MO2: MODEL-BASED OFFLINE OPTIONS

Sasha Salter[*] **Markus Wulfmeier, Dhruva Tirumala,**
Nicolas Heess, Martin Riedmiller, Raia Hadsell, Dushyant Rao
DeepMind, London, UK

ABSTRACT

The ability to discover useful behaviours from past experience and transfer them to new tasks is considered a core component of natural embodied intelligence. Inspired by neuroscience, discovering behaviours that switch at bottleneck states have been long sought after for inducing plans of minimum description length across tasks. Prior approaches have either only supported online, on-policy, bottleneck state discovery, limiting sample-efficiency, or discrete state-action domains, restricting applicability. To address this, we introduce Model-Based Offline Options (MO2), an offline hindsight framework supporting sample-efficient bottleneck option discovery over continuous state-action spaces. Once bottleneck options are learnt offline over source domains, they are transferred online to improve exploration and value estimation on the transfer domain. Our experiments show that on complex long-horizon continuous control tasks with sparse, delayed rewards, MO2's properties are essential and lead to performance exceeding recent option learning methods. Additional ablations further demonstrate the impact on option predictability and credit assignment.

1 INTRODUCTION

While Reinforcement Learning (RL) algorithms have recently achieved impressive feats across a range of domains (Silver et al., 2017; Mnih et al., 2015; Lillicrap et al., 2015), they remain sample-inefficient (Abdolmaleki et al., 2018; Haarnoja et al., 2018b) which makes it challenging applying them to robotics applications. Natural embodied intelligence across their lifetime discover and reuse skills at varying levels of behavioural and temporal abstraction to efficiently tackle new situations. For example, in manipulation domains, beneficial abstractions could include low-level instantaneous motor primitives as well as higher-level object manipulation strategies. Endowing RL agents with a similar ability could be vital for attaining comparable sample efficiency (Parisi et al., 2019).

The options framework (Sutton et al., 1999) presents an appealing approach for discovering and reusing such abstractions, with options representing temporally abstract skills. We are interested in methods supporting sample-efficient embodied transfer learning (Khetarpal et al., 2020), where skills learnt across prior tasks are used to improve performance on downstream ones. Of particular interest is the recent *Hindsight Off-Policy Options (HO2)* (Wulfmeier et al., 2020) supporting off-policy learning over continuous state-action spaces (crucial for sample reuse in embodied applications), training all options across every experience in hindsight (further boosting sample-efficiency). The latter is achieved by computationally-efficiently marginalising across all possible option-segmented trajectories during policy improvement, similar to Shiarlis et al. (2018), leading to state-of-the-art results on transfer learning benchmarks. Whilst Wulfmeier et al. (2020) considered discovering options online, we are interested in offline discovery, further encouraging sample-efficiency (discovering abstractions without any additional environment interactions).

Even though the options framework supports skill discovery, it does not constrain their behaviour. As such, vanilla approaches often lead to degenerate solutions (Harb et al., 2018), with options switching every timestep, maximising policy flexibility and return, at the expense of skill reuse across tasks. Therefore, constraining skill properties may be necessary. Many traditional methods (McGovern & Barto, 2001; Şimşek & Barto, 2004; Solway et al., 2014; Harutyunyan et al., 2019; Ramesh et al., 2019) sought out *bottleneck options* that terminate at bottleneck states. Bottleneck states are defined as the most frequently visited states when considering the shortest distance between any two states in an MDP (Solway et al., 2014), and are considered beneficial for planning by inducing plans of minimum description length across a set of tasks (Harutyunyan et al., 2019; Solway et al., 2014). Intuitively, bottleneck states connect different parts of an environment, and are hence visited more often by successful trajectories (Harutyunyan et al., 2019; McGovern & Barto, 2001; Stolle & Precup, 2002). Unfortunately, these methods do not support off-policy hindsight learning, necessary for sample-efficiency, nor continuous state-action spaces, crucial in robotics.

To address this, we present *Model-Based Offline Options* (MO2), an offline hindsight *bottleneck options* framework supporting continuous (and discrete) state-action spaces, that discovers skills suited for planning (in the form of value

[*] Work done while at DeepMind (sasha.salter@hotmail.com).

estimation) and acting across modular tasks. We refer to modular tasks as a family of MDPs whose optimal policies are obtained by recomposing (a subset of) shared temporal behaviours. For example, navigation tasks with self-driving cars share a set of common temporal abstractions: traversal between junctions. While this modular constraint appears restrictive, options can collapse to the original action-space, transitioning per timestep if necessary. MO2 discovers *bottleneck options* by extending HO2 to the offline context, training options to maximise: 1) the log-likelihood of the offline behaviours; 2) a *predictability objective* for option termination states across the episode, computed as the cumulative 1-step option-transition log-likelihood. Maximising the 1-step option-transition log-likelihood encourages low-entropic, predictable terminations. The cumulative equivalent (across all transitions), encourages minimal switching only at bottleneck states where behaviours diverge. Once options are learnt offline, we freeze them and perform online transfer learning over the option-induced semi-MDP, learning and acting over the temporally abstract skill space. We compare MO2 against state-of-the-art baselines on complex continuous control domains (Fu et al., 2020) and perform an extensive ablation study. We demonstrate that MO2's options are bottleneck aligned (see Figure 2) and improve acting (and exploring), value estimation, and learning of a jumpy option-transition model (defined in Section 3). On the challenging, sparse, long-horizon, AntMaze domain, MO2 drastically outperforms all baselines.

2 Preliminaries

Our goal is to leverage temporally abstract skills extracted from large, unstructured and multi-task datasets (*source domains*) to accelerate the learning of new tasks (*transfer domain*). For skills to be beneficial for learning, they should be suited for planning and acting over them (Solway et al., 2014). We propose an offline *bottleneck options* approach discovering skills with these properties, that supports continuous state-action spaces, for efficient skill transfer from large datasets. We decompose the problem of skill transfer into two sub-problems: 1) the extraction of skills suited for planning and acting from offline data, and 2) learning of transfer tasks over the temporally abstracted skill space.

2.1 Problem Formulation

Our work considers reinforcement learning in Markov Decision Processes (MDPs), defined by $M = (S, A, R, p, p^0, \gamma)$. S, A, R denote state, action and reward spaces. $p(s'|s, a) : S \times S \times A \rightarrow \mathbb{R}_{\geq 0}$ represents the dynamics model. $p^0(s) : S \rightarrow \mathbb{R}_{\geq 0}$ represents the initial state distribution. We denote the history of states $s \in S$ and actions $a \in A$ up to timestep t as $h_t = (s_0, a_0, s_1, a_1, \ldots, s_t)$. In this work, we consider option-augmented policies, taking the *call-and-return* formulation (Sutton et al., 1999), defining options as triple $(I(s_t, o_t), \pi^L(a_t|s_t, o_t), \beta(s_t, o_t))$. I, β represent an option's initial and termination conditions and π^L denotes its behaviour. Following Bacon et al. (2017); Zhang & Whiteson (2019), we define $I = 1 \forall s_t \in S$ and sample a new option o_t from $\pi^C(o_t|s_t)$ (the option-level controller) only if the previous one has terminated. $b(a|h)$ denotes the behaviour policy that collects experiences saved into the offline dataset D in the form (s_t, a_t). We leverage the options framework, trained on a maximum-likelihood behavioural cloning objective, to discover a set of task-agnostic, shared skills across tasks able to reconstruct offline behaviours. During transfer, we freeze these options, learning and acting over them to accelerate online RL on the new tasks. As such, we make the assumption that the offline datasets (*source domains*) are multi-task and diverse enough that they exhibit all the temporal abstractions (skills) necessary for transfer. We refer to this as the *modularity* assumption, that skills from *source domains* suffice and can be recomposed to obtain optimal *transfer domain* performance. In Section 3.3, we discuss this assumption and how to relax it.

2.2 HO2: Hindsight Off-Policy Options

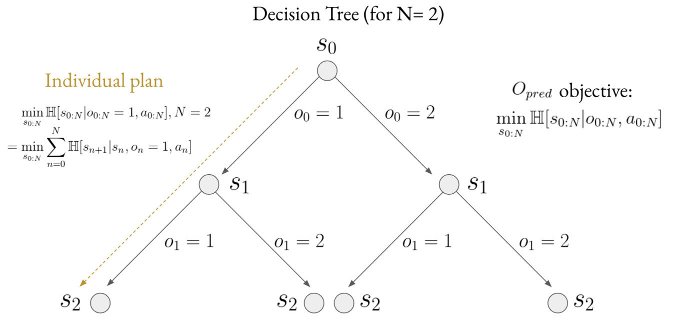
Decision Tree (for N= 2)
s_0
Individual plan
$\min_{s_{0:N}} \mathbb{H}[s_{0:N}|o_{0:N} = 1, a_{0:N}], N = 2$
$= \min_{s_{0:N}} \sum_{n=0}^{N} \mathbb{H}[s_{n+1}|s_n, o_n = 1, a_n]$
O_{pred} objective:
$\min_{s_{0:N}} \mathbb{H}[s_{0:N}|o_{0:N}, a_{0:N}]$
$o_0 = 1$
$o_0 = 2$
s_1
s_1
$o_1 = 1$
$o_1 = 2$
$o_1 = 1$
$o_1 = 2$
s_2
s_2
s_2
s_2

Figure 1: *Option Decision Tree (depth = 2)*: MO2's O_{pred} objective encourages low-entropic, compressed, option-level plans (over s_n) that reconstruct offline behaviours with high confidence. See Section 3.1 for more details.

We build on Hindsight Off-Policy Options (HO2) (Wulfmeier et al., 2020) as a highly sample-efficient, performant, off-policy actor-critic options algorithm that builds on Smith et al. (2018). HO2 achieves higher sample efficiency than alternate options frameworks for two reasons: 1) HO2 trains all options in hindsight across all experiences within a trajectory, not just the current timestep. This is achieved by back-propagating gradients through HO2's dynamic programming inference procedure (discussed below), training all options end-to-end. 2) It uses Maximum-A-Posteriori Policy Optimisation (MPO) (Abdolmaleki et al., 2018), a highly performant, off-policy framework with monotonic improvement guarantees.

