q-values. This is not possible if the LSR is fixed during retraining.

Thus, the naturally occurring differences in Markovianity between LSRs show that learning an LSR that is more similar to the CMSR tends to aid generalization, especially for moderately sized hidden layers. Furthermore, adding an auxiliary loss to the training that pushes a DQN to learn closer to the CMSR in its hidden layer leads to better generalization for all networks except those with very small hidden layers (Fig. 5(d)). The latter occurs because due to their lower flexibility, very small networks need to learn closer to a Q^*-irrelevance abstraction in their hidden layers to be able to learn the new Q-values in their output layers.

6 Conclusions

We analyzed the LSRs deep RL agents form of their environments to gain a better understanding of the learning process and the factors that impact it. Thereby, we suppose that due to its theoretical and especially generalization properties, an agent should ideally learn the CMSR. In the CMSR, distances between states are proportional to how behaviorally different the states are. We find that while LSRs tend to become more similar to the CMSR at the start of training, states are ultimately clustered based on Q-values rather than behavioral similarity. This may cause the LSRs to become less similar to the CMSR again. Moreover, the *precise* CMSR is not learned in any of our experiments. Our standard network architectures and optimization algorithms thus do not lead to ideal LSRs. While our analysis in this paper is based on Gridworld 3x3, we obtained comparable results for the learning process on Gridworld 5x5, FrozenLake 8x8 from OpenAI Gym and the partially observable Hallway domain in [2].

Our analysis of the factors impacting the learning process further reveals that both the state encoding and the network capacity impact the degree to which the CMSR is formed *during* and is still present *at the end of* training. For large hidden layer sizes, for example, networks learn the CMSR to a much lesser extent during training. The reason is that due to their higher flexibility, such networks can learn the Q-values without grouping behaviorally equivalent observations together. Notably, the CMSR is even less learned by such large networks if it is also rather difficult to form the CMSR due to the state encoding. It is thus crucial to carefully choose both network architecture and state encoding if learning closer to the CMSR is desired. Future work should explore the generalization of these findings to environments with more complex observations. For such environments, our proposed correlation coefficients can be made more scalable by approximately computing the bisimulation metric based on the algorithm by [3].

Our claim that deep RL agents should ideally learn the CMSR is supported by our empirical findings. That is, we find that learning closer to the CMSR may improve upon generalization to new irrelevant feature values and modified dynamics. Our results thus show that learning good LSRs is crucial. Rather than selecting architectures and optimization algorithms primarily based on average returns, we should hence strive to make a more informed decision based on the LSRs that are formed. To this end, we need to also report the quality of the LSRs learned in our experiments via measures such as the ones we propose. Moreover, as our current architectures and algorithms do not form ideal LSRs, it is important that we as a community strive to develop scalable methods that address the discrepancies between what is and what should be learned. The auxiliary loss we designed provides a starting point, but has to be made more scalable to be useful in practice. For example, the expensive exact computation of the bisimulation metric could be replaced by an approximation that is incorporated into training in a vein similar to the approach by [3].

Acknowledgments. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 758824 —INFLUENCE).

References

1. Agarwal, R., Machado, M.C., Castro, P.S., Bellemare, M.G.: Contrastive behavioral similarity embeddings for generalization in reinforcement learning. In: International Conference on Learning Representations, ICLR (2021)
2. Albers, N.: Learning what to attend to: Using bisimulation metrics to explore and improve upon what a deep reinforcement learning agent learns. Master thesis (2020)
3. Castro, P.S.: Scalable methods for computing state similarity in deterministic Markov decision processes. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 10069–10076 (Apr 2020)
4. Cobbe, K., Klimov, O., Hesse, C., Kim, T., Schulman, J.: Quantifying generalization in reinforcement learning. In: Proceedings of the 36th International Conference on Machine Learning, ICML. Proceedings of Machine Learning Research, vol. 97, pp. 1282–1289. PMLR (2019)
5. Ferns, N., Castro, P.S., Precup, D., Panangaden, P.: Methods for computing state similarity in Markov decision processes. In: Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence. pp. 174–181. AUAI Press (2006)
6. Ferns, N., Panangaden, P., Precup, D.: Metrics for finite Markov decision processes. In: Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence. pp. 162–169. AUAI Press (2004)
7. Ferns, N., Precup, D.: Bisimulation metrics are optimal value functions. In: Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence. pp. 210–219. AUAI Press (2014)
8. François-Lavet, V., Bengio, Y., Precup, D., Pineau, J.: Combined reinforcement learning via abstract representations. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 3582–3589 (2019)

