Composing Entropic Policies using Divergence Correction

Jonathan J Hunt [1] Andre Barreto [1] Timothy P Lillicrap [1] Nicolas Heess [1]

Abstract

Composing skills mastered in one task to solve novel tasks promises dramatic improvements in the data efficiency of reinforcement learning. Here, we analyze two recent works composing behaviors represented in the form of action-value functions and show that they perform poorly in some situations. As part of this analysis, we extend an important generalization of policy improvement to the maximum entropy framework and introduce an algorithm for the practical implementation of successor features in continuous action spaces. Then we propose a novel approach which addresses the failure cases of prior work and, in principle, recovers the optimal policy during transfer. This method works by explicitly learning the (discounted, future) divergence between base policies. We study this approach in the tabular case and on non-trivial continuous control problems with compositional structure and show that it outperforms or matches existing methods across all tasks considered.

1. Introduction

Reinforcement learning (RL) algorithms coupled with powerful function approximators have recently achieved a series of successes (Mnih et al., 2015; Silver et al., 2016; Lillicrap et al., 2015; Kalashnikov et al., 2018). Unfortunately, all of these approaches still require a large number of interactions with the environment. One reason for this is that the algorithms are typically applied "from scratch," rather than leveraging experience from prior tasks. This reduces their applicability in domains where generating experience is expensive, or learning from scratch is challenging.

In humans, the development of complex motor skills, such as bipedal locomotion or high-speed manipulation, also requires large amounts of experience and practice (Adolph et al., 2012; Haith & Krakauer, 2013) However, once such skills have been acquired humans rapidly put them to work in new contexts and to solve new tasks, suggesting transfer learning as an important mechanism for data efficiency.

Motivated by this observation, we are interested in RL methods for transfer that are suitable for high-dimensional motor control. We focus on model-free approaches which are evident in human motor control (Haith & Krakauer, 2013) and underlie the recent successes of Deep RL cited above.

Transfer may be especially valuable in domains where a small set of skills can be composed, in different combinations, to solve a variety of tasks. Different notions of compositionality have been considered in the RL and robotics literature. For instance, 'options' tend to be associated with discrete units of behavior that can be sequenced, thus emphasizing composition in time (Precup et al., 1998). In this paper we are concerned with a rather distinct notion of compositionality, namely how to combine and blend potentially concurrent behaviors. This form of composition is particularly relevant in high-dimensional continuous action spaces, where it is possible to achieve more than one task simultaneously (e.g. walking somewhere while juggling).

One approach to this challenge is via the composition of task rewards. Specifically, we are interested in the following question: If we have previously solved a set of tasks with similar transition dynamics but different reward functions, how can we leverage this knowledge to solve new tasks which can be expressed as a convex combination of those reward functions?

This question has recently been studied in two independent lines of work: by Barreto et al. (2017; 2018) in the context of successor feature (SF) representations used for Generalized Policy Improvement (GPI) with deterministic policies, and by Haarnoja et al. (2018a); van Niekerk et al. (2018) in the context of maximum entropy (max-ent) policies. These approaches operate in distinct frameworks but both achieve skill composition by combining the Q-functions associated with previously learned skills.

In this paper, we clarify the relationship between the two approaches and show that both perform well in some situations but that they fail in complementary ways. We introduce a novel method of behavior composition that can consistently

[1]DeepMind. Correspondence to: Jonathan J Hunt <jjhunt@google.com>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

achieve good performance.

Our contributions are as follows:

1. We introduce succcessor features (SF) in the context of the maximum entropy framework and extend the GPI theorem to the max-ent objective (max-ent GPI).
2. We provide an analysis of when GPI, and compositional "optimism" (CO) (Haarnoja et al., 2018a) perform poorly. We highlight these failure cases with tabular discrete action tasks and challenging continuous control tasks.
3. We propose a correction term – which we call Divergence Correction (DC)– based on the Rényi divergence between policies which allows us, in principle, to recover the optimal policy for transfer for any convex combination of rewards.
4. We demonstrate a practical algorithm, which relies of adaptive importance sampling, for zero-shot transfer with DC or max-ent GPI in continuous action spaces. We compare the approaches introduced here, max-ent GPI and DC, with compositional optimism (Haarnoja et al., 2018a) and Conditional Q functions (Schaul et al., 2015) in a variety of non-trivial continuous action transfer tasks.

2. Background

2.1. Multi-task RL

We consider Markov Decision Processes defined by the tuple $\mathcal{M}$ containing: a state space $\mathcal{S}$, action space $\mathcal{A}$, start state distribution $p(s_1)$, transition function $p(s_{t+1}|s_t, a_t)$, discount $\gamma \in [0, 1)$ and reward function $r(s_t, a_t, s_{t+1})$. We aim to find an optimal policy $\pi(a|s) : \mathcal{S} \to \mathcal{P}(\mathcal{A})$, which is one that maximises the discounted expected return from any state $J(\pi) = \mathbb{E}_{\pi,\mathcal{M}} \left[\sum_{\tau=t}^{\infty} \gamma^{\tau-t} r_\tau\right]$ where the expected return is dependent on the policy π and the MDP $\mathcal{M}$.

We formalize transfer as in Barreto et al. (2017); Haarnoja et al. (2018a), as the desire to perform well across all tasks in a set $\mathcal{M} \in \mathcal{T}'$ (without direct experience on these tasks) after having learned policies for tasks $\mathcal{M} \in \mathcal{T}$. We assume that $\mathcal{T}$ and $\mathcal{T}'$ are related in two ways: all tasks share the same state transition function, and tasks in $\mathcal{T}'$ can be expressed as convex combinations of rewards associated with tasks in set $\mathcal{T}$. We write the reward functions for tasks in $\mathcal{T}$ as the vector $\boldsymbol{\phi} = (r_1, r_2, \dots)$, so tasks in $\mathcal{T}'$ can be expressed as $r_{\mathbf{w}} = \boldsymbol{\phi} \cdot \mathbf{w}$. Our theoretical results assume that we have learned optimal policies for all tasks in $\mathcal{T}$.