To train all options across all experiences, HO2 unrolls the graphical model for option-based policies and calculates the joint $\pi(a_t, o_t|h_t)$, marginalising across all combinations of option sequences $o_{0:t-1}$. To avoid a combinatoral explosion, HO2 uses the following recursive relation, similar to the one introduced in Shiarlis et al. (2018):

$$p(o_t|s_t, o_{t-1}) = \begin{cases} 1 - \beta(s_t, o_{t-1})(1 - \pi^C(o_t|s_t)) & \text{if } o_t = o_{t-1} \\ \beta(s_t, o_{t-1})\pi^C(o_t|s_t) & \text{otherwise} \end{cases} \tag{1}$$

$$\tilde{\pi}^H(o_t|h_t) = \sum_{o_{t-1}=1}^{M} \left[p(o_t|s_t, o_{t-1})\pi^H(o_{t-1}|h_{t-1})\pi^L(a_{t-1}|s_{t-1}, o_{t-1})\right] \tag{2}$$

The distribution is normalized per timestep with $\pi^H(o_t|h_t) = \tilde{\pi}^H(o_t|h_t)/\sum_{o'_t=1}^{M} \tilde{\pi}^H(o'_t|h_t)$. π^H denotes option probabilities given histories and is not the same as π^C, the option controller for any given state. Intuitively, option probabilities given history $\pi^H(o_t|h_t)$ are not only dependent on the current state s_t and option controller $\pi^C(o_t|s_t)$, but additionally the previous option (according to $\pi^H(o_{t-1}|h_{t-1})$) and whether it terminated (according to $\beta(s_t, o_{t-1})$). The exact relation is shown in Equations (1) and (2). Using π^H for optimisation, allows for all options up to timestep t ($o_{0:t}$), not just the current option (o_t), to be optimised jointly on a control objective at time t (e.g. for behavioural cloning $\log(\pi(a_t|h_t))$). As such, all options are trained across all experiences in the episode in hindsight, improving performance (Wulfmeier et al., 2020). Building on option probabilities we obtain the joint:

$$\pi(a_t, o_t|h_t) = \pi^L(a_t|s_t, o_t)\pi^H(o_t|h_t) \tag{3}$$

While Wulfmeier et al. (2020) uses the joint to effectively train π^L, π^C, β on an online RL objective, we instead use an offline maximum-likelihood behavioural cloning objective, $O_{bc}(\pi^L, \pi^C, \beta) = \mathbb{E}_{a_t, h_t \sim D, o \sim \pi^H(\cdot|h_t)}[\log(\pi(a_t, o|h_t))]$ (omitting t subscript from o to emphasise it is not sampled from D), to distil skills from the offline dataset.

3 MO2: Model-Based Offline Options

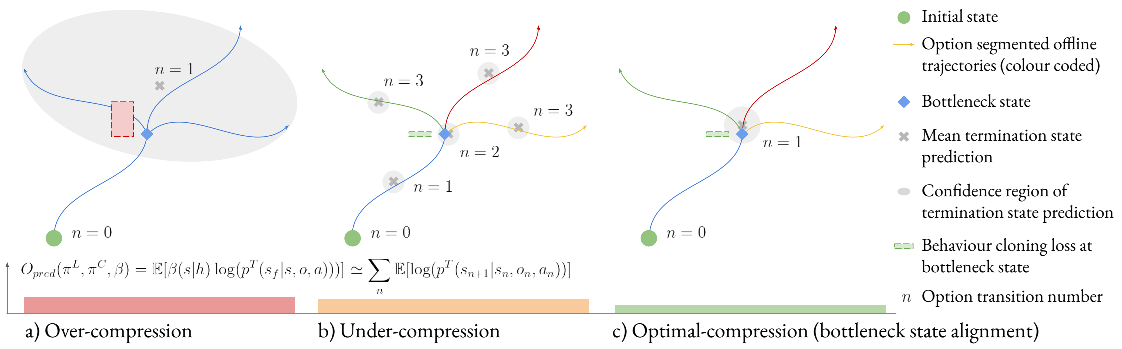
$n = 1$
$n = 0$
$n = 3$
$n = 3$
$n = 3$
$n = 2$
$n = 1$
$n = 0$
$n = 1$
$n = 0$
$O_{pred}(\pi^L, \pi^C, \beta) = \mathbb{E}[\beta(s|h)\log(p^T(s_f|s,o,a)))] \simeq \sum_n \mathbb{E}[\log(p^T(s_{n+1}|s_n,o_n,a_n))]$
Initial state
Option segmented offline trajectories (colour coded)
Bottleneck state
Mean termination state prediction
Confidence region of termination state prediction
Behaviour cloning loss at bottleneck state
n Option transition number
a) Over-compression
b) Under-compression
c) Optimal-compression (bottleneck state alignment)

Figure 2: *Intuition behind MO2.* MO2 trains an option policy and option-transition model, $\tilde{p}^T(s_f|s, o, a)$, predicting terminal state s_f of option o given state s and action a. MO2's *predictability* objective, O_{pred}, represents the predictability of option-transitions, $p^T(s_{n+1}|s_n, o_n, a_n)$, across the episode, with n the option-transition number, s_{n+1}/s_n the n^{th} options termination/initiation states. Individual log-likelihoods (n) encourage predictable, low-entropic transitions, minimising any termination confidence region (individual ellipse area), ruling out *a) over-compression*. Given positive transition entropies, the cumulative log-likelihoods (across ns) lead to transitions only when necessary, to reduce cumulative entropies (intuitively similar to the total area across ellipses), ruling out *b) under-compression*. MO2's *behaviour cloning* objective, O_{bc}, further encourages terminations where multi-modal behaviours exist. Combined, MO2 leads to *c) optimal-compression (bottleneck state alignment)*, transitioning only at *bottlenecks states*.

While HO2 provides a sample-efficient framework for discovering options, no constraints are placed on their behaviour. As such, it is unclear whether HO2's abstractions are suited for *compressed* decision making across tasks, arguably the key reason of temporal abstractions as tools for computation- and sample- efficient learning. Taking inspiration from humans as resource constrained learners, Solway et al. (2014) demonstrate the favouring of abstractions suited for long horizon reasoning (specifically *planning*) and *acting* across tasks. Particularly suited abstractions are those that transition between environment *bottleneck states*, concentrated regions of the state-space that are commonly visited between tasks when traversing distinct and diverse parts of the environment (such as doors or junctions, for distinct navigation tasks, in multi-room or maze domains, respectively) (Solway et al., 2014; Harutyunyan et al., 2019). As such, bottlenecks give rise to a *compressed* decision problem, reasoning only where necessary, over a few key decision making states (predictable due to their concentrated nature) where behaviours diverge across tasks. *Bottleneck options* methods discover such abstractions, suited for long horizon option-level *planning* (due to the high *predictability* and *compressed* nature of option-transitions) and *acting* across tasks. Unfortunately, existing methods (McGovern & Barto, 2001; Stolle & Precup, 2002; Solway et al., 2014; Harutyunyan et al., 2019) do not support offline hindsight learning, necessary for sample-efficiency, nor continuous state-action spaces, crucial in robotics. We propose a method without these shortcomings, *compressing* diverse multi-task offline behaviours into *bottleneck options*.

3.1 Discovering Bottleneck Options Offline

We introduce Model-Based Offline Options (MO2), an offline hindsight *bottleneck options* method supporting continuous state-action spaces. MO2 distils offline behaviours into a set of skills by training on the maximum-likelihood behavioural cloning objective O_{bc} introduced in Section 2.2, using HO2's efficient calculation of the joint $\pi(a, o|h)$ to optimise all options in hindsight. However, in addition to training on O_{bc}, MO2 extends HO2 with an additionally learnt option-level transition model, and a *predictability* objective O_{pred} encouraging option-level transitions across the episode to be predictable, compressed and beneficial for planning, thereby discovering *bottleneck options*:

$$O_{pred} = \sum_{n=0}^{N} \mathbb{E}_{s_n,o_n,a_n,s_{n+1}\sim\pi}[\log(p^T(s_{n+1}|s_n, o_n, a_n))] = -\sum_{n=0}^{N} \mathbb{H}_\pi[s_{n+1}|s_n, o_n, a_n] = -\mathbb{H}_\pi[s_{0:N}|o_{0:N}, a_{0:N}] \tag{4}$$

Here, n is the option-transition number within the episode, N the maximum, s_{n+1} and s_n are the n^{th} option's termination and initiation states respectively, and $p^T(s_f|s, o, a)$ is the option-transition distribution over the terminal state s_f of an option o, given state s and action a. Note that $n \neq t$ as options act at a different timescale. Shown in Equation (4) (for proof see Appendix D), encouraging predictable option-transitions is equivalent to encouraging individual conditional entropies to be low $\mathbb{H}_\pi[s_{n+1}|s_n, o_n, a_n]$, ruling out the *over-compression* of offline behaviours (see Figure 2 for an explanation). Maximising O_{pred} is additionally equivalent to minimising the joint conditional entropy of option transition states $s_{0:N}$ given options $o_{0:N}$ and first option-executed actions $a_{0:N}$ from policy π, $\mathbb{H}_\pi[s_{0:N}|o_{0:N}, a_{0:N}]$ (see Appendix D for proof and Figure 1 for a visual depiction). We subscript entropies by π to emphasise they are dependent on the policy. If individual transition entropies are positive (see Equation (9) for how this is achieved), then O_{pred} encourages minimal option-transitions able to reconstruct offline behaviours using options, as to minimise N and the accumulation of positive entropies. As such, O_{pred} rules out the *under-compression* and leads to the *optimal-compression (bottleneck state alignment)* scenario in Figure 2. For a more detailed explanation as to why this objective discovers *bottleneck options* we refer the reader to Appendix D.1.

