SELECTING DECISION-RELEVANT CONCEPTS IN REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Training interpretable concept-based policies requires practitioners to select which human-understandable concepts an agent should use for sequential decision-making. This process requires domain expertise, scales poorly with the number of candidate concepts, and provides no performance guarantees. To overcome this limitation, we propose the first algorithms for principled automatic concept selection in sequential decision-making. Our key insight is that concept selection can be viewed through the lens of state abstraction: a concept is *decision-relevant* if removing it causes the agent to confuse states that require different actions. That is, states with the same selected concept representation should share the same optimal action, preserving the decision structure of the original state space. This perspective leads to Decision-Relevant Selection (DRS), which selects a subset of concepts from a candidate set and provides performance bounds relating abstraction error to policy performance. Empirically, DRS performs comparably to manually curated concept sets and improves policy performance under test-time concept intervention across reinforcement learning benchmarks and a healthcare simulator.

1 INTRODUCTION

As we assign AI agents tasks with increasingly higher stakes, understanding their decision-making process becomes paramount. One family of interpretable models are *concept-based*. These models enable agents to make decisions using human-understandable attributes by first mapping high-dimensional states (e.g., images) to concepts (e.g., "Paddle Location > 0?") (Kim et al., 2018), then from concepts to actions (Zabounidis et al., 2023). This structure supports interpretable decision-making, diagnosis of concept prediction errors, and human intervention on mispredicted concepts (Zarlenga et al., 2023).

A key step in training these models is choosing the set of concepts for agent decision-making. Selecting a small subset of concepts can reduce both the complexity of the resulting representation (Bhatt et al., 2021) and the number of concepts humans must inspect during intervention. Prior work shows that simpler explanations are easier for humans to reason over (Bhatt et al., 2021; Ramaswamy et al., 2023). Figure 1 depicts a typical pipeline: practitioners distill a non-interpretable baseline policy into an interpretable concept-based policy by selecting an initial set of concepts, then iteratively training and evaluating policies to refine the concept set. This process is costly in both human effort and computational resources: it requires domain expertise and scales poorly with the number of concepts. Despite this effort, this process still provides no performance guarantees for the resulting model.

In this work, we study this problem of *concept selection*: given a bank of human-interpretable concepts, which subset should an agent use for sequential decision-making? To the best of our

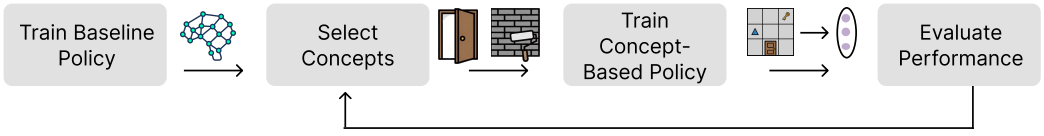

Figure 1: Standard pipeline for training concept-based policies. Practitioners select concepts by iteratively selecting candidate concepts, training concept-based policies, and evaluating their performance.

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.

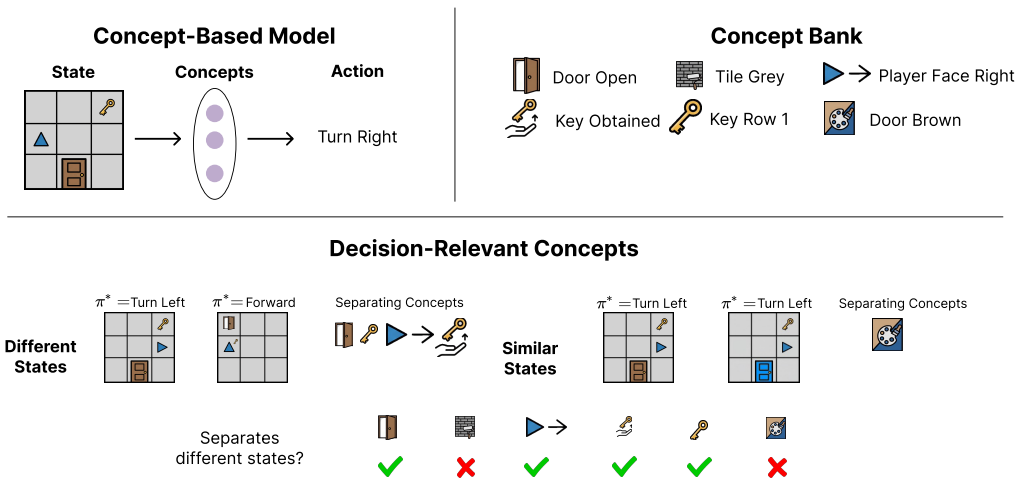

Figure 2: Concept-based models rely on a set of decision-relevant concepts that help distinguish between different states, yet currently these concepts are manually selected. In this work, we study how to identify and select decision-relevant concepts. Our key insight is that decision-relevant concepts best separate "different" states, where difference is defined by their decision consequences. We use this insight to develop algorithms for concept selection with performance guarantees.

knowledge, we propose the first algorithms to tackle this problem (Figure 2).While prior work has explored using large language models (LLMs) to select concepts in supervised settings (Yang et al., 2023; Zhao et al., 2025), there is little work investigating principled selection for sequential decision-making. Such a problem is more complex in sequential decision-making because states must be distinguished according to their long-term decision consequences. For example, the "Door Location" concept is useful in Minigrid only if the agent can first obtain the key.

The challenge is to identify a small subset that preserves decision performance without exhaustively training policies for different concept subsets. Our goal is not to show that fewer concepts always improve policy performance; rather, given a budget on representation complexity or human inspection, we ask which subset of concepts best preserves decision-relevant information. To tackle this, we propose the decision-relevant selection (DRS) algorithm, which automatically selects concepts that best distinguish between states. By viewing concepts as a case of *state abstractions*, we allow DRS to select concepts by minimizing state abstraction error and provide performance guarantees.

Contributions: We (i) design algorithms for selecting decision-relevant concepts in decision-making; (ii) derive performance bounds by connecting concept selection to state abstraction theory; (iii) demonstrate strong empirical performance across RL benchmarks and a healthcare simulator; and (iv) show that well-selected concepts improve policy performance under test-time intervention.

2 Preliminaries

Reinforcement Learning We study decision-making in RL for an agent in an infinite-horizon Markov decision process (MDP). An MDP $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$ is a tuple consisting of a set of states $\mathcal{S}$, a set of actions $\mathcal{A}$, a transition function $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ mapping state-action-state triplets to transition probabilities, a reward function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ mapping state-action pairs to rewards, and a discount factor $\gamma \in [0, 1]$. The agent learns a policy $\pi : \mathcal{S} \to \mathcal{A}$ that maps states to actions. We evaluate π through the value $V^\pi(s)$ and action-value functions $Q^\pi(s, a)$:

$$V^\pi(s) := \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \mid s_0 = s \right]$$

$$Q^\pi(s, a) := \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \,\middle|\, s_0 = s,\ a_0 = a \right],$$

where $a_t = \pi(s_t),\ s_{t+1} \sim P(\cdot \mid s_t, a_t)$. Let π^* denote an optimal policy.

Concept-Based Models Concept-based models map states to human-understandable concepts, then concepts to actions (Figure 2). As in previous work (Koh et al., 2020), each concept, c_i, is a Boolean-valued function, and the concept set is $\mathcal{C} = \{c_1, c_2, \ldots, c_K\}, c_i : \mathcal{S} \to \{0, 1\}$. Concept-based models consist of a concept predictor, $g_{\mathbf{c}}(s) := [c_{i_1}(s), \ldots, c_{i_k}(s)]$ and a policy $\pi_{\mathbf{c}} : \{0, 1\}^k \to \mathcal{A}$, so a state s is mapped to an action $a = \pi_{\mathbf{c}}(g_{\mathbf{c}}(s))$ (Delfosse et al., 2024). Users can intervene on concept-based models during test-time by correcting a subset of concept predictions $g_{\mathbf{c}}(s)$.

State Abstractions We propose viewing concepts as a special case of *state abstractions*. State abstractions group together states via a function $\phi_{\text{SA}} : \mathcal{S} \to \bar{\mathcal{Z}}$ such that states in the same abstraction satisfy certain desiderata. Because concept representations may merge states that still differ in their true dynamics or value, we focus on *ϵ-approximate state abstractions*, which require grouped states to have near-identical action values under the optimal policy (Abel et al., 2016). Formally, if $\phi_{\text{SA}}(s_1) = \phi_{\text{SA}}(s_2)$, then $\max_{a \in \mathcal{A}} \left| Q^{\pi^*}(s_1, a) - Q^{\pi^*}(s_2, a) \right| \ \leq\ \epsilon$. The state abstraction literature allows us to give guarantees based on the level of coverage for a given abstraction:

Theorem 2.1 (Theorem 1, (Abel et al., 2016))**.** *Suppose that ϕ_{SA} is an ϵ-approximate state abstraction. Let $\pi_{\phi_{\text{SA}}}$ be*

$$\pi_{\phi_{\text{SA}}} = \arg\max_{\pi} \mathbb{E}_{s \sim \mathcal{S}} \left[Q^{\pi^*}(\phi_{\text{SA}}(s), \pi(\phi_{\text{SA}}(s))) \right],$$

Then $V^{\pi^*}(s) - V^{\pi_{\phi_{\text{SA}}}}(s) \leq \frac{2\epsilon}{(1-\gamma)^2}$ for all s.

3 Identifying Decision-Relevant Concepts

We now present the first formalism of the concept selection problem. We begin by defining the problem setting, then propose *decision-relevancy* as the key principle for selecting concepts.

