REVIEW **Communicated by Ari Kahn**

Predictive Representations: Building Blocks of Intelligence

Wilka Carvalho
wcarvalho92@gmail.com
Kempner Institute for the Study of Natural and Artificial Intelligence, Harvard University, Cambridge, MA 02134, U.S.A.

Momchil S. Tomov
tomov90@gmail.com
Department of Psychology and Center for Brain Science, Harvard University, Cambridge, MA 02134, U.S.A., and Motional AD LLC, Boston, MA 02210, U.S.A.

William de Cothi
w.decothi@ucl.ac.uk
Caswell Barry
caswell.barry@ucl.ac.uk
Department of Cell and Developmental Biology, University College London, London WC1E 7JE, U.K.

Samuel J. Gershman
gershman@fas.harvard.edu
Kempner Institute for the Study of Natural and Artificial Intelligence, and Department of Psychology and Center for Brain Science, Harvard University, Cambridge, MA 02134, U.S.A., and Center for Brains, Minds, and Machines, MIT, Cambridge, MA 02139, U.S.A.

Adaptive behavior often requires predicting future events. The theory of reinforcement learning prescribes what kinds of predictive representations are useful and how to compute them. This review integrates these theoretical ideas with work on cognition and neuroscience. We pay special attention to the successor representation and its generalizations, which have been widely applied as both engineering tools and models of brain function. This convergence suggests that particular kinds of predictive representations may function as versatile building blocks of intelligence.

1. Introduction

The ability to make predictions has been hailed as a general feature of both biological and artificial intelligence, cutting across disparate perspectives

Wilka Carvalho, Momchil S. Tomov, and William de Cothi contributed equally.

Neural Computation **36, 2225–2298** (2024)
https://doi.org/10.1162/neco_a_01705

on what constitutes intelligence (Ciria et al., 2021; Clark, 2013; Friston & Kiebel, 2009; Ha & Schmidhuber, 2018; Hawkins & Blakeslee, 2004; Littman & Sutton, 2001; Lotter et al., 2016). Despite this general agreement, attempts to formulate the idea more precisely raise many questions: Predict what, and over what timescale? How should predictions be represented? How should they be used, evaluated, and improved? These normative "should" questions have corresponding empirical questions about the nature of prediction in biological intelligence. Our goal is to provide systematic answers to these questions. We will develop a small set of principles that have broad explanatory power.

Our perspective is based on an important distinction between predictive *models* and predictive *representations*. A predictive model is a probability distribution over the dynamics of a system's state. A model can be "run forward" to generate predictions about the system's future trajectory. This offers a significant degree of flexibility: an agent with a predictive model can, given enough computation time, answer virtually any query about the probabilities of future events. However, the "given enough computation time" proviso places a critical constraint on what can be done with a predictive model in practice. An agent that needs to act quickly under stringent computational constraints may not have the luxury of posing arbitrarily complex queries to its predictive model. Predictive representations, however, cache the answers to certain queries, making them accessible with limited computational cost.[1] The price paid for this efficiency gain is a loss of flexibility: only certain queries can be accurately answered.

Caching is a general solution to ubiquitous flexibility-efficiency tradeoffs facing intelligent systems (Dasgupta & Gershman, 2021). Key to the success of this strategy is caching representations that make task-relevant information directly accessible to computation. We will formalize the notion of task-relevant information, as well as what kinds of computations access and manipulate this information, in the framework of reinforcement learning (RL) theory (Sutton & Barto, 2018). In particular, we will show how one family of predictive representation, the *successor representation* (SR) and its generalizations, distills information that is useful for efficient computation across a wide variety of RL tasks. These predictive representations facilitate exploration, transfer, temporal abstraction, unsupervised pretraining, multi-agent coordination, creativity, and episodic control. On the basis of such versatility, we argue that these predictive representations can serve as fundamental building blocks of intelligence.

Converging support for this argument comes from cognitive science and neuroscience. We review a body of data indicating that the brain uses predictive representations for a range of tasks, including decision making,

[1] While we adhere to this definition consistently throughout this review, we recognize that other uses of the phrase "predictive representation" appear in the literature.

navigation, and memory. We also discuss biologically plausible algorithms for learning and computing with predictive representations. This convergence of biological and artificial intelligence suggests that predictive representations may be a widely used tool for intelligent systems.

Several previous surveys on predictive representations have scratched the surface of these connections (Gershman, 2018; Momennejad, 2020). The purpose of this survey is to approach the topic in much greater detail, yielding a comprehensive reference on both technical and scientific aspects. Despite this broad scope, the survey's focus is restricted to predictive representations in the domain of RL; we do not review predictive representations that have been developed for language modeling, vision, and other problems. An important long-term goal will be to fully synthesize the diverse notions of predictive representations across these domains.

2. Theory

In this section, we introduce the general problem setup and a classification of solution techniques. We then formalize the SR and discuss how it fits into the classification scheme. Finally, we describe two key extensions of the SR that make it much more powerful: the successor model and successor features. Due to space constraints, we omit some more exotic variants such as the first-occupancy representation (Moskovitz et al., 2022) or the forward-backward representation (Touati et al., 2022).

2.1 The Reinforcement Learning Problem. We consider an agent situated in a Markov decision process (MDP) defined by the tuple $M = (\gamma, \mathcal{S}, \mathcal{A}, T, R)$, where $\gamma \in [0, 1)$ is a discount factor, $\mathcal{S}$ is a set of states (the state space), $\mathcal{A}$ is a set of actions (the action space), $T(s'|s, a)$ is the probability of transitioning from state s to state s' after taking action a, and $R(s)$ is the expected reward in state s.[2] Following Sutton and Barto (2018), we consider settings where the MDP and an agent give rise to a trajectory of experience,

$$s_0, a_0, r_1, s_1, a_1, \ldots, \tag{2.1}$$

where state s_t and action a_t lead to reward r_{t+1} and state s_{t+1}. The agent chooses actions probabilistically according to a state-dependent policy $\pi(a|s)$.

We consider settings where the agent prioritizes immediate reward over future reward, as formalized by the concept of discounted return. The value

[2] For notational convenience, we will assume that the state and action spaces are both discrete, but this assumption is not essential for many of the algorithms described here.

of a policy is the expected[3] discounted return:

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{t=0}^{H} \gamma^t R(s_{t+1}) \middle| s_0 = s\right]. \tag{2.2}$$

One can also define a state-action value (i.e., the expected discounted return conditional on action a in state s) by

$$Q^{\pi}(s,a) = \mathbb{E}\left[\sum_{t=0}^{H} \gamma^t R(s_{t+1}) \middle| s_0 = s, a_0 = a\right]. \tag{2.3}$$

The optimal policy π^* is then defined by

$$\pi^*(\cdot|s) = \underset{\pi(\cdot|s)}{\operatorname{argmax}} V^{\pi}(s) = \underset{\pi(\cdot|s)}{\operatorname{argmax}} \sum_{a} \pi(a|s) Q^{\pi}(s,a). \tag{2.4}$$

The optimal policy for an MDP is always deterministic (choose the value-maximizing action):

$$\pi^*(a|s) = \mathbb{I}\left[a = \underset{\tilde{a}}{\operatorname{argmax}}\, Q^*(s,\tilde{a})\right], \tag{2.5}$$

where $\mathbb{I}[\cdot] = 1$ if its argument is true and 0 otherwise. This assumes that the agent can compute the optimal values $Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a)$ exactly. In most practical settings, values must be approximated. In these cases, stochastic policies are useful (e.g., for exploration), as discussed later.

The Markov property for MDPs refers to the conditional independence of the past and future given the current state and action. This property allows us to write the value function in a recursive form known as the Bellman equation (Bellman, 1957):

$$\begin{aligned} V^{\pi}(s) &= \sum_{a} \pi(a|s) \sum_{s'} T(s'|s,a)\left[R(s') + \gamma V^{\pi}(s')\right] \\ &= \mathbb{E}[R(s') + \gamma V^{\pi}(s')]. \end{aligned} \tag{2.6}$$

Similarly, the state-action value function obeys a Bellman equation:

$$Q^{\pi}(s,a) = \mathbb{E}[R(s') + \gamma Q^{\pi}(s',a')]. \tag{2.7}$$

These Bellman equations lie at the heart of many efficient RL algorithms, as we discuss next.

[3] To simplify notation, we will sometimes leave implicit the distributions over which the expectation is being taken.

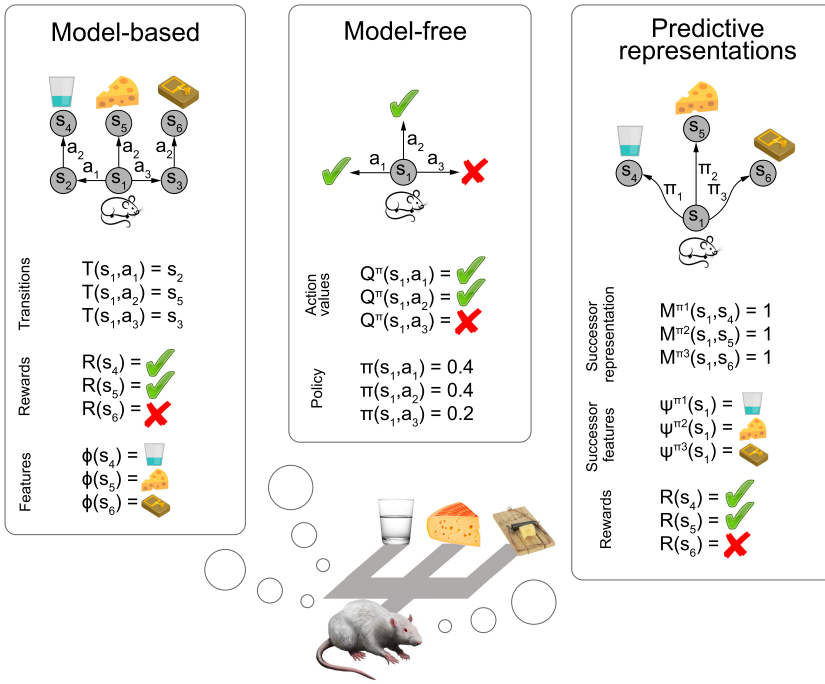

Figure 1: Algorithmic solutions to the RL problem. An agent solving a three-armed maze (bottom) can adopt different classes of strategies (top). Model-based strategies (left) learn an internal model of the environment, including the transition function (T), the reward function (R), and (optionally) the features ($\boldsymbol{\phi}$). At decision time, the agent can run forward simulations to predict the outcomes of different actions. Model-free strategies (middle) learn action values (Q) and/or a policy (π). At decision time, the agent can consult the cached action values and/or policy in the current state. Strategies relying on predictive representations (right) learn the successor representation (SR) matrix ($\mathbf{M}$) mapping states to future states and/or the successor features ($\boldsymbol{\psi}$) mapping states to future features, as well as the reward function (R). At decision time, the agent can consult the cached predictions and cross-reference them with its task (specified by the reward function) to choose an action.

2.2 Classical Solution Methods. We say that an algorithm solves an MDP if it outputs an optimal policy (or an approximation thereof). Broadly speaking, algorithms can be divided into two classes:

- **Model-based algorithms** use an internal model of the MDP to compute the optimal policy (see Figure 1, left).
- **Model-free algorithms** compute the optimal policy by interacting with the MDP (see Figure 1, middle).

These definitions allow us to be precise about what we mean by *predictive model* and *predictive representation*. A predictive model corresponds to $\hat{T}$, the internal model of the transition distribution, and $\hat{R}$, an internal model of the reward function. An agent equipped with $\hat{T}$ can simulate state trajectories and answer arbitrary queries about the future. Of principal relevance to solving MDPs is policy evaluation, an answer to the query, "How much reward do I expect to earn in the future under my current policy?" A simple (but inefficient) way to do this, known as *Monte Carlo policy evaluation*, is by running many simulations from each state (roll-outs) and then averaging the discounted return. The basic problem with this approach stems from the curse of dimensionality (Bellman, 1957): the trajectory space is very large, requiring a number of roll-outs that is exponential in the trajectory length.

A better model-based approach exploits the Bellman equation. For example, the value iteration algorithm starts with an initial estimate of the value function, $\hat{V}^\pi$, then simultaneously improves this estimate and the policy by applying the following update (known as a Bellman backup) to each state:

$$\hat{V}^\pi(s) \leftarrow \max_a \sum_{s'} \hat{T}(s'|s,a)\left[\hat{R}(s') + \gamma \hat{V}^\pi(s')\right]. \tag{2.8}$$

This is a form of dynamic programming, guaranteed to converge to the optimal solution, $V^*(s) = \max_\pi V^\pi(s)$, when the agent's internal model is accurate ($\hat{T} = T, \hat{R} = R$). After convergence, the optimal policy for state s is given by

$$\pi^*(a|s) = \mathbb{I}[a = a^*(s)], \tag{2.9}$$

$$a^*(s) = \operatorname*{argmax}_a Q^*(s,a) \approx \operatorname*{argmax}_a \hat{Q}^*(s,a). \tag{2.10}$$

The approximation becomes an equality when the agent's internal model is accurate.

Value iteration is powerful, but still too cumbersome for large state spaces, since each iteration requires $\mathcal{O}(|\mathcal{A}||\mathcal{S}|^2)$ steps. The basic problem is that algorithms like value iteration attempt to compute the optimal policy for every state, but in an online setting, an agent only needs to worry about what action to take in its current state. This problem is addressed by tree search algorithms, which rely on roll-outs (as in Monte Carlo policy evaluation) but only from the current state. When combined with heuristics for determining which roll-outs to perform (e.g., learned value functions; see below), this approach can be highly effective (Silver et al., 2016).

Despite their effectiveness for certain problems (e.g., games like Go and chess), model-based algorithms have had only limited success in a wider range of problems (e.g., video games) due to the difficulty of learning a good model and planning in complex (possibly infinite/continuous) state

spaces.[4] For this reason, much of the work in modern RL has focused on model-free algorithms.

A model-free agent by definition has no access to $\hat{T}$ (and sometimes no access to $\hat{R}$), but nonetheless can still answer certain queries about the future if it has cached a predictive representation. For example, an agent could cache an estimate of the state-action value function, $\hat{Q}^\pi(s,a) \approx Q^\pi(s,a)$. This predictive representation does not afford the same flexibility as a model of the MDP, but it has the advantage of caching, in a computationally convenient form, exactly the information about the future that an agent needs to act optimally.

Importantly, $\hat{Q}^\pi(s,a)$ can be learned purely from interacting with the environment, without access to a model. For example, temporal difference (TD) learning methods use stochastic approximation of the Bellman backup. Q-learning is the canonical algorithm of this kind:

$$\hat{Q}^\pi(s,a) \leftarrow \hat{Q}^\pi(s,a) + \eta\delta, \tag{2.11}$$

$$\delta = R(s') + \gamma \max_{a'} \hat{Q}^\pi(s',a') - \hat{Q}^\pi(s,a), \tag{2.12}$$

where $\eta \in [0,1]$ is a learning rate and s' is sampled from $T(s'|s,a)$. When the estimate is exact, $\mathbb{E}[\delta] = 0$ and $\hat{Q}^\pi = Q^\pi$. Moreover, these updates converge to $Q^*(s,a)$ with probability 1 provided that the learning rates satisfy the standard Robbins-Monro conditions for stochastic approximation (Watkins & Dayan, 1992).

We have briefly discussed the dichotomy of model-based versus model-free algorithms for learning an optimal policy. Model-based algorithms are more flexible—capable of generating predictions about future trajectories—while model-free algorithms are more computationally efficient—capable of rapidly computing the approximate value of an action. The flexibility of model-based algorithms is important for transfer: when the environment changes locally (e.g., a route is blocked or the value of a state is altered), an agent's model will typically also change locally, allowing it to transfer much of its previously learned knowledge without extensive new learning. In contrast, a cached value function approximation (due to its long-term dependencies) will change nonlocally, necessitating more extensive learning to update all the affected cached values.

One of the questions we aim to address is how to get some aspects of model-based flexibility without learning and computing with a predictive model. This leads us to another class of predictive representations: the SR. In this section, we describe the SR (see section 2.3), its probabilistic variant

[4] Recent work on applying model-based approaches to video games has seen some success (Tsividis et al., 2021), but progress toward scalable and generally applicable versions of such algorithms is still in its infancy.

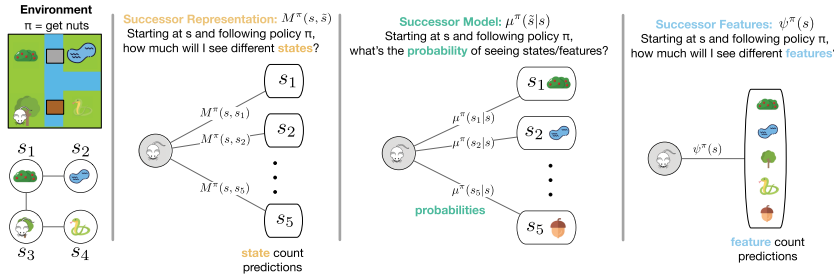

Figure 2: Three kinds of predictive representations: the successor representation (see section 2.3), the successor model (see section 2.4), and successor features (see section 2.5). Their computations are summarized in Table 1. Each of these predictive predictive representations describes a state by a prediction of what will happen when a policy π is followed. With the **successor representation**, one gets a description of how much all states will be visited in the near future when beginning at state s. One limitation of this is that it does not scale well to large state spaces, since it is impractical to maintain predictions about all states in the state space. **Successor models** circumvent this challenge by framing learning as a density estimation problem. By leveraging methods from density estimation, an agent can efficiently learn successor models and scale to high-dimensional state and action spaces (including continuous spaces) with amortized learning procedures (see section 3.2). **Successor features** are another method for circumventing the challenge of representing large state spaces. Here, we do so by describing states with a shared set of state features. Rather than making predictions about how much all states will be visited, we make predictions about how much features will be experienced. Both successor models and successor features have their own pros and cons. Successor models are useful because they open up new possibilities like supporting temporally abstract sampling of future states under a policy. Additionally, methods for learning successor models typically subsume learning of state features, whereas successor features typically need a separate mechanism for learning state features. On the other hand, successor features are easier to learn and more readily enable stitching together policies concurrently (see section 2.5.1) and sequentially (see section 2.5.2) in time—though there is progress on doing this with successor models (see see section 4.2.3).

(the successor model; see section 2.4), and an important generalization (successor features; see section 2.5). A visual overview of these predictive representations is shown in Figure 2. Applications of these concepts are covered in see section 4.

2.3 The Successor Representation. The SR, denoted M^π, was introduced to address the transfer problem described in the previous section (Dayan, 1993; Gershman, 2018). In particular, the SR is well suited for

solving sets of tasks that share the same transition structure but vary in their reward structure; we delve more into this problem setting later when we discuss applications.

Like the value function, the SR is a discounted sum over a quantity of interest known as its cumulant. The value function has reward as its cumulant, whereas the SR has state occupancy as its cumulant:

$$M^{\pi}(s,\tilde{s}) = \mathbb{E}\left[\sum_{t=0}^{H} \gamma^{t}\mathbb{I}[s_{t+1}=\tilde{s}] \,\middle|\, s_0 = s\right]. \tag{2.13}$$

Here $\tilde{s}$ denotes a future state, and $M^{\pi}(s,\tilde{s}) \in \mathbb{R}$ is the expected discounted future occupancy of $\tilde{s}$ starting in state s under policy π. An illustration of the SR and comparison with the value function is shown in Figure 3.

If we define the marginal transition matrix $\mathbf{T}^{\pi} \in \mathbb{R}^{|S|\times|S|}$ by

$$T^{\pi}(s,s') = \sum_{a} \pi(a|s)T(s'|s,a), \tag{2.14}$$

then the SR can be derived analytically from the transition function when $H = \infty$ as

$$\mathbf{M}^{\pi} = \sum_{t=0}^{\infty} \gamma^{t}[\mathbf{T}^{\pi}]^{t+1} = \mathbf{T}^{\pi}(\mathbf{I} - \gamma\mathbf{T}^{\pi})^{-1}, \tag{2.15}$$

where $\mathbf{M}^{\pi} \in \mathbb{R}^{|S|\times|S|}$. Equation 2.15 makes explicit the sense in which the SR (a predictive representation) is a compilation of the transition matrix (a predictive model). The SR discards information about individual transitions, replacing them with their cumulants, analogous to how the value function replaces individual reward sequences with their cumulants.

Like the value function, the SR obeys a Bellman equation:

$$\begin{aligned} M^{\pi}(s,\tilde{s}) &= \sum_{a}\pi(a|s)\sum_{s'}T(s'|s,a)\big[\mathbb{I}[s'=\tilde{s}] + \gamma M^{\pi}(s',\tilde{s})\big] \\ &= \mathbb{E}\big[\mathbb{I}[s'=\tilde{s}] + \gamma M^{\pi}(s',\tilde{s})\big]. \end{aligned} \tag{2.16}$$

This means that it is possible to learn the SR using TD updates similar to the ones applied to value learning:

$$\hat{M}^{\pi}(s,\tilde{s}) \leftarrow \hat{M}^{\pi}(s,\tilde{s}) + \eta\delta_{M}(\tilde{s}), \tag{2.17}$$

where

$$\delta_{M}(\tilde{s}) = \mathbb{I}[s'=\tilde{s}] + \gamma\hat{M}^{\pi}(s',\tilde{s}) - \hat{M}^{\pi}(s,\tilde{s}) \tag{2.18}$$

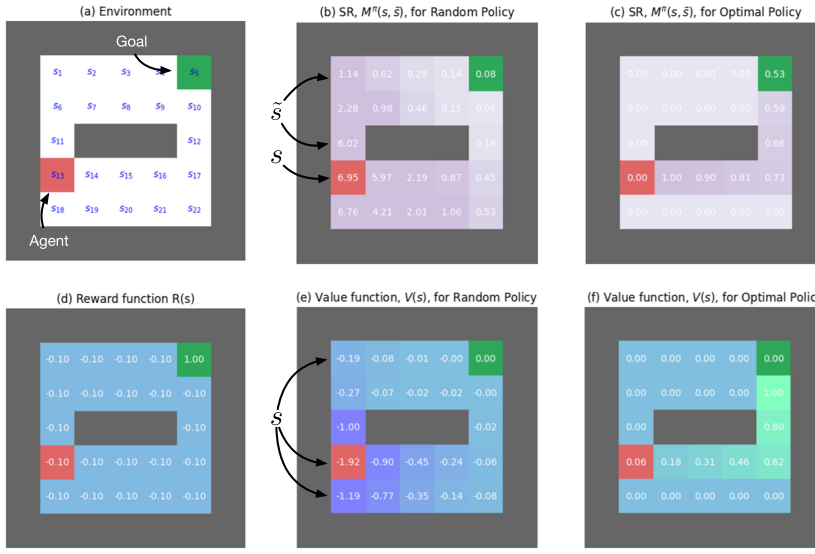

Figure 3: The successor representation (SR). (a) A schematic of an environment where the agent is a red box at state s_{13} and the goal is a green box at state s_5. In general, an SR $M^\pi(s, \tilde{s})$ (see equation 2.13) describes the discounted state occupancy for state $\tilde{s}$ when beginning at state s and following policy π. In panels b and c, we showcase $M^\pi(s_{13}, \tilde{s})$ for a random policy and an optimal policy. (b) The SR under a random policy measures high state occupancy near the agent's current state (e.g., $M^\pi(s_{13}, s_{14}) = 5.97$) and low state occupancy at points farther away from the agent (e.g., $M^\pi(s_{13}, s_{12}) = 0.16$). (c) The SR under the optimal policy has the highest state occupancy along the shortest path to the goal (e.g., here, $M^\pi(s_{13}, s_{12}) = .66$), fading as we get farther from the current state. In contrast to a random policy, states not along that path have 0 occupancy (e.g., $M^\pi(s_{13}, s_{19}) = 0.0$). Once we know a reward function, we can efficiently evaluate both policies (see equation 2.19). (d) An example reward function that has a cost of -0.1 for each state except the goal state where reward is 1. The SR allows us to efficiently compute (e) the value function under a random policy and (f) the value function under the optimal policy.

is the TD error. Notice that unlike in TD learning for value, the error is now vector valued (one error for each state). Once the SR is learned, the value function for a particular reward function under the policy π can be efficiently computed as a linear function of the SR:

$$V^\pi(s) = \sum_{\tilde{s}} M^\pi(s, \tilde{s})\, R(\tilde{s})\,. \tag{2.19}$$

Intuitively, equation 2.19 expresses a decomposition of future reward into immediate reward in each state and the frequency with which those states are visited in the near future.

Just as one can condition a value function on actions to obtain a state-action value function (see equation 2.3), we can condition the SR on actions as well:[5]

$$M^{\pi}(s, a, \tilde{s}) = \mathbb{E}\left[\sum_{t=0}^{H} \gamma^{t} \mathbb{I}[s_{t+1} = \tilde{s}] \,\middle|\, s_0 = s, A_0 = a\right] \tag{2.20}$$

$$= \mathbb{E}\big[\mathbb{I}[s' = \tilde{s}] + \gamma M^{\pi}(s', a', \tilde{s}) \big| s_0 = s, A_0 = a\big]. \tag{2.21}$$

Given an action-conditioned SR, the action value function for a particular reward function can be computed as a linear function of the SR with

$$Q^{\pi}(s, a) = \sum_{\tilde{s}} M^{\pi}(s, a, \tilde{s}) R(\tilde{s}). \tag{2.22}$$

Having established some mathematical properties of the SR, we can now explain why it is useful. First, the SR, unlike model-based algorithms, obviates the need to simulate roll-outs or iterate over dynamic programming updates because it has already compiled transition information into a convenient form: state values can be computed by simply taking the inner product between the SR and the immediate reward vector. Thus, SR-based value computation enjoys efficiency comparable to model-free algorithms.

Second, the SR can, like model-based algorithms, adapt quickly to certain kinds of environmental changes. In particular, local changes to an environment's reward structure induce local changes in the reward function, which immediately propagate to the value estimates when combined with the SR.[6] Thus, SR-based value computation enjoys flexibility comparable to model-based algorithms, at least for changes to the reward structure. Changes to the transition structure, however, require more substantial nonlocal changes to the SR due to the fact that an internal model of the detailed transition structure is not available.

