GOING BEYOND STATE-REACHING: LEARNING ABSTRACTIONS FOR INTRINSICALLY MOTIVATED OPTION DISCOVERY

Anonymous authors
Paper under double-blind review

ABSTRACT

Temporal abstraction via options can improve exploration in large environments. However, existing option discovery algorithms find subgoals that target all aspects of the state simultaneously. This *state-reaching* approach produces options that only apply in narrow regions of the state-space, eventually causing an explosion in the number of options that overwhelms the agent, and impedes progress on its primary task of reward maximization. We introduce an algorithm that instead identifies a small, relevant subset of features for each subgoal, yielding options that generalize broadly and accelerate exploration. Our approach learns abstract, transferrable options and achieves rapid exploration in three sparse-reward, image-based domains, including the Atari game MONTEZUMASREVENGE.

1 INTRODUCTION

Reinforcement learning (RL) agents must interact with the environment at every timestep. Yet, effective decision-making may demand temporal abstractions to ease central challenges of credit assignment, exploration, and transfer (Klissarov et al., 2025). The core question is—how can RL agents autonomously discover temporally extended actions, or *options* (Sutton et al., 1999), solely via environment interaction? This problem—skill or option discovery—is the core bottleneck for scaling hierarchical RL (HRL; Barto & Mahadevan, 2003) to long-horizon, high-dimensional problems (Konidaris, 2019; Gershman, 2017; Sutton et al., 2022). However, most skill discovery algorithms today have a common shortcoming: they resort to the state-reaching objective, i.e., they create skills that target entire states, rather than small, relevant portions of those states.

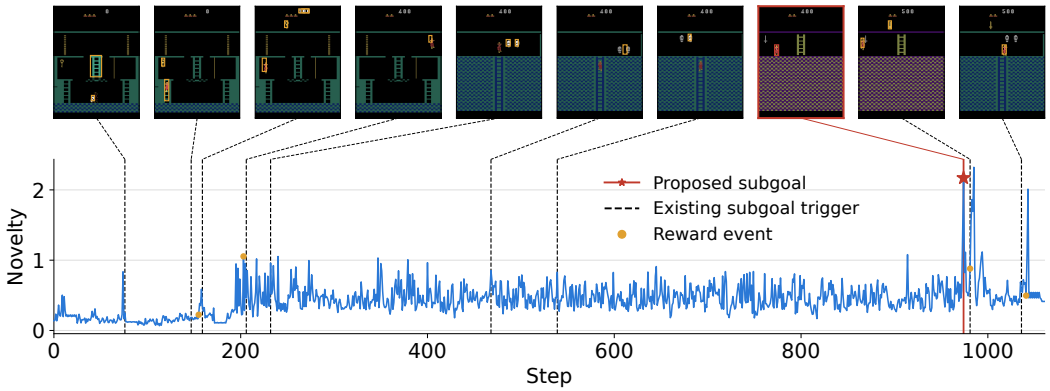


Figure 1: **Abstract subgoal discovery in a MONTEZUMASREVENGE episode.** The blue curve shows novelty over an episode. Frames show achieved option subgoals; yellow bounding boxes isolate the parts of the image the option's subgoal classifier attends to, ignoring all other pixels. The frame with the red box denotes the subgoal for a new option discovered in this episode.

To understand state-reaching, recall the moment you first balanced a bicycle without training wheels and realized you had done something right. Practicing this new skill later, you recreated your speed

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

and balance, but not the color of your t-shirt, the weather, or the exact wear of your tires. By contrast, existing skill discovery algorithms that learn to recreate a previously encountered experience (Chentanez et al., 2005) attempt to recreate *every aspect* of that state. Examples include the ϵ-ball from deep skill graphs (Bagaria et al., 2021b), pixel-wise equality (Veeriah et al., 2018), the perceptual hashing from Go-Explore (Ecoffet et al., 2021), random projections of the target state (Dabney et al., 2021; Farebrother et al., 2023), or a neural classifier that predicts perceptual similarity between states (Vezhnevets et al., 2017; Hafner et al., 2022). This state-reaching strategy hurts hierarchical agents in several ways. First, the resulting skills are nontransferable; consider a robot grabbing a cup from a cluttered table: the skill's subgoal is tied to the position of every object in the state, requiring the agent to learn combinatorially more skills. Beyond stymieing transfer—a key benefit of HRL (Taylor & Stone, 2009)—state-reaching artificially reduces the size of the skill's subgoal region, complicating policy learning and eventually making the skill less useful. Finally, recreating a state is *unscalable* because in large, realistic domains, recreating all aspects of a particular state may simply be impossible (Bagaria & Schaul, 2023; Javed & Sutton, 2024).

Our skill discovery algorithm is inspired by intrinsic motivation (Oudeyer et al., 2008; Colas et al., 2022): during play, when children cause something interesting to happen, they try to recreate it, with increasing efficiency (White, 1959). They practice until they get bored, then move on, retaining the new skill for later reuse (Barto et al., 2004). Similarly, when an AI agent's exploration reveals something particularly interesting, it should learn a skill to reliably achieve it again. Other algorithms (e.g., Bagaria et al., 2021b; 2025; Jinnai et al., 2020) use these ideas, but resort to a state-reaching objective. Instead, when the agent encounters a state that produces a spike in novelty (Chentanez et al., 2005), we use *feature attribution* to identify the features responsible, and construct a subgoal classifier that attends only to those specific features.[1] These classifiers serve as reward signals for training each option's policy, and a high-level policy selects among these resulting options to maximize the agent's overall reward.

We test our algorithm in three challenging exploration problems: MINIGRID-KEYCORRIDOR, VISUALTAXI, and the Atari game MONTEZUMASREVENGE—all with image-based observations and sparse reward functions. We focus on image-based domains to stress-test our algorithm: when a factored state representation is available, each dimension serves as a candidate feature and a streamlined version of our algorithm applies directly. Our method outperforms both non-hierarchical (flat) RL baselines and a state-reaching hierarchical baseline that shares our subgoal discovery algorithm but builds classifiers over the entire image. Our discovered subgoals depend on only a small number of features and are achieved across qualitatively different states, enabling skill reuse.

Contributions. Conceptually, we identify the implicit state-reaching assumption shared by existing option discovery methods and argue that identifying a relevant subset of features per subgoal is a crucial missing step. Methodologically, we introduce an algorithm that detects novelty spikes during exploration, attributes them to a small set of features, and constructs subgoal classifiers that attend only to those features, resulting in options that can be reused in different contexts. Empirically, we show on MINIGRID-KEYCORRIDOR, VISUALTAXI, and MONTEZUMASREVENGE that this abstraction outperforms flat-RL baselines and a state-reaching hierarchical baseline.

2 BACKGROUND AND RELATED WORK

We model the agent-environment interaction as a Markov Decision Process $M = (S, A, R, T, \gamma)$ (Puterman, 1994; Sutton & Barto, 2018). To aid in reward maximization, our agent will discover options $o = (\mathcal{I}_o, \pi_o, \beta_o) \in \mathcal{O}$ (Sutton et al., 1999), where $\mathcal{I}_o : S \to [0, 1]$ is the initiation function, $\pi_o : S \to A$ is the intra-option policy over primitive actions, and $\beta_o : S \to \{0, 1\}$ is the termination condition. Each option has a maximum horizon of H steps.

Subgoal Options. While options need not optimize any objective in general, for option *discovery* it is convenient to think of options as achieving subgoals—indeed, most option discovery algorithms can be viewed this way (Precup, 2001; Bagaria, 2025; Klissarov et al., 2025). A subgoal here is not a state, but is instead a classifier that grounds to a potentially large set of states. A *subgoal option*

[1] We do not assume that state features are given—instead learning subgoal achievement classifiers directly from images—but the language of "features" helps present our ideas simply.

uses this classifier to serve three roles: it defines the option's subgoal, its termination condition $\beta_o : S \to \{0, 1\}$, and provides a binary pseudo-reward $R_o(s) = \beta_o(s)$ for training π_o. The initiation function $\mathcal{I}_o$ identifies states from which the option has a high probability of achieving its subgoal. This formulation reduces option discovery to identifying useful subgoals: once β_o is specified, standard policy optimization can train π_o, and policy evaluation can train $\mathcal{I}_o$.

Universal Value Function Approximators (UVFAs) (Schaul et al., 2015) are a scalable way to learn goal-conditioned value functions $V_g : S \to \mathbb{R}$, $Q_g : S \times A \to \mathbb{R}$ and their corresponding goal-conditioned policies $\pi_g : S \to A$ (Kaelbling, 1993) using function approximation. We use UVFAs to parameterize option value functions and policies: instead of representing each option's policy with a separate function approximator (Sutton et al., 2011), a single function approximator can be shared across all options by conditioning it on each option's own subgoal (Bagaria et al., 2021a): $\pi_o(s) = \arg\max_a Q_g(s, a) = \arg\max_a Q_{\beta_o}(s, a)$.

Novelty-based exploration. In tabular RL, count-based exploration bonuses that decay as $1/\sqrt{N[s, a]}$ can guide efficient learning (Strehl & Littman, 2008), but do not scale to large state spaces. Pseudocounts (Bellemare et al., 2016) generalize counts by assigning similar values to similar states; Coin Flip Networks (CFN; Lobel et al., 2023) are a simple, state-of-the-art technique for estimating pseudocounts from images. Like other pseudocount methods, CFN learns a function f_ϕ^{int} that maps states to an approximate $1/\sqrt{N}$ bonus. We use CFN over alternatives like Random Network Distillation (RND; Burda et al., 2019b) because Lobel et al. (2023) showed it outperforms popular novelty estimators—especially in stochastic domains—on task reward, hyperparameter robustness, and exploration-bonus interpretability. Our framework, however, is agnostic to the choice of novelty estimator. Importantly, while novelty-driven agents demonstrate impressive performance (Kapturowski et al., 2022), they do not learn options, which are key for transfer (Taylor & Stone, 2009) and high-level planning (Konidaris et al., 2018; Sutton et al., 2024). We seek to bridge this gap by using novelty as the substrate for option discovery (Simsek & Barto, 2004).