3.1 Problem Setting

Concept predictors $g_{\mathbf{c}}$ rely on the selection of a set of k concepts $\mathbf{c}$ from a larger bank of K concepts $\mathcal{C}$. Here, k controls the number of concepts selected and serves as a proxy for representation complexity; larger k can improve performance, but increases the number of concepts humans may need to inspect or correct (Bhatt et al., 2021). Our goal is therefore to identify the most decision-relevant concepts subject to this budget. We focus on concept selection rather than concept discovery, and in our setting the concept banks are obtained either manually or through large language models (Oikarinen et al., 2023). The key challenge in concept selection is that even when all candidate concepts are semantically meaningful, their utility for decision-making can vary:

Example 3.1. *Consider a state space $\mathcal{S} = \{1, 2, 3, 4\}$ and an action space $\mathcal{A} = \{\text{left}, \text{stay}, \text{right}\}$, which corresponds to an agent navigating a 1D state space with 4 states that loop around. Let $R(1, \cdot) = R(3, \cdot) = 1$ and $R(2, \cdot) = R(4, \cdot) = 0$. Consider two binary concepts:*

$$c_1(s) = \mathbf{1}\{s \bmod 2 = 0\},$$
$$c_2(s) = \mathbf{1}\{s \bmod 3 = 0\},$$

where $\mathbf{1}$ is the indicator function. When $k = 1$, selecting $\mathbf{c} = \{c_1\}$ ensures that states sharing the same parity have identical rewards and symmetric transitions under π^*. However, selecting $\mathbf{c} = \{c_2\}$ means the reward is not consistent; while $g_{\mathbf{c}}(1) = g_{\mathbf{c}}(2)$, $R(1, \text{stay}) \neq R(2, \text{stay})$. c_1 preserves decision-relevant distinctions that c_2 does not.

Objective Given a concept bank $\mathcal{C}$, our objective is

$$\max_{\mathbf{c}: |\mathbf{c}| \leq k} \mathbb{E}_{s \sim \mu} \left[Q^{\pi^*_{\mathbf{c}}}(s, \pi^*_{\mathbf{c}}(g_{\mathbf{c}}(s))) \right]. \tag{1}$$

Here, $\mu \in \Delta(\mathcal{S})$ is the initial state distribution, and $\pi^*_{\mathbf{c}}$ is the optimal policy given the concept predictor $g_{\mathbf{c}}$. Unlike classical state abstraction, which learns abstract state representations without constraints, we restrict our attention to selecting a subset of concepts from a fixed, human-interpretable concept bank. Our problem requires combinatorial optimization over a fixed concept bank, making it NP-hard to compute the optimal solution (proof in Appendix O).

3.2 Defining Decision-Relevancy

In Equation 1, the set of concepts $\mathbf{c}$ helps distinguish between states through the concept predictor $g_{\mathbf{c}}$. In other words, concepts need to separate states with different optimal actions. As a result, we propose identifying *decision-relevant* concepts as the key principle for concept selection.

Intuitively, concepts are decision-relevant only to the extent they distinguish states in which different actions lead to different outcomes. A natural condition is that if $\pi^*(s) \neq \pi^*(s')$, then $g_{\mathbf{c}}(s) \neq g_{\mathbf{c}}(s')$, where s and s' are pairs of states. A stronger desideratum is that if $Q^{\pi^*}(s, \cdot) \neq Q^{\pi^*}(s', \cdot)$, then $g_{\mathbf{c}}(s) \neq g_{\mathbf{c}}(s')$. The former helps us better learn policies, while the latter is needed to guarantee good performance from the resulting policy. While our main concern is the latter condition, the former guides algorithm development. To formally define decision-relevance, we introduce two ideas:

Definition 3.2 (Q-Distance)**.** For states s, s', the Q-distance is $D_{s,s'} := \max_{a \in \mathcal{A}} |Q^{\pi}(s, a) - Q^{\pi}(s', a)|$.

Definition 3.3 (Abstraction Error)**.** Define the abstraction error for a concept predictor g as

$$\epsilon_{Q^{\pi}}(g) := \max_{s,s' : g(s) = g(s')} D_{s,s'}.$$

Unless otherwise stated, let $\epsilon(g) = \epsilon_{Q^{\pi^*}}(g)$.

3.3 Benefits of Decision-Relevant Concepts

Proposition 3.1. *For a concept predictor $g_{\mathbf{c}}$, we have that*

$$V^{\pi^*}(s) - V^{\pi^*_{\mathbf{c}}}(s) \leq \frac{2\epsilon(g_{\mathbf{c}})}{(1-\gamma)^2}.$$

We prove that $g_{\mathbf{c}}$ induces an ϵ-state abstraction, allowing us to use Theorem 2.1 (see Appendix O). Intuitively, ϵ measures the largest decision-relevant difference between states that the concept representation fails to distinguish; smaller abstraction error yields tighter guarantees.

We additionally consider test-time intervention, where a user corrects an α-fraction of concept predictions, increasing their accuracy from δ^A toward δ^H. Under monotonicity conditions on concept errors and policy performance, lower abstraction error implies weakly higher policy performance at any intervention level α (Proposition A.1; Appendix A). Our theory predicts that decision-relevant concept sets can support strong performance both before and under intervention; it does not imply that concept selection increases the marginal benefit of intervention itself.

4 Concept Selection Algorithms

4.1 Selection with Perfect Concept Predictors

We introduce an algorithm for concept selection based on the intuition from Section 3.2 called **Decision-Relevant Selection (DRS)**. DRS selects concepts $\mathbf{c}$ to minimize abstraction error $\epsilon(g_{\mathbf{c}})$, while requiring them to separate a ρ-fraction of state pairs with different actions. The former controls performance guarantees, while the latter encourages distinguishing states requiring different decisions.

Formally, let $\mathbf{x}$ be a 0-1 vector representing the subset of concepts selected, and $\mathbf{Y}$ a binary matrix representing pairs of states differing in concept representations. We aim to minimize the abstraction error $D_{s,s'}$ across state pairs s, s' with $Y_{s,s'} = 1$, while limiting concept selection to k concepts:

$$\min_{\mathbf{x},\mathbf{Y}} \quad \textstyle\sum_{s,s' \in \mathcal{S}} D_{s,s'}(1 - Y_{s,s'}) \tag{2a}$$

$$\text{s.t.} \quad \textstyle\sum_{j=1}^{K} x_j \leq k \tag{2b}$$

$$Y_{s,s'} \leq \textstyle\sum_{j=1}^{K} x_j \, \mathbf{1}[c_j(s) \neq c_j(s')], \quad \forall\, s, s' \in \mathcal{S} \tag{2c}$$

$$Y_{s,s'} \leq Y_{t,t'}, \quad \forall\, (s, s'), (t, t') \in \mathcal{S}^2 \text{ s.t. } D_{s,s'} \leq D_{t,t'} \tag{2d}$$

$$\textstyle\sum_{s,s' \in \mathcal{S}} \mathbf{1}[\pi(s) \neq \pi(s')] Y_{s,s'} \geq \rho \sum_{s,s' \in \mathcal{S}} \mathbf{1}[\pi(s) \neq \pi(s')] \tag{2e}$$

$$x_j \in \{0, 1\}, \quad Y_{s,s'} \in \{0, 1\}. \tag{2f}$$

Practical Considerations Because the state space can be large, we sample from agent rollouts and consider only abstract states; do not differentiate states with $c_j(s) = c_j(s')\forall j \in [K]$. Let n_d denote the number of *distinct abstract states* observed, i.e., distinct values of $g_{\mathbf{c}}(s)$. Two structural properties keep n_d small in practice: (1) environmental constraints limit which concept combinations are physically realizable, and (2) policy-induced structure concentrates rollouts on a small subset of reachable abstract states. States sharing the same abstract representation can be aggregated, yielding an MILP with $O(n_d^2 + K)$ variables/constraints (runtimes and further discussion in Appendix H).

4.2 Performance Guarantees

We first characterize performance of the idealized algorithm from Equation 2.

Theorem 4.1. *Let $(\mathbf{x}^*, \mathbf{y}^*)$ be an optimal solution to Equation 2a with $\rho = 0$, and let $\mathbf{c} = \{c_j : x_j^* = 1\}$. $g_{\mathbf{c}}$ minimizes $\epsilon(g_{\mathbf{c}})$ among predictors using at most k concepts:*

$$\epsilon(g_{\mathbf{c}}) \;=\; \min_{g:|\mathbf{c}_g|\le k} \epsilon(g).$$

The solution exactly captures the optimization problem over $\epsilon(g_{\mathbf{c}})$, so the resulting concepts have small abstraction error. In practice, DRS differs from the idealized formulation in two ways: we relax constraint 2d and set $\rho = 0.75$. We relax 2d because it imposes too stringent a constraint on which pairs of states need to be separated. Empirically, we show both that the relaxation maintains or improves performance (Appendix E), while maintaining low abstraction error across environments (Appendix L). We next show that we can apply Proposition 3.1 regardless of these choices, as the relaxation affects the guarantee only through the additional abstraction error it introduces.

Corollary 4.1 (Performance of Practical DRS)**.** *Let $\mathbf{c}$ be the concept set selected by the idealized DRS formulation and $\mathbf{c}'$ the concept set selected after relaxing constraint 2d. Define $\epsilon' = \epsilon(g_{\mathbf{c}'}) - \epsilon(g_{\mathbf{c}})$. Then, for all $s \in \mathcal{S}$,*

$$V^{\pi^*}(s) - V^{\pi^*_{\mathbf{c}'}}(s) \le \frac{2(\epsilon(g_{\mathbf{c}}) + \epsilon')}{(1-\gamma)^2}.$$

Proof. By definition, $\epsilon(g_{\mathbf{c}'}) = \epsilon(g_{\mathbf{c}}) + \epsilon'$. Applying Proposition 3.1 to $g_{\mathbf{c}'}$ gives the result. □

We next show that DRS is robust to errors in our estimation of Q^π:

Theorem 4.2. *Assume that $\max_{s,a} |\hat{Q}^\pi(s,a) - Q^\pi(s,a)| \le \delta$. Let $(\mathbf{x}^*, \mathbf{Y}^*)$ be an optimal solution to Equation 2a constructed using $D_{s,s'}$, and let $\mathbf{c}^* = \{c_j : x_j^* = 1\}$. Let $\hat{D}_{s,s'} = \max_a |\hat{Q}^\pi(s,a) - \hat{Q}^\pi(s',a)|$, and let $(\hat{\mathbf{x}}, \hat{\mathbf{Y}})$ be an optimal solution to Equation 2a constructed using $\hat{D}_{s,s'}$, with induced abstraction $g_{\hat{\mathbf{c}}}$. Then for all $s \in \mathcal{S}$*

$$V^{\pi_{\mathbf{c}^*}}(s) - V^{\pi_{\hat{\mathbf{c}}}}(s) \le \frac{2(\epsilon_{Q^\pi}(g_{\mathbf{c}^*}) + 4\delta)}{(1-\gamma)^2}.$$

We prove this in Appendix O by bounding $|\hat{D}_{s,s'} - D_{s,s'}|$. Appendix N complements this empirically; DRS and DRS-log remain effective when Q-values are estimated from undertrained reference policies.