For clarity, we focus on the combination of only two policies, that is $\mathbf{w}$ has only 2 non-zero entries and so we can write $r_b = br_i + (1 - b)r_j$ ($b \in [0, 1]$). As we discuss later, the approaches we consider can be extended to more than two tasks. We refer to a transfer method as optimal, if it results in an optimal policy for the transfer task in $\mathcal{T}'$, using only experience on tasks $\mathcal{T}$. As in most prior work in Deep RL, theoretical guarantees of optimality can only be approximately achieved in practise.

2.2. Successor Features

Successor Features (SF) (Dayan, 1993) and Generalised Policy Improvement (GPI) (Barreto et al., 2017; 2018) provide a principled solution to transfer in the setting defined above. SF make the additional assumption that the reward feature $\boldsymbol{\phi}$ is fully observable, that is, the agent always has access to the rewards of all tasks in $\mathcal{T}$ but not $\mathcal{T}'$ during training.

The key insight of SF is that linearity of the reward $r_{\mathbf{w}}$ with respect to the features $\boldsymbol{\phi}$ implies the following decomposition of the action value of policy π on task $r_{\mathbf{w}}$:

$$
\begin{aligned}
Q^\pi_{\mathbf{w}}(s_t, a_t) &= \mathbb{E}_\pi \left[\sum_{\tau=t}^{\infty} \gamma^{\tau-t} \boldsymbol{\phi}_\tau \cdot \mathbf{w} | a_t\right] \\
&= \mathbb{E}_\pi \left[\sum_{i=t}^{\infty} \gamma^{\tau-t} \boldsymbol{\phi}_\tau | a_t\right] \cdot \mathbf{w} \equiv \boldsymbol{\psi}^\pi(s_t, a_t) \cdot \mathbf{w}
\end{aligned}
$$

where $\boldsymbol{\psi}^\pi$ is the expected discounted sum of features $\boldsymbol{\phi}$ induced by policy π. The SF decomposition allows us to compute the value of an existing policy π on a new task $r_{\mathbf{w}}$.

GPI provides a principled way to use this value information to compose n existing polices $\pi_1, \pi_2, ..., \pi_n$ indexed by i to solve task $r_{\mathbf{w}}$. Namely, we act according to the deterministic GPI policy $\pi^{GPI}_{\mathbf{w}}(s_t) \equiv \arg\max_{a_t} Q^{GPI}_{\mathbf{w}}(s_t, a_t)$ where

$$
Q^{GPI}_{\mathbf{w}}(s_t, a_t) \equiv \max_i Q^{\pi_i}_{\mathbf{w}}(s_t, a_t) = \max_i \boldsymbol{\psi}^{\pi_i}(s, a) \cdot \mathbf{w}
$$

The GPI theorem guarantees the GPI policy has a return at least as good as any component policy, that is, $V^{\pi^{GPI}}_{\mathbf{w}}(s) \geq \max_i V^{\pi_i}_{\mathbf{w}}(s) \, \forall s \in \mathcal{S}$.

Note that SF and GPI are separate concepts, the GPI theorem does not require the use of SFs. SFs provide an efficient mechanism for computing the value of existing policies on a new task, and GPI provides a principled way to make use of this information to compose the existing policies.

2.3. Maximum Entropy RL

The maximum entropy (max-ent) RL objective augments the reward to favor entropic solutions

$$
J(\pi) = \mathbb{E}_{\pi,\mathcal{M}} \left[\sum_{\tau=t}^{\infty} \gamma^{\tau-t} (r_\tau + \alpha H[\pi(\cdot|s_\tau)])\right] \quad (1)
$$

where α is a parameter that determines the relative importance of the entropy term.

This objective has been considered in a number of works including (Kappen, 2005; Todorov, 2009; Haarnoja et al., 2017; 2018a; Ziebart et al., 2008; Fox et al., 2016).

We define the action-value Q^π associated with eq. 1 as

$$Q^\pi(s_t,a_t) \equiv r_t + \mathbb{E}_\pi\left[\sum_{\tau=t+1}^\infty \gamma^{\tau-t}(r_\tau + \alpha H[\pi(\cdot|s_\tau)])\right] \quad (2)$$

(notice $Q^\pi(s_t,a_t)$ does not include the entropy term for the state s_t). Soft Q iteration where the policy $\pi(a_t|s_t) \propto \exp(\frac{1}{\alpha}Q(s_t,a_t))$ is implicitly defined by Q converges to the optimal policy with standard assumptions (Haarnoja et al., 2017).

$$\begin{aligned} Q(s_t,a_t) &\leftarrow r(s_t,a_t,s_{t+1}) + \gamma\mathbb{E}_{p(s_{t+1}|s_t,a_t)}\left[V(s_{t+1})\right] \\ V(s_t) &\leftarrow \mathbb{E}_\pi\left[Q(s_t,a_t)\right] + \alpha H[\pi(\cdot|s_t)] \\ &= \alpha \log \int_{\mathcal{A}} \exp(\frac{1}{\alpha}Q(s_t,a_t))da \equiv \alpha \log Z(s_t) \end{aligned}$$

2.4. Compositional Optimisim (CO)

Haarnoja et al. (2018a) introduced a simple approach to policy composition in the max-ent framework by approximating the optimal action-value for the transfer task $r_b = br_i + (1-b)r_j$ from the optimal action-values of the component tasks Q^i and Q^j

$$Q_b^{CO}(s,a) \equiv bQ^i(s,a) + (1-b)Q^j(s,a). \quad (3)$$

When using Boltzmann policies defined by Q, the resulting policy, $\pi_b^{CO}(a|s) \propto \exp(\frac{1}{\alpha}Q_b^{CO}(s,a))$, is the product distribution of the two component policies. We refer to π_b^{CO} as the compositionally "optimistic" (CO) policy, as it acts according to an over-estimate of the action value ($Q_b^{CO}(s,a) \geq Q_b^*(s,a)$) by assuming the optimal returns of Q^i and Q^j will be, simultaneously, achievable.

3. Composing Policies in Max-Ent Reinforcement Learning

In this section we present two novel approaches for transfer in the max-ent framework. In section 4 we then outline a practical algorithm using these results.

