An Automated Measure of MDP Similarity for Transfer in Reinforcement Learning

Haitham Bou Ammar
University of Pennsylvania

Eric Eaton
University of Pennsylvania

Matthew E. Taylor
Washington State University

Decebal Constantin Mocanu
Technical Univ. of Eindhoven

Kurt Driessens
University of Maastricht

Gerhard Weiss
University of Maastricht

Karl Tuyls
University of Liverpool

Abstract

Transfer learning can improve the reinforcement learning of a new task by allowing the agent to reuse knowledge acquired from other source tasks. Despite their success, transfer learning methods rely on having relevant source tasks; transfer from inappropriate tasks can inhibit performance on the new task. For fully autonomous transfer, it is critical to have a method for automatically choosing relevant source tasks, which requires a similarity measure between Markov Decision Processes (MDPs). This issue has received little attention, and is therefore still a largely open problem.

This paper presents a data-driven automated similarity measure for MDPs. This novel measure is a significant step toward autonomous reinforcement learning transfer, allowing agents to: (1) characterize when transfer will be useful and, (2) automatically select tasks to use for transfer. The proposed measure is based on the reconstruction error of a restricted Boltzmann machine that attempts to model the behavioral dynamics of the two MDPs being compared. Empirical results illustrate that this measure is correlated with the performance of transfer and therefore can be used to identify similar source tasks for transfer learning.

Introduction

Reinforcement learning (RL) methods often learn new problems from scratch. In complex domains, this process of tabula rasa learning can be prohibitively expensive, requiring extensive interaction with the environment. Transfer learning provides a possible solution to this problem by enabling reinforcement learning agents to reuse knowledge from previously learned source tasks when learning a new target task. In situations where the source tasks are chosen incorrectly, inappropriate source knowledge can interfere with learning through the phenomenon of *negative transfer*.

Early methods for RL transfer (Taylor, Stone, and Liu 2007; Torrey et al. 2006) were semi-automated, requiring a human user to define the relevant source tasks and relationships between tasks for the algorithm. To develop fully autonomous transfer learning, the agent must be capable of: *a*) selecting the relevant source task, *b*) learning the relationships (e.g., inter-task mapping) between the source and target tasks, and, *c*) effectively and efficiently transferring knowledge between tasks (Taylor and Stone 2009). Recent work on autonomous transfer (Taylor, Jong, and Stone 2008; Ammar et al. 2012; 2013) has focused on the latter two items, leaving the automated selection of relevant source tasks (item *a*) as a largely unsolved problem, yet a critical one for successful transfer learning.

Transfer learning agents must be able to automatically identify source tasks that are most most similar to and helpful for learning a target task. In RL, where tasks are represented by Markov decision processes (MDPs), agents could use an MDP similarity measure to assess the relatedness of each potential source task to the given target. This measure should *a*) quantify the similarity between a source and a target task, *b*) be capable of predicting the probability of success after transfer, and *c*) be estimated autonomously from sampled data. So far, no mathematical framework has been developed that can achieve these goals successfully.

This paper proposes a novel similarity measure for MDPs within a domain and shows that this measure can be used to predict the performance of transfer learning. Moreover, this approach does not require a model of the MDP, but can estimate this measure from samples gathered through an agent's interaction with the environment. We demonstrate that the proposed measure is capable of capturing and clustering dynamical similarities between MDPs with multiple differences, including different reward functions and transition probabilities. Our experiments also illustrate that the initial performance improvement on a target task from transfer is correlated with the proposed measure — as the measured similarity between MDPs increase, the initial performance improvement on the target task similarly increases.

Background

Reinforcement Learning (RL) is an algorithmic technique for solving sequential decision making problems (Buşoniu et al. 2010). An RL problem is typically formalized as a Markov decision process (MDP) $\langle S, A, P, R, \gamma \rangle$, where S is the (potentially infinite) set of states, A is the set of possible actions that the agent may execute, $P : S \times A \times S \rightarrow [0, 1]$ is a state transition probability function that describes the task dynamics, $R : S \times A \times S \rightarrow \mathbb{R}$ is the reward function that measures the performance of the agent, and $\gamma \in [0, 1)$ is a discount factor. A policy $\pi : S \times A \rightarrow [0, 1]$ is defined as a probability distribution over state-action pairs, where $\pi(s, a)$ represent the probability of selecting action a in state s. The goal of an RL agent is to improve its policy, aiming to reach the optimal policy $\pi^\star$ that maximizes the expected cumulative future rewards $\mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$.