4.3 Selection with Imperfect Concept Predictors

When concept predictors are imperfect, state separation (Equation 2c) is stochastic, requiring new algorithms for concept selection. To tackle this, we (i) introduce a noise model for concept predictors to capture imperfections, (ii) show that no method can guarantee low abstraction error under adversarial perturbations, and (iii) derive a lower bound on state separation under stochastic perturbations.

Noisy Concept Predictors Let $f_{\mathbf{c}}^T = [f_{c_1}^{T_1}, \ldots, f_{c_k}^{T_k}]$ be a vector of perturbation functions so $T_i \subseteq \mathcal{S}$ and $\frac{|T_i|}{|\mathcal{S}|} = 1 - \delta_i$. Here, T_i are the states where we flip concept predictions:

$$f_{c_i}^{T_i}(s) := \begin{cases} 1 - c_i(s), & \text{if } s \in T_i, \\ c_i(s), & \text{if } s \notin T_i. \end{cases}$$

Limits Under Adversarial Noise We note that while we can extend concept selection to imperfect scenarios, any algorithm will perform poorly under adversarial noise.

Lemma 4.1. *Suppose that for each concept i, an adversary may apply a perturbation $f^{(T_i)}$ that flips the value of c_i on a subset $T_i \subseteq \mathcal{S}$ with $|T_i|/|\mathcal{S}| \leq 1 - \delta_i$. Then for any $g_{\mathbf{c}}$ with $\delta_i \leq 1 - \frac{1}{|\mathcal{S}|}$, there exists a collection of perturbations $f = \{f_{c_1}^{T_1}, \ldots, f_{c_k}^{T_k}\}$ such that $\epsilon(f(g_{\mathbf{c}})) = \max_{s,s'} D_{s,s'}$.*

We prove this by constructing adversarial perturbation functions f that fail to distinguish between states s and s' with maximum values of $D_{s,s'}$ (proof in Appendix O).

State Separation Under Noisy Concept Predictors In the DRS algorithm, we enforce state separation through $Y_{s,s'} \leq \sum_{j=1}^{K} x_j \mathbf{1}[c_j(s) \neq c_j(s')]$. When concept predictors are noisy, this constraint holds stochastically. To introduce this, let $\delta = [\delta_1, \ldots, \delta_k]$ be a vector representing the accuracy of each concept, where errors are stochastic and independent. Next, note that the probability that noisy predictions remain distinct is $\mathbb{P}[g_{\mathbf{c}}(s)_j \neq g_{\mathbf{c}}(s')_j \mid c_j(s) \neq c_j(s')] = \delta_j^2 + (1-\delta_j)^2$, since disagreement is preserved when both predictors are either correct or incorrect. For concepts with $c_j(s) = c_j(s')$, we conservatively ignore noise-induced separation. Because the product form is non-convex, we apply a logarithmic transformation to obtain a concave lower bound:

$$\log(1 - Y_{s,s'}) \geq \sum_{j:\mathbf{1}[c_j(s) \neq c_j(s')]}^{K} x_j \log\big(1 - (\delta_j^2 + (1-\delta_j)^2)\big).$$

We construct a new algorithm, called **DRS-log**, which uses a log-based constraint for state separation. In practice, we optimize this efficiently using standard convex nonlinear primitives.

5 Experiments

We evaluate our algorithms by focusing on three questions:

1. **RQ1:** What is the impact of decision-relevant concepts on performance?
2. **RQ2:** How do decision-relevant concepts affect performance under test-time intervention?
3. **RQ3:** How do automatically selected concepts compare with manually selected ones?

5.1 Experimental Setup

We evaluate on CartPole, MiniGrid, Pong, Boxing, and Glucose (Sutton & Barto, 2018; Chevalier-Boisvert et al., 2023; Bellemare et al., 2013; Man et al., 2014). Concepts are discretizations of human-understandable features. For example, in Pong, we discretize "Paddle Y" into "Paddle Y > 0" (see Appendix B). We evaluate with perfect and imperfect concept predictors, where imperfect concept predictors use CNNs to predict concepts. All concept predictors achieve at least 95% accuracy except for CartPole, which achieves 85% due to velocity-related concepts (details in Appendix B). Because concept predictors achieve 100% accuracy for Glucose, we report only the perfect setting for that environment. We run all experiments with 6 seeds (details in Appendix B).

We compare against three baselines: 1) Random, which selects a random k-subset of concepts; 2) Variance, which selects concepts by ranking them according to their variation between being active ($c_i(s) = 1$) and inactive ($c_i(s) = 0$); 3) Greedy, which selects concepts that best explain downstream behavior by conditioning on actions and iteratively choosing those that most reduce variability in Q-values. We normalize all rewards on a 0-100 scale, where the extremes are the min and max per-environment reward (details in Appendix D). We additionally compare against a mutual-information baseline on CartPole and MiniGrid in Appendix M.

5.2 RQ1: Impact on Performance

Decision-relevant concepts improve performance We compare concept selection algorithms in both perfect (top) and imperfect (bottom) settings in Figure 3. For each environment, we fix $k = \frac{K}{4}$.[1] For perfect settings, DRS performs best for four out of five environments, performing

[1] In Appendix F, we vary k and find similar results.

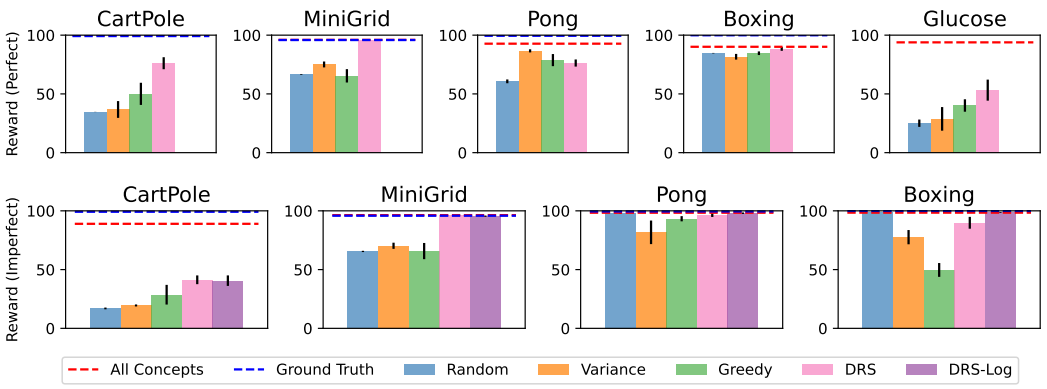

Figure 3: Normalized reward of concept selection algorithms with perfect (top) and imperfect (bottom) concept predictors. Our algorithm, DRS, improves performance compared to the random, variance, and greedy baselines for four out of five environments in the perfect setting. DRS and DRS-log improve performance or match ground-truth performance in all environments in the imperfect setting.

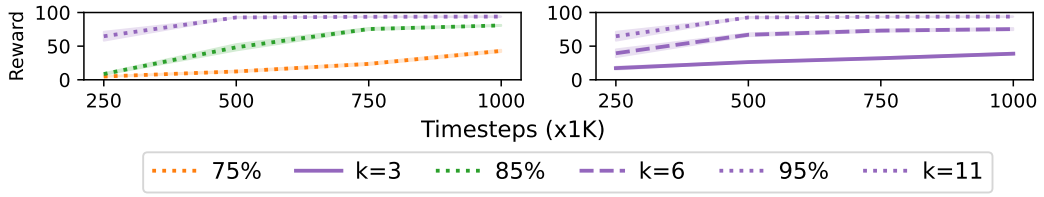

Figure 4: We vary the number of timesteps that we train policies for in MiniGrid, while also varying the accuracy of concept predictors (left) or the number of concepts selected (right). Increasing the accuracy of concept predictors speeds up training, while increasing the number of concepts increases the maximum performance.

50+% better than the best baseline in CartPole, 28% better in MiniGrid, 5% better in Boxing, and 44% better in Glucose. DRS performs similarly to baselines in perfect Pong, where binary concepts provide a coarse representation of fine-grained positional information. In imperfect settings, DRS and DRS-log either outperform baselines or nearly match ground-truth performance across all four environments. In MiniGrid, Pong, and Boxing environments, both DRS and DRS-log match ground-truth performance, while in CartPole, both DRS and DRS-log perform at least 58% better than alternatives. We give further qualitative insights in Appendix G. Importantly, subset selection can preserve the performance of the full concept set, as DRS retains over 99% of the All Concepts performance for three out of four environments in the imperfect setting while using only 25% of the available concepts. DRS is also robust to the quality of the reference policy used to estimate Q-values; DRS and DRS-log remain effective even with substantially undertrained reference policies (Appendix N). We additionally compare against a mutual-information baseline that selects concepts according to their mutual information with discretized Q-values. It performs similarly to or worse than the existing baselines, suggesting that marginal concept-level statistics alone are insufficient to identify decision-relevant concepts (Appendix M).

Decision-relevant concepts increase maximum performance, while concept predictor accuracy increases training efficiency To better understand how well-selected concepts impact performance, in Figure 4, we plot training curves for MiniGrid policies in two settings. On the left, we vary concept predictor accuracies, and on the right, the number of concepts selected. Increases in accuracy improve training efficiency; fewer timesteps are needed to achieve the same performance. Increasing the number of concepts increases the maximum performance (CartPole results in Appendix I).