Equation (4) requires evaluating the agent π in the environment to obtain s_{n+1}, s_n. As such, this objective does not support offline learning, as desired. Additionally, inline with HO2, we wish to train all options across all experiences in hindsight, as to maximise sample-efficiency. Therefore, we present an alternate form for O_{pred} supporting offline hindsight learning, unrolling the graphical model for option-based policies and training across a distribution of option-transitions given histories (by efficiently marginalising over option transitions $o_{0:t-1}$ using Equations (1) and (2)):

$$O_{pred} = \mathbb{E}_{\substack{s_t,h_t\sim D \\ o\sim\pi^H(\cdot|h_t),a\sim\pi^L(\cdot|s_t,o) \\ s_f\sim p^T(\cdot|s_t,o,a)}}[\beta(s_t|h_t)\log(p^T(s_f|s_t, o, a))] \tag{5}$$

With $\beta(s_t|h_t) = \sum_{o_{t-1}=1}^{M} \beta(s_t, o_{t-1})\pi^H(o_{t-1}|h_t)$, representing the probability that state s_t is terminal, for the previous option o_{t-1}, and initial for the subsequent option o_t, given history h_t. $\pi^H(o_{t-1}|h_t) = \pi^H(o_{t-1}|h_{t-1})\pi^L(a_{t-1}|o_{t-1}, s_{t-1})/\pi(a_{t-1}|h_{t-1})$ represents the distribution over o_{t-1} given history h_t. D represents the offline data created by behaviour policy $b(a|h)$. For O_{pred}, options are sampled from $\pi^H(o_t|h_t)$ as the D does not contain options. In Equation (5), we do not subscript o and a with t to emphasise that these samples are not from D. This form of O_{pred} uses β as a weighting for how probable s_t is a starting state (s_n) of an option. The expectation is now taken over offline trajectories, with β also representing the summation over option-transitions, akin to the original O_{pred}. We use p^T to sample terminal states s_f (s_{n+1}) in Equation (5). In practice, we do not have access to p^T and instead learn it offline (Section 3.2). See Appendix D.2 for details under which conditions Equations (4) and (5) are equivalent from an optimisation perspective. Our overall objective combines O_{pred} with O_{bc}:

$$\max_{\pi^L,\pi^C,\beta} O_{mo2}(\pi^L, \pi^C, \beta) = \mathbb{E}_{\substack{s_t,a_t,h_t\sim D \\ o\sim\pi^H(\cdot|h_t),a\sim\pi^L(\cdot|s_t,o) \\ s_f\sim p^T(\cdot|s_t,o,a)}}[\beta(s_t|h_t)\log(p^T(s_f|s_t, o, a)) + \log(\pi(a_t, o|h_t))] \tag{6}$$

While $p^T(s_f|s_t, o, a)$ encourages terminations predictable across all encountered states s_t, the additional weighting $\beta(s_t|h_t)$ focuses predictability only over the distribution of (rather than sampled) option initiation states given histories. In line with the intuition from Figure 2, O_{pred} and O_{bc} together encourage offline hindsight *bottleneck options* discovery over continuous state-action spaces.

3.2 Learning the Option-Transition Model Offline

MO2 assumes access to the option-level transition distribution $p^T(s_f|s, o, a)$ in Equation (6). Given that we do not have access to such a model, we instead learn to approximate it, $\tilde{p}^T(s_f|s, o, a)$, using a two-step self-supervision process. We first sample options $o \sim \pi^H(o_t|h_t)$ and then sample terminations according to $p^T(s_f|s_{t:T}, a_{t:T}, o)$, the termination distribution of the option given future states and actions. We do not subscript o with t to emphasise that these samples are not from D. Specifically, we sample s_f by sampling termination condition $\beta(s_k|o)$ over all future timesteps, $k \in [t, T]$, with T representing episodic length. Each time the termination sample is true, the corresponding state is labelled terminal, $s_f = s_k$.

$$\max_{\tilde{p}^T} \; O_{tran}(\tilde{p}^T) = \mathbb{E}_{\substack{s_{0:T}, a_{0:T}, h_t \sim D \\ o \sim \pi^H(\cdot|h_t) \\ s_f \sim p^T(\cdot|s_{t:T}, a_{t:T}, o)}} \left[\log(\tilde{p}^T(s_f|s_t, o, a_t)))\right] \tag{7}$$

As such, we learn an option-transition model that predicts the termination state distribution of an option for any state during its execution (not just the initial state). We note that this objective is biased, as the future states $s_{t:T}$ and actions $a_{t:T}$ used to sample s_f are not from the option-policy π over which we predict option-transitions, but the behaviour policy b that created the offline behaviours. However, over time, as $\pi(s_{t:T}, a_{t:T}|h_t)$ aligns with $b(s_{t:T}, a_{t:T}|h_t)$, as encouraged by O_{bc}, this bias tends to zero. We train $\tilde{p}^T(s_f|s_t, o, a)$ using Equation (7) while simultaneously using it to train π^L, π^C, β with Equation (6). During transfer, the options aid acting, exploration, and value estimation.

3.3 Online Transfer Learning

Once options are learnt offline (over *source domains*), there are several ways one can use them to improve online learning on a *transfer domain*. We make the *modularity* assumption, that skills exhibited over *source domains* suffice for optimal *transfer domain* performance when recomposed. As such, during transfer, MO2's options are frozen and we train a new high-level categorical controller, $\pi^C(o_t|s_t)$, to recompose them, using MPO (see Appendix B.2). While the *modular* constraint may appear restrictive, the increasingly diverse the *source domains* are (as expected during lifelong learning (Khetarpal et al., 2020)), the increasingly probable that the optimal *transfer* policy can be obtained by recomposing a subset of the *source* abstractions. If this assumption does not hold, one could either fine-tune the existing options during *transfer*, which would require tackling the catastrophic forgetting of skills (Kirkpatrick et al., 2017), or train additional new options. We leave this as future work. MO2's temporal abstractions afford two benefits: 1) temporally-correlated actions and exploration (over bottleneck states); 2) improved credit assignment and value estimation over the option induced semi-MDP (with a lower temporal granularity than the original MDP and occurring over concentrated regions of the state-space deemed beneficial for planning (Solway et al., 2014)).

4 Experimental Setup

In this paper, we explore how beneficial MO2's behavioural abstractions are for: 1) acting and exploration; 2) value estimation; 3) learning an option-transition model; 4) discovering compressed abstractions terminating at bottleneck states. We compare our results against two offline versions of HO2 trained solely to maximise O_{bc} and not on O_{pred}. We call these baselines HO2 (offline) and HO2-lim (offline), a variant introduced in Wulfmeier et al. (2020) (see Appendix B.1) encouraging minimal option switching by penalising switches (yet does not encourage predictable terminations). We compare against these baselines to inspect how important the predictability of options are for all the these questions. Additionally, we compare with HO2 as we build on it as a state-of-the-art, sample-efficient, options framework supporting continuous state-action spaces. Finally, we compare against HO2 trained from scratch on the transfer task (HO2 scratch) to quantify the importance of skill transfer. See Appendix for full experimental details.

4.1 Semi-MDP vs MDP Ablations

To investigate the relative importance of MO2's options for both exploration and value estimation, we run two variants of each method during transfer: 1) value estimation occurs over the semi-MDP; 2) value estimation occurs over the MDP (akin to the original HO2 that we build on (Wulfmeier et al., 2020)). By contrasting these experiments we can infer the importance of MO2's temporal abstractions, in relation to others, for credit assignment. Comparing against the HO2 baselines, we can evaluate the importance of MO2's options for exploration. For all methods with pre-trained options, we act at the option-level during transfer. See Appendix B for further ablation information.

4.2 Domains

We evaluate on two D4RL suite domains (Fu et al., 2020): Maze2D and AntMaze (see Figures 3 and 6 and Appendix C for details). The D4RL suite was designed to test the ability of learners to leverage large, diverse, unstructured, multi-task datasets (*source domains*) to accelerate learning of downstream tasks (*transfer domains*). Availability to such datasets are expected for lifelong learners (Khetarpal et al., 2020) and becoming increasingly abundant in robotics, with their wide-range deployment on streets (Caesar et al., 2020), offices (Mo et al., 2018), and warehouses (Dasari et al., 2019; Cabi et al., 2019), collecting diverse experiences. We evaluate on two navigation domains, with distinct embodiments (pointmass for Maze2D; quadruped for AntMaze), as they are representative of many challenging real-world robotic settings, such as autonomous driving (Caesar et al., 2020). Navigation domains are particularly challenging due to their long-horizons and sparse rewards, making exploration and credit-assignment across extended timescales difficult. We test MO2's ability to discover temporal abstractions beneficial for *transfer* in these settings.

Specifically, the offline data (*source domains*) corresponds to trajectories created by optimal, shortest route, goal reaching policies for randomly initiated agent and goal locations across episodes (see Figure 6). The maze layout is not randomised. Between tasks, behaviours only diverge at corridor intersections, based on which corridor ordering leads to the shortest path to the goal. As such, these are the *bottleneck states* of this setup, leading to plans of minimum description length across tasks (Solway et al., 2014) when planning over them. The *transfer domain* keeps the same maze layout but has an unknown, fixed, goal location. The agent is always spawned on the other side of the maze and must discover where the goal is, and the shortest path to it (by traversing a new ordering of corridor intersections, the *bottleneck states*). The *transfer domain* has sparse rewards (rewarded $+1$ only upon goal reaching) and long-horizons.We compare Maze2D and AntMaze as both have distinct state (2 vs 111) and action (2 vs 8) dimensionalities, and episodic-lengths (200 vs 900 for optimal policies), allowing us to inspect how well MO2 scales (specifically its predictability objective) across these dimensions and to increasingly hard-exploration domains. Additionally, as mentioned in Fu et al. (2020), AntMaze is introduced to mimic a more challenging real-world robotic navigation task.

5 Results

In this section, we perform a qualitative and quantitative analysis of the options belonging to MO2 and the baselines, and in the following sections answer the questions from Section 4. All tables in this section report mean $\pm$ one standard deviation, obtained across four random seeds. We additionally report the maximum and minimum performance, either theoretically or achieved across the learners and their lifetimes. As such, we contextualise the values in each table. It should be clear which performance we report.