Although successful, RL techniques are hampered by the complexity required to attain successful behaviors in challenging domains. Transfer learning is a collection of techniques that have been introduced to remedy this problem.

Transfer Learning (TL) aims to improve learning times and/or behavior of an agent by re-using knowledge from one or more *source tasks* to improve learning on a new *target task*. Let $\mathcal{T}_1 = \langle S_1, A_1, P_1, R_1, \gamma_1 \rangle$ represent a source task and let $\mathcal{T}_2 = \langle S_2, A_2, P_2, R_2, \gamma_2 \rangle$ represent a target task. Each of these tasks is given by an MDP that might differ from the other in all five of its constituents. The differences between two tasks can be divided into: *a*) domain, *b*) reward function, and *c*) dynamical differences. A domain is defined as the state and action spaces of an MDP. When the domains are different from one another, an inter-task mapping (Taylor, Stone, and Liu 2007) is needed to facilitate transfer between the tasks. In this work, the focus is on autonomous transfer within the same domain, i.e., we consider only differences in the reward and dynamics of the tasks.

Although TL has been shown to be successful in many RL domains, the performance of any TL algorithm necessarily depends on the choice of the source and target tasks. When these tasks are similar, transfer is expected to aid the agent's behavior in $\mathcal{T}_2$. On the other hand, if the source and target tasks are too dissimilar, the benefit of transfer is reduced, and transfer may even decrease the target agent's performance. Unfortunately, the choice of these tasks is mostly done in an ad-hoc fashion, often based on the designer's intuition (Taylor and Stone 2009). A key problem is that similarity between MDPs is not always clear from an intuitive inspection. Specifically, even when two tasks do not appear similar at first sight, inter-task mappings can potentially be learned and enable positive transfer. For instance, Bou Ammar et al. (2012) have shown positive transfer between the Mountain Car and Cart-Pole tasks, showing both the potential benefits of transfer between very different tasks, and the potential difficulties in determining which tasks can and cannot be used effectively for transfer.

An Automated MDP Similarity Measure

A similarity measure between MDPs will help practitioners in the field of TL for RL in formalizing and evaluating newly proposed algorithms. First, it will quantify the ad hoc choices made when selecting source and target tasks. Second, such a measure is considered to be the first step in attaining a well-founded performance criterion for TL in RL tasks. For instance, bounds on transfer algorithms can now be attained as a function of this measure. To our knowledge, only a few similarity measures between MDPs have been proposed. One notable example are bisimulation metrics (Ferns et al. 2006; Ferns, Panangaden, and Precup 2011), which quantify differences between MDPs. However, these techniques require the manual definition of a measure between tasks, and either operate in discrete state space MDPs or require high computational effort (e.g., infinite dimensional linear programming). In contrast, our proposed approach is: *a*) data-driven in the sense that the metric is acquired from MDP transitions, *b*) operational in continuous state spaces, and *c*) is computationally tractable.

MDP Similarity Measure

This section introduces the method for computing a similarity measure between MDPs. We first discuss Restricted Boltzmann Machines, as they form the core of our approach. We then introduce *RBDist*, a similarity measure that uses RBMs to relate same-domain MDPs. The main idea is that we can use an RBM to describe different MDPs in a common representation, providing a similarity measure. We first use an RBM to model data collected in the source task, yielding a set of relevant and informative features that characterize the source MDP. These features can then be tested on the target task to assess MDP similarity.

Restricted Boltzmann Machines

Restricted Boltzmann machines (RBMs) are energy-based models for unsupervised learning. They use a generative model of the distribution of training data for prediction. These models employ stochastic nodes and layers, making them less vulnerable to local minima (Salakhutdinov, Mnih, and Hinton 2007). Further, due to multiple layers and the neural configurations, RBMs posses excellent generalization abilities. For example, they have been shown to successfully discover informative hidden features in unlabeled data (Bengio 2009). RBMs are stochastic neural networks with bidirectional connections between the visible and hidden layers (Figure 1). This allows RBMs to posses the capability of regenerating visible layer values, given a hidden layer configuration. The visible layer represents input data, while the hidden layer discovers more informative spaces to describe input instances. Therefore, RBMs could also be seen as density estimators, where the hidden layer approximates a (factored) density representation of the input units.