In Figure 5, we show that accurate concept predictors (y axis) and many concepts (x axis) are needed to achieve near-optimal reward. For example, in CartPole, using only $k = 2$ concepts caps

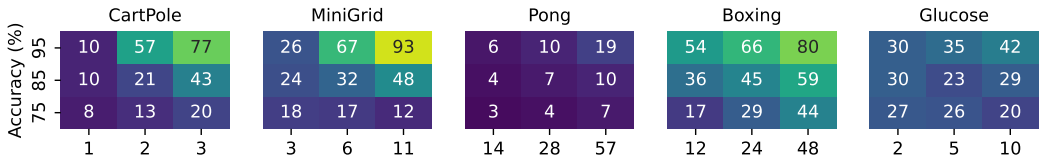

Figure 5: Impact of the number of concepts selected against the accuracy of the underlying concepts. Increasing either number or accuracy of concepts can improve performance, and that sufficiently accurate and many concepts are needed to ensure good performance.

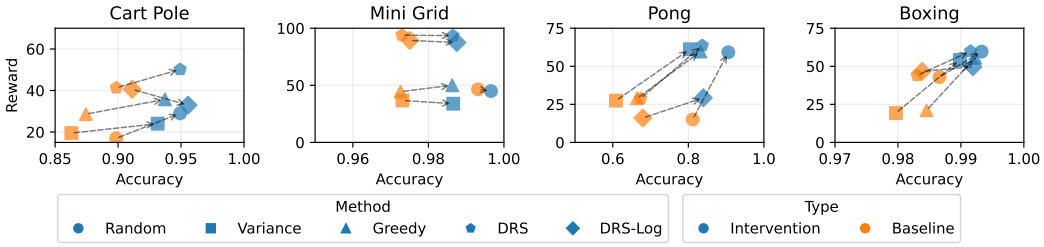

Figure 6: Policy performance before and after test-time intervention across four environments. DRS achieves the highest reward after intervention across all four environments.

performance at 38, while having 85% accurate concept predictors caps performance at 43, compared to 77 with 95% accurate predictors at $k = 3$. While well-selected concepts are necessary to ensure effective concept-based models, they are not sufficient. The exact number of concepts and accuracy level is environment-dependent; in Glucose and Pong even 95% accurate concept predictors are not sufficient, while in MiniGrid, 95% accurate concept predictors achieve a normalized reward of 93.

5.3 RQ2: Impact on Test-Time Intervention

Decision-relevant concepts improve performance under test-time intervention We evaluate concept selection under test-time intervention in the imperfect setting. We weaken policies and concept predictors by training for fewer timesteps to create settings in which intervention can improve performance. In Figure 6, we compare reward before and after intervention, fixing $\alpha = 0.5$ and randomly selecting concepts to intervene on; Appendix J varies α. Across environments, DRS achieves the highest reward after intervention, outperforming alternatives by 40% in CartPole, 87% in MiniGrid, 4% in Pong, and 0.5% in Boxing. DRS-log can perform worse under intervention because it is optimized for the original concept-accuracy profile.

5.4 RQ3: Automatic and Manually Selected Concepts

Decision-relevant concepts match performance of manual selection We use the Caltech-UC Davis Birds (CUB) dataset (Wah et al., 2011), a standard supervised task in concept-based learning. Prior work uses a manually selected subset of 112 of the original 312 concepts, so we investigate how selection algorithms compare with manual selection. We start from the original 312 concepts and compare concept selection algorithms (details on supervised algorithm extensions in Appendix K).

In Figure 7, we find that automatic concept selection can achieve performance comparable to manual selection. Variance, DRS, and DRS-log achieve within 0.6% of manual performance using only $k = 80$ concepts, while selecting concepts that substantially overlap with the manually selected set.

6 Related Work

Interpretable Reinforcement Learning Work in interpretability examines how we can explain model predictions through methods that are trustworthy, informative, and transferable (Lipton, 2018).

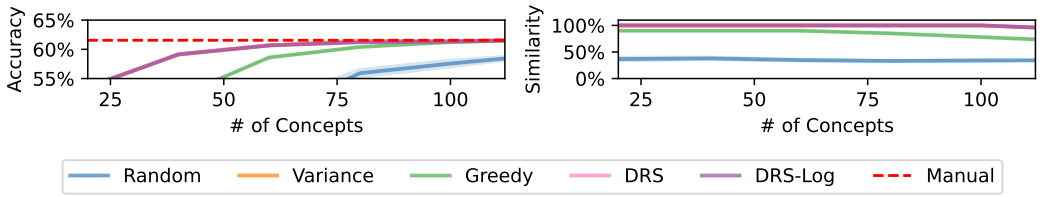

Figure 7: DRS achieves performance comparable to manual concept selection on CUB (left) while selecting a subset of manually selected concepts (right).

In an RL setting, prior work investigates feature importance-based methods (Topin et al., 2021) and policy-level explanations (Topin & Veloso, 2019). Work in the concept-based RL space builds upon policy-level explanations to construct concept policy models (Das et al., 2023) and extends these to multi-agent (Zabounidis et al., 2023) and scarce-label settings (Ye et al., 2025). Our work builds upon policy-level explanations to construct concept-based policies.

Concept-Based Interpretability Concept-based explanations fall into two categories: post-hoc methods (Kim et al., 2018) and interpretable models (Koh et al., 2020; Espinosa Zarlenga et al., 2022). Key to the success of both methods is the selection of concepts. While this parallels the feature selection problem (Li et al., 2017), our work differs by selecting human-interpretable concepts based on decision relevance rather than predictive performance. In the supervised setting, concepts are often selected manually (Koh et al., 2020; Espinosa Zarlenga et al., 2022), while in the unsupervised setting, concepts are selected via LLMs (Oikarinen et al., 2023; Yang et al., 2023). We differentiate ourselves from prior work (Yeh et al., 2020), which analyzes alignment between concept vectors and states in supervised learning, because we 1) formalize the concept selection problem, 2) focus on interpretable models rather than post-hoc, and 3) focus on decision-relevance in RL.

State Abstractions We leverage state abstraction theory to generate performance guarantees for concept selection. Work in state abstraction imposes various desiderata on states within an abstraction group, including *exact bisimulation* (Givan et al., 2003) or ε*–optimality* criteria (Li et al., 2006). Beyond exact bisimulation, other work studies approximate abstractions (Abel et al., 2016) and abstraction selection (Jiang et al., 2015). While both our work and state abstractions aim to aggregate states according to various desiderata, our goal is to ensure interpretability via concept selection, while state abstraction focuses on improving sample complexity for non-interpretable policies.

7 Limitations and Conclusion

Limitations We first note that the DRS algorithm used in practice relaxes constraint 2d, which is required for optimality; Corollary 4.1 instead provides a simple extension based on the resulting abstraction error. Second, our framework requires a pre-trained policy and Q-value estimates before concept selection, which may be limiting when no reference policy exists, though Appendix N shows DRS is robust to substantially suboptimal policies. Third, scalability to very large concept banks or highly stochastic environments is not formally guaranteed, though our runtime experiments suggest this is not a practical barrier. Fourth, extending to continuous or ordinal concepts requires modifications to the underlying optimization. We view these as productive directions for future work.

Conclusion Our work investigates how to reduce manual concept engineering by selecting concepts in a principled manner. Our key intuition is that decision-relevant concepts should distinguish between states that are different. We use results from state abstractions to quantify what "different" means, and use this to bound performance for a selected subset of concepts. In practice, we show that our proposed algorithms can identify concepts most relevant for decision-making, which improves performance both in standalone settings and under test-time concept intervention. Our work takes a step toward understanding how explanation structure shapes decision quality and human–AI collaboration.

LLM USE DISCLOSURE

We used LLMs to assist us with the generation and proving of theorems, along with the writing of code to test experiments. All proofs were checked manually by the authors, and writing in the paper was originally done by the authors. LLM-generated code was manually checked by authors. We additionally used LLMs to assist with proofreading and editing this paper. LLMs assisted with finding additional related work after the authors of the papers did their own related work search first. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We include a link to an anonymous repository in the attached supplementary material. We give additional details on the experiment setup in Appendix C. We additionally discuss proof details in Appendix O.

REFERENCES

David Abel, David Hershkowitz, and Michael Littman. Near optimal behavior via approximate state abstraction. In *International Conference on Machine Learning*, pp. 2915–2923. PMLR, 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.

Umang Bhatt, Adrian Weller, and José MF Moura. Evaluating and aggregating feature-based model explanations. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pp. 3016–3022, 2021.

Maxime Chevalier-Boisvert, Bolun Dai, Mark Towers, Rodrigo Perez-Vicente, Lucas Willems, Salem Lahlou, Suman Pal, Pablo Samuel Castro, and Jordan Terry. Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. *Advances in Neural Information Processing Systems*, 36:73383–73394, 2023.

Devleena Das, Sonia Chernova, and Been Kim. State2explanation: Concept-based explanations to benefit agent learning and user understanding. *Advances in Neural Information Processing Systems*, 36:67156–67182, 2023.

Quentin Delfosse, Sebastian Sztwiertnia, Mark Rothermel, Wolfgang Stammer, and Kristian Kersting. Interpretable concept bottlenecks to align reinforcement learning agents. *Advances in Neural Information Processing Systems*, 37:66826–66855, 2024.

Mateo Espinosa Zarlenga, Pietro Barbiero, Gabriele Ciravegna, Giuseppe Marra, Francesco Giannini, Michelangelo Diligenti, Zohreh Shams, Frederic Precioso, Stefano Melacci, Adrian Weller, et al. Concept embedding models: Beyond the accuracy-explainability trade-off. *Advances in Neural Information Processing Systems*, 35:21400–21413, 2022.

Robert Givan, Thomas Dean, and Michael Greig. Equivalence notions and model minimization in markov decision processes. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2003.

Nan Jiang, Alex Kulesza, and Satinder Singh. Abstraction selection in model-based reinforcement learning. In *International Conference on Machine Learning*, pp. 179–188. PMLR, 2015.

Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pp. 2668–2677. PMLR, 2018.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International Conference on Machine Learning*, pp. 5338–5348. PMLR, 2020.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *ACM computing surveys (CSUR)*, 50(6):1–45, 2017.