Our discussion has already indicated several limitations of the SR. First, the policydependence of its predictions limits its generalization ability. Second, the SR assumes a finite, discrete state space. Third, it does not generalize to new environment dynamics. When the transition structure changes,

[5] Note that we overload M^{π} to also accept actions to reduce the amount of new notation. In general, $M^{\pi}(s, \tilde{s}) = \sum_a \pi(a|s) M^{\pi}(s, a, \tilde{s})$.

[6] At least initially, these are not exactly the correct value estimates, because the SR is policy dependent, and the policy itself requires updating, which may not happen instantaneously (depending on how the agent is optimizing its policy). Nonetheless, these value estimates will typically be an improvement—a good first guess. As we will see, human learning exhibits similar behavior.

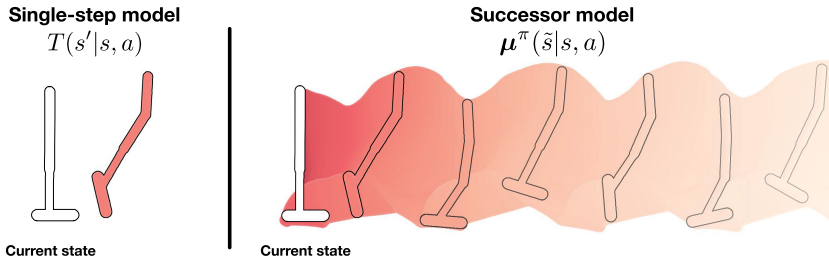

Figure 4: The successor model (SM). A cartoon schematic of a robot leg that can hop forward. Left: A single-step model can only compute likelihoods for states at the next time-step. Right: Multistep successor models can compute likelihoods for states over some horizon into the future. One key difference between the SM and the SR is that the SM defines a valid probability distribution. This means that we can leverage density estimation techniques for learning it over continuous state and action spaces. Additionally, as this figure suggests, we can use it to sample potentially distal states (see section 4.4). Adapted with permission from Janner et al. (2020).

equation 2.15 no longer holds. We discuss in section 3 how the first and second challenges can be addressed and in section 4.2.3 some attempts to address the third challenge.

2.4 Successor Models: A Probabilistic Perspective on the SR. As we've discussed, the SR buys efficiency by caching transition structure while maintaining some model-based flexibility. One thing that is lost, however, is the ability to simulate trajectories through the state space. In this section, we introduce a generalization of the SR—the successor model (SM; Janner et al., 2020; Eysenbach et al., 2020)—that defines an explicit distribution over temporally abstract trajectories (see Figure 4). Here, "temporal abstraction" means a conditional distribution over future states within some time horizon rather than only the next time-step captured by the transition model.

The SM uses a k-step conditional distribution over future occupancy as its cumulant. This k-step conditional distribution $\mathbb{P}(s_k = \tilde{s}|s_0 = s, \pi, T)$ describes the probability that the agent is at state $\tilde{s}$ after it follows policy π for k time steps starting at state s. The SM is then defined as follows:

$$\boldsymbol{\mu}^{\pi}(\tilde{s}|s) = (1-\gamma)\sum_{k=1}^{\infty}\gamma^{k}\mathbb{P}(s_{k+1} = \tilde{s}|s_0 = s, \pi, T), \tag{2.23}$$

where $(1-\gamma)$ ensures that $\boldsymbol{\mu}^{\pi}$ integrates to 1. In the tabular setting, the SM is essentially a normalized SR, since

$$\boldsymbol{\mu}^{\pi}(\tilde{s}|s) = (1-\gamma)M^{\pi}(s,\tilde{s}). \tag{2.24}$$

This relationship becomes apparent when we note that the expectation of an indicator function is the likelihood of the event, that is, $\mathbb{E}[\mathbb{I}[X = x]] = \mathbb{P}(X = x)$. In the general (nontabular) case, the SR and SM are not equivalent; this distinction has practical consequences for learning in realistic problems, where a tabular representation is not feasible.

Since the SM integrates to 1, a key difference to the SR is that it defines a valid probability distribution. This is important because it allows for the SR to generalize to continuous state and action spaces, where we can leverage density estimation techniques for estimating this value on a per state basis. As we will discuss in section 3.2, we can estimate the SM with density estimation techniques such as generative adversarial learning (Janner et al., 2020), variational inference (Thakoor et al., 2022), and contrastive learning (Eysenbach et al., 2020; Zheng et al., 2023).

SMs are interesting because they are a different kind of environment model. Rather than defining transition probabilities over next states, they describe the probability of reaching $\tilde{s}$ within a horizon determined by γ when following policy π. While we don't know exactly when $\tilde{s}$ will be reached, we can answer queries about whether it will be reached within some relatively long time horizon with less computation compared to rolling out the base transition model. Additionally, depending on how the SM is learned, we can also sample from it. This can be useful for policy evaluation and model-based control (Thakoor et al., 2022). We discuss this in more detail in section 4.4.

Like the original SR, the SM obeys a Bellman-like recursion,

$$\boldsymbol{\mu}^{\pi}\left(\tilde{s}|s\right) = \mathbb{E}\left[(1-\gamma)T(\tilde{s}|s,a) + \gamma\boldsymbol{\mu}^{\pi}\left(\tilde{s}|s'\right)\right], \tag{2.25}$$

where the next-state probability $\tilde{s}$ in the first term resembles one-step reward in equation 2.7 and the second term resembles the expected value function at the next time step. As with equation 2.19, we can use the SM to perform policy evaluation by computing

$$V^{\pi}(s) = \frac{1}{1-\gamma}\mathbb{E}_{\tilde{s}\sim\boldsymbol{\mu}^{\pi}(\cdot|s)}\left[R(\tilde{s})\right]. \tag{2.26}$$

Additionally, we can introduce an action-conditioned variant of the SM:

$$\begin{aligned}\boldsymbol{\mu}^{\pi}\left(\tilde{s}|s,a\right) &= (1-\gamma)\sum_{t=0}^{\infty}\gamma^{t}\mathbb{P}\left(s_{t+1}=\tilde{s}|s_0=s,a_0=a,\pi,T\right)\\ &= (1-\gamma)T(\tilde{s}|s,a) + \gamma\mathbb{E}\left[\boldsymbol{\mu}^{\pi}\left(\tilde{s}|s,a\right)\right].\end{aligned} \tag{2.27}$$

We can leverage this to compute an action value:

$$Q^{\pi}(s,a) = \frac{1}{1-\gamma}\mathbb{E}_{\tilde{s}\sim\boldsymbol{\mu}^{\pi}(\cdot|s,a)}\left[R(\tilde{s})\right]. \tag{2.28}$$

2.5 Successor Features: A Feature-Based Generalization of the SR. When the state space is large or unknown, the SR can be challenging to compute or learn because it requires maintaining occupancy expectations over all states. This can also be challenging when dealing with learned state representations, as is common in practical learning algorithms (see section 3.1). In these settings, rather than maintain occupancy measure for states, we can maintain occupancy measure over cumulants that are features shared across states, $\boldsymbol{\phi}(s)$. This generalization of the SR is known as successor features (SFs; Barreto et al., 2017). When $\boldsymbol{\phi}(s)$ is a one-hot vector describing the state the agent is in, SFs are exactly equivalent to the SR. The power of this representation is particularly apparent when reward can be decomposed into a dot product of these successor features and a vector $\mathbf{w}$ describing feature preferences for the current task:

$$R(s,\mathbf{w}) = \boldsymbol{\phi}(s)^{\top}\mathbf{w}. \tag{2.29}$$

SFs are then predictions of accumulated features $\boldsymbol{\phi}$ the agent can expect to encounter when following a policy π:

$$\boldsymbol{\psi}^{\pi}(s) = \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^{t}\boldsymbol{\phi}(s_{t+1}) \mid s_0 = s\right]. \tag{2.30}$$

Like the SR, SFs obey a Bellman equation:

$$\boldsymbol{\psi}^{\pi}(s) = \mathbb{E}\left[\boldsymbol{\phi}(s') + \gamma\boldsymbol{\psi}(s')\right]. \tag{2.31}$$

Under the assumption of equation 2.29, a task-dependent value $V^{\pi}(s,a,\mathbf{w})$ is equivalent to

$$\begin{aligned} Q^{\pi}(s,a,\mathbf{w}) &= \mathbb{E}\left[r(s_1) + \gamma r(s_2) + \ldots \mid s_0 = s, a_0 = a\right] \\ &= \mathbb{E}\left[\boldsymbol{\phi}(s_1)^{\top}\mathbf{w} + \gamma\boldsymbol{\phi}(s_2)^{\top}\mathbf{w} + \ldots \mid s_0 = s, a_0 = a\right] \\ &= \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^{t}\boldsymbol{\phi}(s_{t+1}) \mid s_0 = s, a_0 = a\right]^{\top}\mathbf{w} \\ &= \boldsymbol{\psi}^{\pi}(s)^{\top}\mathbf{w}. \end{aligned} \tag{2.32}$$

As with the SR and SM, we can introduce an action-conditioned variant of SFs:

$$\boldsymbol{\psi}^{\pi}(s,a) = \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^{t}\boldsymbol{\phi}(s_{t+1}) \mid s_0 = s, a_0 = a\right]$$
$$= \mathbb{E}\left[\boldsymbol{\phi}(s') + \gamma\boldsymbol{\psi}(s', a')\right]. \tag{2.33}$$

This provides an avenue to reuse these behavioral predictions for new tasks. Once $\boldsymbol{\psi}^{\pi}(s,a)$ has been learned, the behavior π can be reused to pursue a novel reward function $R(s, \mathbf{w}_{\texttt{new}}) = \boldsymbol{\phi}(s)^{\top}\mathbf{w}_{\texttt{new}}$ by simply changing the corresponding task encoding $\mathbf{w}_{\texttt{new}}$:

$$Q^{\pi}(s,a,\mathbf{w}_{\texttt{new}}) = \boldsymbol{\psi}^{\pi}(s,a)^{\top}\mathbf{w}_{\texttt{new}}. \tag{2.34}$$

As we will see in the next subsection and when we discuss AI applications (see section 4), this becomes even more powerful when combined with an algorithm adaptively combining past policies. We discuss research on successor features over biologically plausible state features in section 5.3.

2.5.1 Generalized Policy Improvement: Adaptively Combining Policies. One limitation of equations 2.19 and 2.26 is that they only enable us to recompute the value of states for a new reward function under a known policy. However, we may want to synthesize a new policy from the other policies we have learned so far. We can accomplish this with SFs by combining them with generalized policy improvement (GPI; Barreto et al., 2017), illustrated in Figure 5.

Assume we have learned (potentially optimal) policies $\{\pi_i\}_{i=1}^{n_{\texttt{train}}}$ and their corresponding SFs $\left\{\boldsymbol{\psi}^{\pi_i}(s,a)\right\}_{i=1}^{n_{\texttt{train}}}$ for $n_{\texttt{train}}$ training tasks $\{\mathbf{w}_i\}_{i=1}^{n_{\texttt{train}}}$. When presented with a new task $\mathbf{w}_{\texttt{new}}$, we can obtain a new policy with GPI in two steps: (1) compute Q-values using the training task SFs and (2) select actions using the highest Q-value. This operation is summarized as follows:

$$a^{*}(s;\mathbf{w}_{\texttt{new}}) = \operatorname*{argmax}_{a}\max_{i\in\{1,\ldots,n_{\texttt{train}}\}}\left\{\boldsymbol{\psi}^{\pi_i}(s,a)^{\top}\mathbf{w}_{\texttt{new}}\right\}$$
$$= \operatorname*{argmax}_{a}\max_{i\in\{1,\ldots,n_{\texttt{train}}\}}\left\{Q^{\pi_i}(s,a,\mathbf{w}_{\texttt{new}})\right\}. \tag{2.35}$$

If $\mathbf{w}_{\texttt{new}}$ is in the span of the training tasks ($\mathbf{w}_{\texttt{new}} = \sum_i \alpha_i \mathbf{w}_i$, where $\alpha_i \in \mathbb{R}$), the GPI theorem states that $\pi(a|s) = \mathbb{I}[a = a^{*}(s,\mathbf{w}_{\texttt{new}})]$ will perform at least as well as any of the training policies—$Q^{\pi}(s,a,\mathbf{w}_{\texttt{new}}) \geq \max_i Q^{\pi_i}(s,a,\mathbf{w}_{\texttt{new}})\,\forall(s,a) \in (\mathcal{S}\times\mathcal{A})$. We discuss how this has been used for transferring policies in artificial intelligence in section 4.2. We discuss empirical evidence that humans transfer policies with GPI in section 6.2.

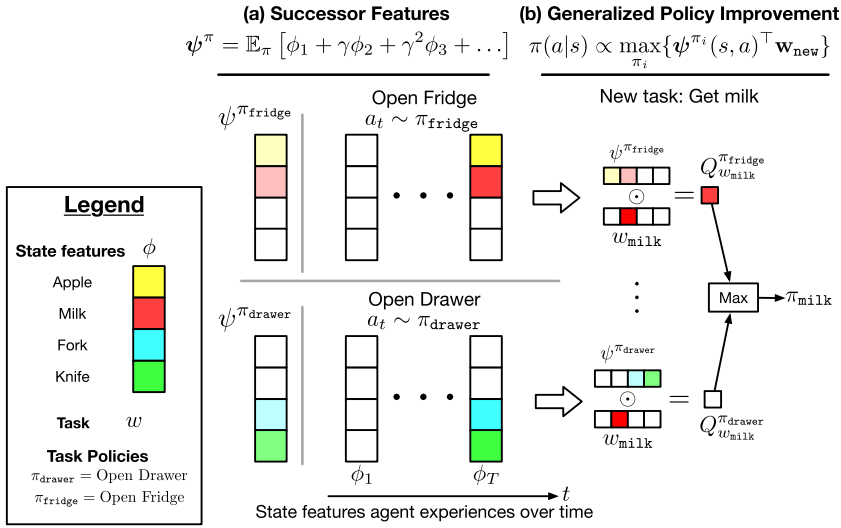

Figure 5: A schematic of Successor features (SFs; see section 2.5) and Generalized policy improvement (GPI; see section 2.5.1). Note that we use the shorthand $\boldsymbol{\phi}_t = \boldsymbol{\phi}(s_t)$ to represent state features that describe what is visible to the agent at time t. π corresponds to policies that the agent knows how to perform. (a) Examples of SFs (see equation 2.30) for the "open drawer" and "open fridge" policies. In this hypothetical scenario, the state features that agent holds describe whether an apple, milk, fork, or knife are present. Beginning for the first time step, the SFs for these policies encode predictions for which of these features will be present when the policies are executed—predicted to be present for apple and milk for the open fridge policy and for the fork and knife when the open drawer policy is executed. (b) The agent can reuse these known policies with GPI (see equation 2.35). When given a new task, say "get milk," it is able to leverage the SFs for the policies to know to decide which behavior will enable it to get milk. In this example, the policy for opening the fridge will also lead to milk. The agent selects actions with GPI by computing Q-values for each known behavior as the dot-product between the current task and each known SF. The highest Q-value is then used to select actions. If the agent wants to execute the option keyboard (OK; see section 2.5.2), they can adaptively set $\mathbf{w}$ based on the current state. For example, at some states, the agent may want to pursue getting milk, while in others, they may want to pursue getting a fork. Adapted from Carvalho et al. (2024) with permission.

2.5.2 Option Keyboard: Chaining Together Policies. One advantage of equation 2.35 is that it facilitates adaptation to linear combinations of training task encodings. However, when transferring to a new task encoding $\mathbf{w}_{\texttt{new}}$, the preferences are constant over time. This becomes problematic when dealing with complex tasks that necessitate different preferences for

Table 1: Summary of Predictive Representations That We Focus On.

Predictive representation	Cumulant	δ: TD update when $a' \sim \pi$ and $s' \sim T$
$Q^\pi(s,a)$ (sec. 2.2)	$r(s)$	$r(s') + \gamma Q^\pi(s',a') - Q^\pi(s,a)$
$M^\pi(s,\tilde{s})$ (SR; sec. 2.3)	$\mathbb{I}(s=\tilde{s})$	$\mathbb{I}(s'=\tilde{s}) + \gamma M^\pi(s',\tilde{s}) - M^\pi(s,\tilde{s})$
$\boldsymbol{\psi}^\pi(s,a)$ (SFs; sec. 2.5)	$\boldsymbol{\phi}(s)$	$\boldsymbol{\phi}(s') + \gamma \boldsymbol{\psi}^\pi(s',a') - \boldsymbol{\psi}^\pi(s,a)$
$\boldsymbol{\mu}^\pi(\tilde{s}\|s,a)$ (SM; sec. 2.4)	$\mathbb{P}(s_{t+1}=\tilde{s}\|s_0,a_0)$	$(1-\gamma)T(s'\|s,a) + \gamma \boldsymbol{\mu}^\pi(\tilde{s}\|s',a') - \boldsymbol{\mu}^\pi(\tilde{s}\|s,a)$

Notes: For each, we also describe the "cumulant" that this predictive representation forms predictions over, along with a corresponding on-policy Bellman update one can use to learn the representation for a policy π. $Q^\pi(s,a)$ is the action-value function, which forms predictions about future reward. $M^\pi(s,\tilde{s})$ is the successor representation (SR), forms predictions about how much a state $\tilde{s}$ will be visited. $\boldsymbol{\psi}^\pi(s,a)$ are successor features (SFs) that form predictions about how much state features $\boldsymbol{\phi}(s)$ will be experienced. $\boldsymbol{\mu}^\pi(\tilde{s}|s,a)$ is the successor model (SM), which predicts the likelihood of experiencing $\tilde{s}$ in the future.

different states—for example, tasks that require both avoidance and approach behaviors at different times.

The Option Keyboard (Barreto et al., 2019, 2020) introduces a solution to this by employing a state-dependent preference vector $\mathbf{w}_s$. This vector is generated through a policy function $g(\cdot)$, which takes as input the current state s and the new task $\mathbf{w}_{\texttt{new}}$, $\mathbf{w}_s = g(s, \mathbf{w}_{\texttt{new}})$. Actions can then be chosen as follows:

$$a^*_{\mathbf{w}_s}(s; \mathbf{w}_{\texttt{new}}) = \operatorname*{argmax}_a \max_{i \in \{1,\ldots,n_{\texttt{train}}\}} \left\{ \boldsymbol{\psi}^{\pi_i}(s,a)^\top \mathbf{w}_s \right\}. \tag{2.36}$$

Note that this policy is adaptive—which known policy π_i is chosen at a time step is dependent on which one maximizes the Q-values at that time step. This is why it's called the Option Keyboard: $\mathbf{w}_s$ will induce a set of policies to be active for a period of time, somewhat like playing a chord on a piano. We discuss how this has been in artificial intelligence applications to chain policies in section 4.3.2.

2.6 Summary. The predictive representations introduced above can be concisely organized in terms of particular cumulants, as summarized in Table 1. These cumulants have different strengths and weaknesses. Value functions (reward cumulants) directly represent the key quantity for RL tasks, but they suffer from poor flexibility. The SR (state occupancy cumulant) and its variations (feature occupancy and state probability cumulants) can be used to compute values but also retain useful information for generalization to new tasks (e.g., using generalized policy improvement).

3 Practical Learning Algorithms and Associated Challenges

Of the predictive representations discussed in section 2, only SFs and SMs have been successfully scaled to environments with large,

high-dimensional state spaces (including continuous state spaces). Thus, these will be our focus of discussion. We first discuss learning SFs in section 3.1 and then learning successor models in section 3.2.

3.1 Learning Successor Features. In this section we discuss practical considerations for learning successor features, including learning a function $\boldsymbol{\phi}_\theta$ that produces cumulants (see section 3.1.1) and estimating SFs $\boldsymbol{\psi}_\theta$ (see section 3.1.2).

3.1.1 Discovering Cumulants. One central challenge to learning SFs is that they require cumulants that are useful for adaptive agent behavior; these are not always easy to define a priori. In most work, these cumulants have been hand-designed, but potentially better ones can be learned from experience. Some general methods for discovering cumulants include leveraging metagradients (Veeriah et al., 2019), discovering features that enable reconstruction (Kulkarni et al., 2016; Machado et al., 2017), and maximizing the mutual information between task encodings and the cumulants that an agent experiences when pursuing that task (Hansen et al., 2019). However, these methods don't necessarily guarantee that the learned cumulants respect a linear relationship with reward (see equation 2.29). To satisfy this, methods typically enforce this by minimizing the L2-norm of their difference (Barreto et al., 2017):

$$\mathcal{L}_r = ||r - \boldsymbol{\phi}_\theta(s)^\top \mathbf{w}||_2^2. \tag{3.1}$$

When learning cumulants that support transfer with GPI, one strategy that can bolster equation 3.1 is to learn an n-dimensional $\boldsymbol{\phi}$ vector for n tasks such that each dimension predicts one task reward (Barreto et al., 2018). Another strategy is to enforce that cumulants describe independent features (Alver & Precup, 2021)—for example, by leveraging a modular architecture with separate parameters for every cumulant dimension (Carvalho et al., 2023) or by enforcing sparsity in the cumulant dimensions (Filos et al., 2021). We discuss research on biologically plausible state features in section 5.3.

3.1.2 Estimating Successor Features.

Learning an estimator that can generalize across policies. The first challenge for learning SFs is that they are defined for a particular policy π. We can mitigate this by learning *universal successor feature approximators* (USFAs; Borsa et al., 2019), which takes as input policy encodings $\mathbf{z}_\mathbf{w} \in \mathcal{Z}_\pi$, which represent an encoding of the policy $\pi_\mathbf{w}$ that is reward maximizing for task $\mathbf{w}$. Concretely, we can estimate $\boldsymbol{\psi}^{\pi_\mathbf{w}}(s, a)$ as

$$\boldsymbol{\psi}^{\pi_\mathbf{w}}(s, a) \approx \boldsymbol{\psi}_\theta(s, a, \mathbf{z}_\mathbf{w}). \tag{3.2}$$

This has several benefits. First, one can share the estimator parameters across policies, which can improve learning. Second, this allows SFs to generalize across different policies. Leveraging a USFA, the GPI operation in equation 2.35 can be adapted to perform a max operation over $\mathcal{Z}_\pi$:

$$a^*(s; \mathbf{w}_{\texttt{new}}) = \arg\max_{a \in \mathcal{A}} \max_{\mathbf{z}_\mathbf{w} \in \mathcal{Z}_\pi} \left\{ \boldsymbol{\psi}_\theta(s, a, \mathbf{z}_\mathbf{w})^\top \mathbf{w}_{\texttt{new}} \right\}. \tag{3.3}$$

Several algorithms exploit this property (see sections 4.1.1, 4.2.2, and 4.5). If the policy encoding space is equivalent to the n training tasks, $\mathcal{Z}_\pi = \{\mathbf{w}_1, \ldots, \mathbf{w}_n\}$, this recovers the original GPI operation (see equation 2.35). In practice, it is common to identify the policy encoding $\mathbf{z}_\mathbf{w}$ for a task with its task encoding $\mathbf{w}$, that is, to use $\mathbf{z}_\mathbf{w} = \mathbf{w}$.

Learning successor features while learning a policy. Often SFs need to be learned simultaneously with policy optimization. This requires the SFs to be updated along with the policy. One strategy is to simultaneously learn an action-value function $Q_\theta(s, a, \mathbf{w}) = \boldsymbol{\psi}_\theta(s, a, \mathbf{z}_\mathbf{w})^\top \mathbf{w}$ that is used to select actions. One can accomplish this with Q-learning over values defined by the SFs and task encoding. Q-values are updated toward a target $\mathbf{y}_Q$ defined as the sum of the reward $R(s'; \mathbf{w})$ and the best next Q-value. We define the learning objective $\mathcal{L}_Q$ as follows:

$$\mathbf{y}_Q = R(s'; \mathbf{w}) + \gamma \boldsymbol{\psi}_\theta(s', a^*, \mathbf{z}_\mathbf{w})^\top \mathbf{w} \quad \mathcal{L}_Q = ||\boldsymbol{\psi}_\theta(s, a, \mathbf{z}_\mathbf{w})^\top \mathbf{w} - \mathbf{y}_Q||, \tag{3.4}$$

where $a^* = \text{argmax}_{a'} \boldsymbol{\psi}_\theta(s', a', \mathbf{z}_\mathbf{w})^\top \mathbf{w}$ is the action that maximizes features determined by $\mathbf{w}$ at the next time step. To ensure that these Q-values follow the structure imposed by SFs (see equation 2.32), we additionally update SFs at a state with a target $\mathbf{y}_\psi$ defined as the sum of the cumulant $\boldsymbol{\phi}(s')$ and the SFs associated with the best Q-value at the next state:

$$\mathbf{y}_\psi = \boldsymbol{\phi}(s') + \gamma \boldsymbol{\psi}_\theta(s', a^*, \mathbf{z}_\mathbf{w}) \qquad \mathcal{L}_\psi = ||\boldsymbol{\psi}_\theta(s, a, \mathbf{z}_\mathbf{w}) - \mathbf{y}_\psi||. \tag{3.5}$$

In an online setting, it is important to learn SFs with data collected by a policy that chooses actions with high Q-values. This is especially important if the true value is lower than the estimated Q-value. Because Q-learning leverages the maximum Q-value when doing backups, it has a bias for overestimating value. This can destabilize learning, particularly in regions of the state space that have been less explored (Ostrovski et al., 2021).

Another strategy to stabilize SF learning is to learn individual SF dimensions with separate modules (Carvalho et al., 2023, 2024). Beyond stabilizing learning, this modularity also enables approximating SFs that generalize better to novel environment configurations (i.e., which are more robust to novel environment dynamics).

Estimating successor features with changing cumulants. In some cases, the cumulant itself will change over time (e.g., when the environment is nonstationary). This is challenging for SF learning because the prediction target

is nonstationary (Barreto et al., 2018). This is an issue even when the environment is stationary but the policy is changing over time: different policies induce different trajectories and different state features induce different descriptions of those trajectories.