Formally, an RBM consists of two binary layers: one visible and one hidden. The visible layer models the data while the hidden layer enlarges the class of distributions that can be represented to an arbitrary complexity (Taylor, Hinton, and Roweis 2011). This paper follows standard notation where i represents the indices of the visible layer, j those of the hidden layer, and $w_{i,j}$ denotes the weight connection between the i^{th} visible and j^{th} hidden unit. We further use v_i and h_j to denote the state of the i^{th} visible and j^{th} hidden unit, respectively. The state energy function is given by:

$$E(v, h) = -\sum_{i,j} v_i h_j w_{ij} - \sum_i v_i b_i - \sum_j h_j b_j \ , \quad (1)$$

where b_i and b_j represent the biases of the visible and invisible nodes respectively. The first term, $\sum_{i,j} v_i h_j w_{ij}$, represents the energy between the hidden and visible units with their associated weights. The second term, $\sum_i v_i b_i$, represents the energy in the visible layer, while the third term represents the energy in the hidden layers. The joint probability of a state of the hidden and visible layers is:

$$P(v,h) \propto \exp\left(-E(v,h)\right). \quad (2)$$

To determine the probability of a data point represented by a state v, the marginal probability is used, summing out the state of the hidden layer:

$$p(v) \propto \sum_h \exp\left(-\sum_{i,j} v_i h_j w_{ij} - \sum_i v_i b_i - \sum_j h_j b_j\right). \quad (3)$$

The above equations can be used for any given input to calculate the probability of either the visible or the hidden configuration, which can then be used to perform inference.

Contrastive Divergence Learning

Learning in RBMs means determining the weight connections and biases such that the likelihood is maximized. To maximize the likelihood of the model, the gradient of the log-likelihood with respect to the weights must be calculated. Unfortunately, computing these gradients is intractable in RBMs. Hinton (2002) proposed an approximative learning method called *contrastive divergence* (CD). In maximum likelihood, the learning phase actually minimizes the Kullback-Leiber (KL) distance between the input data distribution and the approximated model. In CD, however, learning follows the gradient of

$$CD_n = D_{KL}(p_0(\boldsymbol{x})||p_\infty(\boldsymbol{x})) - D_{KL}(p_n(\boldsymbol{x})||p_\infty(\boldsymbol{x})) \ ,$$

where $p_n(\cdot)$ is the distribution of a Markov chain starting from $n = 0$ and running for a small number of n steps. Since the visible units are conditionally independent given the hidden units and vice versa, a step of Gibbs sampling can be carried in two half-steps: (1-forward) update all the hidden units, and (2-backward) update all the visible units. Let $\boldsymbol{v} = [v_1, \ldots, v_{n_v}]$ and $\boldsymbol{h} = [h_1, \ldots, h_{n_h}]$, where v_i and h_j represents the values of the i^{th} visible and j^{th} hidden neuron respectively. Also, let $\boldsymbol{W} \in \mathbb{R}^{n_h \times n_v}$ represent the matrix of all weights. Then, CD_n updates the weights by:

$$w_{ij}^{\tau+1} = w_{ij}^{\tau} + \alpha\left(\left\langle\langle h_j v_i\rangle_{p(\boldsymbol{h}|\boldsymbol{v};\boldsymbol{W})}\right\rangle_0 - \langle h_j v_i\rangle_n\right) \ ,$$

where τ is the iteration, α is the learning rate, $\left\langle\langle h_j v_i\rangle_{p(\boldsymbol{h}|\boldsymbol{v};\boldsymbol{W})}\right\rangle_0 = \frac{1}{N}\sum_{n=1}^{N} v_i^{(n)} P(h_i^{(n)} = 1|\boldsymbol{h};\boldsymbol{W})$, and $\langle h_j v_i\rangle_n = \frac{1}{N}\sum_{n=1}^{N} v_i^{(n)G_l} P(h_j^{(n)G_l}|\boldsymbol{h}^{(n)G_l};\boldsymbol{W})$. N is the total number of input instances and G_l indicates that the states are obtained after l iterations of Gibbs sampling from the Markov chain starting at $p_0(\cdot)$.