Lihong Li, Thomas J. Walsh, and Michael L. Littman. Towards a unified theory of state abstraction for mdps. *Proceedings of the Ninth International Symposium on Artificial Intelligence and Mathematics (ISAIM)*, 2006.

Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, 2018.

Chiara Dalla Man, Francesco Micheletto, Dayu Lv, Marc Breton, Boris Kovatchev, and Claudio Cobelli. The uva/padova type 1 diabetes simulator: new features. *Journal of diabetes science and technology*, 8(1):26–34, 2014.

Tuomas Oikarinen, Subhro Das, Lam M Nguyen, and Tsui-Wei Weng. Label-free concept bottleneck models. In *The Eleventh International Conference on Learning Representations*, 2023.

Vikram V Ramaswamy, Sunnie SY Kim, Ruth Fong, and Olga Russakovsky. Overlooked factors in concept-based explanations: Dataset choice, concept learnability, and human capability. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10932–10941. IEEE, 2023.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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

Nicholay Topin and Manuela Veloso. Generation of policy-level explanations for reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 2514–2521, 2019.

Nicholay Topin, Stephanie Milani, Fei Fang, and Manuela Veloso. Iterative bounding mdps: Learning interpretable policies via non-interpretable methods. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 9923–9931, 2021.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. *California Institute of Technology*, 2011.

Yue Yang, Artemis Panagopoulou, Shenghao Zhou, Daniel Jin, Chris Callison-Burch, and Mark Yatskar. Language in a bottle: Language model guided concept bottlenecks for interpretable image classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 19187–19197, 2023.

Zhuorui Ye, Stephanie Milani, Geoffrey J Gordon, and Fei Fang. Licorice: Label-efficient concept-based interpretable reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

Chih-Kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. On completeness-aware concept-based explanations in deep neural networks. *Advances in neural information processing systems*, 33:20554–20565, 2020.

Renos Zabounidis, Joseph Campbell, Simon Stepputtis, Dana Hughes, and Katia P Sycara. Concept learning for interpretable multi-agent reinforcement learning. In *Conference on Robot Learning*, pp. 1828–1837. PMLR, 2023.

Mateo Espinosa Zarlenga, Katherine M Collins, Krishnamurthy Dj Dvijotham, Adrian Weller, Zohreh Shams, and Mateja Jamnik. Learning to receive help: Intervention-aware concept embedding models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Delong Zhao, Qiang Huang, Di Yan, Yiqun Sun, and Jun Yu. Partially shared concept bottleneck models. *arXiv preprint arXiv:2511.22170*, 2025.

A Human Intervention Theory

During test-time intervention, a (potentially simulated) user intervenes on an α-fraction of concepts, raising their accuracy from δ^A to δ^H. Here, the exact subset of intervened concepts is unknown until test-time. Intuitively, when interventions improve concept accuracy, policies using decision-relevant concepts benefit more from human effort:

Assumption A.1. *For any two concept sets* $\mathbf{c}, \mathbf{c}'$ *and any two accuracy vectors* $\delta' \preceq \delta$*, if*

$$\mathbb{E}[V^{\pi^*_{\mathbf{c},\delta}}(s)] \geq \mathbb{E}[V^{\pi^*_{\mathbf{c}',\delta}}(s)],$$

then

$$\mathbb{E}[V^{\pi^*_{\mathbf{c},\delta'}}(s)] \geq \mathbb{E}[V^{\pi^*_{\mathbf{c}',\delta'}}(s)].$$

Assumption A.2. *For any two concept sets* $\mathbf{c}, \mathbf{c}'$*, when* $\delta = \mathbf{1}$*, if* $\epsilon(g_{\mathbf{c}}) \leq \epsilon(g_{\mathbf{c}'})$*, then*

$$\mathbb{E}[V^{\pi^*_{\mathbf{c},\delta}}(s)] \geq \mathbb{E}[V^{\pi^*_{\mathbf{c}',\delta}}(s)].$$

Assumption A.1 is a rank-preservation condition on policy performance as concept accuracy changes; it may fail when changes in concept accuracy affect different concept sets differently. Assumption A.2 holds when the gap $V^{\pi^*}(s) - V^{\pi^*_{\mathbf{c}}}(s) \leq \frac{2\epsilon}{(1-\gamma)^2}$ is constant, and empirically we find that lower abstraction error correlates with better performance (Figure 5).

Proposition A.1. *Let* $\mathbf{c}$ *and* $\mathbf{c}'$ *be two concept sets, and let* $\epsilon(g_{\mathbf{c}}) \leq \epsilon(g_{\mathbf{c}'})$*. Under Assumptions A.1 and A.2, for any intervention level* $\alpha \in [0, 1]$ *and any accuracy vector* $\delta^{(\alpha)}$ *obtained by improving an* α*-fraction of concepts from* δ^A *toward* $\delta^H = \mathbf{1}$*, we have*

$$\mathbb{E}[V^{\pi^*_{\mathbf{c},\delta^{(\alpha)}}}(s)] \geq \mathbb{E}[V^{\pi^*_{\mathbf{c}',\delta^{(\alpha)}}}(s)].$$

Intuitively, interventions benefit from policies built on decision-relevant concepts.

B Environment Details

We provide details on the construction of glucose environment, concept banks, and the training of concept predictors and groundtruth models in this section.

Glucose Environment Construction We use the glucose environment Man et al. (2014) as a test-bed to understand the performance of our algorithm in real-world healthcare environments. Our observation space is a low-dimensional vector consisting of six normalized features derived from the current blood glucose (BG), its short-term trend, carbohydrate intake, insulin-on-board, and recent dosing history. The observation space is defined as a bounded continuous box, while the action space is discrete and corresponds to a set of clinically realistic bolus doses. Each step corresponds to 5 minutes of simulated time. At every step, a discrete action is mapped to an insulin dose (in units), which is then passed to the underlying simulator. The environment tracks BG history and the time since administration for all prior doses, and uses these quantities to construct the next observation. The reward function penalizes hypo- and hyperglycemia while incorporating dose size and BG rate-of-change. Resetting initializes the patient model, clears all histories, and returns the constructed feature vector for the first time step.

Concept Banks We give details on the concept banks used for each of our five environments:

1. **CartPole** originally has a state space consisting of four numbers: the position, velocity, angle, and angular velocity. We discretize each of these with two thresholds for position and angle, and four thresholds for angular velocity and velocity. This gives a total of 12 concepts.
2. **MiniGrid** is characterized through the agent's position, door location, key obtaining, and door unlock status. For MiniGrid, we use a fully symbolic state representation consisting of 12 discrete features extracted from the final observation frame. These features encode the agent's (x, y) position, orientation, object locations, and local binary indicators. Concretely, the features correspond to: the agent's x-position and y-position; the agent's direction; the

x- and y-positions of the key; the x- and y-positions of the door; a binary indicator for whether the door is open; and four binary indicators denoting the presence of obstacles to the right, left, down, and up of the agent. This totals to 44 concepts, noting that each feature is discretized into multiple binary concepts.

3. **Pong** has concepts derived from object-centric state variables computed from the last one or two frames of the observation stack. Base features include continuous-valued positions, velocities, and relative offsets of the agent paddle, ball, and opponent paddle. Specifically, we extract the agent paddle's vertical position; the ball's horizontal and vertical positions; the opponent paddle's vertical position; the ball's horizontal and vertical velocities; the opponent paddle's vertical velocity; and several relative position features, including paddle–ball and opponent–ball offsets. Position features are centered and normalized to lie approximately in $[-1, 1]$, while velocity features are computed as frame-to-frame differences, clipped to a fixed range, and rescaled. This totals 228 concepts.
4. **Boxing** concepts are constructed from the positions and movements of the player and opponent sprites. Base features include the normalized x- and y-positions of both the player and the enemy in the final observation frame, as well as their velocities computed as differences between the last two frames. We additionally include relative position features capturing the horizontal and vertical offsets between the player and the enemy. These concepts capture spatial dominance, motion direction, and relative positioning between the two agents.
5. **Glucose** has a state which consists of six continuous-valued physiological or control-related variables extracted from the final observation frame. Each feature corresponds to a scalar quantity, such as an absolute level, rate of change, or control signal, and no temporal differencing is applied. Concepts are defined by thresholding each feature at a hand-specified set of values chosen to reflect the feature's numerical range and semantics. The threshold sets vary across features and include both fine-grained thresholds near zero and coarser thresholds for larger-magnitude variables.

Training Models We provide details on model training for each environment below. For all policies, we use Proximal Policy Optimization (PPO) Schulman et al. (2017), with environment-specific hyperparameters. For CartPole, we train using $n_{\text{steps}} = 1024$, a batch size of 256, and 4 epochs per update, with an entropy coefficient of 5×10^{-4}, a learning rate of 10^{-4}, and a value-function loss coefficient of 0.5. For MiniGrid, we again use $n_{\text{steps}} = 1024$ but increase the batch size to 1024, while keeping 4 training epochs per update. For Pong, we use $n_{\text{steps}} = 1024$, a batch size of 256, and 5 epochs per update, with a learning rate of 10^{-4}. Finally, for Boxing, we use $n_{\text{steps}} = 1024$, a batch size of 1024, and 4 epochs per update. All experiments are run on a GPU using CUDA.

Next, we give details on training interpretable policies, which again use PPO Schulman et al. (2017) but with smaller or structured network architectures to encourage interpretability. For CartPole, we use a two-layer policy network with hidden dimensions $[64, 64]$, a rollout length of 256, batch size 128, and 4 epochs per update, with a learning rate of 10^{-4}, zero entropy regularization, a clipping range of 0.1, and a target KL divergence of 0.01. For MiniGrid, we use a learning rate of 3×10^{-4} with $n_{\text{steps}} = 1024$, batch size 1024, and 4 epochs per update. For Pong, we use a larger two-layer architecture with hidden dimensions $[256, 256]$, together with $n_{\text{steps}} = 1024$, batch size 256, 5 epochs per update, and a learning rate of 10^{-4}. For Boxing, we use different hyperparameters depending on the experimental setting: in imperfect or intervention-based settings, we use a wider architecture $[512, 256]$, larger rollouts ($n_{\text{steps}} = 2048$), batch size 512, 10 epochs per update, a learning rate of 2×10^{-4}, entropy coefficient 0.01, and clipping fraction 0.1; otherwise, we use a smaller architecture $[128, 128]$ with $n_{\text{steps}} = 128$, batch size 256, 4 epochs per update, and entropy coefficient 0.01. Finally, for the Glucose environment (including both processed and raw variants), we use $n_{\text{steps}} = 1024$, batch size 256, 5 epochs per update, a learning rate of 3×10^{-4}, and an entropy coefficient of 0.01, which we found necessary for stable training.