Gradient-based feature attribution. When multiple features jointly produce an outcome, how should we attribute credit to each one? Gradient-based attribution methods (Ancona et al., 2017) estimate each feature's contribution directly from a model's gradients. One possible target for such methods is the *Shapley value* (Shapley, 1953) from cooperative game theory, which measures each feature's *marginal contribution*—how much the outcome changes when that feature is added—averaged over all possible orderings of features. Concretely, for some set function v over features $\{1, \ldots, k\}$, the Shapley value of feature i is:

$$\psi_i = \frac{1}{k!} \sum_{\rho} \Big[v\big(B_i^\rho \cup \{i\}\big) - v\big(B_i^\rho\big) \Big], \tag{1}$$

where ρ ranges over permutations of features and B_i^ρ is the set preceding i in ρ. Computing exact Shapley values requires evaluating v on exponentially many subsets, which is intractable for high-dimensional inputs such as images. So, we use the DeepSHAP algorithm (Lundberg & Lee, 2017; Shrikumar et al., 2017), which efficiently approximates Shapley values for neural networks by backpropagating attributions from the output to per-pixel contributions.

Option discovery methods. Option discovery techniques can be broadly categorized by their learning signal. Option-Critic (Bacon et al., 2017) and feudal approaches (Dayan & Hinton, 1993; Vezhnevets et al., 2017) learn options using extrinsic reward, but struggle in sparse-reward settings. Empowerment-based methods (Eysenbach et al., 2019; Gregor et al., 2016) learn diverse skills that increase the agent's control over its environment, showing promise in exploration (Hansen et al., 2021; Campos Camúñez et al., 2020). Spectral methods use the graph Laplacian to find principal directions of the state-space for exploration (Machado et al., 2017; Jinnai et al., 2019; Klissarov & Machado, 2023). Some other algorithms also combine goal-conditioned RL (GCRL) and novelty (Pitis et al., 2020; Pong et al., 2019; Simsek & Barto, 2004; Bagaria et al., 2025). While these approaches differ in their objectives, they share a common limitation: most skills resort to state-reaching because they attend to all features. Our work is orthogonal—we focus on *how* subgoals are represented, showing that identifying relevant features enables broader generalization. For a detailed survey of option discovery, we refer readers to Klissarov et al. (2025).

Learning Subgoal Achievement Classifiers. Most option discovery algorithms either require manually defined subgoal functions or resort to state-reaching, where the benefits of abstraction fade (Colas et al., 2022). Go-Explore (Ecoffet et al., 2021) uses hand-engineered cell representations, and many GCRL methods (Eysenbach et al., 2019; Nachum et al., 2018; Andrychowicz et al., 2017) use domain knowledge to extract a small, relevant set of variables—like the (x, y) location of the agent in navigation tasks—-for defining subgoals. Without privileged information, this is a form of state-reaching and does not produce reusable temporal abstractions. Methods that *learn* classifiers typically still attend to all state features: perceptual similarity approaches threshold distances in observation space (Andrychowicz et al., 2017), use random projections (Dabney et al., 2021) or autoencoders (Tang et al., 2017); temporal distance methods predict whether states are within a fixed number of steps of a goal (Savinov et al., 2018; Mendonca et al., 2021); and mutual information methods (Warde-Farley et al., 2019) abstract away uncontrollable features but not irrelevant, controllable ones. None explicitly identify a small set of relevant features for each subgoal.

Options for feature attainment. The strategy of learning options that target specific features has been explored in HRL (Hengst, 2002). STOMP (Sutton et al., 2024) learns one option per feature, each option maximizing its feature. Proto-goal RL (Bagaria & Schaul, 2023) generalizes this to real-valued targets over combinations of features. Both assume a factored state representation is provided and treat every feature, or every combination, as a subgoal. Our approach starts from candidate features extracted from pixels and selects, for each subgoal, the few that account for a novelty spike. Our subgoal classifiers are also predicates in the sense of symbolic planning—boolean functions of a few state variables (Fikes & Nilsson, 1971; Ghallab et al., 2004). In that field, either the predicates (Yang et al., 2018; Lyu et al., 2019), or skills (Konidaris et al., 2018; Silver et al., 2023; Ahmetoglu et al., 2025), or both (Kaelbling & Lozano-Pérez, 2011; Garrett et al., 2021) are given. Our agent learns both via online exploration, extending this bridge between online RL and symbolic planning.

3 Method: Abstract Subgoal Discovery

Modern RL systems can efficiently compute proxies for intrinsic motivation, like novelty (Kapturowski et al., 2022). However, when used for option discovery, these signals typically drive state-reaching: the agent is rewarded for re-visiting states that previously yielded high novelty or reward, which fails to scale beyond modest navigational problems (Hansen et al., 2021). Instead, our agent measures the *interestingness* of a state using a learned intrinsic reward model f_ϕ^{int} that estimates novelty (Section 2). When the agent encounters a state s_N in which novelty spikes substantially relative to the least interesting states $\mathcal{S}_B$ from the same trajectory—states where nothing *interesting* or novel was happening— it asks: which features of s_N account for the expected spike $\Delta_n = \mathbb{E}_{s_B \sim \mathcal{S}_B}[f_\phi^{\text{int}}(s_N) - f_\phi^{\text{int}}(s_B)]$? Features whose modification substantially changes Δ_n are retained; the rest are discarded. A classifier is then constructed to attend only to these relevant features, serving as an abstract subgoal (illustration in Figure 6, Appendix B). The agent learns an option policy to achieve this subgoal, and the cycle repeats: the agent explores from promising subgoals, discovers new interesting states, and creates new options.

3.1 Option Selection and Exploration

Our agent maintains a set of options $\mathcal{O}$; each option in that set has a subgoal classifier β_o discovered during exploration. Every episode, the agent selects a subgoal to explore from, navigates there using the corresponding option policy, and then executes a novelty-driven exploration policy π_{int} (Ecoffet et al., 2021; Mendonca et al., 2021; Bagaria et al., 2025).

Policy over options. Not all subgoals are equally useful to pursue: some are unreachable from the current state, others lead to already-explored regions. The policy over options $\pi_{\mathcal{O}}(o \mid s)$ balances achievability, task performance, and exploration potential. Given that $V_{\beta_o}(s)$ can be interpreted as a discounted probability of reaching the option's subgoal β_o from state s (Bagaria et al.), the agent restricts option selection using the following initiation function:

$$\mathcal{I}_o(s) = \mathbb{I}\{V_{\beta_o}(s) > \delta\}, \tag{2}$$

where $\delta \in \mathbb{R}$ is a threshold hyperparameter. Among options for which $\mathcal{I}_o(s) = 1$, the agent scores each by balancing two terms: (1) the expected extrinsic reward $\hat{R}_o$ accumulated during option execution, and (2) the value of exploring from the subgoal:

$$U(o) = \alpha \hat{R}_o + \mathbb{E}_{s' \,:\, \beta_o(s')=1}\big[V_{\text{int}}(s')\big], \tag{3}$$

$$\text{where } V_{\text{int}}(s') = \mathbb{E}_{\pi_{\text{int}}}\left[\sum_t \gamma^t \big(r_t + \lambda f_\phi^{\text{int}}(s_t)\big)\right], \tag{4}$$

and $\alpha, \lambda \geq 0$ control the tradeoff between extrinsic and intrinsic rewards; we used $\lambda = 0.01$ in all our experiments. We implement the option reward model $\hat{R}_o$ using a simple average of extrinsic returns over past option executions. The policy is a bandit that samples proportionally to utility:

$$\pi_{\mathcal{O}}(o \mid s) = \frac{U(o)}{\sum_{o' : \mathcal{I}_{o'}(s)=1} U(o')}. \tag{5}$$

Conducting exploration. Once an option o is sampled from $\pi_{\mathcal{O}}$, the agent executes its intra-option policy π_o to reach the subgoal. Upon success, the agent executes the low-level exploration policy π_{int} that maximizes intrinsic and extrinsic reward, producing trajectory $\tau = (s_1, \ldots, s_K)$.

Triggering Option Discovery. After each exploration rollout, the agent checks whether the trajectory τ contains a sufficiently large spike in novelty. Concretely, it tests whether $\max_{s \in \tau} f_\phi^{\text{int}}(s) > \mu + \sigma_{\text{state}}\,\sigma$, where μ and σ are the running mean and standard deviation of f_ϕ^{int}, and σ_{state} is a threshold multiplier (1 in all our experiments; ablated in Appendix D). If the test passes, the agent extracts the most novel state, $s_N = \arg\max_{s \in \tau} f_\phi^{\text{int}}(s)$, encountered at timestep t_N. It then constructs a set of baseline states $\mathcal{S}_B = \{s_B^1, \ldots, s_B^M\}$. Each baseline is drawn (without replacement) from a window of w timesteps around t_N: the agent repeatedly selects the lowest-novelty state whose novelty falls at least σ below $f_\phi^{\text{int}}(s_N)$. Finally, s_N and $\mathcal{S}_B$ are passed to the feature extraction algorithm (Section 3.2).

3.2 Identifying Relevant Features

Given s_N and $\mathcal{S}_B$ from the previous section, the agent identifies which features of s_N explain the spike Δ_n defined as follows:

$$\Delta_n = \mathbb{E}_{s_B \sim \mathcal{S}_B}[f_\phi^{\text{int}}(s_N) - f_\phi^{\text{int}}(s_B)] \approx f_\phi^{\text{int}}(s_N) - \frac{1}{M}\sum_{j=1}^{M} f_\phi^{\text{int}}(s_B^j). \tag{6}$$

This is further decomposed into two steps: extracting candidate features from the image observation, and then measuring each feature's contribution to Δ_n.