Restricted Boltzmann Machine Distance Measure

In this section, we propose *RBDist* as a distance measure between MDPs; Algorithm 1 provides a complete description of the approach. To measure the similarity of two MDPs, we first sample them uniformly to generate data sets $\mathcal{D}_1 = \{\langle s_1^{(j)}, a_1^{(j)}, s_1'^{(j)}\rangle\}_{j=1}^m$ and $\mathcal{D}_2 = \{\langle s_2^{(k)}, a_2^{(k)}, s_2'^{(k)}\rangle\}_{k=1}^n$, where $s_1'^{(j)} \sim P_1(s_1^{(j)}, a_1^{(j)})$, $s_2'^{(k)} \sim P_2(s_2^{(k)}, a_2^{(k)})$, and m and n represent the number of samples from $\mathcal{T}_1$ and $\mathcal{T}_2$, respectively. We then use the source task data set $\mathcal{D}_1$ to train an RBM (line 2) to describe the transitions in a richer feature space. The idea is that if the rich feature space is informative enough, the learned RBM[1] will not only be capable of reconstructing samples from the source MDP, but also from similar MDPs. We then use this learned RBM to reconstruct samples from the other MDP, as shown in lines 4 and 5 of Algorithm 1. The reconstruction error of a sample is defined as the Euclidean distance between the original sample and its reconstruction after a single forward-backward Gibbs step. The difference measure between the two MDPs, referenced from now on as *RBDist*, is defined as the average reconstruction error of all $\mathcal{T}_2$ samples.

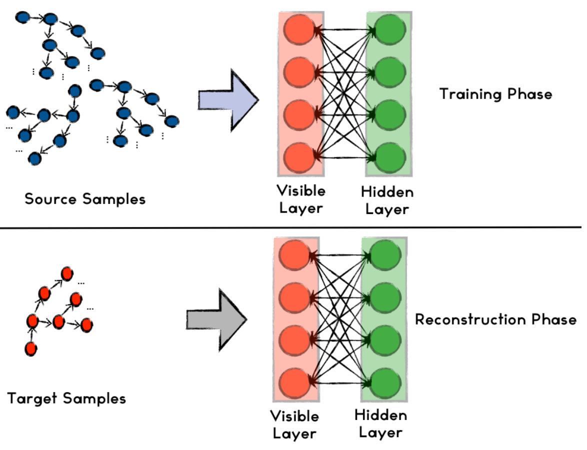


Figure 1: An illustration of the similarity measure between MDPs with shared state-action spaces. Both the training and reconstruction steps use contrastive divergence (CD).

Algorithm 1 RBDist: Shared State and Action Spaces

```
1: Input: $T_1$ samples: $\mathcal{D}_1 = \{\langle s_1^{(j)}, a_1^{(j)}, s_1'^{(j)}\rangle\}_{j=1}^m$,
          $T_2$ samples: $\mathcal{D}_2 = \{\langle s_2^{(k)}, a_2^{(k)}, s_2'^{(k)}\rangle\}_{k=1}^n$
2: Use $\mathcal{D}_1$ to train an RBM yielding $(\boldsymbol{v}, \boldsymbol{h}, \boldsymbol{W})$
3: for k=1 to n do
4:     Reconstruct each sample from $D_2$ in a single forward-backward Gibbs step using:
           $p(\boldsymbol{v}|\boldsymbol{h},\boldsymbol{W}) = \prod_{i=1}^{n_v} \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma})$
           with $\boldsymbol{\mu}_i = \sum_{f=1}^{n_h} w_{i,f} h_f + b_i$
5:     Compute the reconstruction error
           $e_k = L_2(\langle s_2^{(k)}, a_2^{(k)}, s_2'^{(k)}\rangle_0, \langle s_2^{(k)}, a_2^{(k)}, s_2'^{(k)}\rangle_1)$
6: end for
7: Return: the mean of all errors $\boldsymbol{E} = \frac{1}{n}\sum_{k=1}^n e_k$ as the measure between the MDPs.
```

[1]In reinforcement learning problems the input data can potentially be continuous. For RBMs to be capable to deal with continuous data the visible layer units are equipped with Gaussian activations rather than sigmoids. Such an RBM is typically referred to as Gaussian-Bernoulli RBM.

Experiments

Dynamical phases are vital for autonomous transfer. For example, consider the task of controlling an oscillatory system. A car with small mass in the mountain car problem needs a substantial number of oscillation to reach the top of the hill. Given source tasks with their optimal policies, we speculate that transferring from the source that exhibits close dynamical similarities to the target will produce better results compared to transferring from a less similar one.