3.1. Max-Ent Successor Features and Generalized Policy Improvement

We introduce max-ent SF, which provide a mechanism for computing the value of a maximum entropy policy under any convex combination of rewards. We then show that the GPI theorem (Barreto et al., 2017) holds for maximum entropy policies.

We define the action-dependent SF to include the entropy of the policy, excluding the current state, analogous to the max-entropy definition of Q^π in (2):

$$\begin{aligned} \boldsymbol{\psi}^\pi(s_t,a_t) &\equiv \boldsymbol{\phi}_t + \mathbb{E}_\pi\left[\sum_{\tau=i+1}^\infty \gamma^{\tau-t}(\boldsymbol{\phi}_\tau + \alpha\mathbf{1}\cdot H[\pi(\cdot|s)])\right] \\ &= \phi_t + \gamma\mathbb{E}_{p(s_{t+1}|s_t,a_t)}\left[\boldsymbol{\Upsilon}(s_{t+1})\right] \end{aligned}$$

where $\mathbf{1}$ is a vector of ones of the same dimensionality as ϕ and we define the state-dependent successor features as the expected $\boldsymbol{\psi}^\pi$ in analogy with $V^\pi(s)$:

$$\boldsymbol{\Upsilon}^\pi(s) \equiv \mathbb{E}_{a\sim\pi(\cdot|s)}\left[\boldsymbol{\psi}^\pi(s,a)\right] + \alpha\mathbf{1}\cdot H[\pi(\cdot|s)]. \quad (4)$$

The max-entropy action-value of π for any convex combination of rewards $\mathbf{w}$ is then given by $Q_\mathbf{w}^\pi(s,a) = \boldsymbol{\psi}^\pi(s,a)\cdot\mathbf{w}$. Max-ent SF allow us to estimate the action-value of previous policies on a new task. We show that, as in the deterministic case, there is a principled way to combine multiple policies using their action-values on task $\mathbf{w}$.

Theorem 3.1 (Max-Ent Generalized Policy Improvement) *Let $\pi_1,\pi_2,...,\pi_n$ be n policies with α-max-ent action-value functions $Q^1,Q^2,...,Q^n$ and value functions $V^1,V^2,...,V^n$. Define*

$$\pi(a|s) \propto \exp\left(\tfrac{1}{\alpha}\max_i Q^i(s,a)\right).$$

Then,

$$Q^\pi(s,a) \geq \max_i Q^i(s,a) \text{ for all } s\in\mathcal{S},\, a\in\mathcal{A}, \quad (5)$$

$$V^\pi(s) \geq \max_i V^i(s) \text{ for all } s\in\mathcal{S}, \quad (6)$$

where $Q^\pi(s,a)$ and $V^\pi(s)$ are the α-max-ent action-value and value function respectively of π.

Proof: See appendix A.1.

In our setup, we learn $\boldsymbol{\psi}^{\pi_i}(s,a)$, the SFs of policies π_i for each task in $\mathcal{T}$, we define the max-ent GPI policy for task $\mathbf{w}\in\mathcal{T}'$ as $\pi_\mathbf{w}^{GPI}(a|s) \propto \exp(\frac{1}{\alpha}\max_i Q_\mathbf{w}^{\pi_i}(s,a)) = \exp(\frac{1}{\alpha}\max_i \boldsymbol{\psi}^{\pi_i}(s,a)\cdot\mathbf{w})$. In contrast to CO, GPI can be seen as acting "pessimistically" as it always acts according to a lower bound (equation 5) on the action-value.

3.2. Divergence Correction (DC)

Both max-ent GPI we presented above, and CO can, in different ways, fail to transfer well in some situations (fig. 1). Neither approach consistently acts optimally during transfer, even if all component terms are known exactly.

Here we show, at the cost of learning a function conditional on the task weightings b, it is in principle possible to recover the optimal max-ent policy for the transfer tasks, without direct experience on those tasks, by correcting for the compositional optimism bias in Q_b^{CO}.

The correction term for CO uses a property noted, but not exploited in Haarnoja et al. (2018a). The bias in Q^{CO} is related to the the discounted sum of Rényi divergences of the two component policies. Intuitively, if the two policies induce trajectories with low divergence between the policies in each state, the CO assumption that both policies can achieve good returns simultaneously is approximately correct. When the divergences are large (i.e. the two policies do not agree on what action to take), the CO assumption is being overly optimistic.

Theorem 3.2 (DC Optimality) *Let π_i, π_j be α max-ent optimal policies for tasks with rewards r_i and r_j with max-ent action-value functions Q^i, Q^j. Define $C_b^\infty(s_t, a_t)$ as the fixed point of*

$$C_b^{(k+1)}(s_t, a_t) = -\alpha\gamma\mathbb{E}_{p(s_{t+1}|s_t,a_t)}\Big[\log \int_{\mathcal{A}} \pi_i(a_{t+1}|s_{t+1})^b \pi_j(a_{t+1}|s_{t+1})^{(1-b)} \exp\big(-\frac{1}{\alpha} C_b^{(k)}(s_{t+1}, a_{t+1})\big) da_{t+1}\Big]$$

Given the conditions for Soft Q convergence, the max-ent optimal $Q_b^*(s,a)$ for $r_b = br_i + (1-b)r_j$ is

$$Q_b^*(s,a) = bQ^i(s,a) + (1-b)Q^j(s,a) - C_b^\infty(s,a) \quad \forall s \in \mathcal{S}, a \in \mathcal{A}, b \in [0,1]$$

Proof: See appendix A.2.

We call this Divergence Correction (DC) as the quantity C_b^∞ is related to the Rényi divergence between policies (see appendix A.2 for details). Learning C_b^∞ does not require any additional information (in principle) than that required to learn policies π_i and π_j. Unlike with SF, it is not necessary to observe other task rewards while training the policies. On the other hand, SF/GPI can be used to combine any number of tasks with arbitrary weight vectors $\mathbf{w}$, the difficulty of estimating C^∞ increases significantly if more than two tasks are combined (see supplementary theorem A1).

Supplementary Table 1 provides a comparison on the properties of the methods we consider here. We also compare with simply learning a conditional Q function $Q(s,a|b)$ (CondQ) (e.g. Schaul et al., 2015; Andrychowicz et al., 2017). As with GPI, this requires observing the full set of task features $\boldsymbol{\phi}$, in order to compute r_b for arbitrary b.