One technique that has been proposed to facilitate modeling a nonstationary cumulant trajectory is to learn an SF as a probability mass function (pmf) $p(\boldsymbol{\psi}^{(k)}|s, a, \mathbf{w})$ defined over some set of possible values $\mathbb{B} = \{b_1, \ldots, b_M\}$. Specifically, we can estimate an n-dimensional SF $\boldsymbol{\psi}^{\pi_\mathbf{w}}(s, a) \in \mathbb{R}^n$ with $\boldsymbol{\psi}_\theta(s, a, \mathbf{z}_\mathbf{w})$ and represent the kth SF dimension as $\boldsymbol{\psi}^k_\theta(s, a, \mathbf{z}_\mathbf{w}) = \sum_{m=1}^{M} p(\boldsymbol{\psi}^{(k)} = b_m|s, a, \mathbf{z}_\mathbf{w})b_m$. We can then learn SFs with a negative log-likelihood loss where we construct categorical target labels $\mathbf{y}_{\boldsymbol{\psi}^{(k)}}$ from the return associated with the optimal Q-value (Carvalho et al., 2024):

$$\mathbf{y}_{\boldsymbol{\psi}^{(k)}} = \boldsymbol{\phi}^k_\theta(s') + \gamma \boldsymbol{\psi}^k_\theta(s', a^*, \mathbf{z}_\mathbf{w}), \tag{3.6}$$

$$\mathcal{L}_{\boldsymbol{\psi}} = -\sum_{k=1}^{n} f_{\texttt{label}}(\mathbf{y}_{\boldsymbol{\psi}^{(k)}})^\top \log p(\boldsymbol{\psi}^{(k)}|s, a, \mathbf{z}_\mathbf{w}). \tag{3.7}$$

Prior work has found that the two-hot representation is a good method for defining $f_{\texttt{label}}(\cdot)$ (Carvalho et al., 2024; Schrittwieser et al., 2020). In general, estimating predictive representations such as SFs with distributional losses such as equation 3.6 has been shown to reduce the variance in learning updates (Imani & White, 2018). This is particularly important when cumulants are being learned, as this can lead to high variance in $\mathbf{y}_{\boldsymbol{\psi}^{(k)}}$.

3.2 Learning Successor Models. In this section, we focus on estimating $\boldsymbol{\mu}^\pi(\tilde{s}|s, a)$ with $\boldsymbol{\mu}_\theta(\tilde{s}|s, a)$. In a tabular setting, one can leverage TD-learning with the Bellman equation in equation 2.25. However, for very large state spaces (such as with infinite size continuous state spaces), this is intractable or impractical. Depending on one's use case, different options exist for learning. First, we discuss the setting where one wants to learn an SM they can sample from (see section 3.2.1). Then we discuss the setting where one only wants to evaluate an SM for different actions given a target state (see section 3.2.2).

3.2.1 Learning Successor Models That One Can Sample From. Learning an SM that can be sampled from is useful for evaluating a policy (equation 2.26), evaluating a sequence of policies (Thakoor et al., 2022), and in model-based control (Janner et al., 2020). There are two ways to learn such SMs: adversarial learning and density estimation (Janner et al., 2020). Adversarial learning has been found to be unstable, so we focus on density estimation, where the objective is to find parameters θ that maximize the

log-likelihood of states sampled from $\boldsymbol{\mu}^\pi$:

$$\mathcal{L}_\mu = \mathbb{E}_{\substack{(s,a)\sim p(s,a) \\ \tilde{s}\sim \boldsymbol{\mu}^\pi(\cdot|s,a)}} \left[\log \boldsymbol{\mu}_\theta\left(\tilde{s} \mid s, a\right)\right]. \tag{3.8}$$

We can optimize this objective as follows. When sampling targets, we need to sample $\tilde{s}$ in proportion to the discount factor γ. We can accomplish this by first sampling a time step from a geometric distribution, $k \sim \text{Geom}(1-\gamma)$, and then selecting the state at that time step, s_{t+k}, as the target $\tilde{s}$.

While this is a simple strategy, it has several challenges. For values of γ close to 1, this becomes a challenging learning problem requiring predictions over very long time horizons. Another challenge is that you are using s_{t+k} obtained under policy π. In practice, we may want to leverage data collected under a different policy. This happens when, for example, we want to learn from a collection of different data sets, or we are updating our policy over the course of learning. Learning from such off-policy data can lead to high bias, or a high variance learning update from off-policy corrections (Precup et al., 2000).

We can circumvent these challenges as follows. First let's define a Bellman operator T^π:

$$(T^\pi \boldsymbol{\mu}^\pi)(\tilde{s}|s,a) = (1-\gamma)T(s'|s,a) + \gamma \sum_{s'} T(s'|s,a) \sum_{a'} \pi(a'|s') \boldsymbol{\mu}(\tilde{s}|s',a'). \tag{3.9}$$

With this we can define a cross-entropy temporal-difference loss (Janner et al., 2020):

$$\mathcal{L}_\mu = \mathbb{E}_{\substack{(s,a)\sim p(s,a) \\ \tilde{s}\sim (T^\pi \boldsymbol{\mu}^\pi)(\cdot|s,a)}} \left[\log \boldsymbol{\mu}_\theta\left(\tilde{s} \mid s, a\right)\right]. \tag{3.10}$$

Intuitively, $(T^\pi \boldsymbol{\mu}^\pi)(\cdot|s,a)$ defines a random variable obtained as follows. First, sample $s' \sim T(\cdot|s,a)$. Terminate and emit s' with probability $(1-\gamma)$. Otherwise, sample $a' \sim \pi(a'|s')$, and then sample $\tilde{s} \sim \boldsymbol{\mu}^\pi(\cdot|s',a')$.

The most recent promising method for learning equation 3.10 has been to leverage a variational autoencoder (Thakoor et al., 2022). Specifically, we can define an approximate posterior $q_\psi(z|s,a,\tilde{s})$ and then optimize the following evidence lower bound:

$$\mathcal{L}_\mu = \mathbb{E}_{\substack{(s,a)\sim p(s,a) \\ \tilde{s}\sim (T^\pi \boldsymbol{\mu}^\pi)(\cdot|s,a) \\ z\sim q_\psi(\cdot|s,a,\tilde{s})}} \left[\log \frac{\boldsymbol{\mu}_\theta\left(\tilde{s} \mid s, a, z\right)}{q_\psi(z|s,a,\tilde{s})}\right]. \tag{3.11}$$

While Thakoor et al. (2022) were able to scale their experiments to slightly more complex domains than Janner et al. (2020), their focus was on composing policies via geometric policy composition (discussed more in section 4.2.3), so it is unclear how well their method performs in more complex domains. The key challenge for this line of work is in sampling

from $\boldsymbol{\mu}^\pi(\cdot \mid s, a)$, where $\tilde{s}$ can come from a variable next time step after (s, a). In the next section, we discuss methods that address this challenge.

3.2.2 Learning Successor Models That One Can Evaluate. Sometimes we may not need to learn an SM that we can sample from, only one that we can evaluate. This can be used, for example, to generate and improve a policy that achieves some target state (Eysenbach et al., 2020; Zheng et al., 2023). One strategy is to learn a classifier p_θ^μ that, given (s, a), computes how likely $\tilde{s}$ is compared to some set of N random (negative) states $\{s_i^-\}_{i=1}^N$ the agent has experienced:

$$p_\theta^\mu(\tilde{s} \mid s, a, s_{1:N}^-) = \frac{\exp(f_\theta(s, a, \tilde{s}))}{\exp(f_\theta(s, a, \tilde{s})) + \sum_{i=1}^N \exp(f_\theta(s, a, s_i^-))}. \tag{3.12}$$

The function f_θ can be chosen to maximize classification accuracy across random states and actions in the agent's experience, $(s, a) \sim p(s, a)$, target states $\tilde{s}$ drawn from the empirical successor model distribution, $\tilde{s} \sim \boldsymbol{\mu}^\pi(\tilde{s} \mid s, a)$, and negatives drawn from the state marginal, $s_{1:N}^- \sim p(s)$:

$$L_{\boldsymbol{\mu}} = \mathbb{E}_{\substack{(s,a)\sim p(s,a) \\ \tilde{s}\sim \boldsymbol{\mu}^\pi(\tilde{s}|s,a) \\ s_{1:N}^-\sim p(s)}} \left[\log p_\theta^\mu(\tilde{s} \mid s, a, s_{1:N}^-)\right]. \tag{3.13}$$

If we can find such an f_θ, then the resulting classifier is approximately equal to the density ratio between $\boldsymbol{\mu}^\pi(\tilde{s} \mid s, a)$ and the state marginal $p(\tilde{s})$ (Poole et al., 2019; Zheng et al., 2023):

$$\frac{\boldsymbol{\mu}^\pi(\tilde{s} \mid s, a)}{p(\tilde{s})} \approx (N + 1) \cdot p_\theta^\mu(\tilde{s} \mid s, a, s_{1:N}^-). \tag{3.14}$$

Optimizing equation 3.13 is challenging because it requires sampling from $\boldsymbol{\mu}^\pi$. We can circumvent this by instead learning the following TD-like objective, where we replace sampling from $\boldsymbol{\mu}^\pi$ with sampling from the state marginal and reuse p_θ^μ as an importance weight:

$$\begin{aligned} L_{\boldsymbol{\mu}} = \mathbb{E}_{\substack{(s,a)\sim p(s,a) \\ s'\sim T(\cdot|s,a) \\ a'\sim \pi(a|s') \\ \tilde{s}\sim p(s) \\ s_{1:N}^-\sim p(s)}} & \left[(1-\gamma)\log p_\theta^\mu(s'|s, a, s_{1:N}^-)\right. \\ & \left. + \gamma(N+1)p_\theta^\mu(\tilde{s}|s', a', s_{1:N}^-)\log p_\theta^\mu(\tilde{s}|s, a, s_{1:N}^-)\right]. \end{aligned} \tag{3.15}$$

Zheng et al. (2023) show that (under some assumptions) optimizing equation 3.15 leads to the following Bellman-like update:

$$p_\theta^\mu(\tilde{s}|s, a, s_{1:N}^-) \leftarrow (1-\gamma)T(s' = \tilde{s}|s, a) + \gamma \mathbb{E}_{\substack{s'\sim T(\cdot|s,a) \\ a'\sim \pi(\cdot|s')}} \left[p_\theta^\mu(\tilde{s}|s', a', s_{1:N}^-)\right], \tag{3.16}$$

which resembles the original SM Bellman equation (see equation 2.25). However, one key difference to equation 2.25 is that we parameterize $p^{\mu}_{\theta}(\tilde{s}|s, a, s^{-}_{1:N})$ with $N + 1$ random samples (e.g., from a replay mechanism), which is a form of contrastive learning. This provides a nonparametric algorithm for learning SMs. With the SR, one performs the TD update for all states (see equation 2.18); here, one performs this update using a random sample of $N + 1$ states.

We can define f_θ as the dot product between a predictive representation $\varphi_\theta(\cdot, \cdot)$ and label representation $\phi_\theta(\cdot)$, $f_\theta(s, a, \tilde{s}) = \varphi_\theta(s, a)^\top \phi_\theta(\tilde{s})$. $\phi_\theta(s)$ can then be thought of as state features analogous to SFs (see section 2.5). $\varphi_\theta(s, a)$ is a prediction of these future features similar to SFs, $\boldsymbol{\psi}_\theta(s, a)$, with labels coming from future states; however, it doesn't necessarily have the same semantics as a discounted sum (that is, equation 2.32). We use similar notation because of their conceptual similarity. We can then understand equation 3.15 as doing the following. The first term in this objective pushes the prediction $\varphi_\theta(s, a)$ toward the features at the next time step $\phi_\theta(s')$, and the second term pushes $\varphi_\theta(s, a)$ toward the features at arbitrary state features $\phi_\theta(\tilde{s})$. Both terms repel $\varphi_\theta(s, a)$ from arbitrary "negative" state features $\phi_\theta(\tilde{s^-_i})$.

4 Artificial Intelligence Applications

In this section, we discuss how the SR and its generalizations have enabled advances in artificial agents that learn and transfer policies.

4.1 Exploration.

4.1.1 Pure Exploration.

Learning to explore and act in the environment before exposure to reward. In the "pure exploration" setting, an agent can explore its environment for some period of time without external reward. In some cases, the goal is to learn a policy that can transfer to an unknown task $\mathbf{w}_{\texttt{new}}$. SFs can be used to achieve such transfer.

One useful property of SFs is that they encode predictions about what features one can expect when following a policy. Before reward is provided, this can be used to reach different parts of the state space with different policies. One strategy is to associate different parts of the state space with different parts of a high-dimensional task embedding space (Hansen et al., 2019). At the beginning of each episode, an agent samples a goal encoding $\mathbf{w}$ from a high-entropy task distribution $p(\mathbf{w})$. During the episode, the agent selects actions that maximize the features described by $\mathbf{w}$ (e.g., with equation 2.32). As it does this, it learns to predict $\mathbf{w}$ from the states it encounters. If $p(\mathbf{w})$ is parameterized as a von Mises distribution, the agent can learn this prediction by simply maximizing the dot product between the state features $\boldsymbol{\phi}(s)$

and goal encoding $\mathbf{w}$:

$$\mathcal{L}_{\boldsymbol{\phi}} = \log p_\theta(\mathbf{w}|s) = \boldsymbol{\phi}_\theta(s)^\top \mathbf{w}. \tag{4.1}$$

This is equivalent to maximizing the mutual information between $\boldsymbol{\phi}_\theta(s)$ and $\mathbf{w}$. As the agent learns cumulants, it learns SFs as usual (e.g., with equation 3.5). Thus, one can essentially use a standard SF learning algorithm and simply replace the cumulant discovery loss (e.g., equation 3.1) with equation 4.1. Once the agent is exposed to task reward $\mathbf{w}_{\texttt{new}}$, it can freeze $\boldsymbol{\phi}_\theta$ and solve for $\mathbf{w}_{\texttt{new}}$ (e.g., with equation 3.1). The agent can then use GPI to find the best policy for achieving $\mathbf{w}_{\texttt{new}}$ by searching over a gaussian ball defined around $\mathbf{w}_{\texttt{new}}$. This is equivalent to setting $Z = \{z_i \sim \mathcal{N}(\mathbf{w}_{\texttt{new}}, \sigma)\}_{i=1}^n$ for equation 3.3, where σ defines the size of the ball.

Hansen et al. (2019) leveraged this strategy to develop agents that could explore Atari games without any reward for 250 million time steps and then have 100,000 time steps to earn reward. They showed that this strategy was able to achieve superhuman performance across most Atari games, despite not observing any rewards for most of its experience. Liu and Abbeel (2021) improved on this algorithm by adding an intrinsic reward function that favors exploring parts of the state space that are surprising (i.e., that induce high entropy) given a memory of the agent's experience. This dramatically improved sample efficiency for many Atari games.

Exploring the environment by building a map. One strategy for exploring large state spaces systematically is to build a map of the environment defined over landmarks in the environment. With such a map, an agent can systematically explore by planning paths toward the frontier of its knowledge (Ramesh et al., 2019; Hoang et al., 2021). Numerous questions arise in this process. How do we define good landmarks? How do we define policies that traverse between landmarks? How does an agent identify that it has made progress between landmarks after it has set a plan or course-correct if it finds that it accidentally deviated? Hoang et al. (2021) developed an elegant solution to all of these problems with the successor feature similarity (SFS) metric. This metric defines the closeness of two states by the frequency with which an agent starting in each state visits the same parts of the environment. Concretely:

$$S_{\boldsymbol{\psi}}(s_1, a, s_2) = \boldsymbol{\psi}^{\bar{\pi}}(s_1, a)^\top \mathbb{E}_{A\sim\bar{\pi}}[\boldsymbol{\psi}^{\bar{\pi}}(s_2, A)], \tag{4.2}$$

$$S_{\boldsymbol{\psi}}(s_1, s_2) = \mathbb{E}_{A\sim\bar{\pi}}[\boldsymbol{\psi}^{\bar{\pi}}(s_1, a)^\top] \mathbb{E}_{A\sim\bar{\pi}}[\boldsymbol{\psi}^{\bar{\pi}}(s_2, A)], \tag{4.3}$$

where $\bar{\pi}$ is a uniform policy.[7]

Through only learning of SFs over pretrained cumulants $\boldsymbol{\phi}$, Hoang et al. (2021) were able to address all the needs above by exploiting SFS. Let $\mathcal{L} \subset \mathcal{S}$

[7] Note that to avoid excessive notation, we've overloaded the definition of S_ψ.

be the set of landmarks discovered so far. The algorithm works as follows. Each landmark $L \in \mathcal{L}$ has associated with it a count $N(L)$ for how often has been visited. A "frontier" landmark L_F is sampled in proportion to the inverse of this count. The agent makes a shortest-path plan of subgoals $(s_1^*, \ldots, s_n^*)$ toward this landmark where $s_n = L_F$. In order to navigate to the next subgoal s_i^*, it defines a policy with the action of the current successor features that are most aligned with the goal's successor features, $a^*(s, s_i^*) = \text{argmax}_a S_{\boldsymbol{\psi}}(s, a, s_i^*)$. As it traverses toward the landmark, it localizes itself by comparing the current state s to known landmarks $f_{\texttt{loc}}(s, \mathcal{L}) = \text{argmax}_{L \in \mathcal{L}} S_{\boldsymbol{\psi}}(s, L)$. When $s_i^* = f_{\texttt{loc}}(s, \mathcal{L})$, it has reached the next landmark and moves on to the next subgoal. Once a frontier landmark is found, the agent explores with random actions. The states it experiences are added to its replay buffer and used in two ways. First, they update the agent's current SR. Second, if a state is sufficiently different from known landmarks, it is added. In summary, the three key functions that one can compute without additional learning are:

$$\pi(s, s_i^*) = \underset{a}{\text{argmax}}\, S_{\boldsymbol{\psi}}(s, a, s_i^*) \qquad \text{goal-conditioned policy}$$

$$f_{\texttt{loc}}(s, \mathcal{L}) = \underset{L \in \mathcal{L}}{\text{argmax}}\, S_{\boldsymbol{\psi}}(s, L) \qquad \text{localization function}$$

$$f_{\texttt{add}}(s, \mathcal{L}) = \mathcal{L} \leftarrow \mathcal{L} \cup s \quad \text{if} \quad (S_{\boldsymbol{\psi}}(s, L) < \epsilon_{\texttt{add}})\, \forall L \in \mathcal{L}$$

landmark addition function

The process then repeats. This exploration strategy was effective in exploring both minigrid (Chevalier-Boisvert et al., 2023) and the partially observable 3D VizDoom environments (Kempka et al., 2016).

4.1.2 Balancing Exploration and Exploitation.

Cheap uncertainty computations. Balancing exploration and exploitation is a central challenge in RL (Sutton & Barto, 2018; Kaelbling et al., 1996). One method that provides close to optimal performance in tabular domains is posterior sampling (also known as Thompson sampling, after Thompson, 1933), where an agent updates a posterior distribution over Q-values and then chooses the value-maximizing action for a random sample from this distribution (see Russo et al., 2018, for a review). The main difficulties for implementing posterior sampling are associated with representing, updating, and sampling from the posterior when the state space is large. Janz et al. (2019) showed that SFs enable cheap methods for posterior sampling. They assume the following prior, likelihood, and posterior for the environment reward:

$$\mathbf{w} \sim \mathcal{N}(0, \mathbf{I}) \qquad r|\mathbf{w} \sim \mathcal{N}(\boldsymbol{\phi}(s)^\top \mathbf{w}, \beta) \qquad \mathbf{w}|r, s \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{w}}, \Sigma_{\mathbf{w}}), \quad (4.4)$$

where β is the variance of reward around a mean that is linear in the state feature $\boldsymbol{\phi}(s)$; $\boldsymbol{\mu}_{\mathbf{w}}$ and $\Sigma_{\mathbf{w}}$ are given by analytical solutions for the mean and variance of the posterior given the gaussian prior/likelihood assumptions and a set of observations. The posterior distribution for the Q-values is then given by

$$\hat{Q}^{\pi}_{\mathrm{SU}} \sim \mathcal{N}\left(\Psi^{\pi}\boldsymbol{\mu}_{\mathbf{w}},\, \Psi^{\pi}\Sigma_{\mathbf{w}}\left(\Psi^{\pi}\right)^{\top}\right), \tag{4.5}$$

where $\Psi^{\pi} = [\boldsymbol{\psi}^{\pi}(s,a)]^{\top}_{(s,a)\in\mathcal{S}\times\mathcal{A}}$ is a matrix with each row corresponding to an SF for a state-action pair. SFs, cumulants, and task encodings are learned with standard losses (e.g., see equations 3.1 and 3.5).

Count-based exploration. Another method that provides (near) optimal exploration in tabular settings is count-based exploration with a bonus of $1/\sqrt{N(s)}$ (Auer, 2002), where $N(s)$ is the number of times a state s has been visited. When the state space is large, it can be challenging to track this count. Machado et al. (2020) showed that that the L1-norm of SFs is proportional to visitation count and can therefore be used as an exploration bonus:

$$R^{\mathtt{int}}(s) = \frac{1}{||\boldsymbol{\psi}^{\pi}(s)||_1}. \tag{4.6}$$

With this exploration bonus, they were able to improve exploration in sparse-reward Atari games such as *Montezuma's Revenge*. Recent work has built on this idea: Yu et al. (2023) combined SFs with predecessor representations, which encode retrospective information about the agent's trajectory. This ideas, motivated by work in neuroscience (Namboodiri & Stuber, 2021), were shown to more efficiently target exploration toward bottleneck states (i.e., access points between large regions of the state space).

4.2 Transfer. We've already introduced the idea of cross-task transfer in our discussion of GPI. We now review the broader range of ways in which the challenges of transfer have been addressed using predictive representations.

4.2.1 Transferring Policies between Tasks. We first consider transfer across tasks that are defined by different reward functions. In the following two sections, we consider other forms of transfer.

Few-shot transfer between pairs of tasks. SFs can enable transferring policies from one reward function R to another function $R_{\mathtt{new}}$ by exploiting equation 2.34 with learned cumulant $\boldsymbol{\phi}_{\theta}$ and SFs $\boldsymbol{\psi}_{\theta}$ for a source task. At transfer time, one freezes each set of parameters and solves for $\mathbf{w}_{\mathtt{new}}$ (e.g., with equation 3.1). Kulkarni et al. (2016) showed that this enabled an RL agent that learned to play an Atari game to transfer to a new version of that game

where the reward function was scaled. Later, Zhu et al. (2017) showed that this enabled transfer to new "go to" tasks in a photorealistic 3D household environment.

Continual learning across a set of tasks. Beyond transferring across task pairs, an agent may want to continually transfer its knowledge across a set of tasks. Barreto et al. (2017) showed that SFs and GPI provide a natural framework to do this. Consider sequentially learning n tasks. As the agent learns new tasks, they maintain a growing library of SFs $\{\boldsymbol{\psi}^{\pi_i}\}_{i=1}^{m}$ where $m < n$ is the number of tasks learned so far. When the agent is learning the mth task, they can select actions with GPI according to equation 2.35 using $\mathbf{w}_m$ as the current transfer task. The agent learns SFs for the current task according to equation 3.5. Zhang et al. (2017) extended this approach to enable continual learning when the environment state space and dynamics were changing across tasks but still relatively similar. Transferring SFs to an environment with a different state space requires leveraging new state features. Their solution involved reusing old state features by mapping them to the new state space with a linear projection. By exploiting linearity in the Q-value decomposition (see equation 2.34), this allowed reusing SFs for new environments.

Zero-shot transfer to task combinations. Another benefit of SFs and GPI is that they facilitate transfer to task conjunctions when they are defined as weighted sums of training tasks, $\mathbf{w}_{\texttt{new}} = \sum_{i=1}^{n} \alpha_i \mathbf{w}_i$. A clear example is combining "go to" tasks (Barreto et al., 2018, 2020; Borsa et al., 2019; Carvalho et al., 2023). For example, consider four tasks $\{\mathbf{w}_1, \ldots, \mathbf{w}_4\}$ defining by collecting different object types; $\mathbf{w}_{\texttt{new}} = (-1) \cdot \mathbf{w}_1 + 2 \cdot \mathbf{w}_2 + 1 \cdot \mathbf{w}_3 + 0 \cdot \mathbf{w}_4$ defines a new task that tries to avoid collecting objects of type 1, while trying to collect objects of type 2 twice as much as objects of type 3. This approach has been extended to combining policies with continuous state and action spaces (Hunt et al., 2019), though it has so far been limited to combining only two policies. Another important limitation of this general approach is that it can only specify which tasks to prioritize but cannot specify an ordering for these tasks. For example, there is no way to specify collecting object type 1 before object type 2. One can address this limitation by learning a state-dependent transfer task encoding as with the Option Keyboard (see section 2.5.2).

4.2.2 Learning about Nontask Goals. In the previous section, we discussed the transfer setting where an agent learns about a task $\mathbf{w}$ and then subsequently wants to transfer this knowledge to another task $\mathbf{w}_{\texttt{new}}$. In this section, we consider an agent that is learning task $\mathbf{w}$ and wants to concurrently learn policies for other tasks (defined by $\tilde{\mathbf{w}}$). That is, each experience trying to accomplish $\mathbf{w}$ is reused to learn how to accomplish $\tilde{\mathbf{w}}$. This can broadly be categorized as off-task learning.

Borsa et al. (2019) showed that one can reuse experiences accomplishing task $\mathbf{w}$ to learn control policies for tasks $\tilde{\mathbf{w}}$ that are not too far from $\mathbf{w}$ in the

task encoding space by leverage universal SFs. In particular, nearby off-task goals can be sampled from a gaussian ball around $\mathbf{w}$ with standard deviation σ: $Z = \{\tilde{\mathbf{w}}_i \sim \mathcal{N}(\mathbf{w}_{\texttt{new}}, \sigma)\}_{i=1}^{n}$. Then an SF loss following equation 3.5 would be applied for each nontask goal $\tilde{\mathbf{w}}_i$. Key to this is that the optimal action for each $\tilde{\mathbf{w}}_i$ would be the action that maximized the features determined by $\tilde{\mathbf{w}}_i$ at the next time step: $a^* = \text{argmax}'_a \boldsymbol{\psi}_\theta(s', a', \tilde{\mathbf{w}}_i)^\top \tilde{\mathbf{w}}_i$. This enabled an agent to concurrently learn a policy for not only $\mathbf{w}$ but also for nontask goals $\tilde{\mathbf{w}}_i$ with no direct experience on those tasks in a simple 3D navigation environment.