Hypothesis 1 *Given a target task, $\mathcal{T}_2$, and n_S source tasks, $\mathcal{T}_1^{(1)}, \dots, \mathcal{T}_1^{(n_S)}$, transferring from a source task in a similar dynamical phase produces greater positive transfer to $\mathcal{T}_2$.*

We conducted two sets of experiments to test the above hypothesis. In the first experiment, we used RBDist to cluster tasks in their corresponding dynamical phases. Clustering results indeed shows that RBDist is capable of automatically discovering such dynamical phases in each of the inverted pendulum (IP), cart-pole (CP), and mountain car (MC) tasks. In the second experiment, transfer results show that: *a*) RBDist correlates with transfer performance, and *b*) transferring from tasks with similar dynamical phases produces greater positive transfer.

Experimental Domains

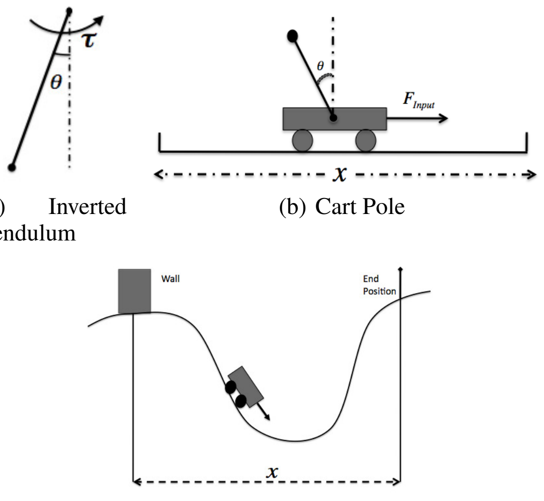
We evaluated RBDist on three domains, shown in Figure 2.

Inverted Pendulum (IP): The state of the IP is characterized by two variables: the angle θ and angular velocity $\dot{\theta}$ of the pendulum. The agent's goal is to balance the pendulum in an upright position by choosing actions consisting of two torques values $\tau = [-10, 10]$ in units of Newton meters (Nm). The agent receives a reward of -1 on every time step the pendulum is outside $-\frac{\pi}{9} < \theta < \frac{\pi}{9}$ and $+1$ for every time step its angle is in the target region.

Cart Pole (CP): The CP's state is described by the angle and angular velocity of the pole and the position and velocity of the cart: $s = \langle \theta, \dot{\theta}, x, \dot{x} \rangle$. The action space[2] is a set of 11 equally distanced linear forces between $[-1, 1]$. The agent's goal is to stabilize the pole in an upright position. A reward of $+1$ is delivered to the agent at each step the angle is between $-\frac{\pi}{9} < \theta < \frac{\pi}{9}$ and the position is $-4 < x < 4$, otherwise the reward is 0.

Mountain Car (MC): The MC state is described via the position x and velocity $\dot{x}$ of the car. The agent can choose from a discrete set of linear forces $F = [-1, 0, 1]$. The car starts at the bottom of the hill and has to drive to the top. The torque is insufficient to drive the car straight to the goal state, and therefore, the agent must oscillate to reach the goal position, where it receives a positive reward. The position of the car is bounded between $-1.5 < x < 1$ and the velocity is bounded between $-0.007 < \dot{x} < 0.007$.

[2]The actions space was discretized to show that RBDist is robust to such representational differences. A set of experiments with different discretization setting attained similar results as the ones shown in Figures 3(a), 3(b), and 3(c).



