Model-based Knowledge Representations

Lucas Lehnert
Computer Science Department
Brown University
Providence, RI 02912, USA
lucas_lehnert@brown.edu

Michael L. Littman
Computer Science Department
Brown University
Providence, RI 02912, USA
michael_littman@brown.edu

Michael J. Frank
Department of Cognitive, Linguistic & Psychological Sciences
Carney Institute for Brain Science
Brown University
Providence, RI 02912, USA
michael_frank@brown.edu

Abstract

One question central to reinforcement learning is which representations – including aspects of the state space, transition function and reward function – can be generalized or re-used across different tasks. Humans are adept at such flexible transfer but existing RL algorithms are much more limited. This paper presents model features, a latent representation that compresses the state space of a control problem by exploiting which states are equivalent in terms of both transition and reward functions. Because model features only extract these equivalences from the transition and reward functions, but are not tied to the functions themselves, this latent state representation generalizes across tasks that change in both their transition and reward functions.

While transferring successor features between different tasks has been shown to improve learning speed, this representation is overly specific and hence needs to be re-learned when the optimal policy or transition function change. Model features link successor features to model reductions, facilitating the design of gradient-based optimization algorithms to approximate model reductions directly from transition data. Learning model features is akin to model-based RL, because from the learned model a linear action model can be extracted to predict future reward outcomes. This paper summarizes theoretical results from our extended paper and presents empirical simulation results showing that model features which minimize prediction errors on future reward outcomes serve as useful state abstractions that afford generalization across tasks that differ in both transition and reward functions.

Because model features construct a latent state representation that supports predictions of future reward outcomes, the presented results motivate further experiments to investigate if humans or animals learn such a representation, and whether neural systems involved in state representation reflect the equivalence abstraction.

Keywords: Model-based Reinforcement Learning, Knowledge Representations, Latent Structure Learning, Successor Representation, Human and Animal Learning

1 Introduction

Reinforcement learning (RL) [19] studies the problem of computing optimal decision strategies from one-step interactions sampled from an environment. Each interaction consists of the agent selecting an action to cause a change in the environment's state. For each state transition the agent receives a reward, a single scalar number. The goal is to compute a policy that maximizes rewards. One question central in RL is how to generalize knowledge across different environments. This paper presents model features, a feature representation that compresses the state space into a lower dimensional space by clustering states that produce identical future reward outcomes. By only exploiting such state equivalences, model features generalize across tasks that differ in reward and transition functions.

A Markov decision processes (MDP) [19] consists of a 5-tuple $M = \langle \mathcal{S}, \mathcal{A}, p, r, \gamma \rangle$, where $\mathcal{S}$ is a set of all possible states and the agent is allowed to chose an action from a set of actions $\mathcal{A}$ to trigger a state transition according to the stochastic transition function p. The reward function r specifies the rewards for each transition. The discount factor $\gamma \in [0, 1)$ determines how strongly short term rewards are emphasized over long term rewards. One key property of model-based RL is the ability to predict a sequence of reward outcomes $(r_1, r_2, ...)$ given an arbitrary sequence of actions $(a_1, a_2, ...)$. Model features compress the state space by assigning two different states (approximately) the same n-dimensional feature vector if, given an arbitrary action sequence $(a_1, a_2, ...)$, both states generate the same reward sequence $(r_1, r_2, ...)$ with equal (or near-equal) probability. Such two states are also called behaviourally equivalent or stochastically bisimilar [9]. Knowing which reward sequence a certain action sequence can generate is sufficient for evaluating any arbitrary policy and identifying the optimal policy. Model features encode model reductions [9] and provide a basis function [18, 11] specifically trained to preserve all information relevant for predicting future reward outcomes.