Another example where off-task learning is useful is in hindsight experience replay. Typically, experiences that don't accomplish a task $\mathbf{w}$ don't contribute to learning unless some form of reward shaping is employed. Hindsight experience replay provides a strategy for automating reward shaping. In this setting, when the agent fails to accomplish $\mathbf{w}$, it relabels one of the states in its experience as a fictitious goal $\tilde{\mathbf{w}}$ for that experience (Andrychowicz et al., 2017). This strategy is particularly effective when tasks have sparse rewards as it leads there to be a dense reward signal. When learning a policy with SMs (see section 2.4), hindsight experience replay naturally arises as part of the learning objective. It has been shown to improve sample efficiency in sparse-reward virtual robotic manipulation domains and long-horizon navigation tasks (Eysenbach et al., 2020, 2022; Zheng et al., 2023). Despite their potential, learning and exploiting SMs is still in its infancy, whereas SFs have been more thoroughly studied. Recently, Schramm et al. (2023) developed an asymptotically unbiased importance sampling algorithm that leverages SFs to remove bias when estimating value functions with hindsight experience replay. This enabled learning for both simulation and real-world robotic manipulation tasks in environments with large state and action spaces.

4.2.3 Other Advances in Transfer.

Generalization to new environment dynamics. One limitation of SFs is that they're tied to the environment dynamics T with which they were learned. Lehnert and Littman (2020) and Han and Tschiatschek (2021) both attempt to address this limitation by learning SFs over state abstractions that respect bisimulation relations (Li et al., 2006). Abdolshah et al. (2021) attempt to address this by integrating SFs with gaussian processes such that they can be quickly adapted to new dynamics given a small amount of experience in the new environment.

Synthesizing new predictions from sets of SFs. While GPI enables combining a set of SFs to produce a novel policy, it does not generate a novel prediction of what features will be experienced from a combination of policies. Some methods attempt to address this by convex combination of SFs (Brantley et al., 2021; Alegre et al., 2022).

Alternatives to generalized policy improvement. Madarasz and Behrens (2019) develop the gaussian SF, which learns a set of reward maps for

different environments that can be adaptively combined to adjudicate between different policies. While this compared favorably to GPI, these results were in toy domains; it is currently unclear if their method scales to more complex settings as gracefully as GPI. A potentially more promising alternative to GPI is geometric policy composition (GPC; Thakoor et al., 2022), which enables estimating Q-values when one follows an ordered sequence of n policies $(\pi_1, \ldots, \pi_n)$. Whereas GPI evaluates how much reward will be obtained by the best of a set of policies, GPC is a form of model-based control where the agent evaluates the path obtained from following a sequence of policies. We discuss this in more detail in section 4.4.

4.3 Hierarchical Reinforcement Learning. Many tasks have multiple timescales, which can be exploited using a hierarchical architecture in which the agent learns and acts at multiple levels of temporal abstraction. The classic example of this is the options framework (Sutton et al., 1999). An option o is a temporally extended behavior defined by (1) an initiation function $\mathcal{I}_o$ that determines when it can be activated, (2) a policy π_o, and (3) a function β_o that determines when the option should terminate: $o = \langle \mathcal{I}_o, \pi_o, \beta_o \rangle$. In this section, we discuss how predictive representations can be used to discover useful options and transfer them across tasks.

4.3.1 Discovering Options to Efficiently Explore the Environment. One key property of the SR is that it captures the long-range structure of an agent's paths through the environment. This has been useful in discovering options. Machado et al. (2017, 2023) showed that if one performs an eigendecomposition on a learned SR, the eigenvectors corresponding to the highest eigenvalues could be used to identify states with high diffusion (i.e., states that tend to be visited frequently under a random walk policy). In particular, for eigenvector e_i, they defined the following intrinsic reward function:

$$R_i^{\texttt{int}}(s, s') = e_i^\top (\boldsymbol{\phi}(s') - \boldsymbol{\phi}(s)), \tag{4.7}$$

which rewards the agent for exploring diverse parts of the state space. Note that $\boldsymbol{\phi}$ is either a one-hot vector in the case of the SR or state features in the case of SFs. Here, we focus on the SR. With this reward function, the agent can iteratively build up a set of options $\mathcal{O}$ that can be leveraged to improve exploration of the environment. The algorithm proceeds as follows:

1. Collect samples with a random policy that selects between between primitive actions and options. The set of options is initially empty $(\mathcal{O} = \{\emptyset\})$.
2. Learn successor representation M^π from the gathered samples.
3. Get new exploration option $o = \langle \mathcal{I}_o, \pi_o, \beta_o \rangle$. π_o is a policy that maximizes the intrinsic reward function in equation 4.7 using the current SR. The initiation function $\mathcal{I}_o$ is 1 for all states. The termination

function β_o is 1 when the intrinsic reward becomes negative (i.e., when the agent begins to go toward more frequent states). This option is added to the overall set of options, $\mathcal{O} \leftarrow \mathcal{O} \cup \{o\}$.

Agents endowed with this strategy were able to discover meaningful options and improve sample efficiency in the four-rooms domain, as well as on challenging Atari games such as *Montezuma's Revenge*.

4.3.2 Transferring Options with the SR.

Instant synthesis of combinations of options. One of the benefits of leveraging SFs is that they enable transfer to tasks that are linear combinations of known tasks (see section 2.5.1). At transfer time, an agent can exploit this when transferring options by defining subgoals using this space of tasks (see section 2.5.2). In continuous control settings where an agent has learned to move in a set of directions (e.g., up, down, left, right), this has enabled generalization to combinations of these policies (Barreto et al., 2019). For example, the agent could instantly move in novel directions (e.g., up right, down left) as needed to complete a task.

Transferring options to new dynamics. One limitation of both options and SFs is that they are defined for a particular environment and don't naturally transfer to new environments with different transition functions. Han and Tschiatschek (2021) addressed this limitation by learning abstract options with SFs defined over abstract state features that respect bisimulation relations (Li et al., 2006). This method assumes transfer from a set of N $\boldsymbol{\psi}$-MDPs $\{\mathcal{M}_i\}_{i=1}^n$ where $\mathcal{M}_i = \langle S_i, O_i, T_i, \boldsymbol{\psi}, \gamma \rangle$. It also assumes the availability of MDP-dependent state-feature functions $\boldsymbol{\phi}_{\mathcal{M}_i}(s, a)$ that map individual MDP state-action pairs to a shared feature space. Assume an agent has learned a (possibly) distinct set of options $\{o_k\}$ for each source MDP. The SFs for each option's policy are defined as usual (see equation 2.30). When we are transferring to a new MDP $\mathcal{M}'$, we can map each option to this environment by mapping its SFs to a policy that produces similar features using an inverse reinforcement learning algorithm (Ng et al., 2000). Once the agent has transferred options to a new environment, it can plan using these options by constructing an abstract MDP over which to plan. Han and Tschiatschek (2021) implement this using successor homomorphisms, which define criteria for aggregating states to form an abstract MDP. In particular, pairs of states (s_1, s_2) and options (o_1, o_2) will map to the same abstract MDP if they follow bisimulation relations (Li et al., 2006) and if their SFs are similar:

$$||\boldsymbol{\psi}^{o_1}(s_1, a) - \boldsymbol{\psi}^{o_2}(s_2, a)|| \leq \epsilon_\psi, \tag{4.8}$$

where ϵ_ψ is a similarity threshold. With this method, agents were able to discover state abstractions that partitioned large grid worlds into intuitive segments and successfully plan in novel MDPs with a small number of learning updates.

4.4 Jumpy Model-Based Reinforcement Learning. The successor model (SM) is interesting because it offers a novel way to do model-based RL. Traditionally, a model-based agent simulates trajectories with a single-step model. While this is flexible, it is also expensive. SMs enable an alternative strategy, where the agent instead samples and evaluates likely (potentially distal) states that will be encountered when following some policy π. As mentioned in section 4.2.3, Thakoor et al. (2022) leverage this property to develop generalized policy composition (GPC), a novel algorithm that enables a jumpy form of model-based RL. Rather than simulate trajectories defined over next states, agents simulate trajectories by using SMs to jump between states using a given set of policies. While this is not as flexible as simulating trajectories with a single-step model, it is much more efficient.

In RL, one typically uses a large discount factor ($\gamma \approx 1$). When learning an SM, this is useful because you can learn likelihoods over potentially very distal states. However, this makes learning an SM more challenging. GPC mitigates this challenge by composing a shorter horizon SM $\boldsymbol{\mu}_\beta^\pi$ with a longer horizon SM $\boldsymbol{\mu}_\gamma^\pi$, where $\beta < \gamma$. Composing two separate SMs with different horizons has the following benefits. An SM with a shorter horizon $\boldsymbol{\mu}_\beta^\pi$ is easier to learn but cannot sample futures as distal as $\boldsymbol{\mu}_\gamma^\pi$; on the other hand, $\boldsymbol{\mu}_\gamma^\pi$ is harder to learn but can make very long-horizon predictions and better avoids compounding errors. By combining the two, Thakoor et al. (2022) studied how these two errors can be traded off.

Intuitively, GPC works as follows. Given a starting state-action pair (s_0, a_0) and policies $(\pi_1, \ldots, \pi_n)$, the agent samples a sequence of $n-1$ next states with our shorter horizon SM $\boldsymbol{\mu}_\beta^{\pi_i}$ and π_i, that is, $s_1 \sim \boldsymbol{\mu}_\beta^{\pi_1}(\cdot|s_0, a_0)$, $a_1 \sim \pi_1(\cdot|s_1)$, $s_2 \sim \boldsymbol{\mu}_\beta^{\pi_1}(\cdot|s_1, a_1)$, and so on. The agent then samples a (potentially more distal) state s_n from the longer-horizon SM, $\boldsymbol{\mu}_\gamma^\pi$, $s_n \sim \boldsymbol{\mu}_\beta^{\pi_n}(\cdot|s_{n-1}, a_{n-1})$. The reward estimates for the sampled state-action pairs can be combined as a weighted sum to compute $Q^\pi(s_0, a_0)$ analogous to equation 2.28 (see Thakoor et al., 2022, for technical details). Leveraging GPC enabled convergence with an order of magnitude fewer samples in the four-rooms domain and in a continuous-control maze navigation domain.

4.5 Multiagent Reinforcement Learning. As we've noted, the SR is conditioned on a policy π. In a single-agent setting, the SR provides predictions about what that agent can expect to experience when executing the policy. In multiagent settings, one can parameterize this prediction with another agent's policy to form predictions about what one can expect to see in the environment when other agents follow their own policies. This is the basis for numerous algorithms that aim to learn about, from, and with other agents (Rabinowitz et al., 2018; Kim et al., 2022; Filos et al., 2021; Gupta et al., 2021; Lee et al., 2019).

4.5.1 Learning about Other Agents. Rabinowitz et al. (2018) showed that an AI agent could learn aspects of theory of mind (including passing a false

belief test) by meta-learning SFs that described other agents. While Rabinowitz did not explicitly compare against humans (and was not trying to directly model human cognition), this remains exciting as a direction for exploring scalable algorithms for human-like theory of mind.

4.5.2 Learning from Other Agents. One nice property of SFs is that they can be learned with TD learning using off-policy data (i.e., data collected from a policy different from the one currently being executed). This can be leveraged to learn SFs for the policies of other agents just as an agent learns SFs for their own policy. Filos et al. (2021) exploited this to design an agent that simultaneously learned SFs for both its own policy and for multiple other agents. They were then able to generalize effectively to new tasks via a combination of all of their policies by exploiting GPI (see equation 2.35).

4.5.3 Learning with Other Agents. One of the key benefits of leveraging universal SFs with GPI (see equation 3.3) is that you can systematically change the Q-value used for action selection by (1) shifting the policy encoding $\mathbf{z}_\pi$ defining the SF or (2) shifting the feature preferences $\mathbf{w}$ that are being maximized. Gupta et al. (2021) exploit this for cooperative multiagent RL. Let $s^{(i)}$ denote the state of the ith agent and $a^{(i)} \in \mathcal{A}_i$ denote the action they take. Now let $\boldsymbol{\psi}_\theta\left(s^{(i)}, a, \mathbf{z}_\pi\right)$ denote that agent's SFs for policy encoding $\mathbf{z}_\pi$, let $\mathcal{C}_{\texttt{tasks}}$ denote the set of possible feature preferences that agents can follow, and let $\mathcal{C}_\pi$ denote the set of possible policy encodings over which agents can make predictions. Gupta et al. (2021) studied the cooperative setting where the overall Q-value is simply the sum of individual Q-values. If $\mathcal{C}_\pi$ denotes policies for separate but related tasks, acting in accordance to

$$a^{(i)} = \underset{a \in \mathcal{A}_i}{\operatorname{argmax}} \max_{\mathbf{w} \in \mathcal{C}_{\texttt{tasks}}} \max_{\mathbf{z}_\pi \in \mathcal{C}_\pi} \left\{ \boldsymbol{\psi}_\theta\left(s^{(i)}, a, \mathbf{z}_\pi\right)^\top \mathbf{w} \right\} \tag{4.9}$$

enables a team of agents to take actions that systematically explore the environment on tasks specified by $\mathcal{C}_{\texttt{tasks}}$. These teams were able to improve exploration and zero-shot generalization in a variety of cooperative multiagent environments including StarCraft (Samvelyan et al., 2019).

4.6 Other Artificial Intelligence Applications. The SR and its generalizations have been broadly applied within other areas of AI. For example, it has been used to define an improved similarity metric in episodic control settings (Emukpere et al., 2021). By leveraging SFs, one can incorporate information from previously experienced states with similar dynamics to the current state. The SR has also been applied toward improving importance sampling in off-policy learning. If the agent learns a density ratio similar to equation 3.13, this can enable simpler marginalized importance sampling algorithms (Liu et al., 2018) that improve off-policy evaluation (Fujimoto

et al., 2021). In addition to these examples, we highlight the following applications of the SR.

4.6.1 Representation Learning. Learning SMs can obviate the need for separate representation losses. In many applications, the reward signal is not enough to drive learning of useful representations. Some strategies to address this challenge include data augmentation and learning of auxiliary tasks. Learning the SM has been shown to enable representation learning with superior sample efficiency without using these additions (Eysenbach et al., 2020, 2022; Zheng et al., 2023). Predictive representations can also be used to define an auxiliary task for representation learning, which has been shown to be helpful in several settings. A simple example comes from inspecting the loss for learning SFs (see equation 3.5). In standard Q-learning, the agent learns only about achieving task-specific reward. When learning SFs, the agent also learns representations that enable achieving state features that are potentially not relevant for the current task (i.e., the agents are by default leaning auxiliary tasks). This important ability is even possible in a continual learning setting where the distribution of state features are nonstationary (McLeod et al., 2021). Another interesting example comes from proto-value networks (Farebrother et al., 2023). The authors show that if one learns a successor measure (a set-inclusion-based generalization of the SR) over random sets, this can enable the discovery of predictive representations that enable very fast learning in the Atari learning environment.

4.6.2 Learning Diverse Policies. A final application of the SR has been in learning diverse policies. In the mujoco environment, Zahavy et al. (2021) showed that SFs enabled discovering a set of diverse policies for controlling a simulated dog avatar. Their approach used SFs to prospectively summarize trajectories. A set of policies was than incrementally learned so that each new policy would be different in its expected features from all policies learned so far. This idea was then generalized to diversify chess playing strategies based on their expected future features (Zahavy et al., 2023).

5 Neuroscience Applications

In this section, we discuss how the computational ideas reviewed above have been used to understand a variety of brain systems. Medial temporal lobe regions, and in particular the hippocampus, appear to encode predictive representations. We review evidence for this claim and efforts to formalize its mechanistic basis in neurobiology. We also discuss how vector-valued dopamine signals may provide an appropriate learning signal for these representations.

5.1 A Brief Introduction to the Medial Temporal Lobe. Before discussing evidence for predictive representations, it is important to take a

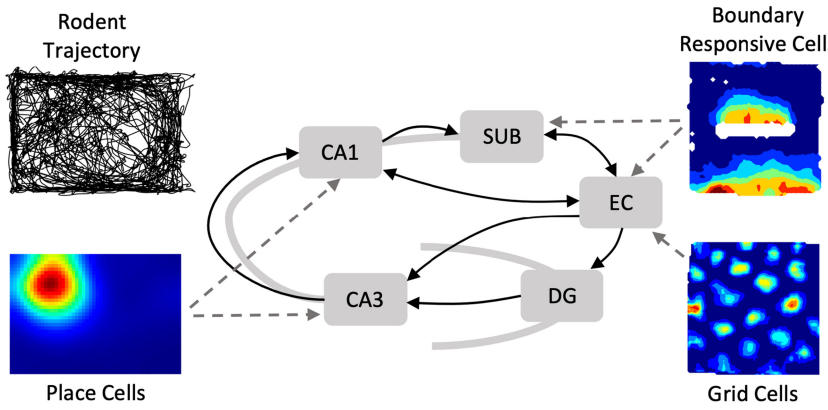

Figure 6: Spatial representations in the medial temporal lobe. As a rodent navigates space (e.g., a rectangular arena; top left), place cells recorded in regions CA1 and CA3 of hippocampus fire in stable, sparse representations of self-location (bottom left; hot colors indicate increased neuronal activity). Conversely, grid cells in neighboring medial entorhinal cortex (EC; bottom right), which provides input to both CA1 and CA3 directly, as well as CA3 via dentate gyrus (DG; solid arrows imply directional connectivity between regions), have spatially periodic hexagonal firing patterns tiling the entire environment. Additionally, boundary-responsive neurons in both medial entorhinal cortex and subiculum (SUB) fire when the animal occupies specific positions relative to external and internal environmental boundaries.

sufficiently broad view of the medial temporal lobe's functional organization. Not everything we know about these regions fits neatly into a theory of predictive representations. Indeed, classical views are quite different, emphasizing spatial representation and episodic memory.

Extensive evidence identifies the hippocampus and associated cortical regions as providing a neural-level representation of space, often conceptualized as a cognitive map (O'Keefe & Nadel, 1978; Morris et al., 1982, see also section 6.4). Integral to this framework are the distinct firing patterns of various cell types found in structures across the hippocampal formation (see Figure 6). Place cells, in regions CA3 and CA1, offer a temporally stable, sparse representation of self-location able to rapidly reorganize in novel environments—the phenomenon of remapping (O'Keefe & Dostrovsky, 1971; Muller & Kubie, 1987; Bostock et al., 1991).

Subiculum and dentate gyrus also contain spatially modulated neurons with broadly similar characteristics, the former tending to be diffuse and elongated along environmental boundaries (Lever et al., 2009) while the latter are extremely sparse (Jung & McNaughton, 1993; Leutgeb et al., 2007). In contrast, in medial entorhinal cortex (mEC), the primary cortical partner

of hippocampus, the spatially periodic firing patterns of grid cells effectively tile the entire environment (see Figure 6) and are organized into discrete functional modules of different scale (Hafting et al., 2005; Barry et al., 2007; Stensola et al., 2012). The highly structured activity of grid cells has provoked a range of theoretical propositions pointing to roles in path integration (McNaughton et al., 2006; Burgess et al., 2007), vector-based navigation (Bush et al., 2015; Banino et al., 2018), and as an efficient basis set for spatial generalization (Whittington et al., 2020). Notably, mEC also contains a "zoo" of other cell types with functional characteristics related to self-location, including head direction cells (Sargolini et al., 2006), border cells (Solstad et al., 2008), speed cells (Kropff et al., 2015), and multiplexed conjunctive responses (Sargolini et al., 2006; Hardcastle et al., 2017).

In summary, the medial temporal lobe exhibits remarkable functional diversity. We now turn to the claim that predictive principles offer a unifying framework for understanding aspects of this diversity.

5.2 The Hippocampus as a Predictive Map. Accumulating evidence indicates that neurons within the hippocampus and its surrounding structures, particularly place and grid cells, demonstrate predictive characteristics consistent with a predictive map of spatial states. Stachenfeld et al. (2017) were the first to systematically explore this perspective, establishing a connection between the responses of hippocampal neurons and the SR.[8] They argued that place cells were not inherently representing the animal's spatial location, but rather its expectations about future spatial locations. Specifically, they argued that the receptive fields of place cells correspond to columns of the SR matrix $\mathbf{M}^{\pi}$ from section 2.3 (see Figure 7, left). This implies that each receptive field is a retrodictive code, in the sense that the cells are more active in locations that tend to precede the cell's "preferred" location (i.e., the location of the peak firing). The population activity of place cells at a given time corresponds to a row of the SR matrix; this is a predictive code, in the sense that they collectively encode expectations about upcoming states.

In line with this hypothesis, the tuning of place fields (see Figure 7, bottom left) is influenced by the permissible transitions afforded by an environment: they do not typically cross environmental boundaries like walls, tending to extend along them, mirroring the trajectories animals follow (Alvernhe et al., 2011; Tanni et al., 2022). Alternations made to an environment's layout, affecting the available paths, influence the activity of adjacent place cells, consistent with the SR (Stachenfeld et al., 2017). Notably, even changes in policy alone, such as training rats to switch between foraging and directed behavior, can markedly alter place cell firing (Markus

[8] Earlier ideas about predictive processing in the hippocampus were explored by several authors (Levy, 1996; Levy et al., 2005; Buckner, 2010; Lisman & Redish, 2009), though these were not framed in terms of the SR or related ideas.

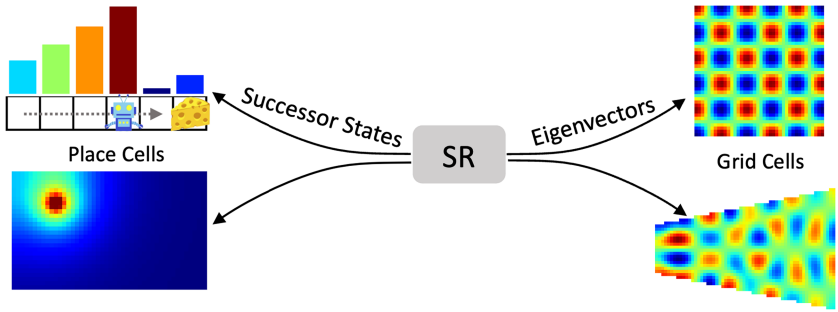

Figure 7: Successor representation model of the hippocampus and medial entorhinal cortex. As an agent explores a linear track environment in a unidirectional manner, the SR skews backward down the track opposite to the direction of motion (top left; hot colors indicate increased predicted occupancy of the agent's depicted state), as observed in hippocampal place cells (Mehta et al., 2000). In a 2D arena, the SR forms place cell–like sparse representations of self-location (bottom left), while the eigenvectors of the SR form spatially periodic activity patterns reminiscent of entorhinal grid cells (top right). Similar to real grid cells (Krupic et al., 2015), the periodicity of these eigenvectors is deformed in polarized environments such as trapezoids (bottom right).

et al., 1995), also broadly consistent with the SR. Further, when animals are trained to generate highly stereotyped trajectories, for example, repeatedly traversing a track in one direction, CA1 place fields increasingly exhibit a backward skew, opposite the direction of travel (Mehta et al., 2000), thereby anticipating the animal's future state. This arises naturally from learning the SR, since the upcoming spatial states become highly predictable when agents consistently move in one direction, resulting in a backward skew of the successor states (see Figure 7, top left). The basic effect is captured in simple grid worlds like those used by Stachenfeld et al. (2017), but when the anchoring is replaced with continuous feature-based methods, the successor features also capture the backward shift in field peak observed in neural data (Mehta et al., 2000; George et al., 2023).

While the properties of place cells are consistent with encoding the SR, grid cells appear to resemble the eigenvectors of the SR (see Figure 7, right). Specifically, eigendecomposition of the SR matrix $\mathbf{M}^{\pi}$ yields spatially periodic structures of varying scales, heavily influenced by environmental geometry (Stachenfeld et al., 2017) while being relatively robust to the underlying policy (De Cothi & Barry, 2020). Broadly, these resemble grid cells, but notably lack the characteristic hexagonal periodicity except when applied to hexagonal environments. This discrepancy, however, is likely not significant because subsequent work indicates that biological constraints, such as nonnegative firing rates (Dordek et al., 2016; Sorscher et al., 2019),

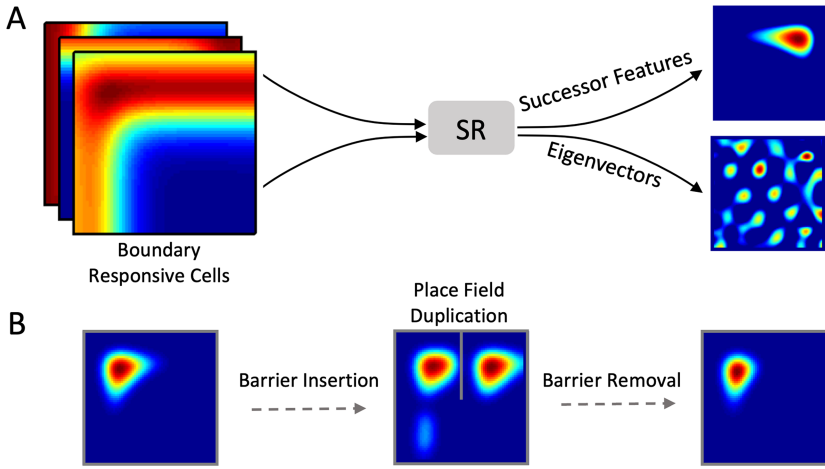

Figure 8: A successor representation with neurobiological state features. (A) Boundary responsive cells, present in subiculum and entorhinal cortex, are used as state features for learning successor features and their eigenvectors (De Cothi & Barry, 2020). These can capture more nuanced characteristics of both place and grid cells compared to one-hot spatial features derived from a grid world. For example, the eigenvectors have increased hexagonal periodicity relative to a grid world SR model. (B) The successor features can convey duplicated fields when additional walls are inserted in the environment, as observed in real place cells (Barry et al., 2006).

efficiency considerations (Dorrell et al., 2023), and neurobiologically plausible state features (De Cothi & Barry, 2020) tend to move these solutions closer to the expected hexagonal activity patterns (see Figure 8A). The key point then is that environmental geometries that polarize the transitions available to an animal produce SR eigenvectors with commensurate distortions, matching observations that grid firing patterns are also deformed under such conditions (Derdikman et al., 2009; Krupic et al., 2015). Notably, this phenomenon is also observed in open-field environments with straight boundaries, where both biological grid cells and SR-derived eigenvectors exhibit a tendency to orient relative to walls (Krupic et al., 2015; De Cothi & Barry, 2020). Complementary evidence comes from virtual reality studies of human subjects, where errors in distance estimates made by participants mirrored distortions in eigenvector-derived grid cells (Bellmund et al., 2020).