(a) Inverted Pendulum (b) Cart Pole

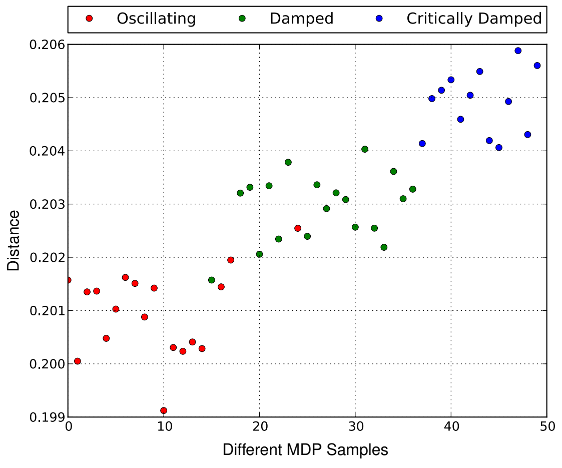
(c) Mountain Car

Figure 2: Experimental domains

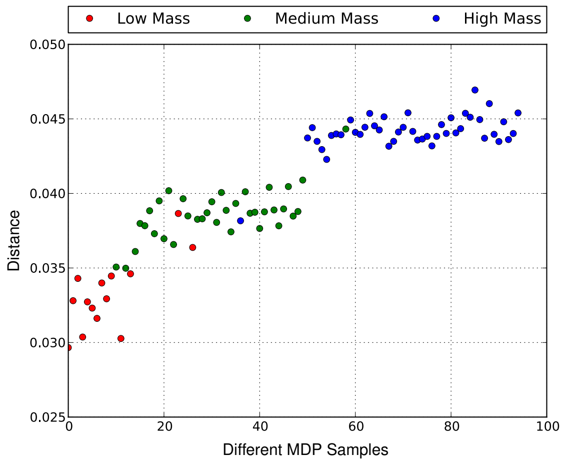
Dynamical Phase Discovery & Clustering

Each of the above systems exhibits different dynamical phases, depending on the parameter values of their transition models, i.e., oscillatory, damped, or critically damped. Each of the phases require a substantially different control policy in order to attain the desired optimal behavior. To test RBDist, each of the above systems was intentionally set to these different phases by varying their dynamical parameters. Samples from each setting were used to measure the similarity to other settings using Algorithm 1.

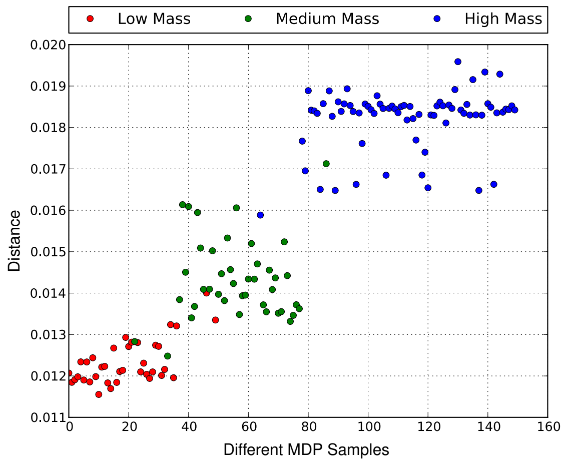
Inverted Pendulum Experiments To set the system in different phases the inertia of the rod, J, and the damping constant, b, between the rod and the wall-pin were varied, leading to three different clusters: (1) low damping with high inertia, and thus high oscillations; (2) medium damping with high inertia, resulting in oscillations but at medium frequencies; and (3) high damping such that the system does not oscillate. A random system belonging to the oscillating phase cluster was chosen as a reference and RBDist was used to determine the similarity to the other systems. Results are shown in Figure 3(a). The x-axis corresponds to 50 different MDPs randomly sampled in each of the three different phases (or clusters) of the system. Each sample set contained 5,000 transitions. The y-axis represents the similarity of these different MDPs to the reference MDP. Different colors (red, green, and blue) show the ground truth of the different phases for all 50 MDPs. Figure 3(a) shows that similar phases result in similar differences in RBDist. The first MDPs with the smallest differences all belong to the highly oscillating phase of the IP system, as does the reference MDP. The second phase, i.e., showing medium oscillation (indicated as green dots in Figure 3(a)) results in a bigger difference than the highly oscillating phase, and a smaller difference than the third, damped phase.