5.1 Temporal Compression Of Offline Behaviours (Bottleneck Options)

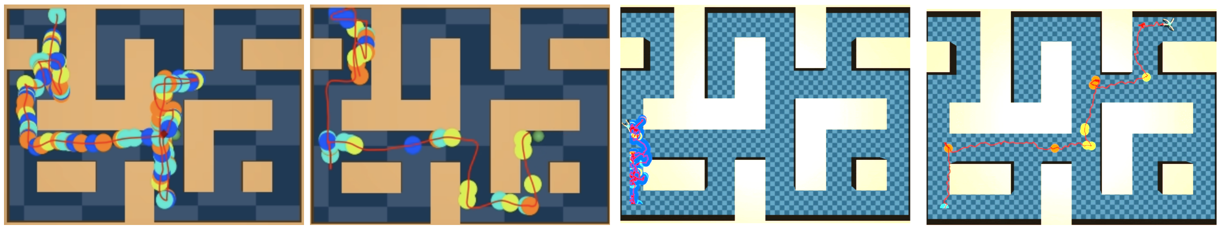

Figure 3: *Policy Rollouts*. We plot policy rollouts across 1 episode for Maze2D (left two plots) and AntMaze (right two plots), for HO2-lim (offline) (inner-left for each domain) and MO2 (inner-right for each domain). These rollouts are obtained early during training on the transfer domain. The centre of mass trajectory is plotted as a red line. The termination locations of options are denoted as coloured circles along the rollout, with the colour denoting the subsequent chosen option. MO2 leads to more temporally compressed options, switching less frequently, and primarily at environment-aligned, bottleneck states, corresponding with corridor intersections. As such, MO2 leads to temporally correlated behaviour, traversing the depths of the maze. In contrast, HO2-lim (offline) exhibits high frequency switching with shallower maze traversal for the longer horizon, higher dimensional, AntMaze domain.

Table 1: Temporal Compression Metrics

(a) Average Switch Rate

Environment	Maze2D	AntMaze
HO2 (offline)	0.48 ± 0.04	0.56 ± 0.04
HO2-lim (offline)	0.15 ± 0.01	0.36 ± 0.01
MO2	$\mathbf{0.03 \pm 0.00}$	$\mathbf{0.01 \pm 0.00}$
Max (min)	1.00 (0.00)	1.00 (0.00)

(b) Expected Behaviour Cloning Performance

Environment	Maze2D	AntMaze
HO2 (offline)	$\mathbf{0.102 \pm 0.004}$	$\mathbf{1.30 \pm 0.05}$
HO2-lim (offline)	0.082 ± 0.005	0.98 ± 0.02
MO2	0.079 ± 0.001	$\mathbf{1.27 \pm 0.04}$
Max (min)	0.102 (-5.000)	1.30 (-50.00)

We analyse the properties of the options for each method. In Figure 3, we visualise representative rollouts of the transfer agent, early during training, using the pre-trained, frozen, options. We plot rollouts for MO2 and HO2-lim (offline). We plot centre-of-mass rollouts for each domain, depicted as a red line, together with option termination locations, depicted as circles colour coded by the subsequent chosen option.

For both domains, we observe that MO2's options primarily align with individual corridor traversal with terminations occurring at the intersection of corridors. MO2's options have therefore discovered the environment bottleneck states (corridor intersections) leading to switching of options at a low-entropic distribution of states that are highly visited and predictable, where behaviours naturally diverge, connecting distinct regions of the environment (individual corridors). This leads to high temporal compression of offline behaviours, with option switches occurring infrequently.

In contrast, HO2-lim (offline) exhibits far lower temporal compression, with option switching occurring more frequently, and at seemingly random locations, not aligning with bottleneck states. This suggests that penalising option switches, as achieved through HO2-lim (offline), is not enough to achieve effective temporal compression and bottleneck discovery. Comparing average switch rates (the inverse of expected option duration over offline trajectories), seen in Table 1a, additionally shows the compression disparity between each approach. Importantly, MO2's increased temporal compression barely influences controllability, as seen by comparing behaviour cloning O_{bc} values in Table 1b (over offline trajectories, the higher the score the better). In Table 1b, max (min) scores refer to the range of values achieved across all methods during training. We report mean $\pm$ one standard deviation, across four random seeds.

5.2 Exploration

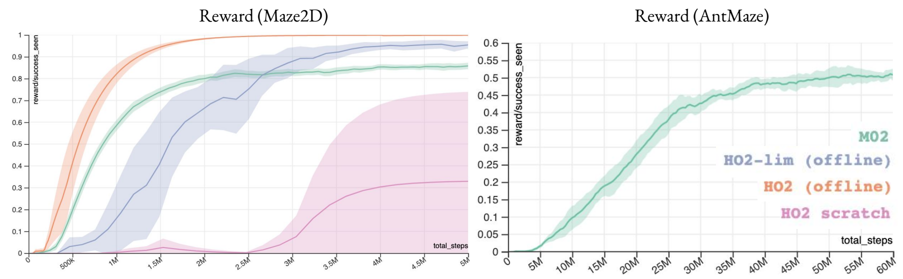

Figure 4: *Return Curves*. We plot return learning curves for both transfer environments and all methods. Experiments were ran with 4 seeds, and we plot mean (solid line) and 1 standard deviation (shaded region). Curves are periodically obtained by evaluating the current policy on the environment. Temporally compressed, bottleneck aligned, abstractions are essential for the sparse, long horizon, AntMaze, with MO2 the only method with non zero reward. For the less sparse, Maze2D, domain, MO2's abstractions are less necessary and hinder asymptotic performance.

In Figure 4, we plot transfer performance for each approach on both domains. For the sparser, longer horizon, higher-dimensional action-space AntMaze domain, we observe that MO2 is the only method able to attain any rewards and solve the task, demonstrating the benefits of MO2's temporally compressed bottleneck options for exploration. In contrast, for Maze2D, temporally compressed behaviours, achieved either by MO2 or HO2-lim (offline), do not yield benefits, demonstrating that for simpler exploration domains, temporal compression is not necessary for effective transfer, and can slightly hinder policy flexibility and asymptotic performance. Comparing initial policy rollouts in Figure 3, we see that MO2's reduced switch rate leads to more directed, temporally correlated, exploration, with wider coverage of the maze for AntMaze. This is not the case for Maze2D, suggesting that for low-dimensional environments with short horizons, compression is not necessary for exploration.

5.3 Value Estimation

To evaluate the importance of temporal abstractions for learning, specifically value estimation, we plot semi-MDP vs MDP policy evaluation learning curves, where value estimation occurs either over the semi-MDP or MDP, both acting at the option-level[1] (see Figure 5). We additionally compare policy performance for both setups, to inspect how important an accurate critic is for policy improvement (see Figure 5). Comparing policy performance and evaluation for each method, it is apparent that value estimation over MO2's more temporally compressed options yields a faster learning, less biased critic (seen when comparing return and value estimate curves). For both domains, improved policy evaluation yields improved policy performance, although the improvement is more significant for the longer horizon AntMaze. Additionally, the more temporally compressed, the larger the improvement for value bias and convergence as well as policy improvement, when value estimation occurs over the semi-MDP. This highlights the importance of temporal compression for policy evaluation and improvement. Interestingly, for Maze2D, an increasingly biased critic, HO2-MDP, does not hinder performance, suggesting that here, reasonable advantage estimation is occurring.

5.4 Option-Transition Predictability

In Section 3.1, we proposed discovering *bottleneck options* by optimising $O_{pred} = \sum_n \mathbb{E}_\pi[\log(p^T(s_{n+1}|s_n, o_n, a_n))]$ (Equation (5)), predictable option-transitions across offline episodes. As outlined in Section 3.2, we use a learnt option-transition model $\tilde{p}^T$ to optimise O_{pred}, as we do not have access to the true p^T. We train such a

[1]The Reward and Value Estimates horizontal axes are aligned despite different metrics due to the asynchronous learner.

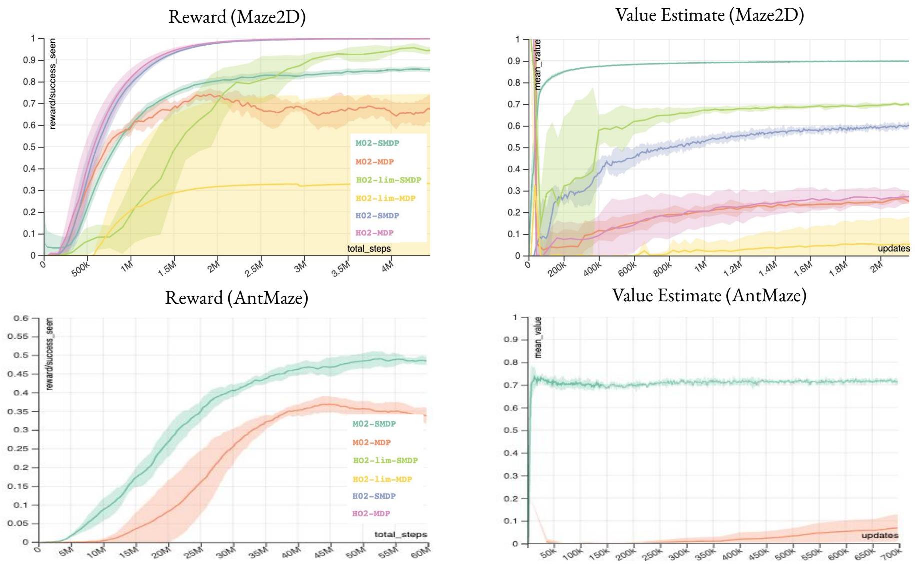

Figure 5: *Semi-MDP vs MDP Learning Curves*. To evaluate the importance of temporal abstraction for policy evaluation and improvement, we compare expected return and value estimates between methods that perform value estimation over the semi-MDP vs MDP, both acting at the option-level. We plot mean (solid line) and standard deviation (shaded region) across 4 seeds. Curves are periodically obtained by evaluating the current policy on the environment. As seen by comparing semi-MDP and MDP experiments, contrasting reward and value estimate curves, abstractions reduce value bias, improve value convergence, and policy performance. The degree of improvement directly correlates to the level of temporal compression (MO2 > HO2-lim > HO2). Nevertheless, for the simpler Maze2D domain, temporal abstraction is unnecessary with HO2-MDP performing best.