Although place and grid cells are predominantly conceptualized as spatial representations, it is increasingly clear that these neurons also represent nonspatial state spaces (Constantinescu et al., 2016; Aronov et al., 2017); in some cases, activity can be interpreted as encoding an SR over such state

spaces. For example, a study by Garvert et al. (2017) showed human participants a series of objects on a screen in what appeared to be a random order. However, unknown to the participants, the sequence was derived from a network of nonspatial relationships, where each object followed certain others in a predefined pattern. Brain imaging found that hippocampal and entorhinal activity mirrored the nonspatial predictive relationships of the objects, as if encoded by an SR (see Brunec & Momennejad, 2022).

5.3 Learning a Biologically Plausible SR. In much of the neuroscience work, SRs are formulated over discrete state spaces, facilitating analysis and enabling direct calculation for diffusive trajectories. In spatial contexts, this corresponds to a grid world with one-hot-location encoding, a method that can produce neurobiologically plausible representations (Stachenfeld et al., 2017). However, the brain must use biologically plausible learning rules and features derived from sensory information, with the choice of state features $\boldsymbol{\phi}(s)$ exerting significant influence on the resultant SFs $\boldsymbol{\psi}^{\pi}(s)$ (described in section 2.5).

De Cothi and Barry (2020) employed idealized boundary vector cells (BVCs)—neurons coding for distance and direction to environmental boundaries—as a basis over which to calculate a spatial SR. BVCs have been hypothesized as inputs to place cells (Hartley et al., 2000). They resemble the boundary-responsive cells found in the mEC (Solstad et al., 2008) and subiculum (Barry et al., 2006; Lever et al., 2009); hence, they are plausibly available to hippocampal circuits (see Figure 8A). The SFs of these neurobiological state features and their eigendecomposition resemble place and grid fields as before, but also captured more of the nuanced characteristics of these spatially tuned neurons—for example, the way in which place fields elongate along environmental boundaries (Tanni et al., 2022) and duplicate when additional walls are introduced (see Figure 8B; Barry et al., 2006). Geerts et al. (2020) employed a complementary approach, using an SR over place cell state features in parallel with a model-free framework trained on egocentric features. The dynamics of the interaction between these two elements mirrored the behavioral preference of rodents, which initially favor a map-based navigational strategy, before switching to one based on body turns (Geerts et al., 2020; Packard & McGaugh, 1996).

Subsequent models expanded on these ideas. While these differ in terms of implementation and focus, they employ the common idea of embedding transition probabilities into the weights of a network using biologically plausible learning rules. Fang et al. (2023) advanced a framework using a bespoke learning rule acting on a recurrent neural network supplied with features derived from experimental recordings. The network was sufficient to calculate an SR and could do so at different temporal discounts, producing SFs that resembled place cells. Bono et al. (2023) applied spike-time-dependent plasticity (STDP; Bi & Poo, 1998; Kempter et al., 1999), a Hebbian learning rule sensitive to the precise ordering of pre- and

postsynaptic spikes, to a single-layer spiking network. Because the ordering of spikes from spatial inputs inherently reflects the sequence of transitions between them, this configuration also learns an SR. Indeed, the authors were able to show that the synaptic weights learned by this algorithm are mathematically equivalent to TD learning with an eligibility trace. Furthermore, temporally accelerated biological sequences, such as replay (Wilson & McNaughton, 1994; Ólafsdóttir et al., 2016), provide a means to quickly acquire SRs for important or novel routes. Finally, George et al. (2023) followed a similar approach, showing that STDP (Bi & Poo, 1998) applied to theta sweeps—the highly ordered sequences of place cell spiking observed within hippocampal theta cycles (O'Keefe & Recce, 1993; Foster & Wilson, 2007)—was sufficient to rapidly learn place field SFs that were strongly modulated by agent behavior, consistent with empirical observations. Additionally, because the speed and range of theta sweeps are directly linked to the size of the underlying place fields, the authors also noted that the gradient of place field sizes observed along the dorsal-ventral axis of the hippocampus inherently approximates SFs with decreasing temporal discounts (Kjelstrup et al., 2008; Momennejad & Howard, 2018).

The formulation used by George et al. (2023) highlights a paradox: while place fields can serve as state features, they are also generated as SFs. This dual role might suggest a functional distinction between areas such as CA3 and CA1, with CA3 potentially providing the spatial basis and CA1 representing SFs. Alternatively, spatial bases could originate from upstream circuits, such as mEC, as proposed in the Fang et al. (2023) model. Furthermore, it is conceivable that the initial rapid formation of place fields is governed by a distinct plasticity mechanism, such as behavioral-timescale plasticity (Bittner et al., 2017). Once established, these fields would then serve as a basis for subsequent SF learning. Such a perspective is compatible with observations that populations of place fields in novel environments do not immediately generate theta sweeps (Silva et al., 2015).

These algorithms learn SFs under the premise that the spatial state is fully observable, for example, by a one-hot encoding in a grid world or the firing of BVCs computed across the distances and directions to nearby walls. However, in reality, states are often only partially observable and inherently noisy due to sensory and neural limitations. Vértes and Sahani (2019) present a mechanism for how the SR can be learned in partially observable noisy environments, where state uncertainty is represented in a feature-based approximation via distributed, distributional codes. This method supports RL in noisy or ambiguous environments, for example, navigating a corridor of identical rooms.

In summary, the SR framework can be generalized to a state space comprising continuous, nonidentical, overlapping state features, and acquired with biological learning rules. As such, it is plausible that hippocampal circuits could in principle instantiate a predictive map, or close approximation, based on the mixed population of spatially modulated neurons

available from its inputs. The models reviewed above demonstrate ways in which the SR could be learned online during active experience. Nonetheless, much learning in the brain is achieved offline, during periods of wakeful rest or sleep. One candidate neural mechanism for this is replay, rapid sequential activity during periods of quiescence.

5.4 Replay. During periods of sleep and awake rest, hippocampal place cells activate in rapid sequences that recapitulate past experiences (Wilson & McNaughton, 1994; Foster & Wilson, 2007). These reactivations, known as replay, often coordinate with activity in the entorhinal (Ólafsdóttir et al., 2016) and sensory cortices (Ji & Wilson, 2007; Rothschild et al., 2017), and are widely thought to be a core mechanism supporting system-level consolidation of experiential knowledge (Girardeau et al., 2009; Ego-Stengel & Wilson, 2010; Ólafsdóttir et al., 2015). In linear track environments, hippocampal place cells typically exhibit directional tuning with firing fields that disambiguate travel in either direction (Navratilova et al., 2012). This directionality enables two functional classes of replay to be distinguished: forward replay where the sequence reflects the order in which the animal experienced the world, and reverse replay where the behavioral sequence is temporally reversed (analogous to the animal walking tail-first down the track). Intriguingly, the phase of a navigation task has been shown to influence the type of replay that occurs; for example, reverse replay is associated with receipt of reward at the end of trials, while forward replay is more abundant at the start of trials prior to active navigation (Diba & Buzsáki, 2007).

Mattar and Daw (2018) modeled the emergence of forward and reverse replays using a reinforcement learning agent that accesses memories of locations in an order determined by their expected utility. Specifically, the agent prioritizes replaying memories as a balance of two requirements: the need to evaluate imminent choices versus the gain from propagating newly encountered information to preceding locations. The need term for a spatial state corresponds to its expected future occupancy given the agent's current location, thus utilizing the definition of the SR (see section 2.3 and equation 2.13) to provide a measure for how often in the near future that state will tend to be visited. The gain term represents the expected increase in reward from visiting that state. This mechanism produces sequences that favor adjacent backups: upon discovery of an unexpected reward, the last action executed will have a positive gain, making it a likely candidate for replay. Thus, value information can be propagated backward by chaining successive backups in the reverse direction, simulating reverse replay. Conversely, at the beginning of a trial, when the gain differences are small and the need term dominates, sequences that propagate value information in the forward direction will be the likely candidates for replay, prioritizing nearby backups that extend forward through the states the agent is expected to visit in the near future.

Sequential neural activity in humans has similarly been observed to exhibit orderings consistent with sampling based on future need. Using a statistical learning task with graph-like dependencies between visual cues (Schapiro et al., 2013; Garvert et al., 2017; Lynn et al., 2020), Wittkuhn et al. (2022) showed participants a series of animal images drawn from a ring-like graph structure in either a uni- or bidirectional manner. Using fMRI data recorded during 10 s pauses between trials, Wittkuhn et al. (2022) found forward and reverse sequential activity patterns in visual and sensorimotor cortices, a pattern well captured by an SR model learned from the graph structure participants experienced.

5.5 Dopamine and Generalized Prediction Errors. As described in section 2.3, TD learning rules provide a powerful algorithm for value estimation. The elegant simplicity of this algorithm led neuroscientists to explore if, and how, TD learning might be implemented in the brain. Indeed, one of the celebrated successes of neuroscience has been the discovery that the activity of midbrain dopamine neurons appears to report reward prediction errors (Schultz et al., 1997) consistent with model-free RL algorithms (see section 2.2, equation 2.11). This successfully accounts for many aspects of dopamine responses in classical and instrumental conditioning tasks (Starkweather & Uchida, 2021).

While elegant, the classical view that dopamine codes for a scalar reward prediction error does not explain more heterogeneous aspects of dopamine responses. For example, the same dopamine neurons also respond to novel and unexpected stimuli (Ljungberg et al., 1992; Horvitz, 2000) and to errors in predicting the features of rewarding events, even when value remains unchanged (Chang et al., 2017; Takahashi et al., 2017; Stalnaker et al., 2019; Keiflin et al., 2019). Russek et al. (2017) highlighted the biological plausibility of the SR TD learning rule (see equations 2.17 and 2.18) in light of its similarity to the model-free TD learning rule (see equation 2.11), while refraining from making any explicit connections between vector-valued SR TD errors and dopamine. Gardner et al. (2018) took this idea further and proposed an extension to the classic view of dopamine, suggesting that it also encodes prediction errors related to sensory features. According to this model, dopamine reports vector-valued TD errors suitable for updating SFs (see section 2.5), using the fact that SFs obey a Bellman equation and hence are learnable by TD. This model explains a number of phenomena that are puzzling under a classic TD model based only on scalar reward predictions.

First, it explains why the firing rate of dopamine neurons increases after a change in reward identity, even when reward magnitude is held fixed (Takahashi et al., 2017): changes in reward identity induce a sensory prediction error that shows up as one component of the error vector. Second, it explains, at least partially, why subpopulations of dopamine neurons encode a range of different nonreward signals (Engelhard et al., 2019; de Jong et al., 2022; Gonzalez et al., 2023). Third, it explains why optogenetic

manipulations of dopamine influence conditioned behavior even in the absence of reward (Chang et al., 2017; Sharpe et al., 2017).

How do dopamine neurons encode a vector-valued error signal? One possibility is that the errors are distributed across population activity. Pursuing this hypothesis, Stalnaker et al. (2019) analyzed the information content of dopamine neuron ensembles. They showed that reward identity can be decoded from these ensembles, but not from single neuron activity. Moreover, they showed that this information content disappeared over the course of training following an identity switch, consistent with the idea that error signals go to 0 as learning proceeds. The question remains how a vector-valued learning system is implemented biophysically. Some progress in this direction (albeit within a different theoretical framework) has been made by Wärnberg and Kumar (2023).

6 Cognitive Science Applications

A rich body of work dating back over a century has linked RL algorithms to reward-based learning processes in humans and nonhuman animals (Niv, 2009). Empirical findings align with the theoretical properties of model-based and model-free control (described in section 2.2), suggesting that model-based control underlies reflective, goal-directed behaviors, while model-free control underlies reflexive, habitual behaviors. The existence of both systems in the brain and their synergistic operation has received extensive support by a wide range of behavioral and neural studies across a number of species and experimental paradigms (see Dolan & Dayan, 2013, for a detailed review).

Recall that the SR (described in section 2.3) occupies an intermediate ground between model-based and model-free algorithms. This can make it advantageous when flexibility and efficiency are both desirable, which is the case in most real-world decision-making scenarios. In the field of cognitive science, several lines of research suggest that human learning and generalization are indeed consistent with the SR and related predictive representations. In this section, we examine studies showing that patterns of responding to changes in the environment (Momennejad et al., 2017), transfer of knowledge across tasks (Tomov et al., 2021), planning in spatial domains (Geerts et al., 2024), and contextual memory and generalization (Gershman et al., 2012; Smith et al., 2013; Zhou et al., 2023) exhibit signature characteristics of the SR-like predictive representations that cannot be captured by pure model-based or model-free strategies.

6.1 Revaluation. Some of the key findings pointing to a balance between a goal-directed system and a habitual system in the brain came from studies of reinforcer revaluation (Adams & Dickinson, 1981; Adams, 1982; Dickinson, 1985; Holland, 2004). In a typical revaluation paradigm, an animal (e.g., a rat) is trained to associate a neutral action (e.g., a lever press)

with an appetitive outcome (e.g., food). The value of that outcome is subsequently reduced (e.g., the rat is satiated, so food is less desirable), and the experimenter measures whether the animal keeps taking the action in the absence of reinforcement. Goal-directed control predicts that the animal would not take the action, since the outcome is no longer valuable, while habitual control predicts that the animal would keep taking the action, since the action itself was not devalued. Experimenters found that under some conditions—such as moderate training, complex tasks, or disruptions to dopamine inputs to striatum—behavior appears to be goal directed (e.g., lever pressing is reduced), while under other conditions—such as extensive training or disruptions to prefrontal cortex—behavior appears to be habitual (e.g., the rat keeps pressing the lever).

A modeling study by Daw et al. (2005) interpreted these findings through the lens of RL (see section 2.2). The authors formalized the goal-directed system as a model-based controller putatively implemented in prefrontal cortex and the habitual system as a model-free controller putatively implemented in dorsolateral stratum. They proposed that the brain arbitrates dynamically between the two controllers based on the uncertainty of their value estimates, preferring the more certain (and hence likely more accurate) estimate for a given action. Under this account, moderate training or complex tasks would favor the model-based estimates, since the model-free estimates may take longer to converge and hence be less reliable. On the other hand, extensive training would favor the model-free estimates, since they will likely have converged and hence be more reliable than the noisy model-based simulations.

One limitation of this account is that it explains sensitivity to outcome revaluation in terms of a predictive model, but it does not rule out the possibility that the animals may instead be relying on a predictive representation. A hallmark feature of predictive representations is that they allow an agent to adapt quickly to changes in the environment that keep its cached predictions relevant, but not to changes that require updating them. In particular, an agent equipped with the SR (see section 2.3) should adapt quickly to changes in the reward structure (R) of the environment but not to changes in the transition structure (T). Since the earlier studies on outcome revaluation effectively only manipulated reward structure, both model-based control and the SR could account for them, leaving open the question of whether outcome revaluation effects could be fully explained by the SR instead.

This question was addressed in a study by Momennejad et al. (2017), which examined how changes in either the reward structure or the transition structure experienced by human participants affect their subsequent choices. The authors used a two-step task consisting of three phases: a learning phase, a relearning (or revaluation) phase, and a test phase (see Figure 9A). During the learning phase, participants were presented with two distinct two-step sequences of stimuli and rewards corresponding to two distinct trajectories through state space. The first trajectory terminated

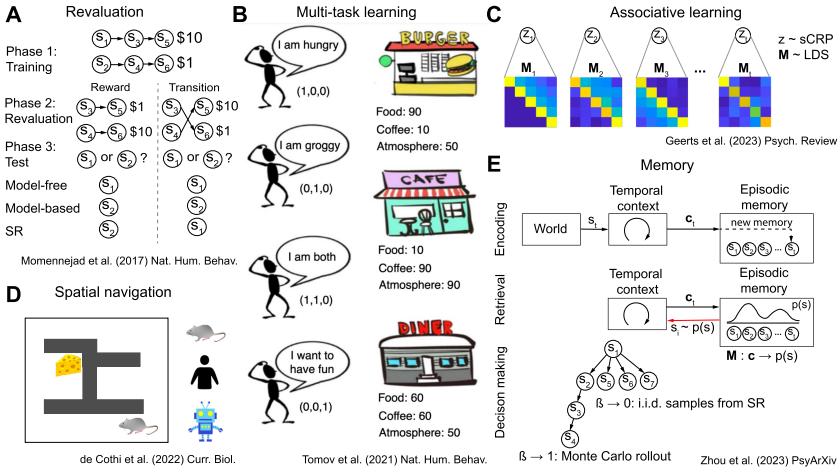

Figure 9: Predictive representations in cognitive science. (A) Revaluation paradigm and predictions from Momennejad et al. (2017, sec. 6.1). (B) Multitask learning paradigm from Tomov et al. (2021, sec. 6.2). A person is trained on tasks where they are either hungry ($\mathbf{w}^1_{train} = [1, 0, 0]$) or groggy ($\mathbf{w}^2_{train} = [0, 1, 0]$), and then tested on a task in which they are hungry, groggy, and looking to have fun ($\mathbf{w}_{new} = [1, 1, 1]$). (C) Context-dependent Bayesian SR from Geerts et al. (2024, sec. 6.3). z, context. $\mathbf{M}$, SR matrix associated with each context. sCRP, sticky Chinese restaurant process. LDS, linear-gaussian dynamical system. (D) Spatial navigation paradigm from de Cothi et al. (2022, sec. 6.4). (E) TCM-SR: Using the temporal context model (TCM) to learn the SR for decision making (Zhou et al., 2023, sec. 6.5). $\mathbf{M}$, SR matrix. $\mathbf{c}$, context vector. s, state and/or item. SR, successor representation.

with high reward ($s_1 \rightarrow s_3 \rightarrow s_5$ \$10), while the second trajectory terminated with a low reward ($s_2 \rightarrow s_4 \rightarrow s_6$ \$1), leading participants to prefer the first one over the second one.

During the revaluation phase, participants had to relearn the second half of each trajectory. Importantly, the structure of the trajectories changed differently depending on the experimental condition. In the reward revaluation condition, the transitions between states remained unchanged, but the rewards of the two terminal states swapped ($s_3 \rightarrow s_5$ \$1; $s_4 \rightarrow s_6$ \$10). In contrast, in the transition revaluation condition, the rewards remained the same, but the transitions to the terminal states swapped ($s_3 \rightarrow s_6$ \$1; $s_4 \rightarrow s_5$ \$10).

Finally, in the test phase, participants were asked to choose between the two initial states of the two trajectories (s_1 and s_2). Note that under both revaluation conditions, participants should now prefer the initial state of the second trajectory (s_2) as it now leads to the higher reward.

Unlike previous revaluation studies, this design clearly disambiguates between the predictions of model-free, model-based, and SR learners. Since the initial states (s_1 and s_2) never appear during the revaluation phase, a pure model-free learner would not update the cached values associated with those states and would still prefer the initial state of the first trajectory (s_1). On the other hand, a pure model-based learner would update its reward ($\hat{R}$) or transition ($\hat{T}$) estimates during the corresponding revaluation phase, allowing it to simulate the new outcomes from each initial state and make the optimal choice (s_2) during the test phase. Critically, both model-free and model-based learners (and any hybrid between them, such as a convex combination of their outputs, as in Daw et al., 2011) would exhibit the same preferences during the test phase in both revaluation conditions.

In contrast, an SR learner would show differential responding in the test phase depending on the revaluation condition, adapting and choosing optimally after reward revaluation but not after transition revaluation. Specifically, during the learning phase, an SR learner would learn the successor states for each initial state (the SR itself, i.e., $s_1 \rightarrow s_5$; $s_2 \rightarrow s_6$). In the reward revaluation condition, it would then update its reward estimates ($\hat{R}$) for the terminal states (s_5 \$1; s_6 \$10) during the revaluation phase, much like the model-based learner. Then, during the test phase, it would combine the updated reward estimates with the SR to compute the updated values of the initial states ($s_1 \rightarrow s_5$ \$1; $s_2 \rightarrow s_6$ \$10), allowing it to choose the better one (s_2). In contrast, in the transition revaluation condition, the SR learner would not have an opportunity to update the SR of the initial states (s_1 and s_2) since they are never presented during the revaluation phase, much like the model-free learner. Then, during the test phase, it would combine the unchanged reward estimates with its old but now incorrect SR to produce incorrect estimates for the initial states ($s_1 \rightarrow s_5$ \$10; $s_2 \rightarrow s_6$ \$1) and choose the worse one (s_1).

The pattern of human responses showed evidence of both model-based and SR learning: participants were sensitive to both reward and transition revaluations, consistent with model-based learning, but they performed significantly better after reward revaluations, consistent with SR learning. To rule out the possibility that this effect can be attributed to pure model-based learning with different learning rates for reward ($\hat{R}$) versus transition ($\hat{T}$) estimates, the researchers extended this Pavlovian design to an instrumental design in which participants' choices (i.e., their policy π) altered the trajectories they experienced. Importantly, this would correspondingly alter the learned SR: unrewarding states would be less likely under a good policy and hence not be prominent (or not appear at all) in the SR for that policy. Such states could thus get overlooked by an SR learner if they suddenly became rewarding. This subtle kind of reward revaluation (dubbed *policy revaluation* by the researchers) also relies on changes in the reward structure R, but induces predictions similar to the transition revaluation condition: SR

learners would not adapt quickly, while model-based learners would adapt just as quickly as in the regular reward revaluation condition.

Human responses on the test phase after policy revaluation were similar to responses after transition revaluation but significantly worse than responses after reward revaluation, thus ruling out a model-based strategy with different learning rates for $\hat{R}$ and $\hat{T}$. Overall, the authors interpreted their results as evidence of a hybrid model-based-SR strategy, suggesting that the human brain can adapt to changes in the environment by both updating its internal model of the world and by learning and leveraging its cached predictive representations (see also Kahn & Daw, 2023, for additional human behavioral data leading to similar conclusions).

6.2 Multitask Learning. In the previous section, we saw that humans can adapt quickly to changes in the reward structure of the environment (reward revaluation), as predicted by the SR (Momennejad et al., 2017). However, that theoretical account alone does not fully explain how the brain can take advantage of the SR to make adaptive choices. Here we take this idea further and propose that humans learn successor features (SFs; see section 2.5) for different tasks and use something like the GPI algorithm (see section 2.5.1) to generalize across tasks with different reward functions (Barreto et al., 2017, 2018, 2020).

In Tomov et al. (2021), participants were presented with different two-step tasks that shared the same transition structure but had different reward functions determined by the reward weights $\mathbf{w}$ (see Figure 9B). Each state s was associated with a different set of features $\boldsymbol{\phi}(s)$, which were valued differently depending on the reward weights $\mathbf{w}$ for a particular task. On each training trial, participants were first shown the weight vector $\mathbf{w}_{train}$ for the current trial and then asked to navigate the environment in order to maximize reward. At the end of the experiment, participants were presented with a single test trial on a novel task $\mathbf{w}_{new}$.

The main dependent measure was participant behavior on the test task, which was designed (along with the training tasks, state features, and transitions) to distinguish among several possible generalization strategies. Across several experiments, Tomov et al. (2021) found that participant behavior was consistent with SF and GPI. In particular, on the first (and only) test trial, participants tended to prefer the training policy that performed better on the new task, even when this was not the optimal policy. This "policy reuse" is a key behavioral signature of GPI. This effect could not be explained by model-based or model-free accounts. Their results suggest that humans rely on predictive representations from previously encountered tasks to choose promising actions on novel tasks.

6.3 Associative Learning. RL provides a normative account of associative learning, explaining how and why agents ought to acquire long-term reward predictions based on their experience. It also provides a

descriptive account of a myriad of phenomena in the associative learning literature (Sutton & Barto, 1990; Niv, 2009; Ludvig et al., 2012). Two recent ideas have added nuance to this story:

- Bayesian learning: Animals represent and use uncertainty in their estimates.
- Context-dependent learning: Animals partition the environment into separate contexts and maintain separate estimates for each context.

We examine each idea in turn and then explore how they can be combined with the SR (see section 2.3). In brief, the key idea is that animals learn a probability distribution over context-dependent predictive representations.

6.3.1 Bayesian RL. While standard RL algorithms (see section 2.2) learn point estimates of different unknown quantities like the transition function ($\hat{T}$) or the value function ($\hat{V}$), Bayesian RL posits that agents treat such unknown quantities as random variables and represent beliefs about them as probability distributions.

Generally Bayesian learners assume a generative process of the environment according to which hidden variables give rise to observable data. The hidden variables can be inferred by inverting the generative process using Bayes's rule. In one example of Bayesian value learning (Gershman, 2015), the agent assumes that the expected reward $R(s, \mathbf{w})$ is a linear combination of the observable state features $\boldsymbol{\phi}(s)$,[9]

$$R(s, \mathbf{w}) = \boldsymbol{\phi}(s)^\top \mathbf{w} \tag{6.1}$$

where the hidden weights $\mathbf{w}$ are assumed to evolve over time according to the following dynamical equations:

$$\mathbf{w}_0 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_0), \tag{6.2}$$

$$\mathbf{w}_t \sim \mathcal{N}(\mathbf{w}_{t-1}, q\mathbf{I}), \tag{6.3}$$

$$r_t \sim \mathcal{N}(R(s_t, \mathbf{w}_t), \sigma_R^2). \tag{6.4}$$

In effect, this means that each feature (or stimulus dimension) is assigned a certain reward weight. The weights are initialized randomly around zero (with prior covariance $\boldsymbol{\Sigma}_0$) and evolve according to a random walk (with volatility governed by the transition noise variance q). Observed rewards are given by the linear model (see equation 6.1) plus zero-mean gaussian noise with variance σ_R^2.

[9] Note that the linear reward model is the same as assumed in much of the SF work reviewed above (see equation 2.29).