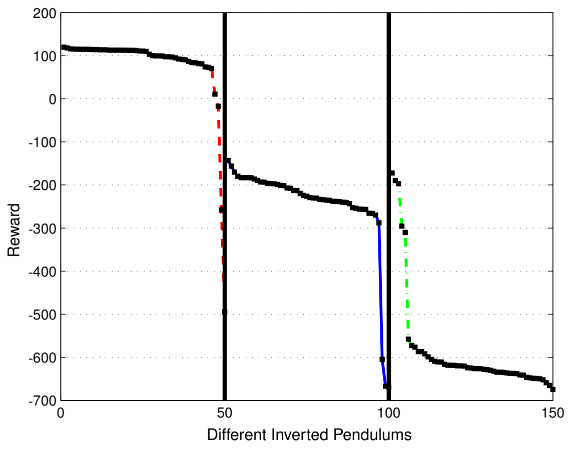
(a) RBDist: Inverted Pendulum



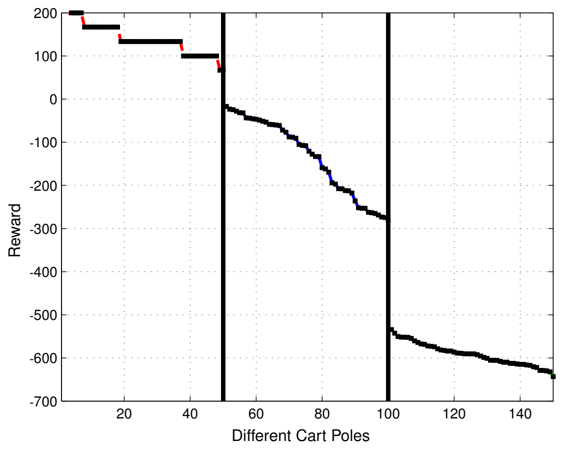
(b) RBDist: Cart Pole



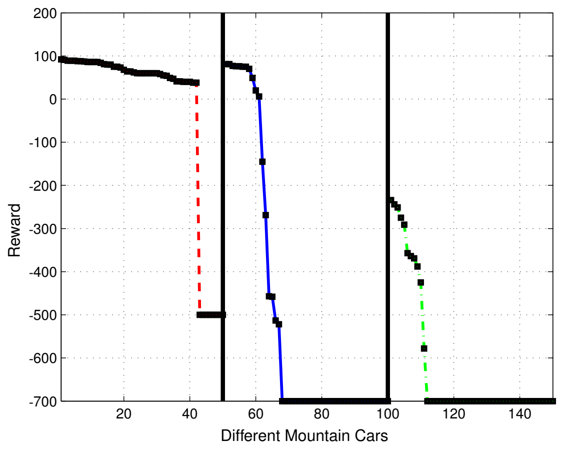
(c) RBDist: Mountain Car



(d) Jumpstart: Inverted Pendulum



(e) Jumpstart: Cart Pole



(f) Jumpstart: Mountain Car

Figure 3: Plots a–c show RBDist values for the three different phases of each domain in reference to an MDP from the red phase of that domain. Plots d–f show the Jumpstart results in each domain, demonstrating a correlation between the attained jumpstart and RBDist values. These figures represent the results of all learning algorithms.

Cart Pole and Mountain Car Experiments Similar experiments were performed on the CP and MC systems. For CP, the length (related to the inertia of the pole) and the damping constant were modified to put the system in the three different phases. For MC, the mass of the car was varied to set the car in one of the three phases, as described above. As before, a random system belonging to the oscillating phase set was chosen as a reference for each domain, and 50 different MDPs were randomly sampled in each of the three different phases for each system. Figures 3(b) and 3(c) again show that the proposed measure was capable of automatically clustering tasks with similar dynamics.

These experimental results lead us to the following conclusion: *RBDist is capable of discovering relevant phases in dynamical systems.*

Predicting Transfer Performance

To determine whether RBDist can be used to predict transfer quality, we tested the correlation between RBDist and the transferability between tasks. We measured transfer performance between tasks based on the jumpstart — the increase in initial performance on the target task from transfer from the source task.

We learned the optimal source policy on the reference MDP in each of the benchmarks using either SARSA with Q tables, Fitted-Q Iteration (FQI), or Least Squares Policy Iteration (LSPI). For transfer learning, we used either the optimal source task Q-values, Q-value parametrization, or optimal policy parametrization to initialize the target task Q tables, Q-function parameters, or policy parameters for each of SARSA, FQI, or LSPI, respectively. The policies specified by these initializations were then greedily followed and the jumpstart (i.e., the performance improvement without additional learning over no transfer) was averaged over 300 episodes.

Figures 3(d)–3(f) show a strong correlation between RBDist and transfer performance, demonstrating that when the source and target task are similar according to RBDist, we obtain high transferability and vice versa. These results also show that there is a decreasing gain from transfer learning when the source tasks become less similar to the target task. In some cases, we see that it is possible for MDPs belonging different clusters to yield high jumpstart gains, but this is relatively rare. These plots show the results of all three RL algorithms, revealing that similar transfer results were attained regardless of the adopted RL method. Therefore, we can also see that RBDist is independent of the learning method.