Feature extraction. Neural representations learned end-to-end are entangled, making it difficult to intervene on individual factors (Rodriguez-Sanchez et al., 2025; Kumar et al., 2025). Our method relies on disentangled candidate features, and can leverage any extraction technique capable of producing them. We explore two off-the-shelf computer vision techniques: a lightweight contour detection algorithm (Suzuki & Abe, 1985), and a pretrained segmentation model (Kirillov et al., 2023). The output of this module is a set of candidate features $\{b_1, ..., b_k\}$ to be used for attribution.

Feature attribution. Given candidate features $\{b_1, ..., b_k\}$, the agent must determine which ones explain the novelty difference. The core question is: how much does each feature b_i contribute to Δ_n? We consider two techniques. *Counterfactual substitution* (see Figure 2) uses a single baseline $\mathcal{S}_B = \{s_B\}$ and constructs a counterfactual state c_i by replacing feature b_i in s_N with the corresponding feature from s_B. If the resulting novelty drop $\Delta_i = f_\phi^{\text{int}}(s_N) - f_\phi^{\text{int}}(c_i)$ exceeds a threshold ϵ, then b_i is relevant. This is effective when features contribute to Δ_n roughly independently. *Gradient-based attribution methods* (Ancona et al., 2017) estimate each feature's contribution directly from f_ϕ^{int}'s gradients, without requiring explicit counterfactual generation. These methods return an attribution, a_p, for each pixel p. We convert this to a feature-level attribution by averaging over the pixels in b_i: $A_i = \frac{1}{|b_i|}\sum_{p \in b_i} a_p$. A feature b_i is relevant if $A_i > \mu_A + \sigma_{\text{patch}}\,\sigma_A$, where μ_A

and σ_A are the mean and standard deviation of $\{A_1, \ldots, A_k\}$ over all candidate features and σ_{patch} is a threshold multiplier (1 in all our experiments; ablated in Appendix D). Many algorithms can do this; we use DeepSHAP (Lundberg & Lee, 2017), with implementation details in Section 4.2.

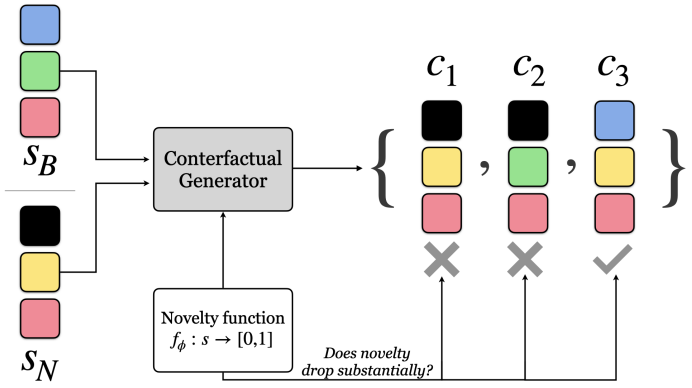


Figure 2: **Illustration of counterfactual substitution.** The most boring and most novel states, s_B and s_N respectively, are input into the Counterfactual Generator, which outputs counterfactual states c_1, c_2, c_3—each with one feature reset to its value in s_B. Only resetting the first feature (black $\to$ blue) substantially lowers novelty, indicating that the first feature accounts for most of the change in novelty Δ_n. This one-at-a-time reset is effective when features contribute independently.

This yields a subset of features $\mathcal{F} \subseteq \{b_1, \ldots, b_k\}$ that explain the jump in novelty from $\mathcal{S}_B$ to s_N. These features define the scope of the subgoal classifier constructed in the next section.

3.3 Subgoal Classifier Construction

Most of s_N is irrelevant to why it was novel. The feature selection step identified which parts matter; everything else can vary freely. The subgoal classifier respects this—it checks only whether the relevant features $\mathcal{F}$ match their values in s_N, and ignores the rest.

The agent constructs a non-parametric classifier $g : S \to \{0, 1\}$ since s_N is the only positive example. The classifier stores s_N and a feature extractor h that returns only the relevant features—selecting relevant indices for factored states, or cropping bounding boxes for images. For state s:

$$g(s) = \mathbb{I}\Big\{\mathcal{D}\big(h(s_N), h(s)\big) < \xi\Big\}, \tag{7}$$

where ξ is a threshold hyperparameter and $\mathcal{D}$ measures dissimilarity (e.g., Euclidean distance for factored states, or bounding-box similarity for images). The single-positive-example setting is inherent to the problem—when novelty first spikes, s_N is all that is available—and the feature selection step is precisely what enables the classifier to generalize from it (Section 4). The classifier g serves as the new option o's termination condition β_o. This new option o is added to $\mathcal{O}$, making it available for future use by the policy over options $\pi_{\mathcal{O}}$ (Section 3.1).

3.4 Putting It Together

Algorithm 1 summarizes the full option discovery loop. The agent alternates between exploiting existing options to reach promising subgoals and exploring from there to discover new ones. When exploration yields a state with sufficient spike in novelty, the agent identifies the relevant features, constructs a new subgoal classifier, and adds the corresponding option to its repertoire.

4 Experiments

Our experiments address three questions: (1) Does our feature attribution method correctly identify the features responsible for novelty? (2) Do the resulting subgoal classifiers generalize across states that differ in irrelevant features? (3) Does this abstraction lead to better exploration and task performance compared to flat RL and state-reaching HRL?

Algorithm 1 Abstract Subgoal Option Discovery

Require: Algorithm $\mathcal{A}_1$ to estimate novelty $f_\phi^{\text{int}} : S \to \mathbb{R}^+$.
Algorithm $\mathcal{A}_2$ to learn intrinsic policy π_{int} and value function V_{int}.
Algorithm $\mathcal{A}_3$ to learn intra-option policy π_o and its value function V_{β_o}.
Algorithm $\mathcal{A}_4$ to extract features from an image (e.g., contour detection).
Algorithm $\mathcal{A}_5$ to attribute credit to features (e.g., Shapley values).

Initialize:
Initialize option set $\mathcal{O}$ with a default exploration option.
while True **do**
  Sample option $o \sim \pi_{\mathcal{O}}(o \mid s)$ using Equation 5.
  Execute π_o until $\beta_o(s) = 1$ or timeout after H steps.
  if timeout **then**
    continue {Resample option}
  end if
  if agent reached subgoal **then**
    Roll out π_{int} to get trajectory τ.
  end if
  if $\max_{s \in \tau} f_\phi^{\text{int}}(s) > \mu + \sigma_{\text{state}}\, \sigma$ **then**
    Extract $s_N = \arg\max_{s \in \tau} f_\phi^{\text{int}}(s)$.
    Construct baseline set $\mathcal{S}_B = \{s_B^1, \ldots, s_B^M\}$ (Section 3.1).
    Extract candidate features $\{b_1, \ldots, b_k\}$ from s_N using algorithm $\mathcal{A}_4$.
    Select relevant features $\mathcal{F}$ using algorithm $\mathcal{A}_5$.
    Construct classifier g from $\mathcal{F}$ (Section 3.3).
    Add new option $o' = (\mathcal{I}_{o'}, \pi_{o'}, \beta_{o'} = g)$ to $\mathcal{O}$.
  end if
  Update mean μ and standard deviation σ of intrinsic rewards.
  Update novelty estimator f_ϕ^{int} using algorithm $\mathcal{A}_1$.
  Update V_{β_o} and π_o using reward $R_o(s) = \beta_o(s)$ and algorithm $\mathcal{A}_3$.
  Update V_{int} and π_{int} using reward $r_t + \lambda f_\phi^{\text{int}}(s_t)$ and algorithm $\mathcal{A}_2$.
end while

4.1 Experimental Design

Domains. We evaluate on three sparse-reward, image-based domains where the agent must explore effectively. MiniGrid-KeyCorridor-S5R3 (Chevalier-Boisvert et al., 2023) requires the agent to navigate through rooms, pick up a key, and unlock a door. Colas et al. (2022) identified it as a challenging exploration problem due to its sparse rewards and combinatorial state space. VisualTaxi (Dietterich, 2000) is a 10×10 grid-world, in which the agent must navigate to a passenger, pick them up, and deliver them to one of 8 possible destinations. Variants of this problem have been widely used to evaluate HRL algorithms (Dietterich, 2000; Allen et al., 2021). In both domains, non-zero reward is only observed upon completing the entire task, with no intermediate rewards for picking up the key or passenger. Random exploration is unlikely to ever see non-zero rewards in these domains, making them good testbeds for directed exploration strategies, like ours. MontezumasRevenge (Bellemare et al., 2013) is a canonical exploration benchmark with a natural hierarchical structure, and we seek to build option discovery algorithms that can autonomously find this hierarchical structure from experience.