This formulation corresponds to a linear-gaussian dynamical system (LDS). The posterior distribution over weights given the observation history $H_t = (s_{1:t}, r_{1:t})$ follows from Bayes's rule:

$$p(\mathbf{w}_t|H_t) = \frac{p(H_t|\mathbf{w}_t)p(\mathbf{w}_t)}{p(H_t)} = \mathcal{N}(\mathbf{w}_t; \hat{\mathbf{w}}_t, \boldsymbol{\Sigma}_t). \tag{6.5}$$

The posterior mean $\hat{\mathbf{w}}_t$ and covariance $\boldsymbol{\Sigma}_t$ can be computed recursively in closed form using the Kalman filtering equations:

$$\hat{\mathbf{w}}_t = \hat{\mathbf{w}}_{t-1} + \mathbf{k}_t\delta_t, \tag{6.6}$$

$$\boldsymbol{\Sigma}_t = \boldsymbol{\Sigma}_{t-1} + q\mathbf{I} - \lambda_t\mathbf{k}\mathbf{k}_t^\top, \tag{6.7}$$

where

- $\delta_t = r_t - \boldsymbol{\phi}(s_t)^\top\hat{\mathbf{w}}_t$ is the reward prediction error.
- $\lambda_t = \boldsymbol{\phi}(s_t)^\top(\boldsymbol{\Sigma}_t + q\mathbf{I})\boldsymbol{\phi}(s_t) + \sigma_R^2$ is the residual variance.
- $\mathbf{k}_t = (\boldsymbol{\Sigma}_t + q\mathbf{I})\boldsymbol{\phi}(s)/\lambda$ is the Kalman gain.

This learning algorithm generalizes the seminal Rescorla-Wagner model of associative learning (Rescorla & Wagner, 1972), and its update rule bears resemblance to the error-driven TD update (see equation 2.11). However, there are a few notable distinctions from its non-Bayesian counterparts:

- Uncertainty-modulated updating: The learning rate corresponds to the Kalman gain $\mathbf{k}_t$, which increases with posterior uncertainty (the diagonals of the posterior covariance, $\boldsymbol{\Sigma}_t$).
- Nonlocal updating: The update is multivariate, affecting the weights of all features simultaneously according to $\mathbf{k}_t$. This means that weights of "absent" features (i.e., where $\phi_i(s) = 0$) can be updated, provided those features have nonzero covariance with observed features.

These properties allow the Kalman filter to explain a number of phenomena in associative learning that elude non-Bayesian accounts (Dayan & Kakade, 2000; Kruschke, 2008; Gershman, 2015). Uncertainty-modulated updating implies that nonreinforced exposure to stimuli, or even just the passage of time, can affect future learning. For example, in a phenomenon known as *latent inhibition*, preexposure to a stimulus reduces uncertainty, which in turn reduces the Kalman gain, retarding subsequent learning for that stimulus. Nonlocal updating implies that learning could occur even for unobserved stimuli if they covary with observed stimuli according to $\boldsymbol{\Sigma}_t$. For example, in a phenomenon known as *backward blocking*, reinforcing two stimuli simultaneously (a compound stimulus, $\boldsymbol{\phi}(s) = [1, 1]$) and subsequently reinforcing only one of the stimuli ($\boldsymbol{\phi}(s') = [1, 0]$) reduces reward expectation for the absent stimulus ($\boldsymbol{\phi}(s'') = [0, 1]$) due to the learned negative covariance between the two reward weights ($\boldsymbol{\Sigma}_{1,2} < 0$).

Kalman filtering can also be extended to accommodate long-term reward expectations in an algorithm known as Kalman TD (Geist & Pietquin, 2010) by replacing the state features $\boldsymbol{\phi}(s)$ with their discounted temporal difference:

$$\mathbf{h}_t = \boldsymbol{\phi}(s_t) - \gamma \boldsymbol{\phi}(s_{t+1}). \tag{6.8}$$

This recovers the Kalman filter model described above when the discount factor γ is set to 0. Importantly, this model provides a link between the Bayesian approach to associative learning and model-free reinforcement learning algorithms; we exploit this link below when we discuss Bayesian updating of beliefs about predictive representations. Gershman (2015) showed that Kalman TD can explain a range of associative learning phenomena that are difficult to explain by Bayesian myopic ($\gamma = 0$) models and nonmyopic ($\gamma > 0$) non-Bayesian models.

6.3.2 Context-Dependent Learning. In addition to accounting for uncertainty in their estimates, animals have been shown to learn different estimates in different contexts (Redish et al., 2007). This kind of context-dependent learning can also be cast in a Bayesian framework (Courville et al., 2006; Gershman et al., 2010) by assuming that the environment is carved into separate contexts according to a hypothetical generative process. One possible formalization is the Chinese restaurant process (CRP), in which each observation is conditioned on a hidden context z_t that is currently active and evolves according to

$$P(z_t = k \mid \mathbf{z}_{1:t-1}) \propto \begin{cases} N_k & \text{if } k \text{ is a previously sampled context} \\ \alpha & \text{otherwise} \end{cases} \tag{6.9}$$

where $N_k = \sum_{i=1}^{t-1} \mathbb{I}[z_i = k]$ is the number of previous time steps assigned to context k and $\alpha \geq 0$ is a concentration parameter, which can be understood as controlling the effective number of contexts.[10]

If each context is assigned its own probability distribution over observations, then inferring a given context k is driven by two factors:

- How likely is the current observation under context k?
- How likely is context k given the previous context assignments $\mathbf{z}_{1:t-1}$ (see equation 6.9)?

In particular, a given context is more likely to be inferred if the current observations are more likely under its observation distribution and/or if it has been inferred more frequently in the past (i.e., if past observations have also been consistent with its observation distribution). Conversely, if the current

[10] Under the CRP, the expected number of contexts after t time steps is $\alpha \log t$.

observations are unlikely under any previously encountered context, a new context is induced with its own observation distribution.

This formulation has accounted for a number of phenomena in the animal learning literature (Gershman et al., 2010). For example, it explains why associations that have been extinguished sometimes reappear when the animal is returned to the context in which the association was first learned, a phenomenon known as *renewal*. It also provides an explanation of why the latent inhibition effect is attenuated if preexposure to a stimulus occurs in one context and reinforcement occurs in a different context.

Recently, a similar formulation has been used to explain variability in the way hippocampal place cells change their firing patterns in response to changes in contextual cues, a phenomenon known as *remapping* (Sanders et al., 2020). On this view, the hippocampus maintains a separate cognitive map of the environment for each context, and hippocampal remapping reflects inferences about the current context.

6.3.3 Bayesian Learning of Context-Dependent Predictive Maps. As discussed in section 5.2, Stachenfeld et al. (2017) showed that many aspects of hippocampal place cell firing patterns are consistent with the SR, suggesting that the hippocampus encodes a predictive map of the environment. In this light, the view that the hippocampus learns different maps for different contexts (Sanders et al., 2020) naturally points to the idea of a context-dependent predictive map.

This idea was formalized by Geerts et al. (2024) in a model that combines SFs (see section 2.5) with Bayesian learning and context-dependent learning. Their model learns a separate predictive map of the environment for each context, where each map takes the form of a probability distribution over SFs (see Figure 9C). The generative model assumes that the SF is given by a linear combination of state features:

$$\boldsymbol{\psi}_{j,t}(s) = \boldsymbol{\phi}(s)^\top \mathbf{m}_{j,t}, \tag{6.10}$$

where $\mathbf{m}_j$ is a vector of weights for predicting successor feature j. Analogous to the Kalman TD model described above, Geerts et al. assume that the weight vectors for all features evolve over time according to the following LDS:

$$\mathbf{m}_{j,0} \sim \mathcal{N}(\mu_0, \boldsymbol{\Sigma}_0), \tag{6.11}$$

$$\mathbf{m}_{j,t} \sim \mathcal{N}(\mathbf{m}_{j,t-1}, q\mathbf{I}), \tag{6.12}$$

$$\phi_{j,t}(s_t) \sim \mathcal{N}(\mathbf{h}_t^\top \mathbf{m}_{j,t-1}, \sigma_\phi^2). \tag{6.13}$$

This means that the weight vector is initialized randomly (with prior mean μ_0 and variance $\boldsymbol{\Sigma}_0$) and evolves randomly according to a random walk (with transition noise variance q). Observed features $\boldsymbol{\phi}(s)$ are noisy

differences in the SFs of successive states (with observation noise variance σ_ϕ^2). As in Kalman TD, the posterior over the weight vector for feature j is also gaussian with mean $\hat{\mathbf{m}}_j$ and variance $\mathbf{\Sigma}$, which can be computed using Kalman filtering equations essentially the same as those given above (see equations 6.6 and 6.7 in section 6.3.1).

The authors generalize this to multiple contexts using a switching LDS: each context has a corresponding LDS and SF weights. The contexts themselves change over time based on a "sticky" version of the Chinese restaurant process, which additionally favors remaining in the most recent context:

$$P(z_t = k \mid \mathbf{z}_{1:t-1}) \propto \begin{cases} N_k + \nu \mathbb{I}[z_{t-1} = k] & \text{if } k \text{ is a previously sampled context} \\ \alpha & \text{otherwise} \end{cases}, \tag{6.14}$$

where $\nu \geq 0$ is the "stickiness" parameter dictating how likely it is to remain in the same context.

Thus, a given context k is more likely to be inferred if:

- The current observations are consistent with its SFs (see equation 6.13).
- It was encountered frequently in the past (driven by the N_k term in equation 6.14).
- It was encountered recently (driven by the $\mathbb{I}[z_{t-1} = k]$ term in equation 6.14).

Conversely, a new context is more likely to be inferred if observations are inconsistent with the current SFs (i.e., when there are large SF prediction errors).

The authors show that this model can account for a number of puzzling effects in the animal learning literature that pose problems for both point estimation (TD learning) of the SR/SF (see equations 2.17 and 2.18 in section 2.3) and Bayesian RL (see equations 6.6 and 6.7 in section 6.3.1). One example is the opposing effect that preexposure to a context can have on learning. Brief exposure to a context can facilitate learning (context preexposure facilitation), while prolonged exposure can inhibit learning (latent inhibition; Kiernan & Westbrook, 1993). Context preexposure facilitation by itself can be accounted for by TD learning of the SR alone (Stachenfeld et al., 2017): during preexposure, the animal learns a predictive representation that facilitates propagation of newly learned values. Latent inhibition by itself can be accounted for by Bayesian RL alone (Gershman, 2015), as discussed previously: prolonged exposure reduces value uncertainty, in turn reducing the Kalman gain $\mathbf{k}_t$ (the effective learning rate) and inhibiting learning of new values. Kalman learning of SFs combines these two processes and can thus resolve the apparent paradox: initially, the

animal learns a predictive representation of the context, which facilitates learning, whereas after prolonged exposure, this effect is offset by a reduction in value uncertainty, which inhibits learning.

Another puzzling effect is the partial transition revaluation observed in Momennejad et al. (2017) and discussed in section 6.1, which cannot be accounted for by TD learning of the SR. This led Momennejad et al. to propose a hybrid model-based-SR strategy that relies on offline simulations. Kalman TD offers a more parsimonious account based on nonlocal updating that does not appeal to model-based simulations. In particular, the covariance matrix $\mathbf{\Sigma}_t$ learned during the learning phase captures the relationship between the initial states (s_1 and s_2) and subsequent states (s_3 and s_4). Updating the transitions from those subsequent states ($s_3 \rightarrow s_6$ and $s_4 \rightarrow s_5$) during the transition revaluation phase therefore also updates the SR for the initial states (s_1 and s_2), even though they are not encountered during the revaluation phase.

Nonlocal updating can similarly explain reward devaluation of preconditioned stimuli, a hallmark of model-based learning (Hart et al., 2020). This is similar to reward devaluation, discussed in section 6.1, except with an additional preconditioning phase during which an association is learned between two neutral stimuli (e.g., light $\rightarrow$ tone). During the subsequent conditioning phase, the second stimulus is associated with a rewarding outcome (e.g., tone $\rightarrow$ food), which is then devalued (e.g., by inducing taste aversion) during the devaluation phase. Finally, the animal is tested on the first neutral stimulus (e.g., light). Note that since the first stimulus is never present during the conditioning phase, TD learning would not acquire an association between the first stimulus and the reward and would thus not exhibit sensitivity to reward devaluation (Gardner et al., 2018). In contrast, during the preconditioning phase, Kalman TD learns that the two stimuli covary, allowing it to update the SF for both stimuli during the conditioning phase and consequently propagate the updated value to both stimuli during the devaluation phase.

Note that the phenomena so far can be explained without appealing to context-dependent learning (see section 6.3.2), since the experiments take place in the same context. Context-dependent Kalman TD can additionally explain a number of intriguing phenomena when multiple contexts are introduced.

One such phenomenon is the context specificity of learned associations (Winocur et al., 2009). In this paradigm, an animal learns an association (e.g., tone $\rightarrow$ shock) in one context (e.g., context A) and is then tested in the same context or in another context (e.g., context B). The amount of generalization of the association across contexts was found to depend on elapsed time: if testing occurs soon after training, the animal responds only in the training context (A), indicating context specificity. However, if testing occurs after a delay, the animal responds equally in both contexts A and B, indicating contextual generalization. Even more intriguing, this effect is reversed if

the animal is briefly reintroduced to the training context (A) before testing, in which case responding is once again context specific—a hippocampus-dependent reminder effect.

The context-dependent model readily accounts for these effects. Shortly after training, the uncertainty of the SR assigned to context A is low (i.e., the animal is confident in its predictive representation of context A). Introduction to context B therefore results in a large prediction error, leading the animal to (correctly) infer a new context with a new SR, leading to context-specific responding. However, as more time elapses, the uncertainty of the SR assigned to context A gradually increases (i.e., the animal becomes less confident in its predictive representation of context A, a kind of forgetting). Introduction to context B then results in a smaller prediction error, making it likely that the new observations are also assigned to context A, leading to generalization across contexts. Brief exposure to context A reverses this effect by reducing the uncertainty of the SR assigned to context A (i.e., the animal's confidence in its predictive representation of context A is restored, a kind of remembering), leading once again to context-specific responding.

Recall that the duration of context preexposure has opposing effects on learning, initially facilitating but subsequently inhibiting learning (Kiernan & Westbrook, 1993). But what if the animal is tested in a different context? In a follow-up experiment, Kiernan and Westbrook (1993) showed that longer preexposure to the training context leads to less responding in the test context, indicating that the learned association is not generalized. That is, longer context preexposure has a monotonic inhibitory effect on generalization across contexts. The context-dependent model can account for this with the same mechanism that accounts for context preexposure facilitation: longer preexposure to the training context reduces the uncertainty of its predictive estimate, leading to greater prediction errors when presented with the text context and increasing the probability that the animal will infer a new context, leading to context-specific responding.

Overall, the results of Geerts et al. (2024) suggest that rather than encoding a single monolithic predictive map of the environment, the hippocampus encodes multiple separate predictive maps, each associated with its own context. Both the context and the predictive map are inferred using Bayesian inference: learning of the predictive map is modulated by uncertainty and supports nonlocal updating. A new context is inferred when the current predictive map fails to account for current observations.

6.4 Spatial Navigation. A rich body of work points to the hippocampus as encoding a kind of cognitive map of the environment that mammals rely on for navigation in physical and abstract state spaces (O'Keefe & Dostrovsky, 1971; O'Keefe & Nadel, 1978). As we discussed in the previous section and in section 5.2, this cognitive map can be usefully interpreted as a predictive map in which states predict future expected states, consistent with the SR (see section 2.3; Stachenfeld et al., 2017). Yet despite many

studies of navigation in humans and rodents—key model species used to study spatial navigation (Epstein et al., 2017; Ekstrom & Ranganath, 2018; Ekstrom et al., 2018; Gahnstrom & Spiers, 2020; Nyberg et al., 2022; Spiers & Barry, 2015)—until recently there was no direct comparison of human and rodent navigation in a homologous task. This left open the question of whether spatial navigation across mammalian species relies on an evolutionarily conserved strategy supported by such a predictive map.

A recent study by de Cothi et al. (2022) filled this gap by designing a homologous navigation task for humans and rats. They devised a configurable open-field maze that could be reconfigured between trials, allowing experimenters to assess a hallmark aspect of spatial navigation: the ability to efficiently find detours and shortcuts. The maze consisted of a 10-by-10 grid in which squares could be blocked off by the experimenters. The maze was instantiated in a physical environment for rats and in a virtual reality environment for humans.

On each trial, the participant was placed at a starting location and had to navigate to a goal location to receive a reward (see Figure 9D). The starting location varied across trials, while the goal remained hidden at a fixed location throughout the experiment. Keeping the goal location unobservable ensured that participants could not rely on simple visual heuristics (e.g., proximity to the goal). At the same time, keeping the goal location fixed ensured that once it is identified, the key problem becomes navigating to it rather than rediscovering it. During the training phase of the experiment, all squares of the grid were accessible, allowing participants to learn an internal map of the environment. During the test phase, participants were sequentially presented with 25 different maze configurations in which various sections of the maze were blocked off. Participants completed 10 trials of each configuration before moving on to the next.

Using this task, the authors compared human and rat navigation with three types of RL algorithms:

- Model-free agent (section 2.2). No internal map of the environment; optimal policy is based on state-action value function Q, which is learned from experience, specifically, using Q-learning (see equation 2.11) with eligibility traces.
- Model-based agent (section 2.2). Full internal map of the environment (transition structure T and reward function R) is learned from experience; optimal policy is computed using tree search at decision time—specifically, using A* search.
- SR agent (section 2.3). Predictive map of the environment (SR matrix $\mathbf{M}$ and reward function R) is learned from experience; optimal policy is computed by combining SR and reward function (see equation 2.19).

The key question that the authors sought to answer was which RL strategy best explains human and rat navigation across the novel test configurations.

Across a wide range of analyses, the authors observed a consistent trend: both human and rodent behavior was most consistent with the SR agent. Humans also showed some similarity to the model-based agent, but neither species was consistent with the model-free agent.

First, the authors simulated each RL agent generatively on the same trials as the participants: they let the RL agent navigate and solve each trial as a kind of simulated participant, learning from its own experience along the way. These closed-loop[11] simulations show what overall participant behavior would look like according to each RL strategy. This revealed that:

- Model-free agents struggle on new maze configurations due to the slow learning of the Q-function, which takes many trials to propagate values from the goal location to possible starting locations.
- Model-based agents generalize quickly to new maze configurations, since local updates to the transition structure T can be immediately reflected in the tree search algorithm.
- SR agents generalize faster than model free but more slowly than model-based agents, since updates to the SR matrix $\mathbf{M}$ reach farther than updates to the Q-function, but still require several trials to propagate all the way to the possible starting locations.

Second, the authors clamped each RL agent to participant behavior—is, they fed the agent the same sequence of states and actions experienced by a given participant. These open-loop simulations show what the participant would do at each step if they were following a given RL strategy.[12] By matching these predictions with participant behavior using maximum likelihood estimation of model parameters, the authors quantified how consistent step-by-step participant behavior is with each RL strategy. For both humans and rats, this analysis revealed the greatest similarity (i.e., highest likelihood) with the SR agent, followed by the model-based agent, with the model-free agent showing the least similarity (i.e., lowest likelihood).

Third, the authors combined the above approaches by first training each fitted RL agent with the state-action sequences observed by a participant on several maze configurations and then simulating it generatively on another configuration. This hybrid open-loop training (on past configurations)/closed-loop evaluation (on a new configuration) provides a more global view than the step-by-step analysis above by allowing comparison of predicted and participant trajectories rather than individual actions. This led to several findings:

[11] We refer to the them as "closed-loop" since, at each step, the output of the RL agent (its action) is fed back to change its position on the grid, influencing the agent's input at the following step, and so on.

[12] We refer to them as "open-loop" since at each step, the output of the RL agent has no effect on its subsequent inputs or outputs.

- Configurations that were challenging for the SR agent were also challenging for biological agents, and vice versa. This pattern was less consistent for model-based and model-free agents.
- Overall directedness and direction of participant trajectories (quantified by linear and angular diffusivity) were most similar to SR trajectories.
- The step-by-step distance between participant and SR trajectories was consistently lower compared to model-based and model-free trajectories.

All of these analyses show that the SR agent best explains both human and rat behavior. Overall, the results of de Cothi et al. (2022) indicate that spatial navigation across mammalian species relies on a predictive map that is updated from experience in response to changes in the environment.

6.5 Memory. The hippocampus and the adjacent medial temporal lobe structures are also involved in another high-level cognitive function: episodic memory (Ranganath, 2010). In this section, we review an influential model of episodic memory, the temporal context model (TCM; Howard & Kahana, 2002), through the lens of RL, and show that it can be partially understood as an estimator for the SR (see section 2.3; Gershman et al., 2012). We then discuss how this property can be used in a powerful decision-making algorithm that bridges episodic memory and reinforcement learning systems (Zhou et al., 2023).

6.5.1 The Temporal Context Model. TCM is an influential model of memory encoding and retrieval originally designed to account for a number of phenomena in free recall experiments (Howard & Kahana, 2002). In these experiments, participants are asked to study a list of items and then recall as many of them as they can, in any order. Experimenters observed that recall order is often not, in fact, arbitrary: participants show better recall for recently studied items (the recency effect) and tend to recall adjacent items in the list one after the other (the contiguity effect).

TCM accounts for these phenomena by positing that the brain maintains a temporal context: a slowly drifting internal representation of recent experience that gets bound to specific experiences (memories) during encoding and can serve as a cue to bring those experiences to mind during retrieval. When participants begin recalling the studied items, the temporal context is most similar to the context associated with recently studied items (due to the slow drift), which is why they are recalled better (the recency effect). Recalling items reactivates the context associated with those items, which is similar to the context for adjacent items (again, due to the slow drift), which is why they tend to be recalled soon after (the contiguity effect). Neural evidence for drifting context comes from the finding that lingering brain activity patterns of recent stimuli predicted whether past and present stimuli are later recalled together (Chan et al., 2017). Human brain recordings

have also provided evidence for temporal context reactivation during recall (Gershman et al., 2013; Folkerts et al., 2018).

Temporal context can be formalized as weighted average of recently encountered stimulus vectors:

$$\mathbf{c}_{t+1} = (1-\omega)\mathbf{c}_t + \omega\boldsymbol{\phi}(s_t), \tag{6.15}$$

where $\mathbf{c}_t$ is the current context vector; $\boldsymbol{\phi}(s_t)$ is the feature vector for the current stimulus s_t, which could correspond to a study item (in an episodic memory setting) or a state (in a decision making setting); and the constant ω determines the drift rate—whether the context evolves slowly (low ω) or quickly (high ω).

Stimuli and contexts are bound using an association matrix $\hat{\mathbf{M}}$ which gets updated using outer-product Hebbian learning: when a new stimulus is presented, its associations with previous stimuli are strengthened in proportion to how active they are in the current context:

$$\Delta\hat{M}(i,\ j) \propto \phi_i(s_t)c_{i,t}. \tag{6.16}$$

In early studies of this model, stimuli were encoded using one-hot vectors, $\phi_i(s) = \mathbb{I}[s = i]$, although more flexible feature representations have been studied (Socher et al., 2009; Howard et al., 2011).

6.5.2 TCM as an Estimator for the SR. Note the similarity between the TCM learning rule (equation 6.16) and the SR TD learning rule (equation 2.17 in section 2.3). There are two main distinctions:

- The TCM update (equation 6.16) is modulated by the context vector $\mathbf{c}$, which is absent from the SR TD update (equation 2.17 in section 2.3).
- The Hebbian update in TCM lacks the prediction error terms from the SR TD error $\delta_M(j)$ (equation 2.18 in section 2.3).

The first distinction can be removed by setting a maximum drift rate of $\omega = 1$ in equation 6.15, which ensures that the context is always updated to the latest stimulus vector, $\mathbf{c}_{t+1} = \boldsymbol{\phi}(s_t)$. For the one-hot encoding, this means that only $\hat{M}(s_{t-1},\ j)$ will be updated in the TCM update, since $c_{i,t} = \mathbb{I}[s_{t-1} = i]$ in equation 6.16), just as in the SR TD update (equation 2.17). Conversely, introducing a context term in the SR TD update (equation 2.17) results in a generalization of TD learning using an eligibility trace (Sutton & Barto, 2018), a running average of recently visited states. This is mathematically equivalent to temporal context (equation 6.15), and can sometimes lead to faster convergence. In this way, the TCM learning rule is a generalization of the vanilla SR TD learning rule that additionally incorporates eligibility traces.

The second distinction suggests a way to, in turn, generalize the TCM update rule by replacing the Hebbian term with the prediction error term

$\delta_M(j)$ (equation 2.18) from the SR TD update. This leads to the following revised update equation for TCM:

$$\Delta\hat{M}(i, j) \propto \delta_M(j)c_{i,t}, \tag{6.17}$$

where $\delta_M(j)$ is the same as in equation 2.18. Gershman et al. (2012) showed that this new variant of TCM can be understood as directly estimating the SR using TD learning, with temporal context serving as an eligibility trace. It differs from the original version of TCM in two keys ways. First, learning is error driven rather than purely Hebbian, which means that association strength only grows if there is a discrepancy between predicted and observed stimuli (see DuBrow et al., 2017, for a discussion of empirical evidence). Second, associations are additionally learned between the context and future expected stimuli, not just the present stimulus.

This new interpretation of TCM posits that the role of temporal context is to learn predictions of future stimuli rather than to merely form associations. This makes several distinct predictions from the original version of TCM, one of which is the context repetition effect: repeating the context in which a stimulus was observed should strengthen memory for that stimulus, even if the stimulus itself was not repeated. This prediction was validated in a study by Smith et al. (2013). The authors showed participants triplets of stimuli (images in one experiment and words in another experiment), with the first two stimuli in each triplet serving as context for the third stimulus. Participants were then presented with an item-recognition test in which they had to indicate whether different stimuli are either "old" or "new." Memory performance was quantified as the proportion of test items correctly recognized as old. The key finding was that memory was better for stimuli whose context was presented repeatedly, even if the stimuli themselves were only presented once. This held for different modalities (images and words) and did not occur when context was generally not predictive of stimuli. These findings (see also Manns et al., 2015) substantiate the predicted context repetition effect and lend credence to idea that TCM learns predictions rather than mere associations.

6.5.3 Combining TCM and the SR for Decision Making. The theoretical links between TCM (see section 6.5.1) and the SR (see section 2.3) point to a broader role for episodic memory in RL. Previous studies have implicated the hippocampus in prediction and imagination (Buckner, 2010), as well as replay of salient events (Momennejad et al., 2018), consistent with some form of model-based RL or a successor model (SM; see section 2.4). More generally, episodic memory is thought to support decision making by providing the ingredients for simulating possible futures (Schacter et al., 2015). This idea is corroborated by studies of patients with episodic memory deficits, who also tend to show deficits on decision-making tasks (Gupta et al., 2009; Gutbrod et al., 2006; Bakkour et al., 2019). A related body of

work focuses on decision-by-sampling algorithms, according to which humans approximate action values by sampling from similar past experiences stored in memory (Stewart et al., 2006; Plonsky et al., 2015; Bornstein et al., 2017; Lieder et al., 2018; Bhui & Gershman, 2018).