9. Frangioni, A., Manca, A.: A computational study of cost reoptimization for min-cost flow problems. INFORMS Journal on Computing **18**(1), 61–70 (2006). https://doi.org/10.1287/ijoc.1040.0081
10. Garcez, A.d., Besold, T.R., De Raedt, L., Földiak, P., Hitzler, P., Icard, T., Kühnberger, K.U., Lamb, L.C., Miikkulainen, R., Silver, D.L.: Neural-symbolic learning and reasoning: Contributions and challenges. In: 2015 AAAI Spring Symposium Series (2015)
11. Gelada, C., Kumar, S., Buckman, J., Nachum, O., Bellemare, M.G.: DeepMDP: Learning continuous latent space models for representation learning. In: Proceedings of the 36th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 97, pp. 2170–2179. PMLR (2019)
12. Givan, R., Dean, T., Greig, M.: Equivalence notions and model minimization in Markov decision processes. Artificial Intelligence **147**(1-2), 163–223 (2003). https://doi.org/10.1016/S0004-3702(02)00376-4
13. Greydanus, S., Koul, A., Dodge, J., Fern, A.: Visualizing and understanding Atari agents. In: Proceedings of the 35th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 80, pp. 1792–1801. PMLR (2018)
14. Hausknecht, M., Stone, P.: The impact of determinism on learning atari 2600 games. In: AAAI Workshop on Learning for General Competency in Video Games (2015)
15. Igl, M., Ciosek, K., Li, Y., Tschiatschek, S., Zhang, C., Devlin, S., Hofmann, K.: Generalization in reinforcement learning with selective noise injection and information bottleneck. In: Advances in Neural Information Processing Systems 32. pp. 13956–13968 (2019)
16. Igl, M., Farquhar, G., Luketina, J., Boehmer, W., Whiteson, S.: The impact of non-stationarity on generalisation in deep reinforcement learning. arXiv preprint arXiv:2006.05826 (2020)
17. Jaderberg, M., Mnih, V., Czarnecki, W.M., Schaul, T., Leibo, J.Z., Silver, D., Kavukcuoglu, K.: Reinforcement learning with unsupervised auxiliary tasks. In: 5th International Conference on Learning Representations, ICLR (2017)
18. Li, L., Walsh, T.J., Littman, M.L.: Towards a unified theory of state abstraction for MDPs. In: International Symposium on Artificial Intelligence and Mathematics, ISAIM (2006)
19. Li, X., Li, L., Gao, J., He, X., Chen, J., Deng, L., He, J.: Recurrent reinforcement learning: A hybrid approach. arXiv preprint arXiv:1509.03044 (2015)
20. Maaten, L.v.d., Hinton, G.: Visualizing data using t-SNE. Journal of machine learning research **9**(Nov), 2579–2605 (2008)
21. McCallum, A.K.: Reinforcement Learning with Selective Perception and Hidden State. Ph.D. thesis (1996)
22. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529 (2015)
23. van der Pol, E., Kipf, T., Oliehoek, F.A., Welling, M.: Plannable approximations to MDP homomorphisms: Equivariance under actions. In: AAMAS (2020)
24. Santoro, A., Raposo, D., Barrett, D.G., Malinowski, M., Pascanu, R., Battaglia, P., Lillicrap, T.: A simple neural network module for relational reasoning. In: Advances in Neural Information Processing Systems 30. pp. 4967–4976 (2017)
25. Zhang, A., McAllister, R., Calandra, R., Gal, Y., Levine, S.: Learning invariant representations for reinforcement learning without reconstruction. arXiv preprint arXiv:2006.10742 (2020)
26. Zhang, C., Vinyals, O., Munos, R., Bengio, S.: A study on overfitting in deep reinforcement learning. arXiv preprint arXiv:1804.06893 (2018)