model using Equation (7). To validate that Equations (6) and (7) lead to predictable semi-MDP transitions on the *source* domains, despite using a learnt transition model to optimise behaviours, we evaluate the following:

$$CE(\tilde{p}^T) = -\sum_{n=0}^{N} \mathbb{E}_{s_n, a_n, o_n, s_{n+1} \sim \pi}[\log(\tilde{p}^T(s_{n+1}|s_n, o_n, a_n)))] \quad (8)$$

The expectation is taken over agent π rollouts in the environment after skills are frozen (as opposed to over the offline dataset as in Equation (5)), but before training on the *transfer* domain. As such, we evaluate predictable semi-MDP transitions according to an agent optimised over the *source* domains, as desired. We use the same notation as in Section 3. $CE(\tilde{p}^T)$ represents how well our model $\tilde{p}^T$ can predict transitions across the semi-MDPs. This metric is influenced by both the individual transition predictability and total episodic number of transitions, N. We report results in Table 2. The lower the score the more predictable the transitions. In Table 2, the max value refers to the initial value achieved before offline training of π and $\tilde{p}^T$ and is shown to contextualise the results. Min refers to minimum value achieved across all methods. Interestingly, for AntMaze, with high dimensional state-action spaces and a long horizon, MO2 leads to the best predictability scores, thanks to its low switch rate transitioning at predictable bottleneck states (Figure 3). Conversely, for Maze2D, MO2's abstractions are not necessary for high predictability. This may partly explain why we do not observe *transfer learning* benefits here (Figure 4). For both domains, even though HO2-lim (offline) also reduces option switch rate when compared with HO2 (offline), this reduction does not lead to a lower CE, suggesting that penalising option-switching alone does not lead to predictable transitions. Altogether, these results suggest that Equations (6) and (7) lead to compressed temporal abstractions with predictable transitions, with benefits favouring longer horizon, higher dimensional domains (arguably where abstractions are most necessary).

Table 2: $CE(\tilde{p}^T)$ (see text)

Environment	Maze2D	AntMaze
HO2 (offline)	$\mathbf{0.45 \pm 0.00}$	7.20 ± 0.01
HO2-lim (offline)	1.00 ± 0.02	13.00 ± 1.00
MO2	0.70 ± 0.05	$\mathbf{4.00 \pm 0.05}$
Max (min)	5.00 (0.45)	50.00 (4.00)

6 Related Work

Sutton et al. (1999) introduced the options framework for discovering temporal abstractions. Multiple works built on this framework to improve option switching degeneracy (Kamat & Precup, 2020; Harb et al., 2018), sample efficiency (Wulfmeier et al., 2020; Riemer et al., 2018), off-policy corrections (Levy et al., 2017; Nachum et al., 2018) and optimisation (Wulfmeier et al., 2020; Smith et al., 2018; Zhang & Whiteson, 2019). Wulfmeier et al. (2020) present HO2, a hindsight off-policy options framework combining many of the previous advancements, enabling sample-efficient intra-option learning from off-policy data. Concurrent to our work, Klissarov & Precup (2021) published a similar method to HO2 for training all options off-policy. While aforementioned methods outperform non-hierarchical equivalents, optimising a purely control objective over *source* domains, without encouraging compressed options suited for planning, often leads to high-frequency skills that maximise return on *source* domains at the expense of effective exploration and credit-assignment during *transfer*.

MO2 is a bottleneck option method similar to McGovern & Barto (2001); Şimşek & Barto (2004); Solway et al. (2014); Harutyunyan et al. (2019); Ramesh et al. (2019) discovering temporal abstractions that induce plans of minimum description length over a set of tasks (Solway et al., 2014). Nevertheless, the following existing approaches do not support continuous state-spaces: Harutyunyan et al. (2019), as the predictability objective only supports tabular TD-learning; Solway et al. (2014), as they require constructing a graph over state-space transitions, thus not easily scalable; Şimşek & Barto (2004), building a partial transition graph with time complexity of $O(T^3)$ (T = horizon length); McGovern & Barto (2001), discovering regions of maximum diverse density using exhaustive search, thus scaling poorly. Whilst Ramesh et al. (2019) supports continuous spaces, they are on-policy, discovering successor options using PPO, thus unable to learn from diverse, past experience, commonly available (Khetarpal et al., 2020; Caesar et al., 2020). In contrast, MO2 is the first bottleneck options approach applied to the offline skill learning setting with continuous state-action spaces.

Hierarchical and variational approaches present alternate methods for discovering skills. Hausman et al. (2018); Haarnoja et al. (2018a); Wulfmeier et al. (2019) discover (multi-level) action abstractions suited for re-purposing, often by encouraging skill distinguishability, but do not support temporal abstractions. Singh et al. (2020); Merel et al. (2018; 2020); Salter et al. (2022) encourage temporally consistent behaviours by regularising against auto-regressive or learnt priors, yet do not inherently leverage temporal abstractions for value estimation. Pertsch et al. (2020); Pertsch et al.; Ajay et al. (2020); Co-Reyes et al. (2018) discover temporally abstract skills, shown essential for exploration, but predefine their temporal resolution. Additionally, unlike bottleneck approaches, prior works do not explicitly optimise for skills beneficial for planning and acting across tasks, potentially limiting their transfer learning benefits.

Similar to MO2 are methods that impose information-theoretic constraints on the discovered skills. Harutyunyan et al. (2019) introduce the termination critic, encouraging compressed options with a low entropic termination state distribution, $\min\{H(s_f|o)\}$. Unlike MO2, the termination critic learns options in a tabular setting, and does not support continuous state-action spaces, limiting their applicability to robotic domains. Furthermore, by building on HO2, MO2 enables intra-option learning, shown to be beneficial. Wang et al. (2019) present TAIC, that maximises mutual information between options and initial and termination states, $\max\{I(o; s_i, s_f)\}$, while simultaneously minimising individual mutual informations, $\min\{I(o; s_i) + I(o; s_f)\}$, thereby encouraging options independent of their initial and termination states. Wang et al. (2019) learn their abstractions via a variational auto-encoder approach. Unlike MO2, TAIC is on-policy and builds on PPO (Schulman et al., 2017) which reduces sample efficiency. Additionally, unlike MO2, TAIC's options have limited ability to specialise, due to their independence to initial and termination states. Shiarlis et al. (2018) introduce TACO, a weakly supervised approach for discovering temporally compressed abstractions aligning with sub-tasks, given a sub-task sketch by the user. MO2 discovers skills unsupervised presenting a more viable approach in many scenarios where supervision is expensive.

Finally, in contrast to MO2, Machado et al. (2017) propose eigen-, instead of bottleneck-, options, with superior performance in *online* settings when the goal is far away from the *bottleneck states* (e.g. room corners for the 4-room domain). Eigen-options minimise diffusion time of an agent across the environment. We consider extending the eigen-options framework to the *offline* setting as future work. Similarly, approaches like Jinnai et al. (2019) discover options (or latent actions) with wide state coverage (Eysenbach et al., 2018; Gregor et al., 2016; Achiam et al., 2018; Kamat & Precup, 2020). These works take the intrinsic-curiosity RL paradigm. Whilst encouraging diversity is important for skill development when learning *online* from scratch (over sparse rewards), it is unclear whether these skills are temporally compressed and suited for value estimation. We consider extending MO2 to the online skill learning context, and learning from sub-optimal *source domain* demonstrations (Peng et al., 2019), as future work.

7 Conclusions

We introduce Model-Based Offline Options, MO2, a novel offline hindsight bottleneck options framework supporting continuous state-action spaces, that discovers skills suited for planning (in the form of value estimation) and acting across modular tasks (MDPs whose optimal policies are obtained by recomposing shared temporal abstractions). MO2 achieves this with a *predictability* objective encouraging predictable option-terminations across an episode. Once options are learnt offline, we freeze them and perform online transfer learning over the option-induced semi-MDP, learning and acting over the temporally abstract skill space. We compare MO2 against state-of-the-art baselines on complex continuous control domains and perform an extensive ablation study. We demonstrate that MO2's options are bottleneck aligned and improve acting (and exploring), value estimation, and learning of a jumpy option-transition model. On the challenging, sparse, long-horizon, AntMaze domain, MO2 drastically outperforms all baselines.

REFERENCES

Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation. *arXiv preprint arXiv:1806.06920*, 2018.

Joshua Achiam, Harrison Edwards, Dario Amodei, and Pieter Abbeel. Variational option discovery algorithms. *arXiv preprint arXiv:1807.10299*, 2018.

Anurag Ajay, Aviral Kumar, Pulkit Agrawal, Sergey Levine, and Ofir Nachum. Opal: Offline primitive discovery for accelerating offline reinforcement learning. *arXiv preprint arXiv:2010.13611*, 2020.

Pierre Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In *31st AAAI Conference on Artificial Intelligence, AAAI 2017*, pp. 1726–1734, 2017.

Serkan Cabi, Sergio Gómez Colmenarejo, Alexander Novikov, Ksenia Konyushkova, Scott Reed, Rae Jeong, Konrad Zolna, Yusuf Aytar, David Budden, Mel Vecerik, et al. Scaling data-driven robotics with reward sketching and batch reinforcement learning. *arXiv preprint arXiv:1909.12200*, 2019.

Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11621–11631, 2020.

John Co-Reyes, YuXuan Liu, Abhishek Gupta, Benjamin Eysenbach, Pieter Abbeel, and Sergey Levine. Self-consistent trajectory autoencoder: Hierarchical reinforcement learning with trajectory embeddings. In *International Conference on Machine Learning*, pp. 1009–1018. PMLR, 2018.

Sudeep Dasari, Frederik Ebert, Stephen Tian, Suraj Nair, Bernadette Bucher, Karl Schmeckpeper, Siddharth Singh, Sergey Levine, and Chelsea Finn. Robonet: Large-scale multi-robot learning. *arXiv preprint arXiv:1910.11215*, 2019.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*, 2018.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control. *arXiv preprint arXiv:1611.07507*, 2016.