For training concept predictors, we use the Nature CNN backbone LeCun et al. (2015). We train all CNNs for 25 epochs using 50k rollout steps. All concept predictors are at least 95% accurate per concept in MiniGrid, Pong, and Boxing. For CartPole, velocity-related concepts are at least 85% accurate, while all other concepts are at least 95% accurate. Concept predictions are kept as logits to preserve information rather than rounding. We train concept-based models under the sequential

paradigm Koh et al. (2020), where we first train concept predictors $g_{\mathbf{c}}$, then learn interpretable policies $\pi^*_{\mathbf{c}}$.

We train all models on a server running Ubuntu 20.04 (Linux kernel 5.15) with 4 NVIDIA RTX A6000 GPUs.

C Experimental Details

We run all experiments for 6 seeds and report the standard error. We normalize rewards per-environment by setting the following min/max ranges for reward: CartPole is from 0 to 500, MiniGrid is from 0 to 1, Pong is from -21 to 21, Boxing is from 0 to 100, and Glucose is from -20 to 40. For the first four environments, the normalization comes from bounds on the reward, while for Glucose, we develop this normalization factor based on the performance of all policies. For intervention experiments, we weaken training for all but the CartPole environment, so concept predictors are trained for only 1 epoch, and policies are trained for 250k, 4M, and 2M timesteps for MiniGrid, Pong, and Boxing, respectively. We assume that intervening on concept j results in perfect accuracy for the concept, $\delta_j = 1$. For all policies, we use Proximal Policy Optimization (PPO) Schulman et al. (2017), with environment-specific hyperparameters. For the DRS and DRS-log algorithms we fix $\rho = 0.75$ and evaluate other choices in Appendix E.

D Algorithm Implementation Details

We estimate Q-values using a stable TD learning procedure Sutton & Barto (2018). A neural Q-network with two ReLU hidden layers and orthogonal initialization is trained together with a slowly updated target network. At each step, the agent stores transitions (s, a, r, s') into a replay buffer and updates the parameters by minimizing the Double–DQN TD loss.

$$\ell \;=\; \Big(Q_\theta(s,a) - \big[r + \gamma\, Q_{\theta^-}\big(s',\, \arg\max_{a'} Q_\theta(s',a')\big)\big]\Big)^2.$$

Gradients are clipped for stability, and the target network is synchronized periodically. After training, the Q-value $Q(s, a)$ for any state–action pair is obtained by a single forward pass through the learned network.

The greedy method measures the within-class standard deviation after splitting on each concept then sorts by this. That is, for each concept, we sum the standard deviation for Q values when the concept is on and off, summed across all actions. The intuition is that if a concept is highly relevant to the reward or dynamics, states where the concept is on should have similar Q-values, and states where it is off should have similar Q-values.

For the DRS and DRS-log algorithms, we fix $\rho = 0.75$; if no solution is found, we decrement ρ by 0.05 until a solution is found. When using the DRS algorithm in practice, we relax 2d by dropping the constraint, as this improves performance in practice (see Appendix E). We sample $D_{s,s'}$ from rollouts of π^* with 200k timesteps, while we compute $\mathbf{1}[\pi^*(s) \neq \pi^*(s')]$ with 10k timesteps from the rollout of π^*.

E Hyperparameter Selection for DRS

We compare our DRS algorithm against two variations of the algorithm: one which incorporates 2d, along with others which vary the value of $\rho \in \{0, 0.5, 0.75, 0.99\}$. We compare these in both the perfect and imperfect MiniGrid and CartPole algorithms as a simple evaluation to understand performance.

In Figure 8, we find that re-introducing 2d leads to worse performance in perfect CartPole, while setting $\rho = 0.99$ or $\rho = 0.0$ also leads to worse performance. Here, enforcing 2d forces an ordering over state pairs to cover, which can be too strict in practice, leading us to relax it when constructing DRS. Similarly, setting $\rho = 0.99$ can be too strict, as it forces DRS to separate 99% of state pairs with different optimal actions.

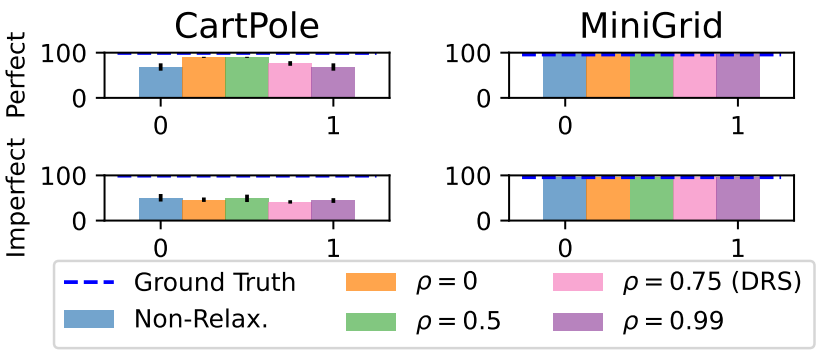

Figure 8: We assess the performance of a) not relaxing constraint 2d for DRS and b) varying ρ for CartPole and MiniGrid. We find that $\rho = 0.75$ performs well across perfect and imperfect settings, while introducing 2d leads to lower performance in CartPole.

F Varied Number of Concepts Selected

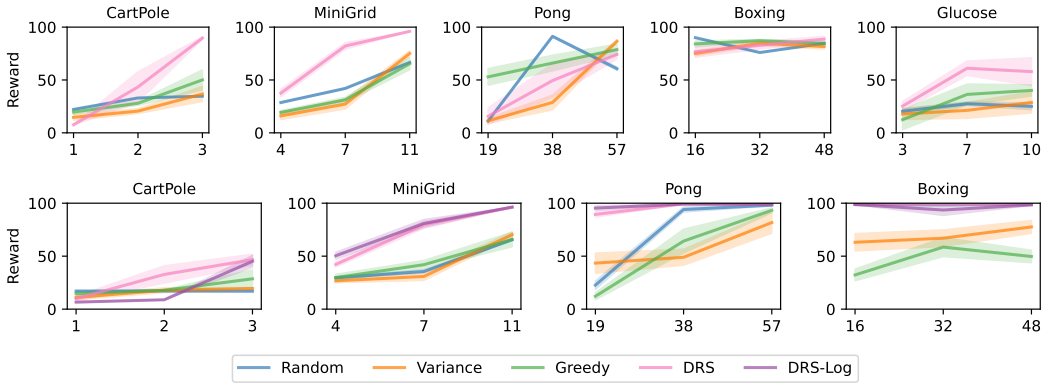

Figure 9: We compare concept selection algorithms while varying the number of concepts selected k with perfect (top) and imperfect (bottom) concept predictors. We find that for CartPole and MiniGrid, DRS outperforms baselines across values of k, while for environments such as Boxing and Glucose, a threshold number of concepts is needed before DRS outperforms baselines.

We vary $k \in \{\frac{K}{12}, \frac{K}{6}, \frac{K}{4}\}$ in Figure 9 for both perfect and imperfect settings. We find that better concept selection algorithms can improve performance across many values of k. For example, in the perfect environment setting for CartPole, we find that the DRS algorithm improves performance even with only $k = 2$ concepts, leading to a 32% improvement over the next best alternative. Similarly, such a trend is also present in MiniGrid, where the DRS algorithm improves performance by 96% when $k = 7$. In some environments, many concepts are necessary to achieve high performance; for example, in Glucose, while the DRS algorithm is 24% better than alternatives when $k = 3$, it achieves 131% better than all alternatives once $k = 10$.

G Qualitative Analysis

We next provide intuition for why DRS improves performance by comparing the set of concepts selected. In MiniGrid, concepts include agent location, door location, and agent direction (see Table 1). DRS consistently selects direction-related concepts, which are not selected by variance or greedy. This occurs because the direction-related concepts have high action variability, but are also decision-relevant. Knowing which direction an agent faces is critical to train effective policies.

Table 1: Average number of concepts selected per group by DRS in MiniGrid, with total concepts per group.

Concept Group	Avg. Selected	Total
Agent		
Position X	1	5
Position Y	2	5
Direction	3	4
Action: Right	0	2
Action: Left	1	2
Action: Down	0	2
Action: Up	1	2
Key		
Position X	0	5
Position Y	1	5
Door		
Position X	0	5
Position Y	1	5
Open State	1	2

H RUNTIME OF ALGORITHMS

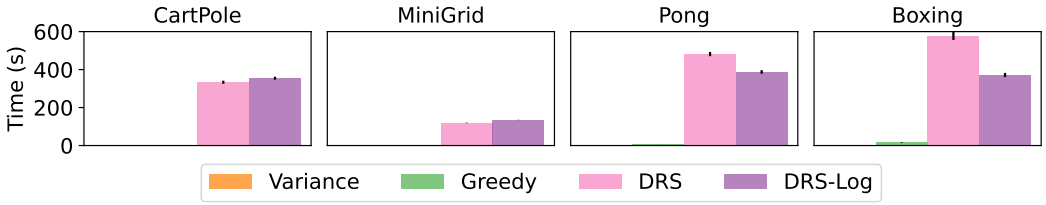

Figure 10: We compare the time to compute across methods and datasets, and find that all algorithms run in under ten minutes. While DRS and DRS-log take longer than variance and greedy, they still run quickly across environments.