We have introduced two new approaches to max-ent transfer composition and described their properties: max-ent SF/GPI and DC. Now we address the question of how to practically learn and sample with these approaches in continuous action spaces.

4. Adaptive Importance Sampling for Boltzman Policies Algorithm

Robotic systems with high-dimensional continuous action spaces are promising use cases for the ideas presented above, particularly as data efficiency is often a paramount concern. Such control problems may allow for multiple solutions, and often contain exploitable compositional structure.

Unfortunately, learning and sampling of general Boltzmann policies defined over continuous action spaces is challenging. One approach is to fit an expressible, tractable sampler, such as a stochastic neural network to approximate π_i (e.g. Haarnoja et al., 2018a). This approach works well when learning a single policy. However, during transfer this may require learning a new sampler for each new transfer task. Here we want to zero-shot transfer by sampling from a newly synthesized action-value function online, which precludes fitting a sampler. To address this issue we introduce Adaptive Importance Sampling for Boltzmann Policies (AISBP), which provides a practical solution to this challenge.

We use parametric approximators (e.g. neural networks) and denote their parameters θ, including the soft action-value for reward i: $Q^i_{\theta_Q}(s,a)$; the associated soft value function $V^i_{\theta_V}(s)$ and a proposal distribution $q^i_{\theta_q}(a|s)$, which is used for importance sampling the policy (we will sometimes drop the task index i for notational simplicity, and write the losses for a single policy).

We use an off-policy algorithm, so that experience generated by training on policy π_i can be used to improve policy π_j. This is important to ensure that $Q^j_{\theta_Q}(s,a)$ is a good approximation of the action-value function in states that are likely under π_j but unlikely under π_i (suppl. F discusses issues of exploration and coverage of the state space under function approximation). Training experience across all tasks is stored in a replay buffer R, and mini-batches of experience are sampled uniformly from the replay.

The proposal distribution is a mixture of M truncated Normal distributions $\mathcal{N}_T$, truncated to the square $a \in [-1,1)^n$ with diagonal covariances.

$$q_{\theta_q}(a|s) = \frac{1}{M}\sum_{m=1}^{M} \mathcal{N}_T(a; \mu^m_{\theta_q}(s), \sigma^m_{\theta_q}(s), -1, 1) \quad (7)$$

The proposal distribution is optimized by minimizing the forward KL divergence with the Boltzmann policy $\pi(a|s) \propto \exp\frac{1}{\alpha}Q_{\theta_Q}(s,a)$. This KL is "zero avoiding" and over-estimates the support of π (Murphy, 2012) which is desirable for a proposal distribution (Gu et al., 2015). The proposal loss is

$$\mathcal{L}(\theta_q) = \mathbb{E}_R\left[\mathbb{E}_{a\sim\pi(\cdot|s)}[\log\pi(a|s_t) - \log q_{\theta_q}(a|s_t)]\right] \quad (8)$$

where the expectation is over the replay buffer state density.

The inner expectation in the proposal loss itself requires sampling from π. We approximate this expectation by self-normalized importance sampling and use a target proposal distribution $p(a_t|s_t)$ which is a mixture distribution consisting of the proposals for all policies along with a uniform distribution. For batchsize B and N proposal action samples the estimator of the proposal loss is then

$$\mathcal{L}(\theta_q) \approx -\frac{1}{B}\sum_{k=1}^{B}\sum_{l=1}^{N} w_{kl} \log q_{\theta_q}(a_{kl}|s_k)$$
$$w'_{kl} = \frac{\exp \frac{1}{\alpha}(Q_{\theta_Q}(s_k, a_{kl}))}{p(a_{kl}|s_k)}; \;\; w_{kl} = \frac{w'_{kl}}{\sum_{m=1}^{N} w'_{km}}$$

By restricting the proposal distribution to mixtures of (truncated) Gaussians, we ensure the product of the proposal distributions is tractable. We make use of this product proposal during transfer (see supplementary C.3).

The policy is improved using Soft Q iteration (eq. 2). The value function loss is defined as the L2 error on the Soft Q estimate of value

$$\mathcal{L}(\theta_V) = \mathbb{E}_R\Big[\frac{1}{2}\Big(V_{\theta_V}(s_t) - \alpha \log \int_{\mathcal{A}} \exp(\frac{1}{\alpha} Q_{\theta_Q}(s_t, a)) da\Big)^2\Big] \quad (9)$$

which is estimated using importance sampling to compute the integral.

$$\mathcal{L}(\theta_V) \approx \frac{1}{2B}\sum_{l=1}^{B} (V_{\theta_V}(s_l) - \alpha \log Z)^2$$
$$Z = \left[\frac{1}{N}\sum_{k=1}^{N} \frac{\exp(\frac{1}{\alpha}Q_{\theta_Q}(s_l, a_{lk}))}{q_{\theta_q}(a_{lk}|s_l)}\right] \quad (10)$$

This introduces bias due to the finite-sample approximation of the expectation inside the (concave) log. In practice we found this estimator sufficiently accurate, provided the proposal distribution was close to π. We also use importance sampling to sample from π while acting.

The action-value loss is the L2 norm with the Soft Q target:

$$\mathcal{L}(\theta_Q) = \mathbb{E}_R\Big[\frac{1}{2}(Q_{\theta_Q}(s_t, a_t) - (r(s_t, a_t, s_{t+1}) + \gamma V_{\theta'_V}(s_{t+1})))^2\Big] \quad (11)$$

To improve stability we employ target networks for the value $V_{\theta_{V'}}$ and proposal $q_{\theta'_q}$ networks (Mnih et al., 2015; Lillicrap et al., 2015) We also parameterize Q as an advantage $Q_{\theta_Q}(s, a) = V_{\theta_V}(s) + A_{\theta_A}(s, a)$ (Baird, 1994; Wang et al., 2015; Harmon et al., 1995) which is more stable when the advantage is small compared with the value. The full algorithm is given in Algorithm 1 and more details are provided in appendix C.