4.2 Implementation details

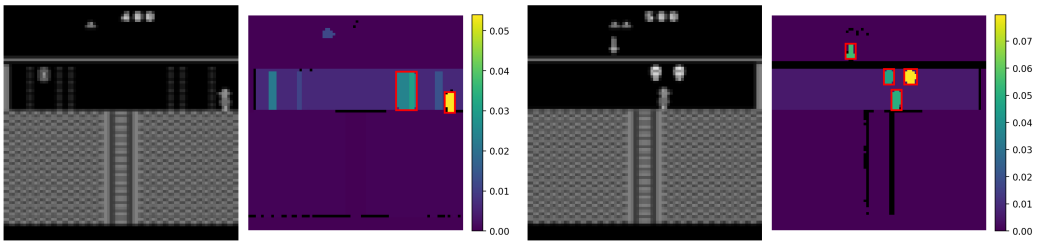
Learning policies and value functions. The classifier g constructed in Section 3.3 serves as the option's termination condition β_o and provides a binary terminating pseudo-reward $R_o(s) = g(s)$. We train the intra-option policy π_o using goal-conditioned R2D2 (Kapturowski et al., 2019) with hindsight experience replay (HER; Andrychowicz et al., 2017), a well-established combination for sparse-reward goal-reaching. Architecturally, we augment the R2D2 CNN encoder with a goal-processing torso; state and goal embeddings are concatenated and passed through an MLP that produces Q-values over actions. The subgoal input to the CNN encoder includes only s_N's ex-

tracted features, and 0 elsewhere. Every episode, the agent uses hindsight relabeling with five subgoals achieved during the trajectory and adds the corresponding sub-trajectories to the replay buffer (Andrychowicz et al., 2017). The initiation function $\mathcal{I}_o(s)$ is derived from the option value function V_{β_o}, as described in Section 3.1. We use CFN (Lobel et al., 2023) for learning f_ϕ^{int} over similar methods (Burda et al., 2019a) because it provides (1) a more stable range of intrinsic rewards, which stabilizes option learning, and (2) more concentrated Shapley attributions (Appendix D.3).

Feature generation and selection. MINIGRID and VISUALTAXI have simple image observations with static backgrounds, so we use contour detection (Suzuki & Abe, 1985) for feature extraction and counterfactual substitution for feature attribution. This provides a low memory and computation version of the algorithm. MONTEZUMASREVENGE is visually more complex—contour detection's static background assumption and counterfactual substitution's independence assumption both break—so, we use a pre-trained Segment Anything Model (SAM, Kirillov et al., 2023) for feature extraction and DeepSHAP for feature attribution (Lundberg & Lee, 2017; Shrikumar et al., 2017). All three domains use the same online training procedure described in Algorithm 1, with subgoal discovery and option learning occurring jointly. We describe additional MONTEZUMASREVENGE-specific implementation details in Appendix C.

4.3 QUALITATIVE RESULTS

Feature attribution. Figure 3 shows two examples of DeepSHAP attributions on discovered subgoals in MONTEZUMASREVENGE. The agent enters a new room for the first time, triggering a novelty spike; the intrinsic-model attribution correctly highlights aspects that uniquely identify this room, while ignoring other aspects like the number of lives remaining or the score. Another shows the agent approaching two skulls while equipped with the sword, a state that precedes extrinsic reward. The method isolates meaningful features rather than attending to the entire observation.



(a) Agent enters room 0.

(b) Agent prepares to kill skull with sword.

Figure 3: **Examples of identifying relevant features in MONTEZUMASREVENGE.** For each state (black and white image), we show the estimated Shapley value attributed to each feature in the image and the bounding boxes selected for the subgoal classifier. The first classifier checks if the player has entered room 0, while the second classifier checks if the player has the sword near the skulls.

Classifier transfer. Figure 4 demonstrates that classifiers built from relevant features transfer across states. The leftmost image shows the state in which a classifier was learned; the bounding box indicates that it attends only to the player's position in a specific screen region. The remaining images show states where this classifier fires later in the agent's lifetime: despite differences in room layout, score, number of lives, and enemy positions, the classifier triggers whenever the player appears in the target region. Appendix E shows examples from MINIGRID, including subgoals that capture the relationship between different objects. Quantitatively, in MONTEZUMASREVENGE, we found that a classifier triggered on a median of 71.4 ± 11 states. Some classifiers are highly selective (e.g., picking up a key while having exactly one life), while others are broadly applicable (e.g., the player appearing in a screen region).

4.4 QUANTITATIVE RESULTS

We compare our method (Abstract Subgoals) against three baselines: R2D2 (Kapturowski et al., 2019), a flat RL method with ϵ-greedy exploration; CFN (Lobel et al., 2023), a flat RL method with

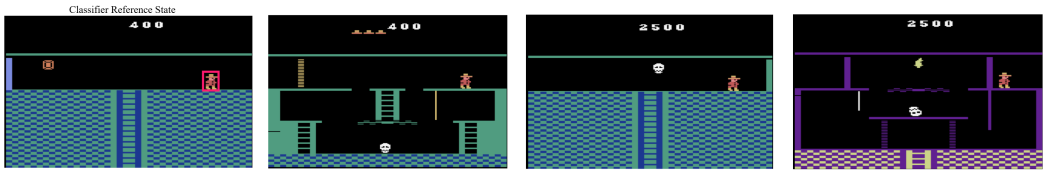


Figure 4: **Example of a discovered subgoal classifier that transfers to many different rooms.** The leftmost image shows the context in which this subgoal classifier was learned—the bounding box shows that the classifier only attends to the player's position. The remaining three images show some states in which that classifier was triggered—as long as the player re-appears in that portion of the screen, other factors are all irrelevant for achieving that option's subgoal.

novelty-based exploration; and Pixel Equality, which uses the same online subgoal discovery and training procedure as Abstract Subgoals but replaces the classifier's dissimilarity test $\mathcal{D}$ (Section 3.3) with $\sum_p (s_N[p] - s[p])^2 < 0.01$, summed over every pixel p of the full image rather than only the selected features $\mathcal{F}$. Figure 5 shows learning curves across all three domains.

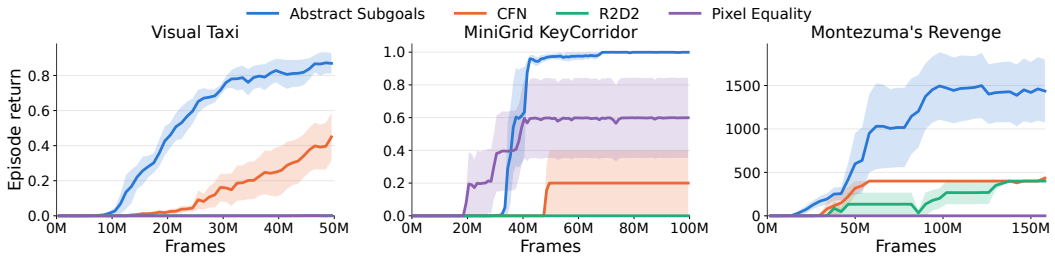


Figure 5: **Learning curves** comparing our agent (Abstract Subgoals) with a non-hierarchical novelty maximizing RL method (CFN), a vanilla RL method (R2D2), and a state-reaching HRL baseline (Pixel Equality). Solid lines denote average undiscounted return; shaded regions denote standard deviation over 5 random seeds.

Our method, Abstract Subgoals, achieves the best final performance in all three domains (Figure 5). Among the baselines, CFN outperforms vanilla R2D2, isolating the importance of exploration in these problems. Pixel Equality improves upon CFN in MiniGrid-KeyCorridor, but underperforms in the other two tasks where it is unable to learn transferrable option policies. Our method's improvement over CFN points to the importance of hierarchies, while its improvement over Pixel Equality additionally isolates the importance of feature selection. Appendix D.2 further isolates the improvement over Pixel Equality.

5 Discussion and Conclusion

We presented an algorithm that discovers abstract subgoal options by identifying which features of a state account for its novelty, and constructing classifiers that attend only to those features. Our experiments show that this abstraction enables more efficient exploration and better task performance than both flat RL and a state-reaching HRL baseline.

Limitations and future work. Our method relies on external feature extraction (contour detection, SAM), which is biased toward object-centric features; end-to-end disentangled representation learning (Rodriguez-Sanchez et al., 2025; Kumar et al., 2025) could serve as a drop-in replacement. Not all discovered subgoals are useful: some attend to uncontrollable features like moving enemies, which off-policy controllability estimation (Bagaria & Schaul, 2023) could filter. Our bandit-based policy over options could be replaced with abstract model-based planning (Konidaris et al., 2018).

Our key insight is simple—when something interesting happens, figure out *why* it was interesting, not just *that* it was interesting. By grounding option discovery in feature attribution, we take a step toward agents that build abstract, reusable skills from experience.

AI USE STATEMENT

In this work, we used generative AI tools for assistance with code edits that added to the algorithm implementation, and assisted in creating visualization plots for the figures. We have not used generative AI tools to help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, or clean and reformat datasets or interpret results. Generating synthetic data sets, assisting with translation, and supporting qualitative and thematic data analysis are not applicable to this work. Additionally, we used generative AI tools to edit the manuscript for readability. We have reviewed all AI-assisted work. LLM-generated code additions were tested and verified for correctness by the authors, and AI-edited manuscript text was checked against the underlying results and figures for accuracy. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Section 3.4 and Algorithm 1 provide complete pseudocode for our option discovery method, Section 4.2 details all specific models and algorithms used for each domain, and Appendix F reports the hyperparameters used to produce every result in the main text, broken down by domain and method (Tables 2–6). MONTEZUMASREVENGE-specific implementation details are further described in Appendix C.

REFERENCES

Alper Ahmetoglu, Steven James, Cameron Allen, Sam Lobel, David Abel, and George Konidaris. Skill-Driven Neurosymbolic State Abstractions. In *Advances in Neural Information Processing Systems 38*, pp. 10750–10785, 2025.

Cameron Allen, Neev Parikh, Omer Gottesman, and George Konidaris. Learning Markov State Abstractions for Deep Reinforcement Learning. *Advances in Neural Information Processing Systems*, 34:8229–8241, 2021.

Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. *arXiv preprint arXiv:1711.06104*, 2017.

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.

Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Akhil Bagaria. *Skill Discovery for Exploration and Planning*. Doctor of philosophy, Brown University, 2025.

Akhil Bagaria and Tom Schaul. Scaling Goal-based Exploration via Pruning Proto-goals. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*, pp. 3451–3460, 2023.