Tuomas Haarnoja, Kristian Hartikainen, Pieter Abbeel, and Sergey Levine. Latent space policies for hierarchical reinforcement learning. In *International Conference on Machine Learning (ICML)*, 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 (ICML)*, 2018b.

Jean Harb, Pierre-Luc Bacon, Martin Klissarov, and Doina Precup. When waiting is not an option: Learning options with a deliberation cost. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Anna Harutyunyan, Will Dabney, Diana Borsa, Nicolas Heess, Remi Munos, and Doina Precup. The termination critic. *arXiv preprint arXiv:1902.09996*, 2019.

Karol Hausman, Jost Tobias Springenberg, Ziyu Wang, Nicolas Heess, and Martin Riedmiller. Learning an embedding space for transferable robot skills. In *International Conference on Learning Representations*, 2018.

Yuu Jinnai, Jee Won Park, Marlos C Machado, and George Konidaris. Exploration in reinforcement learning with deep covering options. In *International Conference on Learning Representations*, 2019.

Anand Kamat and Doina Precup. Diversity-enriched option-critic. *arXiv preprint arXiv:2011.02565*, 2020.

Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards continual reinforcement learning: A review and perspectives. *arXiv preprint arXiv:2012.13490*, 2020.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Martin Klissarov and Doina Precup. Flexible option learning. *Advances in Neural Information Processing Systems*, 34, 2021.

Andrew Levy, George Konidaris, Robert Platt, and Kate Saenko. Learning multi-level hierarchies with hindsight. *arXiv preprint arXiv:1712.00948*, 2017.

T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Marlos C Machado, Marc G Bellemare, and Michael Bowling. A laplacian framework for option discovery in reinforcement learning. In *International Conference on Machine Learning*, pp. 2295–2304. PMLR, 2017.

Amy McGovern and Andrew G Barto. Automatic discovery of subgoals in reinforcement learning using diverse density. 2001.

Josh Merel, Leonard Hasenclever, Alexandre Galashov, Arun Ahuja, Vu Pham, Greg Wayne, Yee Whye Teh, and Nicolas Heess. Neural probabilistic motor primitives for humanoid control. *arXiv preprint arXiv:1811.11711*, 2018.

Josh Merel, Saran Tunyasuvunakool, Arun Ahuja, Yuval Tassa, Leonard Hasenclever, Vu Pham, Tom Erez, Greg Wayne, and Nicolas Heess. Catch & carry: reusable neural controllers for vision-guided whole-body tasks. *ACM Transactions on Graphics (TOG)*, 39(4):39–1, 2020.

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):529–533, 2015.

Kaichun Mo, Haoxiang Li, Zhe Lin, and Joon-Young Lee. The adobeindoornav dataset: Towards deep reinforcement learning based real-world indoor robot visual navigation. *arXiv preprint arXiv:1802.08824*, 2018.

Ofir Nachum, Shixiang Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. *arXiv preprint arXiv:1805.08296*, 2018.

Michael Neunert, Abbas Abdolmaleki, Markus Wulfmeier, Thomas Lampe, Tobias Springenberg, Roland Hafner, Francesco Romano, Jonas Buchli, Nicolas Heess, and Martin Riedmiller. Continuous-discrete reinforcement learning for hybrid control in robotics. In *Conference on Robot Learning*, pp. 735–751. PMLR, 2020.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Karl Pertsch, Youngwoon Lee, Yue Wu, and Joseph Lim. Guided reinforcement learning with learned skills.

Karl Pertsch, Youngwoon Lee, and Joseph J Lim. Accelerating reinforcement learning with learned skill priors. *arXiv preprint arXiv:2010.11944*, 2020.

Rahul Ramesh, Manan Tomar, and Balaraman Ravindran. Successor options: An option discovery framework for reinforcement learning. *arXiv preprint arXiv:1905.05731*, 2019.

Matthew Riemer, Miao Liu, and Gerald Tesauro. Learning abstract options. *arXiv preprint arXiv:1810.11583*, 2018.

Sasha Salter, Kristian Hartikainen, Walter Goodwin, and Ingmar Posner. Priors, hierarchy, and information asymmetry for skill transfer in reinforcement learning. *arXiv preprint arXiv:2201.08115*, 2022.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Kyriacos Shiarlis, Markus Wulfmeier, Sasha Salter, Shimon Whiteson, and Ingmar Posner. Taco: Learning task decomposition via temporal alignment for control. *arXiv preprint arXiv:1803.01840*, 2018.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550 (7676):354–359, 2017.

Özgür Şimşek and Andrew G Barto. Using relative novelty to identify useful temporal abstractions in reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 95, 2004.

Avi Singh, Huihan Liu, Gaoyue Zhou, Albert Yu, Nicholas Rhinehart, and Sergey Levine. Parrot: Data-driven behavioral priors for reinforcement learning. *arXiv preprint arXiv:2011.10024*, 2020.

Matthew J.A. Smith, Herke Van Hoof, and Joelle Pineau. An inference-based policy gradient method for learning options. In *35th International Conference on Machine Learning, ICML 2018*, volume 11, pp. 7481–7490, 2018. ISBN 9781510867963. URL https://pdfs.semanticscholar.org/809f/951c77b5a39e2a9d556e9cf9938de87f2393.pdf?{_}ga=2.168186657.1832697831.1553020853-1357972575.1551696370.

Alec Solway, Carlos Diuk, Natalia Córdova, Debbie Yee, Andrew G Barto, Yael Niv, and Matthew M Botvinick. Optimal behavioral hierarchy. *PLoS computational biology*, 10(8):e1003779, 2014.

Martin Stolle and Doina Precup. Learning options in reinforcement learning. In *International Symposium on abstraction, reformulation, and approximation*, pp. 212–223. Springer, 2002.

Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.

Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

Wenshan Wang, Yaoyu Hu, and Sebastian Scherer. Learning temporal abstraction with information-theoretic constraints for hierarchical reinforcement learning. 2019.

Markus Wulfmeier, Abbas Abdolmaleki, Roland Hafner, Jost Tobias Springenberg, Michael Neunert, Tim Hertweck, Thomas Lampe, Noah Siegel, Nicolas Heess, and Martin Riedmiller. Compositional Transfer in Hierarchical Reinforcement Learning. (1), 2019. URL http://arxiv.org/abs/1906.11228.

Markus Wulfmeier, Dushyant Rao, Roland Hafner, Thomas Lampe, Abbas Abdolmaleki, Tim Hertweck, Michael Neunert, Dhruva Tirumala, Noah Siegel, Nicolas Heess, et al. Data-efficient hindsight off-policy option learning. *arXiv preprint arXiv:2007.15588*, 2020.

Shangtong Zhang and Shimon Whiteson. DAC: The Double Actor-Critic Architecture for Learning Options. (NeurIPS), 2019. URL http://arxiv.org/abs/1904.12691.

Table 3: Feedforward Module, $\pi^{\{C,L\}}, \beta, \tilde{p}^T, Q$

hidden layers	$(256, 256)$
hidden layer activation	elu
output activation	linear

Table 4: Offline Skill Discovery Setup

Environment	Maze2D	AntMaze
$\pi^{\{C,L\}}, \beta$ learning rate	$3e^{-4}$	$3e^{-4}$
$\tilde{p}^T$ learning rate	$3e^{-4}$	$3e^{-4}$
number of options	4	4
α predictability weight	1.0	0.2
m $\log(\tilde{p}^T)$ offset	13	103
batch size	256	128
sequence length	100	100

APPENDIX

We provide the reader with experimental details required to reproduce the results in the main paper.

A ARCHITECTURE

We continue by outlining the model architectures across domains and experiments. Each policy network (π^C, π^L, β) and the option-transition model $\tilde{p}^T$ are comprised of a feedforward module outlined in Table 3. π^C then has a component layer of size 128 for each option. We apply a tanh or softmax activations over network outputs, where appropriate, to match the predefined output ranges of any given module. The critic, Q, is also a feedforward module. For π^C we use a categorical latent space of size 4 (number of options). We find this dimensionality suffices for expressing the diverse behaviours in our domains. $\tilde{p}^T$ is also a gaussian mixture model with 4 heads. On the transfer experiments we freeze π^L, β and train a newly instantiated π^C to recompose prior skills. $\tilde{p}^T$ is not used during online transfer.

B ALGORITHMIC DETAILS

B.1 OFFLINE SKILL DISCOVERY

In Equation (6) we introduce MO2's objective O_{mo2} for discovering bottleneck options. In practice, we actually use the following slightly modified objective:

$$\max_{\pi^L, \pi^C, \beta} O_{mo2}(\pi^L, \pi^C, \beta) = \mathop{\mathbb{E}}_{\substack{s_t, a_t, h_t \sim D \\ o \sim \pi^H(\cdot|h_t), a \sim \pi^L(\cdot|s_t, o) \\ s_f \sim \tilde{p}^T(\cdot|s_t, o, a)}} [\alpha\beta(s_t|h_t)\log(\tilde{p}^T(s_f|s_t, o, a) - m) + \log(\pi(a_t, o|h_t))] \quad (9)$$

Table 5: Online Transfer Learning Setup

Environment	Maze2D	AntMaze
Q learning rate	$1e^{-4}$	$1e^{-4}$
π^C learning rate	$1e^{-4}$	$1e^{-4}$
number options	4	4
ϵ	1	1
ϵ_{KL}	1	1
data-grad ratio	5000	50
batch size	256	256
sequence length	10	10

m and α are hyperparameters. m is set to the maximum possible $\tilde{p}^T(s_f|s_t, o, a)$ value and is a function of the minimum permitted covariance Σ of our model $\tilde{p}^T$ (σ set to $1e^{-3}$ for $\Sigma = \sigma\mathbb{I}$). As such, m ensures O_{pred} is always negative, thus encouraging minimal switches that align with bottleneck states, as outlined in Figure 2. α is set using grid search (specifically $[1e^{-1}, 2e^{-1}, 4e^{-1}, 6e^{-1}, 8e^{-1}, 1e^{0}]$), and weighs the importance of the two objects O_{pred} and O_{bc}. The offline data collected by behavioural policy, $b(a|h)$, consists of tuples of experience in the form (s_t, a_t).