Conclusion

We have proposed RBDist as a similarity measure for MDPs based on restricted Boltzmann machines. Our experiments have shown that RBDist: *a*) automatically discovers dynamical phases of MDPs, and *b*) predicts transfer performance between the source and target tasks. In addition to its usefulness for ensuring high performance in transfer learning, we believe that the RBDist measure is of importance to the general RL community. There have been numerous RL algorithms proposed, typically evaluated on a handful of benchmarks, using a set of parameters tuned on each MDP. Unfortunately, this can lead to the problem of empirical overfitting (Whiteson et al. 2011) (i.e., an algorithm can work well given specific MDP parameters values, but not when using other parameter values or different MDPs). With the help of RBDist, RL algorithms can be evaluated over a well-defined set of MDPs. A natural extension of this work is to consider MDPs with different domains. In this case, different domains might have different dimensionalities, and so deep belief networks (DBNs) could be used in place of RBMs. As another direction for future work, RBDist could be used to dynamically choose relevant source tasks, enabling autonomous transfer scenarios where the agent must learn long sequences of tasks.

Acknowledgements

This work was supported in part by ONR N00014-11-1-0139, AFOSR FA8750-14-1-0069, and NSF IIS-1149917. We thank the reviewers for their helpful suggestions.

References

Ammar, H. B.; Taylor, M. E.; Tuyls, K.; Driessens, K.; and Weiss, G. 2012. Reinforcement learning transfer via sparse coding. In *Proceedings of the 11th Conference on Autonomous Agents and Multiagent Systems* (AAMAS).

Ammar, H. B.; Mocanu, D. C.; Taylor, M. E.; Driessens, K.; Weiss, G.; and Tuyls, K. 2013. Automatically mapped transfer between reinforcement learning tasks via three-way restricted Boltzmann machines. In *Proceedings of the European Conference on Machine Learning* (ECML).

Bengio, Y. 2009. Learning deep architectures for AI. *Foundations and Trends in Machine Learning* 2(1):1–127.

Buşoniu, L.; Babuška, R.; De Schutter, B.; and Ernst, D. 2010. *Reinforcement Learning and Dynamic Programming Using Function Approximators*. Boca Raton, Florida: CRC Press.

Ferns, N.; Castro, P. S.; Precup, D.; and Panangaden, P. 2006. Methods for computing state similarity in Markov decision processes. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence* (UAI), 174–181.

Ferns, N.; Panangaden, P.; and Precup, D. 2011. Bisimulation metrics for continuous Markov decision processes. *SIAM J. Computing* 40(6):1662–1714.

Hinton, G. E. 2002. Training products of experts by minimizing contrastive divergence. *Neural Computation* 14(8):1771–1800.

Salakhutdinov, R.; Mnih, A.; and Hinton, G. 2007. Restricted Boltzmann machines for collaborative filtering. In *Proceedings of the 24th International Conference on Machine Learning* (ICML), 791–798.

Taylor, M. E., and Stone, P. 2009. Transfer learning for reinforcement learning domains: a survey. *Journal of Machine Learning Research* 10(1):1633–1685.

Taylor, G. W.; Hinton, G. E.; and Roweis, S. T. 2011. Two distributed-state models for generating high-dimensional time series. *Journal of Machine Learning Research* 12:1025–1068.

Taylor, M. E.; Jong, N. K.; and Stone, P. 2008. Transferring instances for model-based reinforcement learning. In the *Adaptive Learning Agents and Multi-Agent Systems (ALA-MAS+ALAG) workshop at AAMAS*.

Taylor, M. E.; Stone, P.; and Liu, Y. 2007. Transfer learning via inter-task mappings for temporal difference learning. *Journal of Machine Learning Research* 8(1):2125–2167.

Torrey, L.; Shavlik, J. W.; Walker, T.; and Maclin, R. 2006. Skill acquisition via transfer learning and advice taking. In *Proceedings of the European Conference on Machine Learning* (ECML), 425–436.

Whiteson, S.; Tanner, B.; Taylor, M. E.; and Stone, P. 2011. Protecting against evaluation overfitting in empirical reinforcement learning. In the *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*.