Akhil Bagaria, Ben M Abbatematteo, Omer Gottesman, Matt Corsaro, Sreehari Rammohan, and George Konidaris. Effectively Learning Initiation Sets in Hierarchical Reinforcement Learning. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Akhil Bagaria, Jason Senthil, Matthew Slivinski, and George Konidaris. Robustly Learning Composable Options in Deep Reinforcement Learning. In *30th International Joint Conference on Artificial Intelligence*, 2021a.

Akhil Bagaria, Jason K. Senthil, and George Konidaris. Skill Discovery for Exploration and Planning using Deep Skill Graphs. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 521–531, 2021b.

Akhil Bagaria, Anita De Mello Koch, Rafael Rodriguez-Sanchez, Sam Lobel, and George Konidaris. Intrinsically Motivated Discovery of Temporally Abstract Graph-based Models of the World. In *Reinforcement Learning Journal*, 2025.

Andrew G Barto and Sridhar Mahadevan. Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems*, 13(1-2):41–77, 2003.

Andrew G Barto, Satinder Singh, and Nuttapong Chentanez. Intrinsically motivated learning of hierarchical collections of skills. In *Proceedings of the 3rd International Conference on Development and Learning*, pp. 112–19, 2004.

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, pp. 1471–1479, 2016.

Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47: 253–279, 2013.

Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A. Efros. Large-Scale Study of Curiosity-Driven Learning. In *International Conference on Learning Representations*, 2019a.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *International Conference on Learning Representations*, 2019b.

Víctor Campos Camúñez, Alex Trott, Caiming Xiong, Richard Socher, Xavier Giró Nieto, and Jordi Torres Viñals. Explore, discover and learn: unsupervised discovery of state-covering skills. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 1317–1327, 2020.

Nuttapong Chentanez, Andrew G Barto, and Satinder P Singh. Intrinsically motivated reinforcement learning. In *Advances in neural information processing systems*, pp. 1281–1288, 2005.

Maxime Chevalier-Boisvert, Bolun Dai, Mark Towers, Rodrigo de Lazcano, Lucas Willems, Salem Lahlou, Suman Pal, Pablo Samuel Castro, and Jordan Terry. Minigrid & Miniworld: Modular & Customizable Reinforcement Learning Environments for Goal-Oriented Tasks. *CoRR*, abs/2306.13831, 2023.

Cédric Colas, Tristan Karch, Olivier Sigaud, and Pierre-Yves Oudeyer. Autotelic agents with intrinsically motivated goal-conditioned reinforcement learning: a short survey. *Journal of Artificial Intelligence Research*, 74:1159–1199, 2022.

Will Dabney, André Barreto, Mark Rowland, Robert Dadashi, John Quan, Marc G. Bellemare, and David Silver. The Value-Improvement Path: Towards Better Representations for Reinforcement learning. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI*, pp. 7160–7168. AAAI Press, 2021.

Peter Dayan and Geoffrey E Hinton. Feudal reinforcement learning. In *Advances in neural information processing systems*, pp. 271–278, 1993.

Thomas G Dietterich. Hierarchical reinforcement learning with the MAXQ value function decomposition. *Journal of Artificial Intelligence Research*, 13:227–303, 2000.

Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O Stanley, and Jeff Clune. First return, then explore. *Nature*, 590(7847):580–586, 2021.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is All You Need: Learning Skills without a Reward Function. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=SJx63jRqFm.

Jesse Farebrother, Joshua Greaves, Rishabh Agarwal, Charline Le Lan, Ross Goroshin, Pablo Samuel Castro, and Marc G Bellemare. Proto-Value Networks: Scaling Representation Learning with Auxiliary Tasks. In *The Eleventh International Conference on Learning Representations* , 2023.

Richard Fikes and Nils J. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. *Artif. Intell.*, 2(3/4):189–208, 1971. doi: 10.1016/0004-3702(71)90010-5. URL https://doi.org/10.1016/0004-3702(71)90010-5.

Caelan Reed Garrett, Rohan Chitnis, Rachel M. Holladay, Beomjoon Kim, Tom Silver, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Integrated task and motion planning. *Annu. Rev. Control. Robotics Auton. Syst.*, 4:265–293, 2021. doi: 10.1146/ANNUREV-CONTROL-091420-084139. URL https://doi.org/10.1146/annurev-control-091420-084139.

Samuel J Gershman. On the blessing of abstraction. *SAGE Publications Sage UK: London, England*, 2017.

Malik Ghallab, Dana S. Nau, and Paolo Traverso. *Automated planning - theory and practice*. Elsevier, 2004. ISBN 978-1-55860-856-6.

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

Danijar Hafner, Kuang-Huei Lee, Ian Fischer, and Pieter Abbeel. Deep Hierarchical Planning from Pixels. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

Steven Hansen, Guillaume Desjardins, Kate Baumli, David Warde-Farley, Nicolas Heess, Simon Osindero, and Volodymyr Mnih. Entropic desired dynamics for intrinsic control. *Advances in Neural Information Processing Systems*, 34:11436–11448, 2021.

Bernhard Hengst. Discovering Hierarchy in Reinforcement Learning with HEXQ. In Claude Sammut and Achim G. Hoffmann (eds.), *Machine Learning, Proceedings of the Nineteenth International Conference (ICML 2002), University of New South Wales, Sydney, Australia, July 8-12, 2002*, pp. 243–250, 2002.

Khurram Javed and Richard S. Sutton. The big world hypothesis and its ramifications for artificial intelligence. In *Finding the Frame: An RLC Workshop for Examining Conceptual Frameworks*, 2024. URL https://openreview.net/forum?id=Sv7DazuCn8.

Yuu Jinnai, Jee Won Park, David Abel, and George Konidaris. Discovering options for exploration by minimizing cover time. In *Proceedings of the 36th International Conference on Machine Learning*. PMLR, 2019. URL http://proceedings.mlr.press/v97/jinnai19b.html.

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*, 2020. URL https://openreview.net/forum?id=SkeIyaVtwB.

Leslie Pack Kaelbling. Learning to achieve goals. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1094–1099. Citeseer, 1993.

Leslie Pack Kaelbling and Tomás Lozano-Pérez. Hierarchical task and motion planning in the now. In *IEEE International Conference on Robotics and Automation, ICRA 2011, Shanghai, China, 9-13 May 2011*, pp. 1470–1477. IEEE, 2011. doi: 10.1109/ICRA.2011.5980391. URL https://doi.org/10.1109/ICRA.2011.5980391.

Steven Kapturowski, Georg Ostrovski, Will Dabney, John Quan, and Remi Munos. Recurrent Experience Replay in Distributed Reinforcement Learning. In *International Conference on Learning Representations*, 2019.

Steven Kapturowski, Víctor Campos, Ray Jiang, Nemanja Rakićević, Hado van Hasselt, Charles Blundell, and Adrià Puigdomènech Badia. Human-level Atari 200x faster. *arXiv preprint arXiv:2209.07550*, 2022.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023.

Martin Klissarov and Marlos C Machado. Deep laplacian-based options for temporally-extended exploration. *arXiv preprint arXiv:2301.11181*, 2023.

Martin Klissarov, Akhil Bagaria, Ziyan Luo, George Dimitri Konidaris, Doina Precup, and Marlos C. Machado. Discovering temporal structure: An overview of hierarchical reinforcement learning. *ArXiv*, abs/2506.14045, 2025. URL https://api.semanticscholar.org/CorpusID:279410938.

George Konidaris. On the necessity of abstraction. *Current Opinion in Behavioral Sciences*, 29: 1–7, 2019.

George Konidaris, Leslie Pack Kaelbling, and Tomas Lozano-Perez. From skills to symbols: Learning symbolic representations for abstract high-level planning. *Journal of Artificial Intelligence Research*, 61:215–289, 2018.

Akarsh Kumar, Jeff Clune, Joel Lehman, and Kenneth O. Stanley. Questioning representational optimism in deep learning: The fractured entangled representation hypothesis. *ArXiv*, abs/2505.11581, 2025. URL https://api.semanticscholar.org/CorpusID:278740993.

Sam Lobel, Akhil Bagaria, and George Konidaris. Flipping coins to estimate pseudocounts for exploration in reinforcement learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 22594–22613. PMLR, 23–29 Jul 2023.

Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Neural Information Processing Systems*, 2017. URL https://api.semanticscholar.org/CorpusID:21889700.

Daoming Lyu, Fangkai Yang, Bo Liu, and Steven Gustafson. SDRL: interpretable and data-efficient deep reinforcement learning leveraging symbolic planning. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pp. 2970–2977. AAAI Press, 2019. doi: 10.1609/AAAI.V33I01.33012970. URL https://doi.org/10.1609/aaai.v33i01.33012970.

Marlos C. Machado, Marc G. Bellemare, and Michael Bowling. A laplacian framework for option discovery in reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, pp. 2295–2304, 2017.

Russell Mendonca, Oleh Rybkin, Kostas Daniilidis, Danijar Hafner, and Deepak Pathak. Discovering and Achieving Goals via World Models. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 24379–24391, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/cc4af25fa9d2d5c953496579b75f6f6c-Abstract.html.

Ofir Nachum, Shixiang Shane Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 3303–3313, 2018.

Pierre-Yves Oudeyer, Frederic Kaplan, et al. How can we define intrinsic motivation. In *Proc. of the 8th Conf. on Epigenetic Robotics*, volume 5, pp. 29–31, 2008.

Silviu Pitis, Harris Chan, Stephen Zhao, Bradly C. Stadie, and Jimmy Ba. Maximum Entropy Gain Exploration for Long Horizon Multi-goal Reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 7750–7761. PMLR, 2020. URL http://proceedings.mlr.press/v119/pitis20a.html.