We compare the runtime of the Variance, Greedy, DRS and DRS-log algorithms across different environments (while we note that the random selection algorithm essentially takes no time). In Figure 10, we show that all algorithms run in under ten minutes across all methods. This runtime is much smaller than the time needed to train the policies themselves, which can take hours. While more complex environments such as Pong or Boxing take longer for the DRS and DRS-log algorithms, they still run quickly. Much of this time is spent solving the optimization problem and to run rollouts to compute optimal actions. Moreover, we note that n_d, or the effective dimension, is small in practice. For example, in our rollouts for Pong, we find that $n_d = 1951$ when rolled out for 200,000 steps, which is much less than $2^k = 2^{228}$. Additionally, even when we double the number of concepts in Pong (using redundant concepts), we find that all algorithms run in under 10 minutes (with only a 23% increase in the time taken) even with 400+ candidate concepts. We note that we precomputed Q values for all methods, though the computation of Q values takes under ten minutes as well.

I TRAINING CURVES FOR CARTPOLE

In Figure 11, we plot training curves when varying the concept predictor accuracy (left) and number of concepts (right). Our results are noisier compared to those for MiniGrid, yet in both, we find that when concept predictor accuracy is low or the number of concepts is low, policies fail to train. Concepts train slightly faster when going from 75% to 85% accurate concept predictors, and similarly the maximum performance increases when increasing the number of concepts selected. However, both of these trends are noisy in CartPole due to the large amount of stochasticity; even small differences in the set of concepts selected or the training dynamics can greatly impact policy performance.

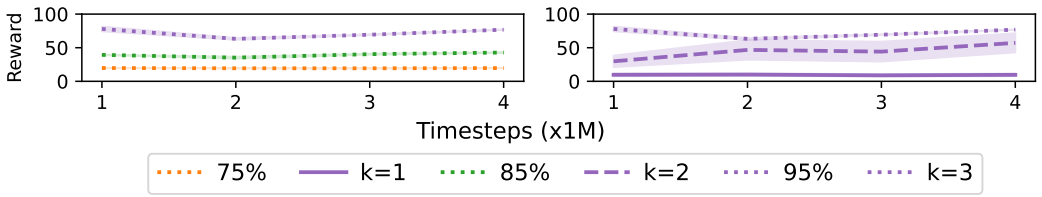

Figure 11: We vary the number of timesteps that we train policies for in CartPole, while also varying the accuracy of concept predictors (left) or the number of concepts selected (right). When the number of concepts is small or the accuracy of the concepts is low, policies fail to train, with the efficiency of training increasing with both parameters.

J Varying the Intervention Percentage

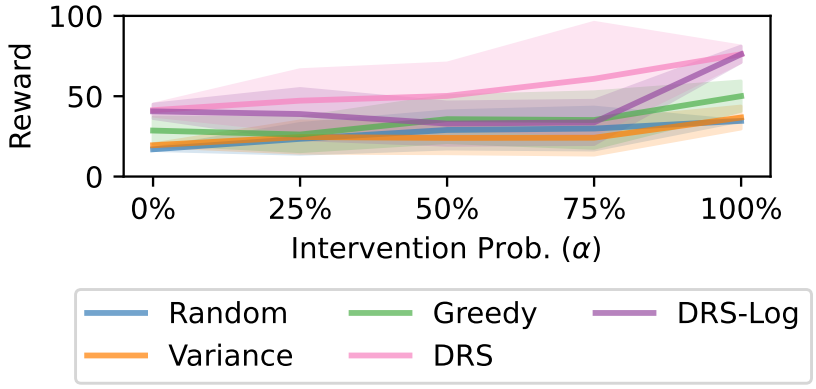

Figure 12: We assess test-time intervention performance when varying the level of intervention α in CartPole. Across all α, the DRS algorithm performs best, with steadily improving performance as α is increased.

In Figure 12, we vary α and analyze the reward in CartPole. We find that the DRS algorithm improves performance across values of α, and steadily improves as α increases. When $\alpha = 0.5$, DRS achieves 40% better than baselines and at $\alpha = 0.75$, DRS achieves a 73% better performance. DRS also improves as α increases; at $\alpha = 0.5$ DRS improves performance compared to $\alpha = 0$ by 7%, and at $\alpha = 0.75$, DRS improves by 30%.

K Concept Selection Algorithms in Supervised Settings

We extend the algorithms from Section 4 in supervised settings to use with the CUB dataset. In this setting, rollouts are replaced by labeled examples, and concept selection is performed using empirical statistics computed over the training set. The variance method selects concepts whose presence most evenly partitions the data, favoring concepts with high empirical variability. The greedy method ranks concepts by their association with the target label, selecting those with the strongest marginal predictive signal. The DRS method formulates concept selection as a weighted hitting-set problem over pairs of training examples with differing labels, selecting a fixed-size subset of concepts that maximizes deterministic coverage of label-separating pairs. Here, weights capture whether s and s' have the same label; $D_{s,s'} = 1$ if s and s' have different labels, essentially capturing a one-step difference in Q values. Essentially, this extends coverage over Q values to the supervised setting. Finally, DRS-log extends this formulation by incorporating concept accuracies into the coverage constraints, replacing hard coverage with a probabilistic notion of separation that accounts for uncertainty in concept predictions while retaining the same budgeted optimization structure.

Method	Cart Pole (k=3)	MiniGrid (k=11)	Pong (k=57)	Boxing (k=48)
Random	2.254 ± 0.349	0.901 ± 0.093	0.995 ± 0.147	2.142 ± 0.281
Greedy	1.602 ± 0.211	0.733 ± 0.104	$\mathbf{0.838 \pm 0.099}$	3.696 ± 0.741
Variance	2.113 ± 0.519	0.796 ± 0.124	1.014 ± 0.118	4.284 ± 0.883
DRS	1.703 ± 0.533	0.138 ± 0.199	0.879 ± 0.136	$\mathbf{0.248 \pm 0.122}$
DRS w/o relaxation	$\mathbf{1.492 \pm 0.219}$	$\mathbf{0.000 \pm 0.000}$	0.861 ± 0.154	$\mathbf{0.248 \pm 0.122}$

Table 2: State abstraction error $\epsilon(g)$ of the selected concept set (lower is better), mean $\pm$ std over seeds.

L State Abstraction Errors

In Table 2, we compare the state abstraction error for different algorithms. In addition to the baselines, we compare both DRS, and DRS without $\rho = 0.75$ and the relaxation of 2d, to better understand what the impact of these two relaxations is. As expected, DRS without the relaxation has the lowest state abstraction error in three of the four environments; in Pong, the large number of states leads DRS to subsample rollouts, leading Greedy to outperform DRS in state abstraction error, though with both performing comparably. In Pong and Boxing, we find that the state abstraction errors for DRS and DRS without relaxation are within 2% of each other.

M Mutual Information Baseline

In addition to the baselines in Section 5, we compare against a mutual information (MI) baseline on a smaller subset of 3 seeds, which measures the mutual information between each concept and discretized Q-values and selects the k concepts with highest mutual information. Table 3 shows that MI performs similarly to or worse than our existing baselines. In particular, MI underperforms even Variance on MiniGrid, suggesting that marginal concept-level statistics may be insufficient to identify decision-relevant concepts. In contrast, DRS jointly reasons about which state distinctions matter for the policy.

Table 3: Normalized reward for the mutual information baseline compared with existing concept selection methods.

Method	CartPole	MiniGrid
All Concepts	100.0	96.0
Random	34.6	66.6
Variance	36.7	75.1
Greedy	50.1	65.4
DRS	**76.1**	**95.9**
Mutual Info	39.4	55.1

N Selecting Concepts with Imperfect Policies

Throughout our work, we have assumed access to a perfect model π^*. In this section, we relax that assumption by training reference policies π in MiniGrid for varied numbers of timesteps. We compare the performance of two baselines (variance and greedy) and our two algorithms (DRS and DRS-log) when varying the number of gold timesteps. We do not test the random selection algorithm because it is independent of π^*.

In Figure 13, we find that the DRS and DRS-log algorithms perform well across the number of timesteps π^* is trained for. Both algorithms achieve ground-truth level performance regardless of the number of timesteps π^* is trained for. The greedy and variance algorithms perform at least 18% worse no matter how many timesteps π^* is trained for. Therefore, even with access to an approximate policy $\hat{\pi}$, the DRS and DRS-log algorithms perform well.

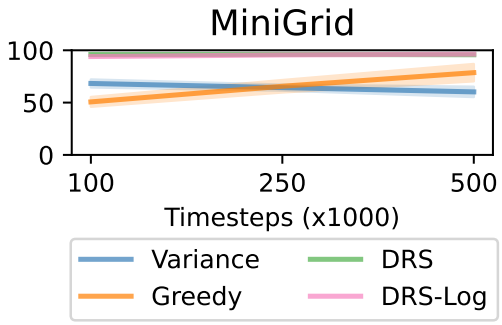

Figure 13: We vary the number of timesteps we use to train the policy π without concepts in MiniGrid. Typically, we have $\pi = \pi^*$, but here, we demonstrate that the DRS and DRS-log algorithms perform well even when π is trained for only 100k timesteps.

O PROOFS

Theorem O.1. *Optimizing the objective in Equation 1 is NP-hard, even under the following restrictions: (i) concept predictors are perfect, and (ii) the optimal action-value function Q^* is known.*

Proof. We prove the claim via a polynomial-time reduction from the *Weighted Maximum Coverage* problem.

An instance of weighted maximum coverage consists of a universe $U = \{e_1, \ldots, e_n\}$ with weights $w_i \geq 0$, a collection of sets $\mathcal{S} = \{S_1, \ldots, S_m\}$, and a budget k. The goal is to select at most k sets maximizing $\sum_{e_i \in \cup_{S_j \in \mathbf{c}} S_j} w_i$.

Given such an instance, we construct a deterministic episodic MDP. For each element e_i, introduce two one-decision states s_i^L and s_i^R (states that transition to a terminal state post-action) and an initial state s_0. From s_0, the MDP transitions uniformly to one of the states $\{s_i^L, s_i^R\}_{i=1}^n$. The action space is $\mathcal{A} = \{a_L, a_R\}$, and rewards are defined by $R(s_i^L, a_L) = w_i, \quad R(s_i^L, a_R) = 0, \qquad R(s_i^R, a_R) = w_i, \quad R(s_i^R, a_L) = 0$. Thus the optimal action-value function Q^* is known.