Algorithm 1 AISBP training algorithm

Initialize proposal network parameters θ_q
Initial value network parameters θ_V
Initialize action-value network parameters θ_Q
Initialize replay R
while training **do** ▷ in parallel on each actor
  Obtain parameters θ from learner
  Sample task $i \sim \mathcal{T}$
  Importance sample $\pi_i(a|s) \propto \exp \frac{1}{\alpha} Q^i_{\theta_Q}(s, a)$
    Using proposal $q^i_{\theta_q}$
  Add experience to replay R
end while
while training **do** ▷ in parallel on the learner
  Sample SARS tuple from R
  Improve $\mathcal{L}(\theta_q)$, $\mathcal{L}(\theta_V)$, $\mathcal{L}(\theta_Q)$
  Improve additional losses for transfer
    $\mathcal{L}(\theta_{\Upsilon})$, $\mathcal{L}(\theta_{\psi})$, $\mathcal{L}(\theta_C)$, $\mathcal{L}(\theta_{V_b})$ $\mathcal{L}(\theta_{Q_b})$,
  if target update period **then**
    Update target network parameters
    $\theta_{V'} \leftarrow \theta_V, \theta_{q'} \leftarrow \theta_q, \theta_{\Upsilon'} \leftarrow \theta_{\Upsilon}, \theta_{V'_b} \leftarrow \theta_{V_b}$
  end if
end while

4.1. Importance Sampled Max-Ent GPI

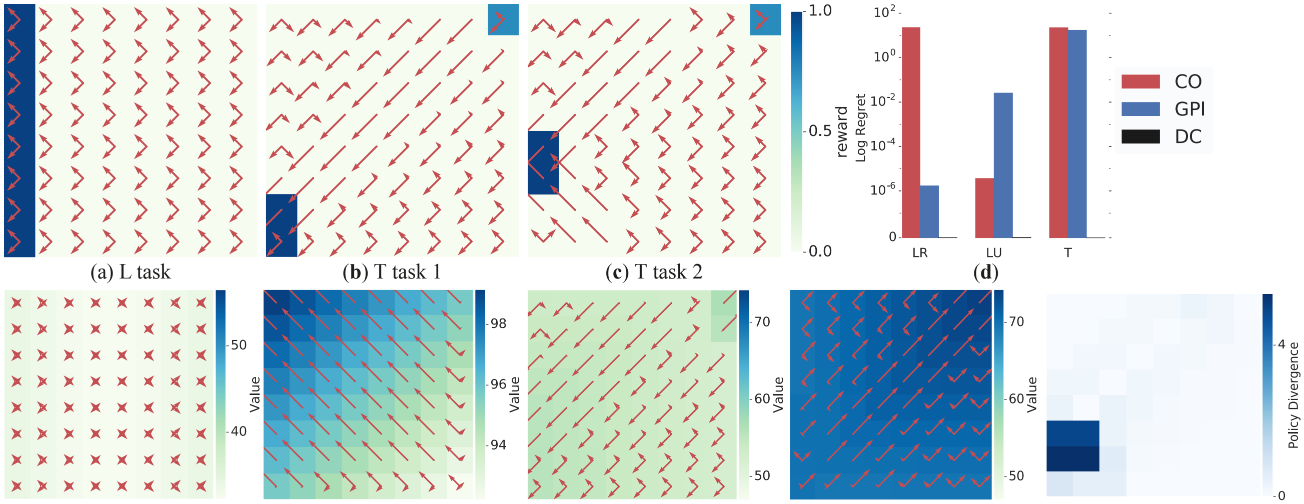
The same importance sampling approach can also be used to estimate max-ent SFs. Max-ent GPI requires us to learn the expected (maximum entropy) features ψ_i for each policy π_i, in order to estimate its (entropic) value under a new convex combination task $\mathbf{w}$. This requires that the experience tuples in the replay contain the full feature vector ϕ, rather than just the reward for the policy which generated the experience r_i. Given this information ψ_{θ_ψ} and $\Upsilon_{\theta_\Upsilon}$ can be learned with analogous updates to V and Q.

As with V_{θ_V}, we use a target network for $\Upsilon_{\theta'_\Upsilon}$ and advantage parametrization. We found it more stable to using a larger target update period than for V. Full details are of the losses and samplers are in appendix C.

4.2. Divergence Correction

Transfer using compositional optimism (equation 3, Haarnoja et al. (2018a)) only requires the max-ent action values of each task, so no additional training is required beyond the base policies. In section 3.2 we have shown that if we can learn the fixed point of $C^{\infty}_b(s, a)$ we can correct this compositional optimism bias and recover the optimal action-value $Q^*_b(s, a)$ for the transfer task r_b.

We exploit the recursive relationship in $C^{\infty}_b(s, a)$ to fit a neural net $C_{\theta_C}(s, a, b)$ with a TD(0) estimator. This requires learning a conditional estimator for any value of b, so as to support arbitrary task combinations. Fortunately, the same experience can be used to learn an estimator for all values of b, by sampling b during each TD update. We learn an estimator $C_{\theta_C}(s, a, b)$ for each pair of policies π_i, π_j with



(a) L task (b) T task 1 (c) T task 2 (d)

(e) LR CO (f) LU GPI (g) T GPI (h) T DC (i)

Figure 1. **Policy composition in the tabular case**. All tasks are in an infinite-horizon tabular 8x8 world. The action space is the 4 diagonal movements (actions at the boundary transition back to the same state) (**a**-**c**) shows the reward functions for the L(eft) task and the two T(ricky) tasks (color indicates reward, dark blue $r = +1$, light blue $r = 0.75$). The arrows indicate the action likelihoods for the max-ent optimal policy for each task. (**d**) The log regret on the compositional task $r_b = 1/2r_i + 1/2r_j$ using the different methods for transfer. This is shown for three, qualitatively distinct compositional tasks: left-right (LR), left-up (LU) and the "tricky" tasks (T). GPI performs well on LR, where the subtasks are incompatible, meaning the optimal policy on the transfer task is similar to one of the existing policies. In LR CO fails to commit to a particular direction (**e** shows the CO policy and value) and performs poorly. Conversely, on the LU task when the base policies are compatible, CO transfers well while the GPI policy (**f**) does not consistently take advantage of the compatibility of the two tasks to simultaneously achieve both base rewards (unlike the CO policy suppl. figure 5c). Neither GPI nor CO policies (**g** shows the GPI policy, but CO is similar) perform well when the optimal transfer policy is unlike either existing task policy. The two tricky task policies are compatible in many states but have a high-divergence in the bottom-left corner since the rewards are non-overlapping there (**i** shows the divergence in each state of the two tricky base policies), thus the optimal policy on the compositional transfer task is to move to the top right corner where there are overlapping rewards. By learning, and correcting for, this future divergence between policies, DC results in optimal policies for all task combinations including tricky (**h**). Additional details in suppl. figure 5.