Vitchyr H Pong, Murtaza Dalal, Steven Lin, Ashvin Nair, Shikhar Bahl, and Sergey Levine. Skew-fit: State-covering self-supervised reinforcement learning. *Proceedings of the 37th International Conference on Machine Learning, ICML*, 2019.

Doina Precup. *Temporal abstraction in reinforcement learning.* PhD thesis, University of Massachusetts at Amherst, 2001.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley Series in Probability and Statistics. Wiley, 1994.

Rafael Rodriguez-Sanchez, Cameron S. Allen, and George Dimitri Konidaris. From pixels to factors: Learning independently controllable state variables for reinforcement learning. *ArXiv*, abs/2510.02484, 2025. URL https://api.semanticscholar.org/CorpusID:281829527.

Nikolay Savinov, Alexey Dosovitskiy, and Vladlen Koltun. Semi-parametric topological memory for navigation. In *International Conference on Learning Representations*, 2018.

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

Lloyd S Shapley. A value for n-person games. In Harold W. Kuhn and Albert W. Tucker (eds.), *Contributions to the Theory of Games II*, pp. 307–317. Princeton University Press, Princeton, 1953.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International conference on machine learning*, pp. 3145–3153. PMlR, 2017.

Tom Silver, Rohan Chitnis, Nishanth Kumar, Willie McClinton, Tomás Lozano-Pérez, Leslie Pack Kaelbling, and Joshua B. Tenenbaum. Predicate invention for bilevel planning. In Brian Williams, Yiling Chen, and Jennifer Neville (eds.), *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pp. 12120–12129. AAAI Press, 2023. doi: 10.1609/AAAI.V37I10.26429. URL https://doi.org/10.1609/aaai.v37i10.26429.

Özgür Simsek and Andrew G. Barto. Using relative novelty to identify useful temporal abstractions in reinforcement learning. In Carla E. Brodley (ed.), *Machine Learning, Proceedings of the Twenty-first International Conference (ICML 2004), Banff, Alberta, Canada, July 4-8, 2004*, volume 69 of *ACM International Conference Proceeding Series*. ACM, 2004. doi: 10.1145/1015330.1015353. URL https://doi.org/10.1145/1015330.1015353.

Alexander L Strehl and Michael L Littman. An analysis of model-based interval estimation for Markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, 2008.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Richard S Sutton, Joseph Modayil, Michael Delp, Thomas Degris, Patrick M Pilarski, Adam White, and Doina Precup. Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pp. 761–768. International Foundation for Autonomous Agents and Multiagent Systems, 2011.

Richard S Sutton, Michael Bowling, and Patrick M Pilarski. The Alberta plan for AI research. *arXiv preprint arXiv:2208.11173*, 2022.

Richard S. Sutton, Marlos C. Machado, G. Zacharias Holland, David Szepesvari, Finbarr Timbers, Brian Tanner, and Adam White. Reward-Respecting Subtasks for Model-Based Reinforcement Learning (abstract reprint). In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI*, pp. 22713. AAAI Press, 2024.

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

Satoshi Suzuki and Keiichi Abe. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32–46, 1985. ISSN 0734-189X. doi: https://doi.org/10.1016/0734-189X(85)90016-7. URL https://www.sciencedirect.com/science/article/pii/0734189X85900167.

Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, OpenAI Xi Chen, Yan Duan, John Schulman, Filip DeTurck, and Pieter Abbeel. # Exploration: A study of count-based exploration for deep reinforcement learning. In *Advances in neural information processing systems*, pp. 2753–2762, 2017.

Matthew E Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(7), 2009.

Vivek Veeriah, Junhyuk Oh, and Satinder Singh. Many-goals reinforcement learning. *arXiv preprint arXiv:1806.09605*, 2018.

Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, and Koray Kavukcuoglu. Feudal networks for hierarchical reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3540–3549, 2017.

David Warde-Farley, Tom Van de Wiele, Tejas D. Kulkarni, Catalin Ionescu, Steven Hansen, and Volodymyr Mnih. Unsupervised Control Through Non-Parametric Discriminative Rewards. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=r1eVMnA9K7.

Robert W White. Motivation reconsidered: the concept of competence. *Psychological review*, 66 (5):297, 1959.

Fangkai Yang, Daoming Lyu, Bo Liu, and Steven Gustafson. PEORL: integrating symbolic planning and hierarchical reinforcement learning for robust decision-making. In Jérôme Lang (ed.), *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pp. 4860–4866. ijcai.org, 2018. doi: 10.24963/IJCAI.2018/675. URL https://doi.org/10.24963/ijcai.2018/675.

A NOTATION

Table 1 summarizes the notation used throughout the paper.

Table 1: Summary of notation.

Symbol	Meaning	Introduced
MDP and options		
$M = (S, A, R, T, \gamma)$	Markov decision process	Sec. 2
$o = (\mathcal{I}_o, \pi_o, \beta_o)$	Option: initiation function, policy, termination condition	Sec. 2
$\mathcal{O}$	The agent's set of discovered options	Sec. 2
H	Option timeout (maximum steps per execution)	Sec. 2
$R_o(s) = \beta_o(s)$	Option pseudo-reward	Sec. 2
V_{β_o}, Q_{β_o}	Option (goal-conditioned) value functions	Sec. 2
Exploration and option selection		
f_ϕ^{int}	Learned novelty estimator with parameters ϕ	Sec. 2
$\pi_{\text{int}}, V_{\text{int}}$	Novelty-driven exploration policy and its value function	Sec. 3.1
$\pi_{\mathcal{O}}(o \mid s)$	Policy over options	Sec. 3.1
$U(o)$	Utility of option o under $\pi_{\mathcal{O}}$	Eq. 3
δ	Initiation threshold on $V_{\beta_o}(s)$	Eq. 2
α, λ	Extrinsic / intrinsic reward coefficients	Eq. 3
$\tau = (s_1, \ldots, s_K)$	Exploration trajectory	Sec. 3.1
μ, σ	Running mean and std. of f_ϕ^{int}	Sec. 3.1
σ_{state}	Subgoal-creation threshold multiplier	Sec. 3.1
Feature attribution and subgoal classifiers		
s_N	Most novel state in τ (single positive example)	Sec. 3.1
$\mathcal{S}_B = \{s_B^1, \ldots, s_B^M\}$	Set of M baseline (boring) states	Sec. 3.1
w	Baseline sampling window around t_N	Sec. 3.1
Δ_n	Expected novelty spike of s_N over $\mathcal{S}_B$	Eq. 6
$\{b_1, \ldots, b_k\}$	Candidate features extracted from s_N	Sec. 3.2
c_i	Counterfactual state (b_i reset to its value in s_B)	Sec. 3.2
Δ_i	Novelty drop from counterfactual c_i	Sec. 3.2
ϵ	Counterfactual relevance threshold	Sec. 3.2
ψ_i	Shapley value of feature i	Sec. 2
a_p	Per-pixel attribution returned by gradient-based methods	Sec. 3.2
A_i	Feature-level attribution (mean of a_p over pixels in b_i)	Sec. 3.2
μ_A, σ_A	Mean and std. of $\{A_1, \ldots, A_k\}$	Sec. 3.2
σ_{patch}	DeepSHAP attribution threshold multiplier	Sec. 3.2
$\mathcal{F}$	Selected subset of relevant features	Sec. 3.2
g	Subgoal classifier (becomes β_o of the new option)	Sec. 3.3
h	Feature extractor used by g	Sec. 3.3
$\mathcal{D}, \xi$	Dissimilarity measure and match threshold of g	Sec. 3.3
κ	Classifier trigger window for subgoal filtering	App. C

B Illustration of Core Insight

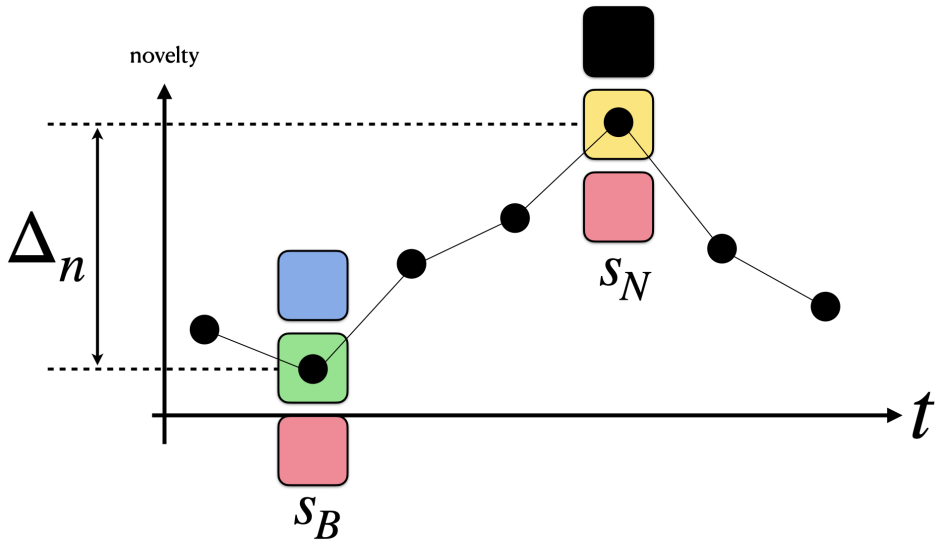


Figure 6: **Illustration of the core insight.** Suppose an agent observes a state trajectory of length 7 and a novelty estimator assigns each state s_t in that trajectory a novelty score $f_\phi^{\text{int}}(s_t)$. s_N denotes the most novel state, and s_B denotes the most boring state in that trajectory. Suppose that each state has 3 features; the colored boxes represent feature values for those two states. Rather than simply treating s_N as a target state, we seek to identify which features in particular were responsible for the large difference in novelty $\Delta_n = f_\phi^{\text{int}}(s_N) - f_\phi^{\text{int}}(s_B)$. Our algorithm can handle input images, but we show state features here for simplicity.