These loosely connected ideas were knitted together in a theoretical proposal by Zhou et al. (2023) that builds on the links between TCM and the SR, showing precisely how a predictive version of TCM can support adaptive decision making. Their model incorporates two key ideas (see Figure 9E):

- During encoding (see Figure 9E, top), incoming feature vectors $\boldsymbol{\phi}(s_t)$ update a slowly drifting context $\mathbf{c}_t$ (equation 6.15). This context vector serves as an eligibility trace in a TD update rule (equation 6.17) that learns the SR estimate $\hat{\mathbf{M}}$ (Gershman et al., 2012).
- During retrieval (i.e., at decision time; see Figure 9E, middle and bottom), possible future stimuli $\mathbf{x}$ are sampled for each action using a tree-search algorithm that uses the SR as a world model, effectively turning it into a SM (see section 2.4): $p(\tilde{s}_\tau) \propto \hat{\mathbf{M}}\mathbf{c}_\tau$, where τ indexes time steps at retrieval and $\tilde{s}_0$ corresponds to the initial state of the retrieval process (the query stimulus, defined as the root of the tree). Retrieval unfolds by recursively sampling states from this process. The corresponding rewards then are averaged to compute a Monte Carlo value estimate for each action.

During the tree search, the context vector $\mathbf{c}_\tau$ can be updated with the sampled feature vector $\boldsymbol{\phi}(s_t)$ to varying degrees, dictated by the drift rate ω (see equation 6.15). This spans a continuum between updating and retrieval regimes.

At one extreme, if the drift rate during retrieval is set to its lowest value ($\omega = 0$), the context is never updated after being initialized with the query stimulus ($\mathbf{c}_\tau = \mathbf{x}_0 = \boldsymbol{\phi}(\tilde{s}_0) \forall \tau$). This results in independent and identically distributed samples from the normalized SR (i.e., the SM). In the limit of infinite samples, this reduces to simply computing action values by combining the reward function and the SR (see equation 2.19 in section 2.3), as discussed in sections 6.1 and 6.2. For finite samples, this produces an unbiased estimator of action values. Note that this estimate is only as accurate as the SR matrix $\hat{\mathbf{M}}$, which is itself an estimate of the true SR matrix $\mathbf{M}$. Hence, this regime inherits all the pros and cons of using the SR (see sections 2.3 and 6.1): it can be used to efficiently compute action values, and it can adapt quickly to changes in the reward structure R but not the transition structure T of the environment.

At the other extreme, if the drift rate during retrieval is set to its highest value ($\omega = 1$), the context is always updated to the latest sampled stimulus ($\mathbf{c}_\tau = \boldsymbol{\phi}(\tilde{s}_\tau)$). If the discount factor is minimal, $\gamma = 0$, the SR reduces to the one-step transition matrix (i.e., $\mathbf{M} = \mathbf{T}$) and the sampled stimuli s_τ are no longer independent and identically distributed but instead form a trajectory through state space that follows the transition structure T and

corresponds to a single Monte Carlo rollout. Averaging rewards from such Monte Carlo rollouts also produces an unbiased estimator of value (Sutton & Barto, 2018). This regime thus corresponds to a fully model-based algorithm (see section 2.2) and inherits all the pros and cons of that approach: it takes longer to compute action values (since trajectories need to be fully rolled out to produce unbiased estimates, requiring more samples), but it can adapt quickly to changes in both the reward structure R and the transition structure T of the environment.

Between these extremes lies a continuum ($0 < \omega < 1$) that trades off between a sampling approximation of the SR ($\omega \to 0$) and model-based Monte Carlo rollouts ($\omega \to 1$). Indeed, results from free recall experiments are consistent with such an intermediate regime (Howard & Kahana, 2002), indicating that context is partially updated during retrieval. This also raises the intriguing possibility that the brain navigates this continuum by dynamically adjusting the drift rate in a way that balances the pros and cons of both regimes, similarly to the way in which the brain arbitrates between model-based and model-free RL systems (Kool et al., 2018).

The authors also demonstrate how emotionally salient stimuli, such as high rewards, can modulate learning by producing a higher learning rate for the SR update (see equation 6.17). This introduces a kind of bias-variance trade-off: the resulting SR skews toward stimuli that were previously rewarded, which could speed up convergence (lower variance) but also induce potentially inaccurate action values (higher bias). Finally, the authors show how initiating the tree search with a retrieval context $\mathbf{c}_0$ that is associated with but different from the query stimulus feature vector $\boldsymbol{\phi}(\tilde{s}_0)$ can lead to bidirectional retrieval. This is consistent with bidirectional recall in human memory experiments and can be advantageous in problems where state transitions can be bidirectional, such as spatial navigation.

In summary, the modeling and simulation results of Zhou et al. (2023) demonstrate how a variant of TCM can be viewed as an estimator of the SR and can serve as the basis for a flexible sampling-based decision algorithm that spans the continuum between vanilla SR and fully model-based search. This work illustrates how episodic memory can be integrated with predictive representations to explain cognitive aspects of decision making.

7 Conclusion

The goal of this survey was to show how predictive representations can serve as the building blocks for intelligent computation. Modern work in AI has demonstrated the power of good representations for a variety of downstream tasks; our contribution builds on this insight, focusing on what makes representations useful for RL tasks. The idea that representing predictions is particularly useful has appeared in many different forms over the past few decades, but has really blossomed only in the past few years.

We now understand much better *why* predictive representations are useful, *what* they can be used for, and *how* to learn them.

Acknowledgments

This work was supported by the Kempner Institute for the Study of Natural and Artificial Intelligence, ARO MURI under grant W911NF-23-1-0277, and the Wellcome Trust under grant 212281/Z/18/Z.

References

Abdolshah, M., Le, H., George, T. K., Gupta, S., Rana, S., & Venkatesh, S. (2021). A new representation of successor features for transfer across dissimilar environments. In *Proceedings of the International Conference on Machine Learning* (pp. 1–9).

Adams, C. D. (1982). Variations in the sensitivity of instrumental responding to reinforcer devaluation. *Quarterly Journal of Experimental Psychology*, *34*, 77–98. 10.1080/14640748208400878

Adams, C. D., & Dickinson, A. (1981). Instrumental responding following reinforcer devaluation. *Quarterly Journal of Experimental Psychology Section B*, *33*, 109–121. 10.1080/14640748108400816

Alegre, L. N., Bazzan, A., & Da Silva, B. C. (2022). Optimistic linear support and successor features as a basis for optimal policy transfer. In *Proceedings of the International Conference on Machine Learning* (pp. 394–413).

Alver, S., & Precup, D. (2021). *Constructing a good behavior basis for transfer using generalized policy updates.* arXiv:2112.15025.

Alvernhe, A., Save, E., & Poucet, B. (2011). Local remapping of place cell firing in the Tolman detour task. *European Journal of Neuroscience*, *33*, 1696–1705. 10.1111/j.1460-9568.2011.07653.x

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., . . . Zaremba, W. (2017). Hindsight experience replay. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems*, *30*. Curran.

Aronov, D., Nevers, R., & Tank, D. W. (2017). Mapping of a non-spatial dimension by the hippocampal–entorhinal circuit. *Nature*, *543*, 719–722. 10.1038/nature21692

Auer, P. (2002). Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, *3*, 397–422.

Bakkour, A., Palombo, D. J., Zylberberg, A., Kang, Y. H., Reid, A., Verfaellie, M., . . . Shohamy, D. (2019). The hippocampus supports deliberation during value- based decisions. *eLife*, *8*, e46080. 10.7554/eLife.46080

Banino, A., Barry, C., Uria, B., Blundell, C., Lillicrap, T., Mirowski, P., . . . Kumaran, D. (2018). Vector-based navigation using grid-like representations in artificial agents. *Nature*, *557*, 429–433. 10.1038/s41586-018-0102-6

Barreto, A., Borsa, D., Hou, S., Comanici, G., Aygün, E., Hamel, P., . . . Precup, D. (2019). The option keyboard Combining skills in reinforcement learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems*, *32*. Curran.

Barreto, A., Borsa, D., Quan, J., Schaul, T., Silver, D., Hessel, M., . . . Munos, R. (2018). Transfer in deep reinforcement learning using successor features and generalised policy improvement. In *Proceedings of the International Conference on Machine Learning* (pp. 501–510).

Barreto, A., Dabney, W., Munos, R., Hunt, J. J., Schaul, T., van Hasselt, H. P., & Silver, D. (2017). Successor features for transfer in reinforcement learning. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems*, *30*. Curran.

Barreto, A., Hou, S., Borsa, D., Silver, D., & Precup, D. (2020). Fast reinforcement learning with generalized policy updates. *Proceedings of the National Academy of Sciences*, *117*(48), 30079–30087. 10.1073/pnas.1907370117

Barry, C., Hayman, R., Burgess, N., & Jeffery, K. J. (2007). Experience-dependent rescaling of entorhinal grids. *Nature Neuroscience*, *10*, 682–684. 10.1038/nn1905

Barry, C., Lever, C., Hayman, R., Hartley, T., Burton, S., O'Keefe, J., . . . Burgess, N. (2006). The boundary vector cell model of place cell firing and spatial memory. *Reviews in the Neurosciences*, *17*, 71–98. 10.1515/REVNEURO.2006.17.1-2.71

Bellman, R. E. (1957). *Dynamic programming*. Princeton University Press.

Bellmund, J. L., de Cothi, W., Ruiter, T. A., Nau, M., Barry, C., & Doeller, C. F. (2020). Deforming the metric of cognitive maps distorts memory. *Nature Human Behaviour*, *4*(2), 177–188. 10.1038/s41562-019-0767-3

Bhui, R., & Gershman, S. (2018). Decision by sampling implements efficient coding of psychoeconomic functions. *Psychological Review*, *125*, 985–1001. 10.1037/rev0000123

Bi, G.-q., & Poo, M.-m. (1998). Synaptic modifications in cultured hippocampal neurons: Dependence on spike timing, synaptic strength, and postsynaptic cell type. *Journal of Neuroscience*, *18*, 10464–10472. 10.1523/JNEUROSCI.18-24-10464.1998

Bittner, K. C., Milstein, A. D., Grienberger, C., Romani, S., & Magee, J. C. (2017). Behavioral time scale synaptic plasticity underlies CA1 place fields. *Science*, *357*, 1033–1036. 10.1126/science.aan3846

Bono, J., Zannone, S., Pedrosa, V., & Clopath, C. (2023). Learning predictive cognitive maps with spiking neurons during behavior and replays. *eLife*, *12*, e80671. 10.7554/eLife.80671

Bornstein, A. M., Khaw, M. W., Shohamy, D., & Daw, N. D. (2017). Reminders of past choices bias decisions for reward in humans. *Nature Communications*, *8*, 15958. 10.1038/ncomms15958

Borsa, D., Barreto, A., Quan, J., Mankowitz, D., Munos, R., Van Hasselt, H., . . . Schaul, T. (2019). Universal successor features approximators. In *Proceedings of the International Conference on Learning Representations*.

Bostock, E., Muller, R. U., & Kubie, J. L. (1991). Experience-dependent modifications of hippocampal place cell firing. *Hippocampus*, *1*, 193–205. 10.1002/hipo.450010207

Brantley, K., Mehri, S., & Gordon, G. J. (2021). Successor feature sets: Generalizing successor representations across policies. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35 (pp. 11774–11781). 10.1609/aaai.v35i13.17399

Brunec, I. K., & Momennejad, I. (2022). Predictive representations in hippocampal and prefrontal hierarchies. *Journal of Neuroscience*, *42*, 299–312. 10.1523/JNEUROSCI.1327-21.2021

Buckner, R. L. (2010). The role of the hippocampus in prediction and imagination. *Annual Review of Psychology*, *61*, 27–48. 10.1146/annurev.psych.60.110707.163508

Burgess, N., Barry, C., & O'Keefe, J. (2007). An oscillatory interference model of grid cell firing. *Hippocampus*, *17*, 801–812. 10.1002/hipo.20327

Bush, D., Barry, C., Manson, D., & Burgess, N. (2015). Using grid cells for navigation. *Neuron*, *87*, 507–520. 10.1016/j.neuron.2015.07.006

Carvalho, W., Filos, A., Lewis, R. L., & Singh, S. (2023). *Composing task knowledge with modular successor feature approximators*. arXiv:2301.12305.

Carvalho, W. C., Saraiva, A., Filos, A., Lampinen, A., Matthey, L., Lewis, R. L., . . . Zoran, D. (2024). Combining behaviors with the successor features keyboard. *Advances in neural information processing systems*, *36*. Curran.

Chan, S. C., Applegate, M. C., Morton, N. W., Polyn, S. M., & Norman, K. A. (2017). Lingering representations of stimuli influence recall organization. *Neuropsychologia*, *97*, 72–82. 10.1016/j.neuropsychologia.2017.01.029

Chang, C. Y., Gardner, M., Di Tillio, M. G., & Schoenbaum, G. (2017). Optogenetic blockade of dopamine transients prevents learning induced by changes in reward features. *Current Biology*, *27*, 3480–3486. 10.1016/j.cub.2017.09.049

Chevalier-Boisvert, M., Dai, B., Towers, M., de Lazcano, R., Willems, L., Lahlou, S., . . . Terry, J. (2023). Minigrid and Miniworld: Modular and customizable reinforcement learning environments for goal-oriented tasks. *CoRR*. abs/2306.13831.

Ciria, A., Schillaci, G., Pezzulo, G., Hafner, V. V., & Lara, B. (2021). Predictive processing in cognitive robotics: A review. *Neural Computation*, *33*, 1402–1432. 10.1162/neco_a_01383

Clark, A. (2013). Whatever next? Predictive brains, situated agents, and the future of cognitive science. *Behavioral and Brain Sciences*, *36*, 181–204. 10.1017/S0140525X12000477

Constantinescu, A. O., O'Reilly, J. X., & Behrens, T. E. (2016). Organizing conceptual knowledge in humans with a gridlike code. *Science*, *352*, 1464–1468. 10.1126/science.aaf0941

Courville, A. C., Daw, N. D., & Touretzky, D. S. (2006). Bayesian theories of conditioning in a changing world. *Trends in Cognitive Sciences*, *10*, 294–300. 10.1016/j.tics.2006.05.004

Dasgupta, I., & Gershman, S. J. (2021). Memory as a computational resource. *Trends in Cognitive Sciences*, *25*, 240–251. 10.1016/j.tics.2020.12.008

Daw, N. D., Gershman, S. J., Seymour, B., Dayan, P., & Dolan, R. J. (2011). Model-based influences on humans' choices and striatal prediction errors. *Neuron*, *69*, 1204–1215. 10.1016/j.neuron.2011.02.027

Daw, N. D., Niv, Y., & Dayan, P. (2005). Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control. *Nature Neuroscience*, *8*, 1704–1711. 10.1038/nn1560

Dayan, P. (1993). Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, *5*, 613–624. 10.1162/neco.1993.5.4.613

Dayan, P., & Kakade, S. (2000). Explaining away in weight space. In T. Leen, T. Dietterich, and V. Tresp (Eds.), *Advances in neural information processing systems*, *13*. Curran.

De Cothi, W., & Barry, C. (2020). Neurobiological successor features for spatial navigation. *Hippocampus*, *30*, 1347–1355. 10.1002/hipo.23246

de Cothi, W., Nyberg, N., Griesbauer, E.-M., Ghanamé, C., Zisch, F., Lefort, J. M., . . . Spiers, H. J. (2022). Predictive maps in rats and humans for spatial navigation. *Current Biology*, *32*, 3676–3689. 10.1016/j.cub.2022.06.090

de Jong, J. W., Fraser, K. M., & Lammel, S. (2022). Mesoaccumbal dopamine heterogeneity: What do dopamine firing and release have to do with it? *Annual Review of Neuroscience*, *45*, 109–129. 10.1146/annurev-neuro-110920-011929

Derdikman, D., Whitlock, J. R., Tsao, A., Fyhn, M., Hafting, T., Moser, M.-B., & Moser, E. I. (2009). Fragmentation of grid cell maps in a multicompartment environment. *Nature Neuroscience*, *12*, 1325–1332. 10.1038/nn.2396

Diba, K., & Buzsáki, G. (2007). Forward and reverse hippocampal place-cell sequences during ripples. *Nature Neuroscience*, *10*, 1241–1242. 10.1038/nn1961

Dickinson, A. (1985). Actions and habits: The development of behavioural autonomy. *Philosophical Transactions of the Royal Society of London. B, Biological Sciences*, *308*, 67–78. 10.1098/rstb.1985.0010

Dolan, R. J., & Dayan, P. (2013). Goals and habits in the brain. *Neuron*, *80*, 312–325. 10.1016/j.neuron.2013.09.007

Dordek, Y., Soudry, D., Meir, R., & Derdikman, D. (2016). Extracting grid cell characteristics from place cell inputs using non-negative principal component analysis. *eLife*, *5*, e10094. 10.7554/eLife.10094

Dorrell, W., Latham, P. E., Behrens, T. E. J., & Whittington, J. C. R. (2023). Actionable neural representations: Grid cells from minimal constraints. In *Proceedings of the Eleventh International Conference on Learning Representations*.

DuBrow, S., Rouhani, N., Niv, Y., & Norman, K. A. (2017). Does mental context drift or shift? *Current Opinion in Behavioral Sciences*, *17*, 141–146. 10.1016/j.cobeha.2017.08.003

Ego-Stengel, V., & Wilson, M. A. (2010). Disruption of ripple-associated hippocampal activity during rest impairs spatial learning in the rat. *Hippocampus*, *20*, 1–10. 10.1002/hipo.20707

Ekstrom, A. D., & Ranganath, C. (2018). Space, time, and episodic memory: The hippocampus is all over the cognitive map. *Hippocampus*, *28*, 680–687. 10.1002/hipo.22750

Ekstrom, A. D., Spiers, H. J., Bohbot, V. D., & Rosenbaum, R. S. (2018). *Human spatial navigation*. Princeton University Press.

Emukpere, D., Alameda-Pineda, X., & Reinke, C. (2021). *Successor feature neural episodic control.* arXiv:2111.03110.

Engelhard, B., Finkelstein, J., Cox, J., Fleming, W., Jang, H. J., Ornelas, S., . . . Witten, I. B. (2019). Specialized coding of sensory, motor and cognitive variables in VTA dopamine neurons. *Nature*, *570*, 509–513. 10.1038/s41586-019-1261-9

Epstein, R. A., Patai, E. Z., Julian, J. B., & Spiers, H. J. (2017). The cognitive map in humans: Spatial navigation and beyond. *Nature Neuroscience*, *20*, 1504–1513. 10.1038/nn.4656

Eysenbach, B., Salakhutdinov, R., & Levine, S. (2020). *C-learning: Learning to achieve goals via recursive classification*. arXiv:2011.08909.

Eysenbach, B., Zhang, T., Levine, S., & Salakhutdinov, R. R. (2022). Contrastive learning as goal-conditioned reinforcement learning. In S. Koyejo, S. Mohamed, A.

Agarwal, D. Belgrave, K. Cho, & A. Oh (Eds.), *Advances in neural information processing systems*, *35* (pp. 35603–35620). Curran.

Fang, C., Aronov, D., Abbott, L., & Mackevicius, E. L. (2023). Neural learning rules for generating flexible predictions and computing the successor representation. *eLife*, *12*, e80680. 10.7554/eLife.80680

Farebrother, J., Greaves, J., Agarwal, R., Lan, C. L., Goroshin, R., Castro, P. S., & Bellemare, M. G. (2023). Proto-value networks. *Scaling representation learning with auxiliary tasks*. arXiv:2304.12567.

Filos, A., Lyle, C., Gal, Y., Levine, S., Jaques, N., & Farquhar, G. (2021). PsiPhi-learning Reinforcement learning with demonstrations using successor features and inverse temporal difference learning. In *Proceedings of the International Conference on Machine Learning* (pp. 3305–3317).

Folkerts, S., Rutishauser, U., & Howard, M. W. (2018). Human episodic memory retrieval is accompanied by a neural contiguity effect. *Journal of Neuroscience*, *38*, 4200–4211. 10.1523/JNEUROSCI.2312-17.2018

Foster, D. J., & Wilson, M. A. (2007). Hippocampal theta sequences. *Hippocampus*, *17*, 1093–1099. 10.1002/hipo.20345

Friston, K., & Kiebel, S. (2009). Predictive coding under the free-energy principle. *Philosophical Transactions of the Royal Society B, Biological Sciences*, *364*, 1211–1221. 10.1098/rstb.2008.0300

Fujimoto, S., Meger, D., & Precup, D. (2021). A deep reinforcement learning approach to marginalized importance sampling with the successor representation. In *Proceedings of the International Conference on Machine Learning* (pp. 3518–3529).

Gahnstrom, C. J., & Spiers, H. J. (2020). Striatal and hippocampal contributions to flexible navigation in rats and humans. *Brain and Neuroscience Advances*, *4*, 2398212820979772. 10.1177/2398212820979772

Gardner, M. P., Schoenbaum, G., & Gershman, S. J. (2018). Rethinking dopamine as generalized prediction error. *Proceedings of the Royal Society B*, *285*(1891), 20181645. 10.1098/rspb.2018.1645

Garvert, M. M., Dolan, R. J., & Behrens, T. E. (2017). A map of abstract relational knowledge in the human hippocampal–entorhinal cortex. *eLife*, *6*, e17086. 10.7554/eLife.17086

Geerts, J. P., Chersi, F., Stachenfeld, K. L., & Burgess, N. (2020). A general model of hippocampal and dorsal striatal learning and decision making. *Proceedings of the National Academy of Sciences*, *117*(49), 31427–31437. 10.1073/pnas.2007981117

Geerts, J. P., Gershman, S. J., Burgess, N., & Stachenfeld, K. L. (2024). A probabilistic successor representation for context-dependent learning. *Psychological Review*, *131*, 578–597. 10.1037/rev0000414

Geist, M., & Pietquin, O. (2010). Kalman temporal differences. *Journal of Artificial Intelligence Research*, *39*, 483–532. 10.1613/jair.3077

George, T. M., de Cothi, W., Stachenfeld, K. L., & Barry, C. (2023). Rapid learning of predictive maps with STDP and theta phase precession. *eLife*, *12*, e80663. 10.7554/eLife.80663

Gershman, S. J. (2015). A unifying probabilistic view of associative learning. *PLOS Computational Biology*, *11*(11), e1004567. 10.1371/journal.pcbi.1004567

Gershman, S. J. (2018). The successor representation: Its computational logic and neural substrates. *Journal of Neuroscience*, *38*, 7193–7200. 10.1523/JNEUROSCI.0151-18.2018

Gershman, S. J., Blei, D. M., & Niv, Y. (2010). Context, learning, and extinction. *Psychological Review*, *117*, 197–209. 10.1037/a0017808

Gershman, S. J., Moore, C. D., Todd, M. T., Norman, K. A., & Sederberg, P. B. (2012). The successor representation and temporal context. *Neural Computation*, *24*, 1553–1568. 10.1162/NECO_a_00282

Gershman, S. J., Schapiro, A. C., Hupbach, A., & Norman, K. A. (2013). Neural context reinstatement predicts memory misattribution. *Journal of Neuroscience*, *33*, 8590–8595. 10.1523/JNEUROSCI.0096-13.2013

Girardeau, G., Benchenane, K., Wiener, S. I., Buzsáki, G., & Zugaro, M. B. (2009). Selective suppression of hippocampal ripples impairs spatial memory. *Nature Neuroscience*, *12*, 1222–1223. 10.1038/nn.2384

Gonzalez, L. S., Fisher, A. A., D'Souza, S. P., Cotella, E. M., Lang, R. A., & Robinson, J. E. (2023). Ventral striatum dopamine release encodes unique properties of visual stimuli in mice. *eLife*, *12*, e85064. 10.7554/eLife.85064

Gupta, R., Duff, M. C., Denburg, N. L., Cohen, N. J., Bechara, A., & Tranel, D. (2009). Declarative memory is critical for sustained advantageous complex decision-making. *Neuropsychologia*, *47*, 1686–1693. 10.1016/j.neuropsychologia.2009.02.007

Gupta, T., Mahajan, A., Peng, B., Böhmer, W., & Whiteson, S. (2021). Uneven, Universal value exploration for multi-agent reinforcement learning. In *Proceedings of the International Conference on Machine Learning* (pp. 3930–3941).

Gutbrod, K., Kroužel, C., Hofer, H., Müri, R., Perrig, W., & Ptak, R. (2006). Decision-making in amnesia: Do advantageous decisions require conscious knowledge of previous behavioural choices? *Neuropsychologia*, *44*, 1315–1324. 10.1016/j.neuropsychologia.2006.01.014

Ha, D., & Schmidhuber, J. (2018). Recurrent world models facilitate policy evolution. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), *Advances in neural information processing systems*, *31*. Curran.

Hafting, T., Fyhn, M., Molden, S., Moser, M.-B., & Moser, E. I. (2005). Microstructure of a spatial map in the entorhinal cortex. *Nature*, *436*, 801–806. 10.1038/nature03721

Han, D., & Tschiatschek, S. (2021). *Option transfer and SMDP abstraction with successor features.* arXiv:2110.09196.

Hansen, S., Dabney, W., Barreto, A., Van de Wiele, T., Warde-Farley, D., & Mnih, V. (2019). Fast task inference with variational intrinsic successor features. In *Proceedings of the International Conference on Learning Representations*.