We train the options framework (π^C, π^L, β) and option-transition model $\tilde{p}^T$ in parallel using Equation (9) and Equation (7) respectively. Algorithmic hyperparameter details are outlined in Table 4. We run four seeds. During transfer, we select the seed that traverses the maze deepest. We note, however, that there is minimal difference between each seed. *Batch size* refers to the number of independently sampled trajectory snippets from the offline dataset, each of length *sequence length*. We sample these batches uniformly, according to Wulfmeier et al. (2020). We run experiments until O_{mo2} convergence ($1e^6$ gradients). For the baselines, *HO2 (offline), HO2-lim (offline)*, we set α to zero, yet still train $\tilde{p}^T$ for evaluation purposes. The other hyperparameters are kept constant. Finally, for *HO2-lim (offline)*, we run a sweep over N (specifically $[1, 5, 10]$), the number of permitted option switches, when training on Equation (17) from the Appendix of Wulfmeier et al. (2020). We report results for $N = 1$, as this is the experiment that achieves the lowest switch rate.

B.2 Online Transfer Learning

During this stage of training we freeze the option-policies (π^L, β). The categorical option controller, π^C is initialized randomly on the transfer task (inline with Wulfmeier et al. (2020)) and is trained to reorder options to solve the task. We note that while some skill transfer approaches (Salter et al., 2022; Pertsch et al., 2020) note performance gains by additionally transferring π^C, we did not observe this. The RL setup is akin to the CMPO (Neunert et al., 2020). We note any changes in Table 5. Hyper-parameter sweeps are performed for *data collection to gradients ratio* (data-grad ratio) (specifically $[5e^1, 5e^2, 5e^3]$) for each method as we find this makes a significant difference. We report the most performant setting for each approach. For HO2-scratch, we train an HO2 agent from scratch, using the same architecture as the other methods, and use the default setting/hyperparameters in Wulfmeier et al. (2020).

B.2.1 MDP learning

For this setup, the agent acts in the environment at the option-level. Learning occurs at the original MDP level, however. The following, per timestep, tuple of experiences are stored in the replay-buffer: $\{s_t, o_t, r_t, s_{t+1}\}$. We use these samples to train the critic and policy, using CMPO. We perform temporal-difference TD(0) learning (Sutton, 1988) for training the critic. Crucially, this occurs per timestep, at the original temporal resolution of the MDP. As such, the option's temporal abstractions are not used to perform more efficient TD learning across long horizons.

B.2.2 Semi-MDP learning

For this setup, the agent acts in the environment at the option-level. Learning also occurs at the option-level (over the option-induced semi-MDP). The following tuple of experiences are stored in the replay-buffer: $\{s_t, o_t, g_t, s_{t+k}\}$, where k is specified by the option's sampled termination condition in the environment. s_t represents the option's o_t initiation state, s_{t+k} the termination state, and $g_t = \sum_{i=t}^{t+k} \gamma^{i-t} r_i$ the discounted cumulative rewards during its execution. We use these samples to train the critic and policy, using CMPO. We perform temporal-difference TD(0) learning (Sutton, 1988) for training the critic. Crucially, this occurs at the option-level, over the semi-MDP. As such, the option's temporal abstractions are used to perform more efficient TD learning across long horizons.

C Environments

In this section, we describe in detail the D4RL (Fu et al., 2020) domains used in our experiments.

C.1 Maze2D

For this D4RL domain (Fu et al., 2020) a 2D pointmass agent must reach a fixed goal location. We test our algorithm on the 'maze2D-large' variant with the maze layout as depicted in Figure 6.

C.1.1 Source Domains

The offline data D is generated by randomly selecting goal locations in the maze and then using a planner that generates sequences of waypoints (as shown in Figure 6) that are followed using a PD controller to reach the goal. Once the goal is reached, a new goal is sampled and the process continues. Distinct goals require distinct corridor traversals. As such, corridor intersections represent the bottleneck states for these source domains. The maze layout remains static and is not randomised between tasks. The offline dataset contains $4 * 10^6$ environment transitions (in the form of (s, a) tuples). The state-space corresponds to 2D Cartesian coordinates of the agent and the actions correspond to its velocity, clipped within the range of $(-1, 1)$ per dimension. Crucially, goal location is not provided to the agent, as the agent must learn task agnostic behaviours. This dataset is publically available from Fu et al. (2020).

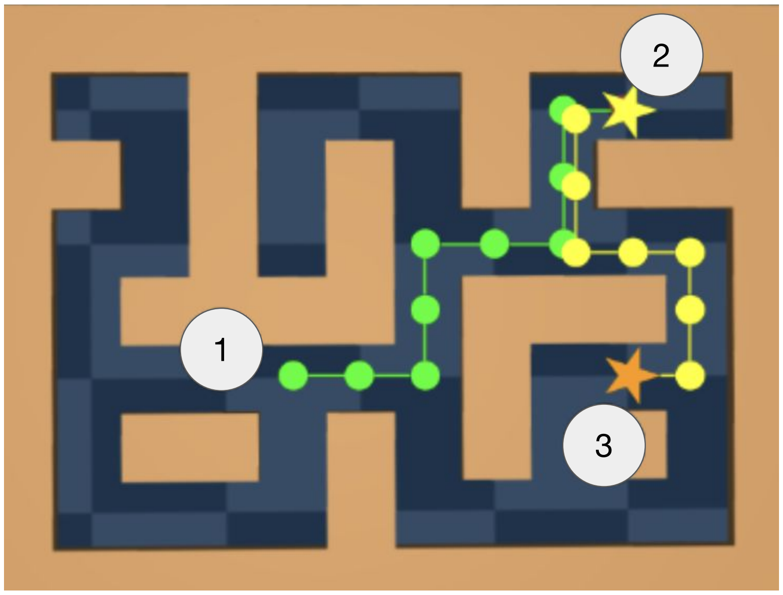

Figure 6: *Offline Data Generation Example.* The agent randomly spawns in (1). Goal location (2) is then randomly sampled (denoted by a star). A planner generates waypoints to reach the goal (shown as circles). For *Maze2D*, a PD controller is used to follow the waypoints. For *AntMaze*, a trained goal-reaching policy (Fu et al., 2020) is used to follow them. Once the goal location (2) is reached, the next goal location (3) is randomly sampled and (2) becomes the spawning location. The maze layout remains static, as displayed, between *source* and *transfer* domains. During *transfer*, an unknown goal location must be reached. See text for further details. Figure modified from Fu et al. (2020).

C.1.2 Transfer Domains

We evaluate our algorithms on a modified version of 'maze2d-eval-large' from Fu et al. (2020). The maze layout remains the same as the source domains'. The goal location now remains static, is not randomised, and the agent must discover its location and how to consistently reach it as quickly as possible. The original Maze2D transfer domain spawns the agent (at the start of the episode) uniformly across the maze. The agent is rewarded only upon goal reaching (+1). The agent sometimes spawns close to the goal, other times far away. As such, credit assignment and exploration are not particularly problematic as the spawn locations provide a form of curriculum for learning values and exploring the environment across all starting locations in the maze. Therefore, this domain does not necessitate temporal abstractions for acting and learning. To increase the difficulty, we modify the environment such that the agent always spawns at the furthest distance from the goal (with respect to steps required to reach target). Now, abstractions are more important for credit assignment and exploration. Nevertheless, the state-action space for this domain is low dimensional, and the episodic horizon is not long, meaning the task is still not particularly sparse. Additionally, to keep with the modular task setting (as detailed in Section 1), we terminate the episode early when the goal is reached (unlike in the original domain). Below we detail the maze layout for our modified setting:

```
// Modified Maze2D
// R = spawn location, G = goal location, 1 = walls, 0 = empty space

Maze2D = [[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
          [1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1],
          [1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1],
          [1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1],
          [1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1],
          [1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1],
          [1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1],
          [1, R, 0, 1, 0, 0, 0, 1, 0, 0, G, 1],
          [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]]
```

C.2 AntMaze

This D4RL domain (Fu et al., 2020) is a navigation domain that replaces the 2D pointmass from Maze2D with a complex 8-DoF 'Ant' quadraped robot. This domain is introduced to mimic more challenging real-world robotic nagivation tasks. We test our algorithm on 'ant-large-diverse' variant using the same maze layout as depicted in Figure 6.

C.2.1 Source Domains

The offline data D is created by training a goal reaching policy and using it in conjunction with the same high-level waypoint generator from Maze2D to provide sub-goals that guide the agent to the goal. For the 'ant-large-diverse' dataset, the ant is commanded to a random goal location from a random start location. Both locations are randomly sampled across the maze. Akin to Maze2D, distinct goals require distinct corridor traversals. As such, corridor intersections represent the bottleneck states for these source domains. The maze layout remains static and is not randomised between tasks. The offline dataset contains 10^6 environment transitions (in the form of (s, a) tuples). The state-space is 111-dimensional corresponding to position and velocity of each joint, and contact forces. The action space is a 8-dimensional continuous space corresponding to the torque applied to each actuator, clipped between $(-1, 1)$ per dimension. Crucially, goal location is not provided to the agent, as the agent must learn task agnostic behaviours. This dataset is publically available from Fu et al. (2020).

C.2.2 Transfer Domains

We evaluate our algorithms on a modified version of 'ant-eval-large-diverse' from Fu et al. (2020). The maze layout remains the same as the source domains'. The goal location now remains static, is not randomised, and the agent must discover its location and how to consistently reach it as quickly as possible. Unlike the source domains, the agent's spawning location is not randomised between episodes and consistently spawns in the corner of the maze. The agent is rewarded only upon goal reaching $(+1)$. For this transfer domain, we experimentally find that reaching the original target location is tricky for all methods. Whilst MO2 strongly outperforms all other approaches, its success rate is still low (within the allocated online training budget). We suspect this may be an issue with the number of (sub-)trajectories offline reaching the goal location. All our methods discover skills by maximising the log-likelihood of offline data. If the goal is rarely reached offline, neither method will favour discovering skills that reach it. As such, we change the goal location to a slightly closer location, more representative of offline behaviours (which we visually inspected). We used the following maze layout:

```
// Modified AntMaze
// R = spawn location, G = goal location, 1 = walls, 0 = empty space

AntMaze = [[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
           [1, R, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1],
           [1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1],
           [1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1],
           [1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1],
           [1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1],
           [1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1],
           [1, 0, 0, 1, 0, 0, G, 1, 0, 0, 0, 1],
           [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]]
```

D Proofs and Discussions

We continue by highlighting why MO2's objective leads to bottleneck state discovery. We provide proofs coupled with explanations.