C MontezumasRevenge Implementation Details

Initiation function. In MontezumasRevenge, instead of using a strict threshold for the initiation function (Equation 2), we instead scale the utility function of the policy over options, $U(o)$ in Equation 3, by $V_{\beta_o}(s)$. This can be seen as a soft way of modulating the policy over options using the option's initiation function (Klissarov et al., 2025).

Observation normalization. Following Burda et al. (2019b), we whiten CFN's input observations using:

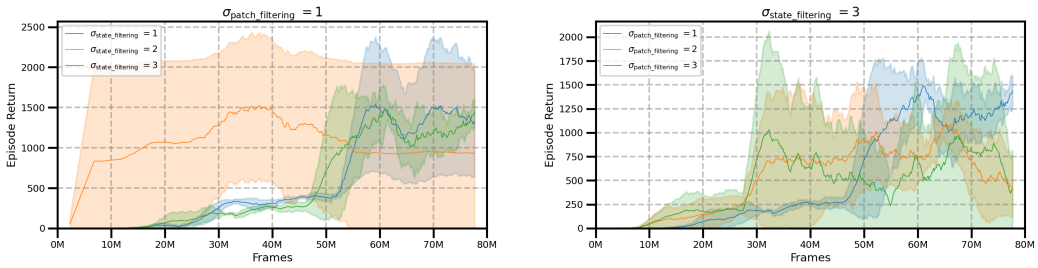
$$\hat{s} = \text{clip}\left(\frac{s - \mu_{\text{obs}}}{\sigma_{\text{obs}}}, -5, 5\right) \tag{8}$$

where μ_{obs} and σ_{obs} are the running mean and standard deviation of the observations. This normalization is applied only to CFN and does not affect the policy or value networks. This reduces the effect bright objects—such as a white skull—have during pixel attribution, which can bias the contribution of those features.

Window filtering. In larger, long-horizon domains, our method risks creating too many classifiers. We add a κ-window filter: for a novel state s_N encountered at t_N in τ, we discard s_N if any state within $(t_N - \kappa, t_N + \kappa)$ already triggers an existing classifier, avoiding redundant coverage. This filter's effectiveness depends on classifier generality: a narrowly applicable classifier rarely triggers near other novel states, so few candidate subgoals, if any, are filtered. It is possible to create a classifier that triggers for a large number of states—for example, a classifier that looks for a specific room or item in the inventory—which would prevent the creation of new subgoals. We prevent this by excluding classifiers that trigger on every state of the full window duration from consideration during filtering.

D ABLATIONS

D.1 HYPERPARAMETER SENSITIVITY



(a) Effect of σ_{state} at fixed $\sigma_{\text{patch}} = 1$.

(b) Effect of σ_{patch} at fixed $\sigma_{\text{state}} = 3$.

Figure 7: **Hyperparameter sensitivity sweep in MONTEZUMASREVENGE.** Each curve is the average undiscounted task return across 3 random seeds; shaded regions denote standard deviation.

We swept the two key threshold multipliers—σ_{state} (the multiplier in the subgoal-creation threshold of Algorithm 1, for the intrinsic signal) and σ_{patch} (the multiplier in the DeepSHAP attribution threshold of Section 3.2)—each over $\{1\sigma, 2\sigma, 3\sigma\}$ in MONTEZUMASREVENGE, with 3 random seeds per configuration. (Our main experiments use $\sigma_{\text{state}} = \sigma_{\text{patch}} = 1$.) Figure 7 shows that most configurations reach meaningful scores (1000+), confirming that the method is not overly sensitive to these choices. Lower values of σ_{patch} consistently perform better: more permissive thresholds yield classifiers that generalize more broadly, easing option policy learning. Results for σ_{state} are more mixed, with different values leading at different points during training, suggesting robustness rather than a single optimal setting.

D.2 DECOUPLING DISCOVERY FROM REACHABILITY

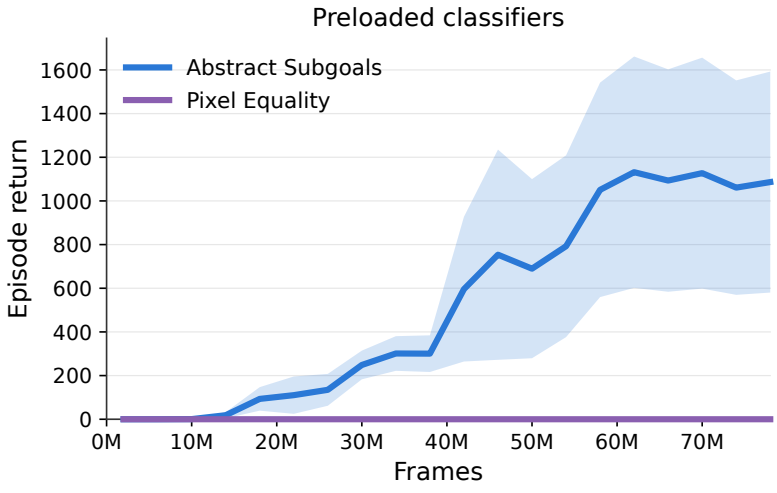


Figure 8: **Learning curves comparing Abstract Subgoals and Pixel Equality given subgoals from a separate discovery-only stage.** Both agents are trained on the same fixed set of novel trigger states for MONTEZUMASREVENGE, with subgoal discovery disabled; each method constructs its own classifier for these states, so the comparison isolates the effect of the classifier alone (averaged over 4 seeds).

In Algorithm 1, the agent only explores for new subgoals after reaching its currently sampled subgoal. This means a method that reaches its subgoals less reliably also explores less, and so may

simply discover fewer or worse subgoals as a result. The comparisons between Pixel Equality and Abstract Subgoals in Section 4 conflate two distinct effects: (1) how easily each method's classifiers can be reached by the option policy, and (2) which subgoals each method happens to discover.

To isolate effect (1), we run an additional experiment on MONTEZUMASREVENGE in which both methods are given an identical, fixed set of subgoals obtained from a separate discovery-only stage: an agent explores purely via RND novelty-driven exploration, with no option selection, producing a fixed set of novel trigger states. Each method then constructs its own classifier for these states using its own procedure—the SAM/DeepSHAP feature-attribution pipeline for Abstract Subgoals, full-image matching for Pixel Equality—so subgoal discovery does not depend on either method's ability to reach previous subgoals. Subgoal discovery is disabled for the remainder of the experiment, so both methods are trained only to reach the same fixed set of subgoals. Figure 8 shows that Pixel Equality still achieves an episode return of 0, while Abstract Subgoals finishes with a final return of $\approx$ 1,100, confirming the advantage of learning abstract subgoals even when both methods are evaluated independently of discovery. The reason for this is likely that the subgoal regions are too small for the non-abstract options to support effective policy learning.

D.3 SENSITIVITY TO NOVELTY ESTIMATOR

Our experiments use CFN to estimate novelty but our approach is agnostic to the choice of novelty estimator. To test whether this choice affects which features get selected, we run an exploration-only discovery stage, using Random Network Distillation (RND; Burda et al., 2019b) as the novelty estimator and compare the resulting subgoals from RND and CFN. Since each estimator independently determines its own most-novel states during discovery, RND and CFN are unlikely to agree on which state is most novel at a given point in exploration; we therefore compare the subgoals each estimator creates, rather than forcing a comparison on identical states, which would not reflect each estimator's own judgment of novelty.

Figure 9 shows that the two novelty estimators produce similar subgoal classifiers: the relevant features largely agree between CFN- and RND-based attribution. However, the attribution maps differ in how concentrated they are. CFN's attribution is close to 0 over most features for all shown states, with elevated attribution around a few features. RND shows a similar pattern for the starting room, room 1; however, for states lying outside this initial room, we see a majority of the features register as moderately novel (light blue), displaying a more diffused attribution map. Both novelty estimators point to the same relevant features on average, but CFN separates relevant from irrelevant features more cleanly, while RND spreads novelty attribution across more features.

D.4 GROWTH IN NUMBER OF DISCOVERED OPTIONS

Figure 10 tracks the option count in MONTEZUMASREVENGE. The option count follows a logarithmic trend—rapidly growing initially then continuously decelerating—starting to saturate around 200 options.

E DISCOVERED SUBGOALS IN MINIGRID

Figure 11 shows three subgoal classifiers discovered by our agent in MINIGRID-KEYCORRIDOR, each attending only to the patches inside its bounding boxes. Together they illustrate a range of abstraction levels: the left classifier combines the agent's location with the state of the yellow door; the middle one couples the agent's position with the state of the key (which must be in the hallway), capturing a relational subgoal across objects; and the right classifier fires whenever the agent enters the locked room with the ball, regardless of where the key is or whether the door is open. This last example illustrates the kind of generalization that state-reaching subgoals cannot represent—the same subgoal is satisfied across many combinations of irrelevant features.