Hardcastle, K., Maheswaranathan, N., Ganguli, S., & Giocomo, L. M. (2017). A multiplexed, heterogeneous, and adaptive code for navigation in medial entorhinal cortex. *Neuron*, *94*, 375–387. 10.1016/j.neuron.2017.03.025

Hart, E. E., Sharpe, M. J., Gardner, M. P., & Schoenbaum, G. (2020). Responding to preconditioned cues is devaluation sensitive and requires orbitofrontal cortex during cue-cue learning. *eLife*, *9*, e59998. 10.7554/eLife.59998

Hartley, T., Burgess, N., Lever, C., Cacucci, F., & O'Keefe, J. (2000). Modeling place fields in terms of the cortical inputs to the hippocampus. *Hippocampus*, *10*, 369–379. 10.1002/1098-1063(2000)10:4⟨369::AID-HIPO3⟩3.0.CO;2-0

Hawkins, J., & Blakeslee, S. (2004). *On intelligence*. Macmillan.
Hoang, C., Sohn, S., Choi, J., Carvalho, W., & Lee, H. (2021). Successor feature landmarks for long-horizon goal-conditioned reinforcement learning. In M. A. Beygelzimer, Y. Dauphin, P. S. Liang, & J. Wortman Vaughan (Eds.), *Advances in neural information processing systems*, *34* (pp. 26963–26975). Curran.
Holland, P. C. (2004). Relations between Pavlovian-instrumental transfer and reinforcer devaluation. *Journal of Experimental Psychology: Animal Behavior Processes*, *30*, 104–117. 10.1037/0097-7403.30.2.104
Horvitz, J. C. (2000). Mesolimbocortical and nigrostriatal dopamine responses to salient non-reward events. *Neuroscience*, *96*, 651–656. 10.1016/S0306-4522(00)00019-1
Howard, M. W., & Kahana, M. J. (2002). A distributed representation of temporal context. *Journal of Mathematical Psychology*, *46*, 269–299. 10.1006/jmps.2001.1388
Howard, M. W., Shankar, K. H., & Jagadisan, U. K. (2011). Constructing semantic representations from a gradually changing representation of temporal context. *Topics in Cognitive Science*, *3*, 48–73. 10.1111/j.1756-8765.2010.01112.x
Hunt, J., Barreto, A., Lillicrap, T., & Heess, N. (2019). Composing entropic policies using divergence correction. In *Proceedings of the International Conference on Machine Learning* (pp. 2911–2920).
Imani, E., & White, M. (2018). Improving regression performance with distributional losses. In *Proceedings of the International Conference on Machine Learning* (pp. 2157–2166).
Janner, M., Mordatch, I., & Levine, S. (2020). Gamma-models: Generative temporal difference learning for infinite-horizon prediction. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, & H. Lin (Eds.), *Advances in neural information processing systems*, *33* (pp. 1724–1735). Curran.
Janz, D., Hron, J., Mazur, P., Hofmann, K., Hernández-Lobato, J. M., & Tschiatschek, S. (2019). Successor uncertainties: Exploration and uncertainty in temporal difference learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems*, *32*. Curran.
Ji, D., & Wilson, M. A. (2007). Coordinated memory replay in the visual cortex and hippocampus during sleep. *Nature Neuroscience*, *10*, 100–107. 10.1038/nn1825
Jung, M., & McNaughton, B. (1993). Spatial selectivity of unit activity in the hippocampal granular layer. *Hippocampus*, *3*, 165–182. 10.1002/hipo.450030209
Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, *4*, 237–285. 10.1613/jair.301
Kahn, A. E., & Daw, N. D. (2023). *Humans rationally balance detailed and temporally abstract world models*. bioRxiv:2023–11.
Keiflin, R., Pribut, H. J., Shah, N. B., & Janak, P. H. (2019). Ventral tegmental dopamine neurons participate in reward identity predictions. *Current Biology*, *29*, 93–103. 10.1016/j.cub.2018.11.050
Kempka, M., Wydmuch, M., Runc, G., Toczek, J., & Jaśkowski, W. (2016). VizDoom: A doom-based AI research platform for visual reinforcement learning. In *Proceedings of the 2016 IEEE Conference on Computational Intelligence and Games* (pp. 1–8).

Kempter, R., Gerstner, W., & Van Hemmen, J. L. (1999). Hebbian learning and spiking neurons. *Physical Review E*, *59*, 4498. 10.1103/PhysRevE.59.4498

Kiernan, M., & Westbrook, R. (1993). Effects of exposure to a to-be-shocked environment upon the rat's freezing response: Evidence for facilitation, latent inhibition, and perceptual learning. *Quarterly Journal of Experimental Psychology*, *46*(3), 271–288. 10.1080/14640749308401089

Kim, S. H., Van Stralen, N., Chowdhary, G., & Tran, H. T. (2022). *Disentangling successor features for coordination in multi-agent reinforcement learning*. arXiv:2202.07741.

Kjelstrup, K. B., Solstad, T., Brun, V. H., Hafting, T., Leutgeb, S., Witter, M. P., . . . Moser, M.-B. (2008). Finite scale of spatial representation in the hippocampus. *Science*, *321*, 140–143. 10.1126/science.1157086

Kool, W., Cushman, F. A., & Gershman, S. J. (2018). Competition and cooperation between multiple reinforcement learning systems. In R. Morris, A. Bornstein, & A. Shenhav (Eds.), *Goal-directed decision making* (pp. 153–178). Elsevier.

Kropff, E., Carmichael, J. E., Moser, M.-B., & Moser, E. I. (2015). Speed cells in the medial entorhinal cortex. *Nature*, *523*, 419–424. 10.1038/nature14622

Krupic, J., Bauza, M., Burton, S., Barry, C., & O'Keefe, J. (2015). Grid cell symmetry is shaped by environmental geometry. *Nature*, *518*, 232–235. 10.1038/nature14153

Kruschke, J. K. (2008). Bayesian approaches to associative learning: From passive to active learning. *Learning and Behavior*, *36*, 210–226. 10.3758/LB.36.3.210

Kulkarni, T. D., Saeedi, A., Gautam, S., & Gershman, S. J. (2016). *Deep successor reinforcement learning*. arXiv:1606.02396.

Lee, D., Srinivasan, S., & Doshi-Velez, F. (2019). *Truly batch apprenticeship learning with deep successor features*. arXiv:1903.10077.

Lehnert, L., & Littman, M. L. (2020). Successor features combine elements of model-free and model-based reinforcement learning. *Journal of Machine Learning Research*, *21*, 8030–8082.

Leutgeb, J. K., Leutgeb, S., Moser, M.-B., & Moser, E. I. (2007). Pattern separation in the dentate gyrus and CA3 of the hippocampus. *Science*, *315*, 961–966. 10.1126/science.1135801

Lever, C., Burton, S., Jeewajee, A., O'Keefe, J., & Burgess, N. (2009). Boundary vector cells in the subiculum of the hippocampal formation. *Journal of Neuroscience*, *29*, 9771–9777. 10.1523/JNEUROSCI.1319-09.2009

Levy, W. B. (1996). A sequence predicting CA3 is a flexible associator that learns and uses context to solve hippocampal-like tasks. *Hippocampus*, *6*(6), 579–590. 10.1002/(SICI)1098-1063(1996)6:6⟨579::AID-HIPO3⟩3.0.CO;2-C

Levy, W. B., Hocking, A. B., & Wu, X. (2005). Interpreting hippocampal function as recoding and forecasting. *Neural Networks*, *18*, 1242–1264. 10.1016/j.neunet.2005.08.005

Li, L., Walsh, T. J., & Littman, M. L. (2006). Towards a unified theory of state abstraction for MDPs. In *Proceedings of the International Symposium on Artificial Intelligence and Mathematics*.

Lieder, F., Griffiths, T. L., & Hsu, M. (2018). Overrepresentation of extreme events in decision making reflects rational use of cognitive resources. *Psychological Review*, *125*, 1–32. 10.1037/rev0000074

Lisman, J., & Redish, A. D. (2009). Prediction, sequences and the hippocampus. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *364*, 1193–1201. 10.1098/rstb.2008.0316

Littman, M., & Sutton, R. S. (2001). Predictive representations of state. In T. G. Dietterich, S. Becker, & Z. Ghahramani (Eds.), *Advances in neural information processing systems*, *14*. MIT Press.

Liu, H., & Abbeel, P. (2021). APS, Active pretraining with successor features. In *Proceedings of the International Conference on Machine Learning* (pp. 6736–6747).

Liu, Q., Li, L., Tang, Z., & Zhou, D. (2018). Breaking the curse of horizon, infinite-horizon off-policy estimation. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), *Advances in neural information processing systems*, *31*. Curran.

Ljungberg, T., Apicella, P., & Schultz, W. (1992). Responses of monkey dopamine neurons during learning of behavioral reactions. *Journal of Neurophysiology*, *67*, 145–163. 10.1152/jn.1992.67.1.145

Lotter, W., Kreiman, G., & Cox, D. (2016). Deep predictive coding networks for video prediction and unsupervised learning. In *Proceedings of the International Conference on Learning Representations*.

Ludvig, E. A., Sutton, R. S., & Kehoe, E. J. (2012). Evaluating the TD model of classical conditioning. *Learning and Behavior*, *40*, 305–319. 10.3758/s13420-012-0082-6

Lynn, C. W., Kahn, A. E., Nyema, N., & Bassett, D. S. (2020). Abstract representations of events arise from mental errors in learning and memory. *Nature Communications*, *11*, 2313. 10.1038/s41467-020-15146-7

Machado, M. C., Barreto, A., Precup, D., & Bowling, M. (2023). Temporal abstraction in reinforcement learning with the successor representation. *Journal of Machine Learning Research*, *24*, 1–69.

Machado, M. C., Bellemare, M. G., & Bowling, M. (2020). Count-based exploration with the successor representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34 (pp. 5125–5133). 10.1609/aaai.v34i04.5955

Machado, M. C., Rosenbaum, C., Guo, X., Liu, M., Tesauro, G., & Campbell, M. (2017). *Eigenoption discovery through the deep successor representation*. arXiv:1710.11089.

Madarasz, T., & Behrens, T. (2019). Better transfer learning with inferred successor maps. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems*, *32*. Curran.

Manns, J. R., Galloway, C. R., & Sederberg, P. B. (2015). A temporal context repetition effect in rats during a novel object recognition memory task. *Animal Cognition*, *18*, 1031–1037. 10.1007/s10071-015-0871-3

Markus, E. J., Qin, Y.-L., Leonard, B., Skaggs, W. E., McNaughton, B. L., & Barnes, C. A. (1995). Interactions between location and task affect the spatial and directional firing of hippocampal neurons. *Journal of Neuroscience*, *15*, 7079–7094. 10.1523/JNEUROSCI.15-11-07079.1995

Mattar, M. G., & Daw, N. D. (2018). Prioritized memory access explains planning and hippocampal replay. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), *Nature Neuroscience*, *21*, 1609–1617. 10.1038/s41593-018-0232-z

McLeod, M., Lo, C. Schlegel, M., Jacobsen, A., Kumaraswamy, R., White, M., & White, A. (2021). Continual auxiliary task learning. In M. A. Beygelzimer, Y. Dauphin, P. S. Liang, & J. Wortman Vaughan (Eds.), *Advances in neural information processing systems*, *34*, 12549–12562. Curran.

McNaughton, B. L., Battaglia, F. P., Jensen, O., Moser, E. I., & Moser, M.-B. (2006). Path integration and the neural basis of the "cognitive map." *Nature Reviews Neuroscience*, *7*, 663–678. 10.1038/nrn1932

Mehta, M. R., Quirk, M. C., & Wilson, M. A. (2000). Experience-dependent asymmetric shape of hippocampal receptive fields. *Neuron*, *25*, 707–715. 10.1016/S0896-6273(00)81072-7

Momennejad, I. (2020). Learning structures: Predictive representations, replay, and generalization. *Current Opinion in Behavioral Sciences*, *32*, 155–166. 10.1016/j.cobeha.2020.02.017

Momennejad, I., & Howard, M. W. (2018). *Predicting the future with multi-scale successor representations*. bioRxiv:449470.

Momennejad, I., Otto, A. R., Daw, N. D., & Norman, K. A. (2018). Offline replay supports planning in human reinforcement learning. *eLife*, *7*, e32548. 10.7554/eLife.32548

Momennejad, I., Russek, E. M., Cheong, J. H., Botvinick, M. M., Daw, N. D., & Gershman, S. J. (2017). The successor representation in human reinforcement learning. *Nature Human Behaviour*, *1*(9), 680–692. 10.1038/s41562-017-0180-8

Morris, R. G., Garrud, P., Rawlins, J. A., & O'Keefe, J. (1982). Place navigation impaired in rats with hippocampal lesions. *Nature*, *297*, 681–683. 10.1038/297681a0

Moskovitz, T., Wilson, S. R., & Sahani, M. (2022). A first-occupancy representation for reinforcement learning. In *Proceedings of the 10th International Conference on Learning Representations*.

Muller, R. U., & Kubie, J. L. (1987). The effects of changes in the environment on the spatial firing of hippocampal complex-spike cells. *Journal of Neuroscience*, *7*, 1951–1968. 10.1523/JNEUROSCI.07-07-01951.1987

Namboodiri, V. M. K., & Stuber, G. D. (2021). The learning of prospective and retrospective cognitive maps within neural circuits. *Neuron*, *109*, 3552–3575. 10.1016/j.neuron.2021.09.034

Navratilova, Z., Hoang, L. T., Schwindel, C. D., Tatsuno, M., & McNaughton, B. L. (2012). Experience-dependent firing rate remapping generates directional selectivity in hippocampal place cells. *Frontiers in Neural Circuits*, *6*, 6. 10.3389/fncir.2012.00006

Ng, A. Y., & Russell, S. (2000). Algorithms for inverse reinforcement learning. In *Proceedings of the International Conference on Machine Learning* (p. 2).

Niv, Y. (2009). Reinforcement learning in the brain. *Journal of Mathematical Psychology*, *53*, 139–154. 10.1016/j.jmp.2008.12.005

Nyberg, N., Duvelle, É., Barry, C., & Spiers, H. J. (2022). Spatial goal coding in the hippocampal formation. *Neuron*, *110*(3), 394–422. 10.1016/j.neuron.2021.12.012

O'Keefe, J., & Dostrovsky, J. (1971). The hippocampus as a spatial map: Preliminary evidence from unit activity in the freely-moving rat. *Brain Research*, *34*(1), 171–175

O'Keefe, J., & Nadel, L. (1978). *The hippocampus as a cognitive map*. Clarendon Press.

O'Keefe, J., & Recce, M. L. (1993). Phase relationship between hippocampal place units and the EEG theta rhythm. *Hippocampus*, *3*, 317–330.

Ólafsdóttir, H. F., Barry, C., Saleem, A. B., Hassabis, D., & Spiers, H. J. (2015). Hippocampal place cells construct reward related sequences through unexplored space. *eLife*, *4*, e06063.

Ólafsdóttir, H. F., Carpenter, F., & Barry, C. (2016). Coordinated grid and place cell replay during rest. *Nature Neuroscience*, *19*, 792–794.

Ostrovski, G., Castro, P. S., & Dabney, W. (2021). The difficulty of passive learning in deep reinforcement learning. In M. A. Beygelzimer, Y. Dauphin, P. S. Liang, & J. Wortman Vaughan (Eds.), *Advances in neural information processing systems*, *34* (pp. 23283–23295). Curran.

Packard, M. G., & McGaugh, J. L. (1996). Inactivation of hippocampus or caudate nucleus with lidocaine differentially affects expression of place and response learning. *Neurobiology of Learning and Memory*, *65*, 65–72. 10.1006/nlme.1996.0007

Plonsky, O., Teodorescu, K., & Erev, I. (2015). Reliance on small samples, the wavy recency effect, and similarity-based learning. *Psychological Review*, *122*, 621–647. 10.1037/a0039413

Poole, B., Ozair, S., Van Den Oord, A., Alemi, A., & Tucker, G. (2019). On variational bounds of mutual information. In *Proceedings of the International Conference on Machine Learning* (pp. 5171–5180).

Precup, D., Sutton, R. S., & Singe, S. P. (2000). Eligibility traces for off-policy policy evaluation. In *Proceedings of the 17th International Conference on Machine Learning*.

Rabinowitz, N., Perbet, F., Song, F., Zhang, C., Eslami, S. A., & Botvinick, M. (2018). Machine theory of mind. In *Proceedings of the International Conference on Machine Learning* (pp. 4218–4227).

Ramesh, R., Tomar, M., & Ravindran, B. (2019). Successor options: An option discovery framework for reinforcement learning. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*.

Ranganath, C. (2010). A unified framework for the functional organization of the medial temporal lobes and the phenomenology of episodic memory. *Hippocampus*, *20*, 1263–1290. 10.1002/hipo.20852

Redish, A., Jensen, S., Johnson, A., & Kurth-Nelson, Z. (2007). Reconciling reinforcement learning models with behavioral extinction and renewal: Implications for addiction, relapse, and problem gambling. *Psychological Review*, *114*, 784–805. 10.1037/0033-295X.114.3.784

Rescorla, R. A., & Wagner, A. R. (1972). A theory of Pavlovian conditioning: Variations in the effectiveness of reinforcement and nonreinforcement. In A. Black & W. Prokasy (Eds.), *Classical conditioning II: Current research and theory* (pp. 64–99). Appleton-Century-Crofts.

Rothschild, G., Eban, E., & Frank, L. M. (2017). A cortical–hippocampal–cortical loop of information processing during memory consolidation. *Nature Neuroscience*, *20*, 251–259. 10.1038/nn.4457

Russek, E. M., Momennejad, I., Botvinick, M. M., Gershman, S. J., & Daw, N. D. (2017). Predictive representations can link model-based reinforcement learning to model-free mechanisms. *PLOS Computational Biology*, *13*, e1005768. 10.1371/journal.pcbi.1005768

Russo, D. J., Van Roy, B., Kazerouni, A., Osband, I., & Wen, Z. (2018). *A tutorial on Thompson sampling*. Now Publishers.

Samvelyan, M., Rashid, T., De Witt, C. S., Farquhar, G., Nardelli, N., Rudner, T. G., . . . Whiteson, S. (2019). *The StarCraft multi-agent challenge.* arXiv:1902.04043.

Sanders, H., Wilson, M. A., & Gershman, S. J. (2020). Hippocampal remapping as hidden state inference. *eLife*, *9*, e51140. 10.7554/eLife.51140

Sargolini, F., Fyhn, M., Hafting, T., McNaughton, B. L., Witter, M. P., Moser, M.-B., & Moser, E. I. (2006). Conjunctive representation of position, direction, and velocity in entorhinal cortex. *Science*, *312*, 758–762. 10.1126/science.1125572

Schacter, D. L., Benoit, R. G., De Brigard, F., & Szpunar, K. K. (2015). Episodic future thinking and episodic counterfactual thinking: Intersections between memory and decisions. *Neurobiology of Learning and Memory*, *117*, 14–21. 10.1016/j.nlm.2013.12.008

Schapiro, A. C., Rogers, T. T., Cordova, N. I., Turk-Browne, N. B., & Botvinick, M. M. (2013). Neural representations of events arise from temporal community structure. *Nature Neuroscience*, *16*, 486–492. 10.1038/nn.3331

Schramm, L., Deng, Y., Granados, E., & Boularias, A. (2023). Usher: Unbiased sampling for hindsight experience replay. In *Proceedings of theConference on Robot Learning* (pp. 2073–2082).

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., . . . Silver, D. (2020). Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, *588*, 604–609. 10.1038/s41586-020-03051-4

Schultz, W., Dayan, P., & Montague, P. R. (1997). A neural substrate of prediction and reward. *Science*, *275*, 1593–1599. 10.1126/science.275.5306.1593

Sharpe, M. J., Chang, C. Y., Liu, M. A., Batchelor, H. M., Mueller, L. E., Jones, J. L., . . . Schoenbaum, G. (2017). Dopamine transients are sufficient and necessary for acquisition of model-based associations. *Nature Neuroscience*, *20*, 735–742. 10.1038/nn.4538

Silva, D., Feng, T., & Foster, D. J. (2015). Trajectory events across hippocampal place cells require previous experience. *Nature Neuroscience*, *18*, 1772–1779. 10.1038/nn.4151

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., . . . Hassibis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, *529*, 484–489. 10.1038/nature16961

Smith, T. A., Hasinski, A. E., & Sederberg, P. B. (2013). The context repetition effect: Predicted events are remembered better, even when they don't happen. *Journal of Experimental Psychology General*, *142*(4), 1298–1308. 10.1037/a0034067

Socher, R., Gershman, S., Sederberg, P., Norman, K., Perotte, A., & Blei, D. (2009). A Bayesian analysis of dynamics in free recall. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, & A. Culotta (Eds.), *Advances in neural information processing systems*, *22*. Curran.

Solstad, T., Boccara, C. N., Kropff, E., Moser, M.-B., & Moser, E. I. (2008). Representation of geometric borders in the entorhinal cortex. *Science*, *322*, 1865–1868. 10.1126/science.1166466

Sorscher, B., Mel, G., Ganguli, S., & Ocko, S. (2019). A unified theory for the origin of grid cells through the lens of pattern formation. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems*, *32*. Curran.

Spiers, H. J., & Barry, C. (2015). Neural systems supporting navigation. *Current Opinion in Behavioral Sciences*, *1*, 47–55. 10.1016/j.cobeha.2014.08.005

Stachenfeld, K. L., Botvinick, M. M., & Gershman, S. J. (2017). The hippocampus as a predictive map. *Nature Neuroscience*, *20*(11), 1643–1653. 10.1038/nn.4650

Stalnaker, T. A., Howard, J. D., Takahashi, Y. K., Gershman, S. J., Kahnt, T., & Schoenbaum, G. (2019). Dopamine neuron ensembles signal the content of sensory prediction errors. *eLife*, *8*, e49315. 10.7554/eLife.49315

Starkweather, C. K., & Uchida, N. (2021). Dopamine signals as temporal difference errors: Recent advances. *Current Opinion in Neurobiology*, *67*, 95–105. 10.1016/j.conb.2020.08.014

Stensola, H., Stensola, T., Solstad, T., Frøland, K., Moser, M.-B., & Moser, E. I. (2012). The entorhinal grid map is discretized. *Nature*, *492*, 72–78. 10.1038/nature11649

Stewart, N., Chater, N., & Brown, G. D. (2006). Decision by sampling. *Cognitive Psychology*, *53*, 1–26. 10.1016/j.cogpsych.2005.10.003

Sutton, R., & Barto, A. (1990). Time-derivative models of Pavlovian reinforcement. In M. Gabriel & J. Moore (Eds.), *Learning and computational neuroscience. Foundations of adaptive networks* (pp. 497–537). MIT Press.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT Press.

Sutton, R. S., Precup, D., & Singh, S. (1999). Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, *112*, 181–211. 10.1016/S0004-3702(99)00052-1

Takahashi, Y. K., Batchelor, H. M., Liu, B., Khanna, A., Morales, M., & Schoenbaum, G. (2017). Dopamine neurons respond to errors in the prediction of sensory features of expected rewards. *Neuron*, *95*, 1395–1405. 10.1016/j.neuron.2017.08.025

Tanni, S., De Cothi, W., & Barry, C. (2022). State transitions in the statistically stable place cell population correspond to rate of perceptual change. *Current Biology*, *32*, 3505–3514. 10.1016/j.cub.2022.06.046

Thakoor, S., Rowland, M., Borsa, D., Dabney, W., Munos, R., & Barreto, A. (2022). Generalised policy improvement with geometric policy composition. In *Proceedings of the International Conference on Machine Learning* (pp. 21272–21307).

Thompson, W. R (1933). On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, *25*, 285–294. 10.1093/biomet/25.3-4.285

Tomov, M. S., Schulz, E., & Gershman, S. J. (2021). Multi-task reinforcement learning in humans. *Nature Human Behaviour*, *5*(6), 764–773. 10.1038/s41562-020-01035-y

Touati, A., Rapin, J., & Ollivier, Y. (2022). *Does zero-shot reinforcement learning exist?* arXiv:2209.14935.

Tsividis, P. A., Loula, J., Burga, J., Foss, N., Campero, A., Pouncy, T., . . . Tenenbaum, J. B. (2021). *Human-level reinforcement learning through theory-based modeling, exploration, and planning*. arXiv:2107.12544.

Veeriah, V., Hessel, M., Xu, Z., Rajendran, J., Lewis, R. L., Oh, J., . . . Singh, S. (2019). Discovery of useful questions as auxiliary tasks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems*, *32*. Curran.

Vértes, E., & Sahani, M. (2019). A neurally plausible model learns successor representations in partially observable environments. In H. Wallach, H. Larochelle, A.

Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems*, *32*. Curran.

Wärnberg, E., & Kumar, A. (2023). Feasibility of dopamine as a vector-valued feedback signal in the basal ganglia. *Proceedings of the National Academy of Sciences*, *120*, e2221994120.

Watkins, C. J., & Dayan, P. (1992). Q-learning. *Machine Learning*, *8*, 279–292.

Whittington, J. C., Muller, T. H., Mark, S., Chen, G., Barry, C., Burgess, N., & Behrens, T. E. (2020). The Tolman-Eichenbaum machine: Unifying space and relational memory through generalization in the hippocampal formation. *Cell*, *183*, 1249–1263.

Wilson, M A., & McNaughton, B. L. (1994). Reactivation of hippocampal ensemble memories during sleep. *Science*, *265*, 676–679. 10.1126/science.8036517

Winocur, G., Frankland, P. W., Sekeres, M., Fogel, S., & Moscovitch, M. (2009). Changes in context-specificity during memory reconsolidation: Selective effects of hippocampal lesions. *Learning and Memory*, *16*(11), 722–729. 10.1101/lm.1447209

Wittkuhn, L., Krippner, L. M., & Schuck, N. W. (2022). *Statistical learning of successor representations is related to on-task replay.* bioRxiv:2022–02.

Yu, C., Burgess, N., Sahani, M., & Gershman, S. (2023). Successor-predecessor intrinsic exploration. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, & S. Levine (Eds.), *Advances in neural information processing systems*, *36*. Curran.

Zahavy, T., O'Donoghue, B., Barreto, A., Mnih, V., Flennerhag, S., & Singh, S. (2021). *Discovering diverse nearly optimal policies with successor features*. arXiv:2106.00669.

Zahavy, T., Veeriah, V., Hou, S., Waugh, K., Lai, M., Leurent, E., . . . Singh, S. (2023). *Diversifying AI: Towards creative chess with AlphaZero*. arXiv:2308.09175.

Zhang, J., Springenberg, J. T., Boedecker, J., & Burgard, W. (2017). Deep reinforcement learning with successor features for navigation across similar environments. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 2371–2378).

Zheng, C., Salakhutdinov, R., & Eysenbach, B. (2023). *Contrastive difference predictive coding.* arXiv:2310.20141.

Zhou, C. Y., Talmi, D., Daw, N., & Mattar, M. G. (2023). *Episodic retrieval for model-based evaluation in sequential decision tasks*. PsyArXiv.

Zhu, Y., Gordon, D., Kolve, E., Fox, D., Fei-Fei, L., Gupta, A., . . . Farhadi (2017). Visual semantic planning using deep successor representations. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 483–492).

Received February 9, 2024; accepted June 10, 2024.