the loss

$$\begin{aligned}\mathcal{L}(\theta_C) =& \mathbb{E}_{s\sim R, b\sim U(0,1)}[\frac{1}{2}(C_{\theta_C}(s,a,b)+\\ &\alpha\gamma\mathbb{E}_{p(s'|s,a)}[\log\int_{\mathcal{A}}\exp(b\log\pi_i(a'|s')+\\ &(1-b)\pi_j(a'|s') - \frac{1}{\alpha}C_{\theta_{C'}}(s',a',b))da'])^2]\end{aligned} \tag{12}$$

As with other integrals of the action space, we approximate this loss using importance sampling to estimate the integral. As before, we use target networks and an advantage parametrization for $C_{\theta_C}(s,a,b)$ Note that, unlike GPI and CondQ (next section), learning C_b^{∞} does not require observing ϕ while training.

We also considered a heuristic approach where we learned C only for $b = \frac{1}{2}$ (this is typically approximately the largest divergence). This avoids the complexity of a conditional estimator and then we we estimate C_b^{∞} during transfer as $\hat{C}_b^{\infty}(s,a) \sim 4b(1-b)C_{1/2}^{\infty}(s,a)$. This heuristic, we denote DC-Cheap, can be motivated by considering Gaussian policies (see appendix D) The max-ent GPI bound can be used to correct for over-estimates of the heuristic C_b^{∞}, $Q^{DC-Cheap+GPI}(s,a) = \max(Q^{CO}(s,a) - \hat{C}_b^{\infty}(s,a), Q^{GPI}(s,a))$.

4.3. Cond Q

As a baseline, we learn a conditional Q function using a similar approach to DC of sampling b each update $Q(s,a,b)$ (Schaul et al., 2015). This, like GPI but unlike DC, requires observing ϕ during training so the reward on task b can be estimated. Full details provided in appendix C.

4.4. Sampling Compositional Policies

During transfer we need to sample from the Boltzmann policy defined by our estimate of the transfer action-value Q_b (the estimate is computed using the methods we enumerated above). We wish to avoid needing to, offline, learn a new proposal or sampling distribution first (which is the approach employed by Haarnoja et al. (2018a)).

As outlined earlier (and detailed in appendix C.3), we chose the proposal distributions so that the product of propos-

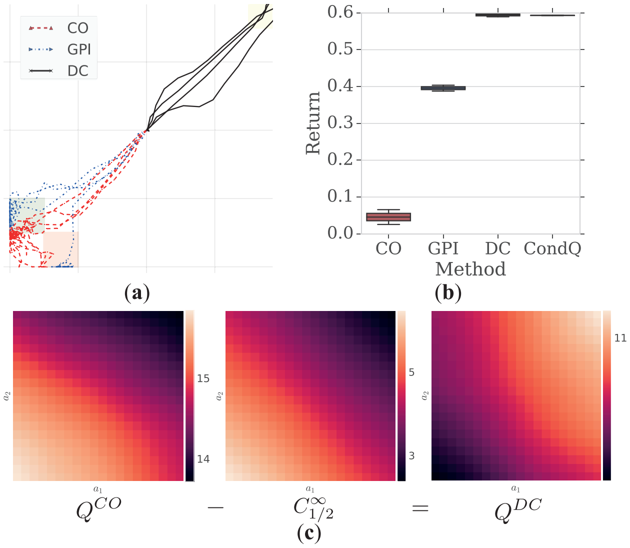


Figure 2. **Tricky point mass**. The continuous "tricky" task with a 2-D velocity controlled pointed mass. (**a**) Environment and example trajectories. The rewards are $(r_1 = 1, r_2 = 0)$, $(0, 1)$ and $(0.75, 0.75)$ for the green, red and yellow squares respectively. Lines show sampled trajectories (starting in the center) for the compositional task $r_{1/2}$ with different transfer methods. Only DC and CondQ (not shown) navigate to the yellow reward area during transfer, which is the optimal reward. (**b**) Box plot of returns for each transfer method (5 seeds). DC and CondQ methods perform significantly better than GPI, and the CO policy performs poorly. (**c**) Q^{CO} at the center position for the transfer task. As both base policies prefer moving left and down, most of the energy (brighter color) is on these actions. However, the future divergence $C^{\infty}_{1/2}$ under these actions is high, which results in the Q^{DC} differing qualitatively from CO and favoring the upward trajectory. Additional details in supplementary figure 6.

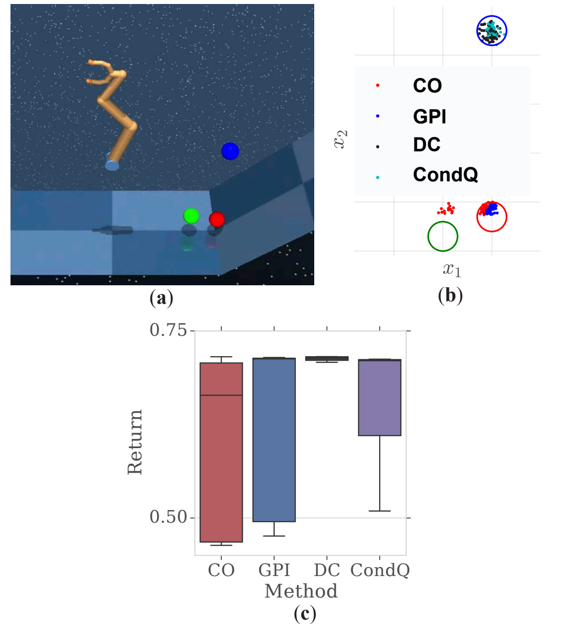


Figure 3. **"Tricky" task with planar manipulator**. (**a**) The "tricky" tasks with a 5D torque-controlled planar manipulator. The training tasks consists of (mutually exclusive) rewards of $(1, 0), (0, 1)$ when the finger is at the green and red targets respectively and reward $(0.75, 0.75)$ at the blue target. (**b**) Finger position at the end of the trajectories starting from randomly sampled start states) for the transfer task with circles indicating the positions of the targets. DC and CondQ trajectories reach towards the blue target (the optimal solution) while CO and GPI trajectories primarily reach towards one of the suboptimal partial solutions. (**c**) Box plot of returns on the transfer tasks, DC outperforms other methods. Additional details in supplementary figure 7.