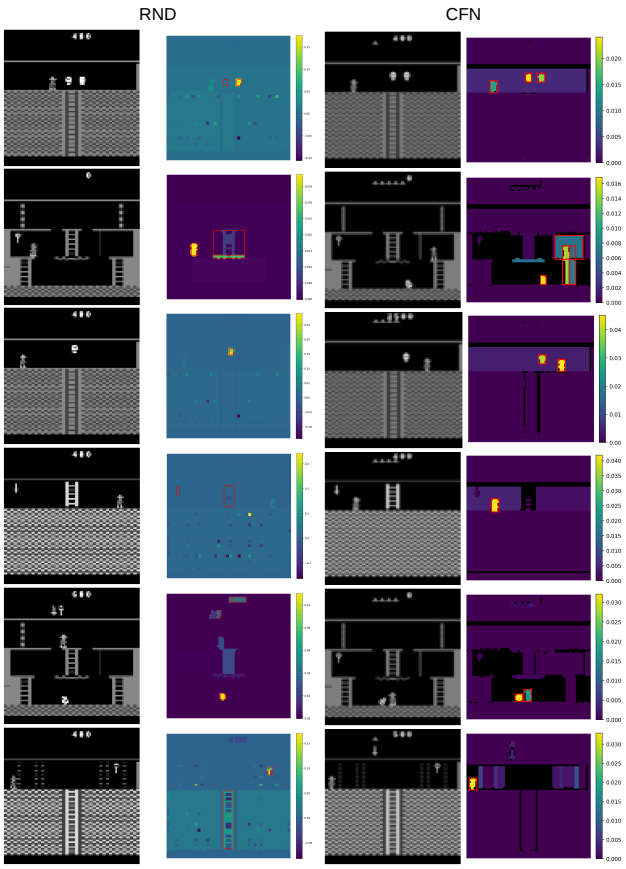


Figure 9: **DeepSHAP attribution is broadly consistent between CFN and RND novelty estimators, but CFN attribution is more concentrated.** Each row compares the most-novel state independently identified by RND and by CFN at a similar stage of exploration; because the two estimators judge novelty differently, these are not necessarily the same state. The left pair of columns shows RND's most-novel state alongside its DeepSHAP attribution map, and the right pair shows CFN's most-novel state alongside its attribution map. Noticeably, CFN's attributions are more concentrated around a few features, while RND's attributions are more diffused across all features. Despite this, the resulting classifiers attend to similar features.

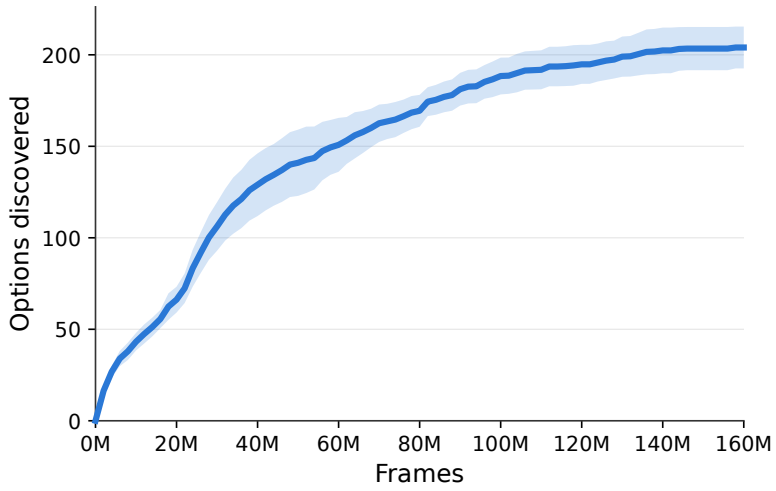


Figure 10: **Number of options discovered in MONTEZUMASREVENGE over training for the learning curves reported in Figure 5.** Solid line denotes the mean number of options discovered; shaded area denotes standard error.

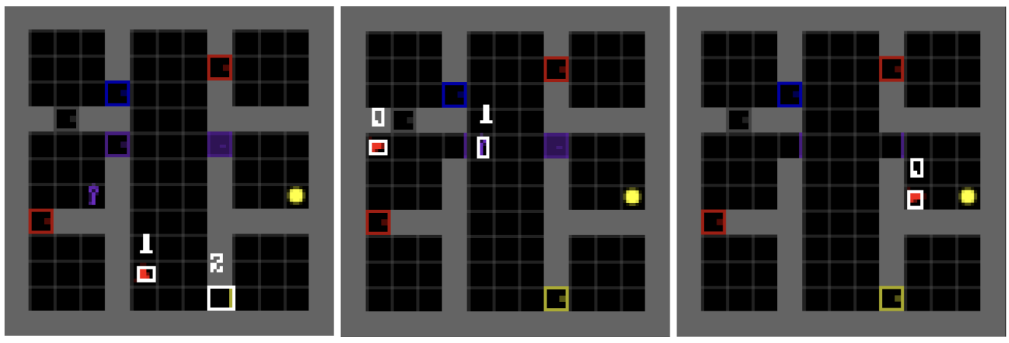


Figure 11: **Discovered subgoal classifiers in MINIGRID-KEYCORRIDOR.** Bounding boxes show the patches each classifier attends to; all other pixels are ignored. *(Left)* Subgoal: reach a location and open the yellow door. *(Middle)* Subgoal: reach the mid-left room while the key is in the hallway. *(Right)* Subgoal: enter the locked room containing the ball, regardless of key position or door configuration.

F HYPERPARAMETERS

Hyperparameters for each method were tuned independently, with the exception of Pixel Equality, which intentionally reuses Abstract Subgoals' hyperparameters (Table 4) to isolate the effect of the classifier.

Table 2: Hyperparameters shared across all agents, domains, and methods, unless overridden in Tables 3–5.

Hyperparameter	**Value**
R2D2 agent	
Number of actors	32
Actor backend	CPU
Sequence batch size	32
Trace length	40
Sequence period	20
Burn-in length	0
Bootstrap n-step	5
Prefetch size	1
Samples per insert (default, unless noted)	2
Replay buffer	
Max. replay size	100,000
Min. replay size	1,000
Priority exponent	0.9
Importance-sampling exponent	0.6
Max. priority weight	0.9
CFN module (architecture)	
Output dimension	20
Batch size	1,024
Max. replay size	2,000,000
Priority exponent	1.0
Importance-sampling exponent	0.6
Max. priority weight	0.9
Predictor learning rate	1×10^{-3}
Variable update period	400
Extrinsic coefficient	1.0
*Feature extraction & attribution (*KEYCORRIDOR / VISUALTAXI*)*	
Pixel intensity threshold (foreground mask & contour)	30
Contour retrieval / approximation mode	`RETR_EXTERNAL` / `CHAIN_APPROX_NONE`
Box size filter (min. width & height)	$> 0.25\times$ tile size (image height / 13)
Counterfactual attribution threshold (novelty drop), ϵ	0.1
Number of attribution baselines, M	1
Baseline window, w	episode length
Classifier match: mean pixel difference $\leq$	60
Classifier match: template score $>$	0.5

Table 3: Flat RL baselines: core hyperparameters per domain. TUP = target update period; SPI = samples-per-insert ratio. "Reward tx." is the transform applied to the R2D2 target (identity or signed-hyperbolic). "Intrinsic coeff." is the coefficient on the CFN novelty bonus added to the training reward.

Domain	Method	γ	LR	TUP	SPI	Reward tx.	Intrinsic coeff.	CFN LR	CFN SPI	CFN min. replay
MONTEZUMA	CFN	0.99	1×10^{-4}	600	2	signed-hyperbolic	0.01	1×10^{-3}	Unlimited	2,048
	R2D2	0.99	1×10^{-4}	600	2	signed-hyperbolic	0.0	—	—	—
KEYCORRIDOR	CFN	0.99	3×10^{-4}	1,200	8	identity	0.001	1×10^{-4}	Unlimited	12,500
	R2D2	0.99	3×10^{-4}	600	2	identity	—	—	—	—
VISUALTAXI	CFN	0.99	3×10^{-4}	600	2	identity	0.03	1×10^{-4}	8	2,048
	R2D2	0.99	3×10^{-4}	600	2	identity	—	—	—	—

Table 4: Abstract Subgoals / Pixel Equality: policy hyperparameters. TUP = target update period; SPI = samples-per-insert ratio. Pixel Equality uses identical values (it differs only on the classifier). λ is the CFN exploration-bonus coefficient from Section 3.1; it is 0.01 in all three domains.

Domain	Policy	γ	LR	TUP	SPI	λ	CFN LR	CFN TUP	CFN SPI	CFN min. replay
MONTEZUMA	Intra-option (π_o)	0.997	1×10^{-4}	600	2	—	—	—	—	—
	Exploration (π_{int})	0.99	3×10^{-4}	600	8	0.01	1×10^{-3}	600	Unlimited	12,500
KEYCORRIDOR	Intra-option (π_o)	0.997	1×10^{-4}	500	2	—	—	—	—	—
	Exploration (π_{int})	0.99	3×10^{-4}	600	8	0.01	1×10^{-3}	600	Unlimited	12,500
VISUALTAXI	Intra-option (π_o)	0.997	1×10^{-4}	600	2	—	—	—	—	—
	Exploration (π_{int})	0.99	3×10^{-4}	600	8	0.01	1×10^{-3}	600	Unlimited	12,500

Table 5: Goal-space and option-discovery hyperparameters for Abstract Subgoals.

Hyperparameter	MONTEZUMA	KEYCORRIDOR	VISUALTAXI
Option timeout, H	100	50	50
Goal-creation threshold multiplier, σ_{state}	1	1	1
Number of goals to replay	5	5	5
Option-initiation threshold, δ	N/A (scaled by $V_{\beta_o}(s)$, Appendix C)	0.1	0.1
Extrinsic-reward weight in $U(o)$, α	0.25	0	0
Classifier match threshold, ξ	0.7 (with $\mathcal{D} = 1 - \text{IOU}$)	see Table 2	see Table 2

IOU (intersection-over-union) is the ratio between the overlapping area of two bounding boxes and the area of their union.

Table 6: MONTEZUMASREVENGE-specific classifier discovery hyperparameters.

Hyperparameter	**Value**
Saliency method	DeepSHAP
SAM checkpoint	ViT-H (`sam_vit_h_4b8939.pth`)
Patch attribution threshold, σ_{patch}	1.0
Number of attribution baselines, M	15
Baseline window, w	50
Classifier trigger window size, κ	8