This paper first summarizes results from our extended paper [12] and shows that model features, and by extension model reductions and bisimulation relations, can be extracted by learning the successor features of a single arbitrary policy. Model features construct a low dimensional representation of the state space by utilizing which states are equivalent to the transition and reward function. We then show that a model feature representation learned for one MDP can be re-used on another MDP with different transition and reward functions, assuming that state equivalences are preserved. Such "deep transfer" across tasks, even the absence of prior experience with specific transition or reward functions, is predicted by behavioral and neural signatures of human structure learning [3, 1, 8] but not afforded by alternative algorithms that compress the transition function directly [17, 16]. As a lay example, an expert musician can immediately transfer a song learned in one key to another, or on a guitar to a piano, despite the very different transition functions [7].

2 Successor Features encode Model Features

A basis function is a function ϕ mapping states s to a real valued vector $\boldsymbol{\phi}_s$. Basis functions perform transforms on the state space and can be used to construct different feature-to-feature transition dynamics. Model features are designed to preserve the dynamics of the underlying MDP. For example, a bijection between the state space $\mathcal{S}$ and $\mathbb{R}^n$ would resemble a valid model feature. However, the aim is to construct model features that also compress the state space. Suppose $\phi : \mathcal{S} \to \{\boldsymbol{e}_1, ..., \boldsymbol{e}_n\}$ is a model feature mapping states to one-hot bit vectors $\boldsymbol{e}_i$ where the ith entry is set to one and all other entries are zero. If ϕ preserves the transition dynamics, then for any action sequence and start state s_0

$$\Pr\{s_1, s_2, ...|s_0, a_1, a_2, ...\} = \Pr\{\boldsymbol{e}_1, \boldsymbol{e}_2, ...|\boldsymbol{e}_1, a_1, a_2, ...\}, \tag{1}$$

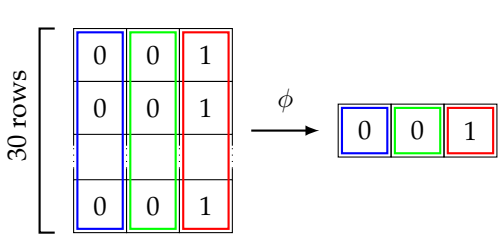
where ϕ maps s_0 to $\boldsymbol{e}_1$. Meaning any two states s and $\tilde{s}$ mapped to the same $\boldsymbol{e}_i$ have to generate identical feature trajectories $(\boldsymbol{e}_1, \boldsymbol{e}_2, ...)$ with equal probability. If ϕ would not preserve the transition dynamics of the underlying MDP, then the probability of observing $\boldsymbol{e}_1, \boldsymbol{e}_2, ...$ would depend on the actual start state s_1. In this case, two states s and $\tilde{s}$ mapped to the same $\boldsymbol{e}_i$ would not generate identical feature trajectories $(\boldsymbol{e}_1, \boldsymbol{e}_2, ...)$ with equal probability. However, if ϕ is a model feature, then the probability $\Pr\{\boldsymbol{e}_1, \boldsymbol{e}_2, ...|\boldsymbol{e}_1, a_1, a_2, ...\}$ is conditioned on $\boldsymbol{e}_1$, a state partition, and is not conditioned on a single state s_1. Figure 1(b) further explains the connection between state and feature transition probabilities. If ϕ is also required to preserve information about rewards, then any two such states s and $\tilde{s}$ mapped to the same vector $\boldsymbol{e}_i$ also need to have equal one-step rewards. Figure 1(a) shows an example of a 90 state grid world that can be compressed into 3 states, because only the column information is relevant for predicting when a reward of zero or one is observed.