als is tractable, meaning we can sample from $q_b^{ij}(a|s) \propto (q^i_{\theta_q}(a|s))^b (q^j_\theta(a|s))^{(1-b)}$. This is a good proposal distribution when the CO bias is low, since Q_b^{CO} defines a Boltzmann policy which is the product of the base policies[1]. However, when $C_b^\infty(s, a)$ is large, meaning the CO bias is large, q^{ij} may not be a good proposal, as we show in the experiments. In this case none of the existing proposal distributions may be a good fit. Therefore we sample from a mixture distribution of all policies, all policy products and the uniform distribution, with the hope that at least one component of this mixture will be sampling from high-value parts of the action space.

$$p_b(a|s) \equiv \tfrac{1}{4}(q^i_{\theta_q}(a|s) + q^j_{\theta_q}(a|s) + q_b^{ij}(a|s) + \tfrac{1}{\mathcal{V}^A}) \quad (13)$$

where $\mathcal{V}^A$ is the volume of the action space. Empirically, we find this is sufficient to result in good performance during transfer. The transfer algorithm is given in supplementary algorithm 2.

[1] $\pi_b^{CO}(a|s) \propto \exp \frac{1}{\alpha} Q^{CO}(s, a) = \exp(\frac{1}{\alpha}(Q^1(s, a) + Q^2(s, a)) = \pi_1(a|s)\pi_2(a|s)$.

5. Experiments

5.1. Discrete, tabular environment

We first provide some illustrative tabular cases of compositional transfer. These highlight situations in which GPI and CO transfer can perform poorly (Figure 1). As expected from theory, we find that GPI performs well when the optimal transfer policy is close to one of the existing policies; CO performs well when both subtask policies are compatible, meaning there is some part of the action space that has a high likelihood under both policies. The task we refer to as "tricky" is illustrative of challenging tasks we will focus on in the next section, namely scenarios in which the optimal policy for the transfer task does not resemble either existing policy: In the grid world non-overlapping rewards for each task are provided in one corner of the grid world, while lower value overlapping rewards are provided in the

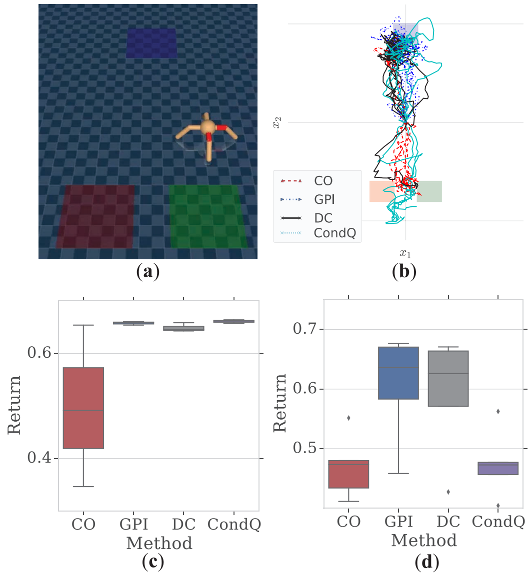


Figure 4. **"Tricky" task with mobile bodies.** "Tricky" task with two bodies: a 3 DOF jumping ball (supplementary figure 8a) and (**a**) 8 DOF ant (both torque controlled). The task has rewards $(1, 0), (0, 1)$ in the green and red boxes respectively and $(0.75, 0.75)$ in the blue square. (**b**) Sampled trajectories of the ant on the transfer task ($b = \frac{1}{2}$) starting from a neutral position. GPI and DC almost always go to the blue square (optimal), CondQ and CO do not. Box plot of returns for the jumping ball (**c**) and ant (**d**) when started in the center position. CO does not recover a good transfer policy for the compositional task while the other approaches largely succeed, although CondQ does not learn a good policy on the ant. Additional details in supplementary figure 8.

other corner (cf. Fig. 1). As a consequence both GPI and CO perform poorly while DC performs well in all cases.

5.2. Continuous action spaces

In the tabular environments we demonstrated some challenging "tricky" transfer tasks that previous methods perform poorly on. Here, we test our approach on continuous control tasks with the same challenging properties. We train max-ent policies to solve individual tasks and then compare transfer performance using the different approaches. All approaches use the same experience, proposal distributions and base policies.

Figure 2 examines the transfer policies in detail in a simple point-mass version of the "tricky" tasks and shows how the estimated C_b^∞ corrects Q^{CO} and results in a qualitatively better transfer policy.

We now examine conceptually similar tasks in more difficult domains: a 5 DOF planar manipulator reaching task (figure 3), 3 DOF jumping ball and 8 DOF ant (figure 4). We see that DC recovers a qualitatively better policy in all cases. The performance of GPI depends noticeably on the choice of α. DC-Cheap, which is a simpler heuristic, performs almost as well as DC in the tasks we consider except for the point mass task. When bounded by GPI (DC-Cheap+GPI) it performs well for the point mass task as well, suggesting these heuristics may be sufficient in some cases[2].

We focused on "tricky" tasks as they are a particularly challenging form of transfer. There are two other cases we considered in the tabular analysis. One: when the base tasks are compatible, CO performs well. We expect that DC would also perform well in this situation since, in this case, the correction term C_b^∞ that DC must learn is inconsequential (CO is equivalent to assuming $C_b^\infty = 0$). Two: At the other extreme, supplementary figure 9 demonstrates on a task with incompatible base tasks (i.e. C_b^∞ is large and potentially challenging to learn), DC continues to perform as well as GPI, slightly better than CondQ and much better than CO. In principle, without function approximator error, DC always recovers the optimal policy during transfer. These experiments provide empirical evidence that our approximate algorithm performs well in a range of situations.