D.1 Why does Equation (4) lead to bottleneck options?

We continue by demonstrating why Equation (4) leads to the minimal number of low entropic, transition state distributions, able to reconstruct offline behaviours. To achieve this we first demonstrate why maximising Equation (4) is equivalent to minimising the conditional entropy of plans. $\mathbb{H}_\pi[s_{0:N}|o_{0:N}, a_{0:N}]$ represents the conditional entropy of plans over sequential option-transition states $s_{0:N}$ conditioned on sequential options $o_{0:N}$ and their corresponding first initiated action $a_{0:N}$.

$$\begin{aligned}\mathbb{H}_\pi[s_{0:N}|o_{0:N},a_{0:N}] &= -\mathbb{E}_{s_{0:N},o_{0:N},a_{0:N}\sim\pi}[\log(p_\pi(s_{0:N}|o_{0:N},a_{0:N}))]\\ &= -\mathbb{E}_{s_{0:N},o_{0:N},a_{0:N}\sim\pi}[\prod_{n=0}^{N}\log(p^T(s_{n+1}|s_n,o_n,a_n))]\\ &= -\sum_{n=0}^{N}\mathbb{E}_{s_{0:N},o_{0:N},a_{0:N}\sim\pi}[\log(p^T(s_{n+1}|s_n,o_n,a_n))]\\ &= -\sum_{n=0}^{N}\mathbb{E}_{s_n,o_n,a_n,s_{n+1}\sim\pi}[\log(p^T(s_{n+1}|s_n,o_n,a_n))]\\ &= \sum_{n=0}^{N}\mathbb{H}_\pi[s_{n+1}|s_n,o_n,a_n]\\ &:= -O_{pred}\ \text{(Equation (4))}\end{aligned} \tag{10}$$

We subscript $\mathbb{H}$ with π to emphasise that this entropy (and plans) is dependent on our option-policy π that defines the option-transition model $p^T(s_{n+1}|s_n,o_n,a_n)$ and how options and actions are chosen (from $\pi^C(o|s)$ and $\pi^L(a|s,o)$). $s_{0:N},o_{0:N},a_{0:N}\sim\pi$ represents that the expectation over sequential transition states, options and actions are sampled from our policy π acting in the environment (i.e. from $p_\pi(s_{0:N},o_{0:N},a_{0:N})$).

The second line of Equation (10) holds due to the product rule and independence between s_{n+1} and all other variables after conditioning on s_n,o_n,a_n, given our graphical model of the agent π and environment. The fourth line holds due to the integral of all extraneous variables, that p^T is not dependent on, integrating to 1. The final line is equivalent to the fourth by the definition of Equation (4).

We therefore demonstrate that maximising the O_{pred} objective in Equation (4) is equivalent to minimising the conditional entropy of plans over sequential option-transition states $\mathbb{H}_\pi[s_{0:N}|o_{0:N},a_{0:N}]$ as well as the cumulative sum of individual conditional option-transition entropies $\sum_{n=0}^{N}\mathbb{H}_\pi[s_{n+1}|s_n,o_n,a_n]$. If individual entropies are enforced to be positive (achievable as shown in Appendix B) then to minimise Equation (10), transitions should be as sparse as possible as to minimise the accumulation of positive option-transition entropies.

If we assume for now that $\pi(a|h)=b(a|h)$ (with b representing the behavioural distribution that created the offline data), and constant environment dynamics, then the distribution over trajectories induced by π equates to the offline data distribution over the *source* domains. As such, Equation (10) encourages a decomposition of offline behaviours into compressed temporal abstractions between sequential decision states able to reconstruct offline behaviours by their recomposition (though options) with high confidence (low $\mathbb{H}_\pi[s_{0:N}|o_{0:N},a_{0:N}]$). Additionally, individual option-transitions entropies $\mathbb{H}_\pi[s_{n+1}|s_n,o_n,a_n]$ should be low, thus aligning with bottleneck states as depicted in Figure 2. Altogether, Equation (4) leads to the *optimal-compression (bottleneck state alignment)* scenario in Figure 2.

In practice, $\pi(a|h)\neq b(a|h)$. However, the O_{bc} objective in Equations (6) and (9) encourages both to align with each other. The α hyper-parameter in Equation (9) is set low enough such that the O_{pred} component of MO2's objective does not hinder π's ability to align with b. Thus, over time, as π distils the offline behaviours, Equations (6) and (9) will discover bottleneck options able to reconstruct the offline modular behaviours across the *source* MDPs.

D.2 Under what assumptions does optimising O_{pred} from Equation (4) = Equation (5)?

Equations (6) and (9) do not use the form of O_{pred} from Equation (4) but instead the alternate definition in Equation (5). We continue by highlighting under what assumptions optimising either form is equivalent.

Our final proof relies on two concepts. Firstly, instead of considering individual n^{th} option initiation state distributions $p_\pi(s_n)$ across the episode, we consider the stationary option initiation state distribution $p_\pi(s_i)$ by marginalising over the transition number dimension n:

$$p_\pi(s_i) = E_{n\sim\pi}[p_\pi(s_n)] \tag{11}$$

s_i denotes the marginal initiation state distribution, independent of option transition number n. The distribution over n is dependent on π which is why $n\sim\pi$ in the expectation. The second concept that we rely on is how to obtain the same marginal initiation state distribution by marginalising over the t rather than n. As highlighted in Section 3.1 we have a closed form solution to a related quantity $\beta(s_t|h_t)$ that represents the probability that s_t is initial given history h_t. Given this quantity, we can calculate $p_\pi(s_i)$ by marginalising across t and h_t as follows:

$$p_\pi(s_i) = E_{t\sim U\{0,T\},h_t\sim\pi}[\beta(s_t|h_t)p_\pi(s_t|h_t)] \tag{12}$$

$U\{0, T\}$ denotes a uniform categorical distribution between 0 and T, the episodic length. $p_\pi(s_t|h_t)$ denotes the probability state distribution at timestep t given history h_t, for policy π. Equation (12) without the $\beta(s_t|h_t)$ term corresponds to the stationary state visitation distribution of policy π (e.g. $p_\pi(s) = E_{t\sim U\{0,T\},h_t\sim\pi}[p_\pi(s_t|h_t)]$ which is independent of t). The additional weights weigh how probable that any s_t is an option initiation state given h_t according to our option-policy π. We continue by using Equations (11) and (12) to show to relation between Equations (4) and (5):

$$\begin{aligned} O_{pred} \text{ (Equation (4))} &= \sum_{n=0}^{N} \mathbb{E}_{s_n,o_n,a_n,s_{n+1}\sim\pi}[\log(p^T(s_{n+1}|s_n,o_n,a_n))] \\ &= N\mathbb{E}_{\substack{s_i\sim\pi \\ o\sim\pi^C(\cdot|s_i),a\sim\pi^L(\cdot|s_i,o) \\ s_f\sim p^T(\cdot|s_i,o,a)}}[\log(p^T(s_f|s_i,o,a))] \\ &\propto \mathbb{E}_{\substack{s_i\sim\pi \\ o\sim\pi^C(\cdot|s_i),a\sim\pi^L(\cdot|s_i,o) \\ s_f\sim p^T(\cdot|s_i,o,a)}}[\log(p^T(s_f|s_i,o,a))] \\ &= \mathbb{E}_{\substack{t\sim U\{0,T\},s_t,h_t\sim\pi \\ o\sim\pi^H(\cdot|h_t),a\sim\pi^L(\cdot|s_t,o) \\ s_f\sim p^T(\cdot|s_t,o,a)}}[\beta(s_t|h_t)\log(p^T(s_f|s_t,o,a))] \\ &= \mathbb{E}_{\substack{s_t,h_t\sim D \\ o\sim\pi^H(\cdot|h_t),a\sim\pi^L(\cdot|s_t,o) \\ s_f\sim p^T(\cdot|s_t,o,a)}}[\beta(s_t|h_t)\log(p^T(s_f|s_t,o,a))] \text{ if } \pi(a|h) = b(a|h) \\ &:= O_{pred} \text{ (Equation (5))} \end{aligned} \tag{13}$$

The second line of Equation (13) is equivalent to the first by marginalising over the depth dimension n, akin to Equation (11), and then grouping into marginal initial transition state distribution $s_i \sim \pi$ of options and their conditional terminal state distribution $s_f \sim p^T(\cdot|s_i, o, a)$. $s_i \sim \pi$ is shorthand for sampling from $p_\pi(s_i)$. In the expectation, we explicitly report how options and actions are sampled. N represents the expected number of transitions across episodes. The third line is proportional to the second by a factor of N. The fourth line is obtained by applying Equation (12) and grouping $p_\pi(s_t|h_t)$ into the expectation (i.e. $s_t \sim \pi$).

In practice, we do not have access to s_t and h_t samples from option-policy π and thus cannot calculate the expectation in the fourth line. Nevertheless, if we assume $\pi(a|h) = b(a|h)$, then we can use offline data samples (from D) instead. As such, we obtain the fifth line from Equation (13). We omit $t \sim U\{0, T\}$ for simplicity and to keep notation consistent with Section 3.1. However, we are still sampling t as aforementioned. The final line holds by definition. We have therefore proved that Equation (4) $\propto$ Equation (5) under the assumption that $\pi(a|h) = b(a|h)$. Proportionality does not influence optimisation and therefore both are equivalent from a learning perspective. As mentioned in Appendix D.1, $\pi(a|h) \neq b(a|h)$. However, O_{bc} encourages both to align over time, ensuring O_{pred} leads to bottleneck options.