Successor features (SFs) [5, 2, 16] predict the visitation frequencies over future state features and these visitation frequencies implicitly encode the feature-to-feature transition probabilities. For a policy $\pi : \mathcal{S} \to \mathcal{A}$, SFs are defined as

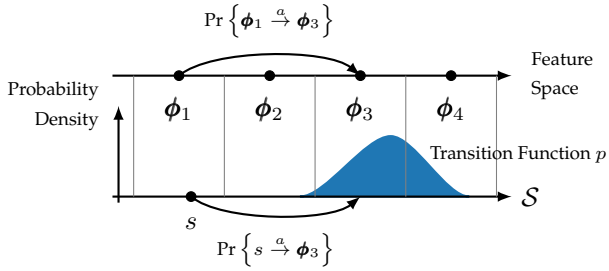
$$\boldsymbol{\psi}^{\pi}_{s,a} = \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} \boldsymbol{\phi}_t \middle| s = s_1, a = a_1, \pi\right], \tag{2}$$

where the expectation in Eq. (2) ranges over all possible sequences that start with state s and action a and then follow the policy π. As discussed above, if ϕ is a model feature, the probability of a feature sequence $(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, ...)$ can to be conditioned on the first feature $\boldsymbol{\phi}_1$ itself. Similarly, SFs can be conditioned on a particular feature vector by assuming

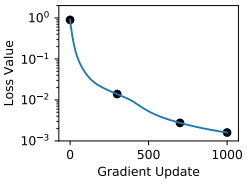
$$\boldsymbol{\psi}^{\pi}_{s,a} = \boldsymbol{\phi}_s^{\top} \boldsymbol{F}^a, \tag{3}$$



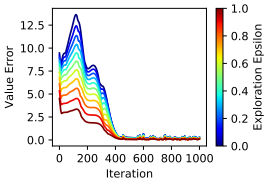
(a) Column Grid World. The 90 grid states can be compressed into 3 states, because only the column position is relevant for predicting rewards.



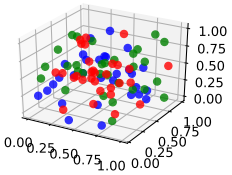
(b) Relationship between transition probabilities. Eq. (1) holds if the probability of transitioning between two feature vectors $\Pr\{\boldsymbol{\phi}_1 \xrightarrow{a} \boldsymbol{\phi}_3\}$ equals the probability of transitioning from one state into the state partition $\boldsymbol{\phi}_3$, $\Pr\{s \xrightarrow{a} \boldsymbol{\phi}_3\}$.



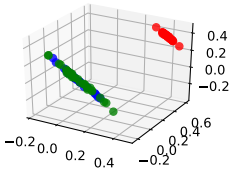
(c) Loss Objective Eq. (5)



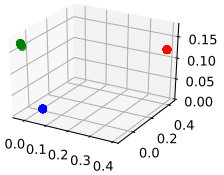
(d) Value Error



(e) Initialization



(f) Iteration 300



(g) Iteration 1000

Figure 1: Learning Model Features. Figure 1(a) shows the Column Grid World example. Figure 1(b) shows the connection between feature-to-feature and state-to-state transition probabilities. Figures 1(c) to 1(g) plot how the loss objective evolves during optimization for the column world MDP. As the value of the loss objective decreases, the feature vectors for equivalent state merge into the same cluster and the different cluster centers move apart. Figure 1(d) shows the error for predicting the expected discounted return for different ε-greedy policy. An ε-greedy policy chooses with $1-\varepsilon$ probability the optimal action and with ε probability actions uniformly at random. As the approximation of model features improves, the same state representation can be used to predict the value of a range of different policies (Figure 1(d)).

meaning two states mapping to the same feature vector are mapped to the same SF vector. Hence the visitation frequencies over future features is conditioned on a feature vector $\boldsymbol{\phi}_s$, rather than a single state. Substituting Eq. (3) into Eq. (2) results in the first identity in line (4). SFs obey a bellman fix-point equation similar to value functions.

$$\boldsymbol{\phi}_s^\top \boldsymbol{F}^a = \boldsymbol{\phi}_s^\top + \gamma \mathbb{E}\left[\boldsymbol{\phi}_{s'}^\top \boldsymbol{F}^{\pi(s')} \middle| s', \pi\right] \quad \boldsymbol{\phi}_s^\top \boldsymbol{r}_\phi^a = r(s,a) \tag{4}$$