6. Discussion

We have presented two approaches to transfer learning via convex combinations of rewards in the maximum entropy framework: max-ent GPI and DC. We have shown that, under standard assumptions, the max-ent GPI policy performs at least as well as its component policies, and that DC recovers the optimal transfer policy. Todorov (2009) and (Pan et al., 2015; Saxe et al., 2017; van Niekerk et al., 2018) previously considered optimal composition of max-ent policies. However, these approaches require stronger assumptions about the class of MDPs. By contrast, DC does not restrict the class of MDPs and *learns* how compatible policies are, allowing approximate recovery of optimal transfer policies both when the component rewards are jointly achievable (AND), and when only one sub-goal can be achieved (OR).

We have compared our methods with conditional action-value functions (CondQ) (Schaul et al., 2015, e.g.) and optimistic policy combination (Haarnoja et al., 2018a). Further, we have presented AISBP, a practical algorithm for training DC and max-ent GPI models in continuous action spaces using adaptive importance sampling. We have compared these approaches, along with heuristic approximations of DC, and demonstrated that DC recovers an approximately optimal policy during transfer across a variety of high-dimensional control tasks. Empirically we have found CondQ may be harder to learn than DC, and it requires additional observation of ϕ during training.

[2] Videos of the tasks and supplementary information at `https://tinyurl.com/yaplfwaq`.

Acknowledgements

We thank Alexander Pritzel, Alistair Muldal, Dhruva TB, Siqi Liu and the rest of the DeepMind team for helpful discussions and assistance with this work.

References

Karen E Adolph, Whitney G Cole, Meghana Komati, Jessie S Garciaguirre, Daryaneh Badaly, Jesse M Lingeman, Gladys LY Chan, and Rachel B Sotsky. How do you learn to walk? thousands of steps and dozens of falls per day. *Psychological science*, 23(11):1387–1394, 2012.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Advances in Neural Information Processing Systems*, pp. 5048–5058, 2017.

Leemon C Baird. Reinforcement learning in continuous time: Advantage updating. In *Neural Networks, 1994. IEEE World Congress on Computational Intelligence., 1994 IEEE International Conference on*, volume 4, pp. 2448–2453. IEEE, 1994.

André Barreto, Will Dabney, Rémi Munos, Jonathan J Hunt, Tom Schaul, Hado P van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. In *Advances in neural information processing systems*, pp. 4055–4065, 2017.

Andre Barreto, Diana Borsa, John Quan, Tom Schaul, David Silver, Matteo Hessel, Daniel Mankowitz, Augustin Zidek, and Remi Munos. Transfer in deep reinforcement learning using successor features and generalised policy improvement. In *Proceedings of the International Conference on Machine Learning*, pp. 501–510, 2018.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

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

Roy Fox, Ari Pakman, and Naftali Tishby. Taming the noise in reinforcement learning via soft updates. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, UAI'16, pp. 202–211, Arlington, Virginia, United States, 2016. AUAI Press. ISBN 978-0-9966431-1-5. URL http://dl.acm.org/citation.cfm?id=3020948.3020970.

Manuel Gil, Fady Alajaji, and Tamas Linder. Rényi divergence measures for commonly used univariate continuous distributions. *Information Sciences*, 249:124–131, 2013.

Shixiang Gu, Zoubin Ghahramani, and Richard E Turner. Neural adaptive sequential monte carlo. In *Advances in Neural Information Processing Systems*, pp. 2629–2637, 2015.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *International Conference on Machine Learning*, 2017.

Tuomas Haarnoja, Vitchyr Pong, Aurick Zhou, Murtaza Dalal, Pieter Abbeel, and Sergey Levine. Composable deep reinforcement learning for robotic manipulation. In *International Conference on Robotics and Automation*, 2018a.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, 2018b.

Adrian M Haith and John W Krakauer. Model-based and model-free mechanisms of human motor learning. In *Progress in motor control*, pp. 1–21. Springer, 2013.

Mance E Harmon, Leemon C Baird III, and A Harry Klopf. Advantage updating applied to a differential game. In *Advances in neural information processing systems*, pp. 353–360, 1995.

Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

Hilbert J Kappen. Path integrals and symmetry breaking for optimal control theory. *Journal of statistical mechanics: theory and experiment*, 2005(11):P11011, 2005.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012. ISBN 0262018020, 9780262018029.

Yunpeng Pan, Evangelos Theodorou, and Michail Kontitsis. Sample efficient path integral control under uncertainty. In *Advances in Neural Information Processing Systems*, pp. 2314–2322, 2015.

Doina Precup, Richard S Sutton, and Satinder Singh. Theoretical results on reinforcement learning with temporally abstract options. In *European conference on machine learning*, pp. 382–393. Springer, 1998.

Andrew M. Saxe, Adam C. Earle, and Benjamin Rosman. Hierarchy through composition with multitask LMDPs. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3017–3026, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL http://proceedings.mlr.press/v70/saxe17a.html.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International Conference on Machine Learning*, pp. 1312–1320, 2015.

Oliver C Schrempf, Olga Feiermann, and Uwe D Hanebeck. Optimal mixture approximation of the product of mixtures. In *International Conference on Information Fusion*, volume 1, pp. 8–pp. IEEE, 2005.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587): 484, 2016.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Emanuel Todorov. Compositionality of optimal control laws. In *Advances in Neural Information Processing Systems*, pp. 1856–1864, 2009.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS)*, pp. 5026–5033. IEEE, 2012.

Benjamin van Niekerk, Steven James, Adam Earle, and Benjamin Rosman. Will it blend? composing value functions in reinforcement learning. *arXiv preprint arXiv:1807.04439*, 2018.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Freitas. Dueling network architectures for deep reinforcement learning. *arXiv preprint arXiv:1511.06581*, 2015.

Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, volume 8, pp. 1433–1438. Chicago, IL, USA, 2008.