The second identity in line (4) requires that two states mapped to the same feature have equal one-step rewards. A theoretical analysis proving that line (4) is sufficient for encoding model reductions as defined by [9] can be found in our extended paper [12]. Using line (4) a new objective function can be defined to learning model features[1]:

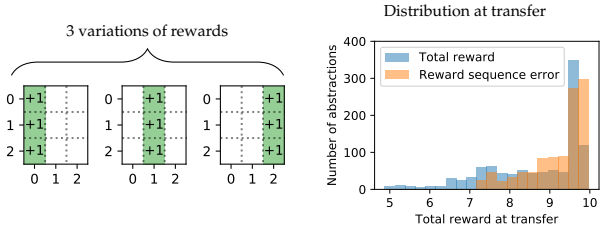
$$\mathcal{L}(\boldsymbol{\phi}, \{\boldsymbol{F}^a\}_{a\in\mathcal{A}}, \{\boldsymbol{r}_\phi^a\}_{a\in\mathcal{A}}) = \mathbb{E}\Big[\left(\boldsymbol{\phi}_s^\top \boldsymbol{r}_\phi^a - r(s,a)\right)^2 + \alpha \left|\left|\boldsymbol{\phi}_s^\top + \gamma \boldsymbol{\phi}_{s'}^\top \boldsymbol{F}^{\overline{\pi}} - \boldsymbol{\phi}_s^\top \boldsymbol{F}^a\right|\right|_2^2\Big]. \tag{5}$$

Figures 1(c) to 1(g) illustrate how minimizing Eq. (5) approximates model features. Such an approximation minimizes the prediction errors of future reward outcomes and states that are only approximately bisimilar may be clustered. Figure 1(d) shows how the learned feature representation allows for generalization across a range of different policies in the column grid world (see below for transfer to different MDPs). A linear action model [20] can be analytically extracted from a solution to Eq. (5) to evaluate any arbitrary policy. Hence, learning model features is akin to model-based RL [12].

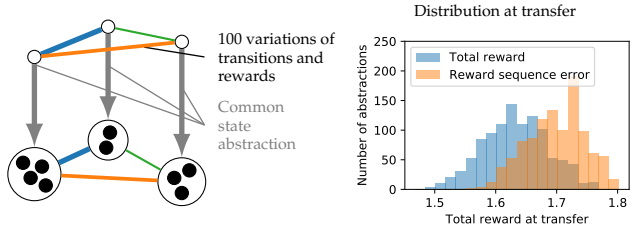
3 Model Features encode Task Knowledge

Model features construct a low dimensional representation of the state space by clustering states that are equivalent to the transition and reward function. On three transfer examples, Figure 2 shows that a state abstraction's ability to predict future reward outcomes is indicative of the abstraction's ability to be re-used in a previously unseen MDP, and moreover, is more useful than abstractions that maximize total reward in the original MDP. If a state abstraction incorrectly clusters states, the resulting compressed MDP does not resemble the original MDP and thus a policy optimal in the compressed MDP is not necessarily optimal in the original MDP and cannot generate a high total reward. For each example, all possible state abstractions were enumerated and scored on their reward sequence prediction error and the total reward

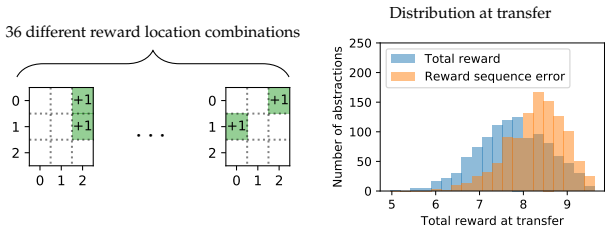
[1]The expectation in Eq. (5) ranges over transitions (s, a, r, s') from some fixed transition data set and α is a scalar hyper-parameter.



(a) Transfer between different column worlds.



(b) Transfer between random MDPs compressible to 3 states.



(c) Transfer between grid worlds with different goal locations.

	p-value
Fig 2(a)	$3.29 \cdot 10^{-30}$
Fig 2(b)	$8.63 \cdot 10^{-176}$
Fig 2(c)	$9.60 \cdot 10^{-75}$

(d) One sided Welch's t-test.

Figure 2: Low reward sequence prediction errors identify state abstractions amenable for "deep transfer". For each experiment all possible state abstractions were enumerated using Algorithm U [10]. State abstractions were scored by compressing an MDP using the state abstraction of interest [14]. The total reward score was computed by solving for the optimal policy using value iteration [19, Chapter 4.4] and running the computed policy 20 times for 10 time steps in the MDP from a randomly selected start state. The reward sequence error was computed by selecting 20 random start states and then performing a random walk for 10 time steps. The histograms report averages over all repeats and transfer MDPs. Figure 2(d) lists the p-values of the difference in mean total reward being insignificant for each histogram.

generated by a policy optimal in the compressed MDP, using only one randomly selected MDP. The top 5% scoring state abstractions were then re-evaluated on 99 other MDPs and the total reward generated by these state abstractions is plotted as a histogram for each transfer example in Figure 2. In all cases a state abstraction that produces low reward sequence prediction error generates a higher total reward at transfer than state abstractions that were selected based on their ability to construct a well performing policy on any given original MDP. This result indicates that model features which are designed to produce low reward sequence prediction errors, encode information about an MDP that can be generalized across different MDPs.

Figure 2(b) presents a transfer example between 100 random MDPs that share the same state abstraction compressing a 9 state problem into a 3 "abstract" state problem. In this experiment, each MDP has a randomly generated transition and reward function, but state equivalences are preserved across all MDPs. The histogram shows that state abstractions with a low reward sequence prediction error perform on average significantly better on the remaining 99 MDPs, because these state abstractions approximately capture the underlying state equivalences common to all 100 MDPs. State abstractions selected based on the total reward generated on one MDP do not generalize as well, because they are tied to the optimal policy and to one specific transition and reward function, which change when transferring the state representation to a different MDP. This pattern repeats in Figure 2(a), but the difference in distribution is not as significant because the transition function remains fixed between different MDPs (only the reward function differs). Figure 2(c) shows an example where no state compression is possible without incurring some reward sequence prediction error. Only the identity map is optimal, because the location information in the grid is necessary to predict the +1 reward cells. However, the histogram in Figure 2(c) shows that abstractions selected based on minimizing reward sequence prediction error criterion still perform better than selecting the abstraction by their total reward. Because grid worlds have a specific topology of the state space, a state abstraction clustering only neighbouring states can produce relatively low reward sequence prediction errors. Such a state abstraction would be expected to perform relatively well across all different reward locations.

4 Conclusion

In model-based RL, the agent has the ability to predict reward outcomes over multiple time steps into the future. Model features approximate state abstractions that compress an MDP while preserving the ability to predict future reward outcomes using only the compressed model. Such a feature representation exploits which states are equivalent for both the reward and transition function and thus model features can be understood as a model-based knowledge representation.

Model-based basis functions have been studied previously [4, 6], but the presented connection to SFs allows us to design an gradient-based optimization algorithm that can construct bisimulation relations directly from transition data.

Because model features only leverage state equivalences, these representations can generalize across environments that differ in both reward and transition functions. Previous work on transfer with SFs has shown that re-using SFs on an MDP with a different reward function can provide faster learning [2, 15, 16]. However, SFs are fragile to changes in the optimal policy [13] and transition function, whereas latent state representations are more abstract and thus are not tied to particular transitions [3, 7]. In related work, Stachenfeld et al. [17] compress the SR of an MDP using PCA and show how this compressed representation can be used for transfer and is tied to place cells and grid cells in the hippocampus. However, in that case model features construct a lower dimensional representation of the transition function itself, and hence transfer is limited to environments that share the same transition function. In contrast to Stachenfeld et al., model features separate the transition dynamics (and the SR) from the compression on the state space itself, and thus generate a latent state representation of a task exploiting task equivalences. Collins and Frank [3] and Badre and Frank [1] show how reward prediction errors and prediction errors on the structure of the state space identify latent state representations that accelerate learning in humans when transferring knowledge across tasks. While this work considers contextual multi-armed bandits, model features may allow these results can be extended to sequential decision making, motivating further experiments to investigate if humans or animals learn such a feature representation.

References

[1] D. Badre and M. J. Frank. Mechanisms of hierarchical reinforcement learning in cortico–striatal circuits 2: Evidence from fmri. *Cerebral cortex*, 22(3):527–536, 2011.

[2] A. Barreto, W. Dabney, R. Munos, J. J. Hunt, T. Schaul, H. P. van Hasselt, and D. Silver. Successor features for transfer in reinforcement learning. In *Advances in neural information processing systems*, pages 4055–4065, 2017.

[3] A. G. E. Collins and M. J. Frank. Neural signature of hierarchically structured expectations predicts clustering and transfer of rule sets in reinforcement learning. *Cognition*, 152:160–169, 2016.

[4] G. Comanici, D. Precup, and P. Panangaden. Basis refinement strategies for linear value function approximation in mdps. In *Advances in Neural Information Processing Systems*, pages 2899–2907, 2015.

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

[6] N. Ferns and D. Precup. Bisimulation metrics are optimal value functions. In *UAI*, pages 210–219. Citeseer, 2014.

[7] N. T. Franklin and M. J. Frank. Compositional clustering in task structure learning. *PLoS computational biology*, 14 (4):e1006116, 2018.

[8] N. T. Franklin and M. J. Frank. Generalizing to generalize: when (and when not) to be compositional in task structure learning. *bioRxiv*, 2019. doi: 10.1101/547406.

[9] R. Givan, T. Dean, and M. Greig. Equivalence notions and model minimization in markov decision processes. *Artificial Intelligence*, 147(1):163–223, 2003.

[10] D. E. Knuth. *The Art of Computer Programming, Volume 4, Fascicle 3: Generating All Combinations and Partitions*. Addison-Wesley, 2005.

[11] G. Konidaris, S. Osentoski, and P. Thomas. Value function approximation in reinforcement learning using the fourier basis. *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, pages pages 380–385, August 2011.

[12] L. Lehnert and M. L. Littman. Successor features support model-based and model-free reinforcement learning. *arXiv preprint arXiv:1708.00102*, 2019.

[13] L. Lehnert, S. Tellex, and M. L. Littman. Advantages and limitations of using successor features for transfer in reinforcement learning. *arXiv preprint arXiv:1708.00102*, 2017.

[14] L. Li, T. J. Walsh, and M. L. Littman. Towards a unified theory of state abstraction for mdps. In *ISAIM*, 2006.

[15] I. Momennejad, E. M. Russek, J. H. Cheong, M. M. Botvinick, N. Daw, and S. J. Gershman. The successor representation in human reinforcement learning. *Nature Human Behaviour*, 1(9):680, 2017.

[16] E. M. Russek, I. Momennejad, M. M. Botvinick, S. J. Gershman, and N. D. Daw. Predictive representations can link model-based reinforcement learning to model-free mechanisms. *PLoS computational biology*, 13(9):e1005768, 2017.

[17] K. L. Stachenfeld, M. M. Botvinick, and S. J. Gershman. The hippocampus as a predictive map. *Nature Neuroscience*, 20:1643 EP –, 10 2017. URL `https://doi.org/10.1038/nn.4650`.

[18] R. S. Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. *Advances in neural information processing systems*, pages 1038–1044, 1996.

[19] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[20] H. Yao and C. Szepesvári. Approximate policy iteration with linear action models. In *AAAI*, 2012.