We define a concept bank as follows. For each set $S_j \in \mathcal{S}$, introduce a binary concept c_j such that

$$c_j(s_i^L) = 0, \quad c_j(s_i^R) = 1 \;\text{ if } e_i \in S_j,$$

and $c_j(s_i^L) = c_j(s_i^R) = 0$ otherwise.

Fix any subset of concepts $\mathbf{c}$ with $|\mathbf{c}| \leq k$, and let $\pi_{\mathbf{c}}^*$ be the optimal policy that conditions only on $g_{\mathbf{c}}$. If no selected concept distinguishes (s_i^L, s_i^R), then both states map to the all-zeros vector, as does every s_j^L; since $\sum_j w_j \geq \sum_{j:\, s_j^R \mapsto \mathbf{0}} w_j$, $\pi_{\mathbf{c}}^*$ takes a_L there, earning w_i at s_i^L and 0 at s_i^R, i.e., $w_i/2$ averaged over the pair. If the pair is distinguished by at least one selected concept, the policy can take the optimal action in each state and obtain reward w_i.

Let $C(\mathbf{c}) \subseteq U$ be the elements whose corresponding state pairs are distinguished by $\mathbf{c}$. The expected return of $\pi_{\mathbf{c}}^*$ is

$$\sum_{e_i \in C(\mathbf{c})} w_i + \sum_{e_i \notin C(\mathbf{c})} \frac{w_i}{2} = \frac{1}{2} \sum_i w_i + \frac{1}{2} \sum_{e_i \in C(\mathbf{c})} w_i.$$

Since the first term is constant, maximizing Equation 1 over $|\mathbf{c}| \leq k$ is equivalent to maximizing $\sum_{e_i \in \cup_{S_j \in \mathbf{c}} S_j} w_i$, i.e., solving weighted maximum coverage.

Therefore, a polynomial-time algorithm for optimizing Equation 1 would imply a polynomial-time algorithm for weighted maximum coverage. Since weighted maximum coverage is NP-hard, the concept selection problem is NP-hard as well. □

Proposition 3.1. *For a concept predictor $g_{\mathbf{c}}$, we have that*

$$V^{\pi^*}(s) - V^{\pi^*_{\mathbf{c}}}(s) \leq \frac{2\epsilon(g_{\mathbf{c}})}{(1-\gamma)^2}.$$

Proof. By assumption, whenever $g_{\mathbf{c}}(s) = g_{\mathbf{c}}(s')$ we have $\max_a |Q(s,a) - Q(s',a)| \leq \epsilon$. Thus, $g_{\mathbf{c}}$ induces an ϵ-approximate state abstraction in the sense of Theorem 2.1. The claimed bound on the value loss of the induced policy $\pi^*_{\mathbf{c}}$ follows directly from that theorem. □

Proposition A.1. *Let $\mathbf{c}$ and $\mathbf{c}'$ be two concept sets, and let $\epsilon(g_{\mathbf{c}}) \leq \epsilon(g_{\mathbf{c}'})$. Under Assumptions A.1 and A.2, for any intervention level $\alpha \in [0,1]$ and any accuracy vector $\delta^{(\alpha)}$ obtained by improving an α-fraction of concepts from δ^A toward $\delta^H = \mathbf{1}$, we have*

$$\mathbb{E}[V^{\pi^*_{\mathbf{c},\delta^{(\alpha)}}}(s)] \geq \mathbb{E}[V^{\pi^*_{\mathbf{c}',\delta^{(\alpha)}}}(s)].$$

Proof. By Assumption A.2, a smaller abstraction error implies a higher value under perfect concept accuracy. Therefore, under perfect intervention, $\epsilon(g_{\mathbf{c}}) \leq \epsilon(g_{\mathbf{c}'})$.

Human test-time intervention improves concept accuracy coordinate-wise, so for any α, the resulting accuracy vector $\delta^{(\alpha)}$ satisfies $\delta^{(\alpha)} \preceq \mathbf{1}$. By Assumption A.1, relative performance rankings between policies induced by $\mathbf{c}$ and $\mathbf{c}'$ are preserved as accuracy improves. Therefore, the ordering induced at perfect accuracy extends to all intermediate levels of intervention, completing the proof. □

Theorem 4.1. *Let $(\mathbf{x}^*, \mathbf{y}^*)$ be an optimal solution to Equation 2a with $\rho = 0$, and let $\mathbf{c} = \{c_j : x^*_j = 1\}$. $g_{\mathbf{c}}$ minimizes $\epsilon(g_{\mathbf{c}})$ among predictors using at most k concepts:*

$$\epsilon(g_{\mathbf{c}}) \;=\; \min_{g:|\mathbf{c}_g|\leq k} \epsilon(g).$$

Proof. By constraint 2b, the abstraction $g_{\mathbf{c}}$ uses at most k concepts.

By construction, $Y^*_{s,s'} = 0$ if and only if $g_{\mathbf{c}}(s) = g_{\mathbf{c}}(s')$ or $Y^*_{t,t'} = 0$ where $D_{t,t'} \geq D_{s,s'}$. Let the state pairs be indexed so that $D_1 \geq D_2 \geq \cdots$, and note that constraint 2d enforces $Y^*_1 \geq Y^*_2 \geq \cdots$. Since $Y^*_{s,s'} \in \{0,1\}$, there exists a threshold τ such that $Y^*_{s,s'} = 0$ for all pairs with $D_{s,s'} \leq \tau$, and $Y^*_{s,s'} = 1$ for all pairs with $D_{s,s'} > \tau$.

Therefore,

$$\epsilon(g_{\mathbf{c}}) = \max_{s,s':Y^*_{s,s'}=0} D_{s,s'} = \tau.$$

Now consider any abstraction g using at most k concepts. For g to achieve abstraction error strictly less than τ, it must separate all state pairs with $D_{s,s'} \geq \tau$, which would contradict the optimality of $(\mathbf{x}^*, \mathbf{y}^*)$. Hence, no such abstraction exists, and $g_{\mathbf{c}}$ is minimax optimal. □

Theorem 4.2. *Assume that $\max_{s,a} |\hat{Q}^\pi(s,a) - Q^\pi(s,a)| \leq \delta$. Let $(\mathbf{x}^*, \mathbf{Y}^*)$ be an optimal solution to Equation 2a constructed using $D_{s,s'}$, and let $\mathbf{c}^* = \{c_j : x^*_j = 1\}$. Let $\hat{D}_{s,s'} = \max_a |\hat{Q}^\pi(s,a) - \hat{Q}^\pi(s',a)|$, and let $(\hat{\mathbf{x}}, \hat{\mathbf{Y}})$ be an optimal solution to Equation 2a constructed using $\hat{D}_{s,s'}$, with induced abstraction $g_{\hat{\mathbf{c}}}$. Then for all $s \in \mathcal{S}$*

$$V^{\pi_{\mathbf{c}^*}}(s) - V^{\pi_{\hat{\mathbf{c}}}}(s) \leq \frac{2(\epsilon_{Q^\pi}(g_{\mathbf{c}^*}) + 4\delta)}{(1-\gamma)^2}.$$

Proof. Since $\|\hat{Q} - Q^{\pi^*}\|_\infty \leq \delta$, for all s, s' we have

$$|\hat{D}_{s,s'} - D_{s,s'}| \tag{3}$$

$$= \left|\max_a |\hat{Q}(s,a) - \hat{Q}(s',a)| - \max_a |Q(s,a) - Q(s',a)|\right| \tag{4}$$

$$\leq 2\delta. \tag{5}$$

Here, this holds because of the Lipschitzness of the max operator and triangle inequality.

Fix any abstraction g. Then

$$\left| \max_{s,s':g(s)=g(s')} \hat{D}_{s,s'} - \max_{s,s':g(s)=g(s')} D_{s,s'} \right| \leq 2\delta.$$

In particular,

$$\epsilon_{\hat{Q}}(g) \leq \epsilon_Q(g) + 2\delta.$$

Let $g_{\mathbf{c}^*}$ be the abstraction minimizing $\epsilon_Q(g)$, and $g_{\hat{\mathbf{c}}}$ be the abstraction minimizing $\epsilon_{\hat{Q}}(g)$. By optimality of $g_{\hat{\mathbf{c}}}$ under $\hat{D}$,

$$\epsilon_{\hat{Q}}(g_{\hat{\mathbf{c}}}) \leq \epsilon_{\hat{Q}}(g_{\mathbf{c}^*}) \leq \epsilon_Q(g_{\mathbf{c}^*}) + 2\delta.$$

Applying the bound in the opposite direction yields

$$\epsilon_Q(g_{\hat{\mathbf{c}}}) \leq \epsilon_{\hat{Q}}(g_{\hat{\mathbf{c}}}) + 2\delta \leq (\epsilon^* + 2\delta) + 2\delta = \epsilon^* + 4\delta.$$

The value-function bound then follows directly from Proposition 3.1. □

Lemma 4.1. *Suppose that for each concept i, an adversary may apply a perturbation $f^{(T_i)}$ that flips the value of c_i on a subset $T_i \subseteq \mathcal{S}$ with $|T_i|/|\mathcal{S}| \leq 1 - \delta_i$. Then for any $g_{\mathbf{c}}$ with $\delta_i \leq 1 - \frac{1}{|\mathcal{S}|}$, there exists a collection of perturbations $f = \{f_{c_1}^{T_1}, \ldots, f_{c_k}^{T_k}\}$ such that $\epsilon(f(g_{\mathbf{c}})) = \max_{s,s'} D_{s,s'}$.*

Proof. Since $\delta_i < 1 - \frac{1}{\mathcal{S}}$ for all i, we may choose $T_i = \{s\}$ for any state $s \in \mathcal{S}$. Let $(s^*, s'^*) \in \arg\max_{s,s'} D_{s,s'}$, and define

$$P = \{i : c_i(s^*) \neq c_i(s'^*)\}.$$

For each $i \in P$, let $T_i = \{s^*\}$ and apply $f^{(T_i)}$, which flips the value of c_i at s^*; for $i \notin P$, let $f^{(T_i)}$ be the identity. After applying all perturbations, we have

$$f(g_{\mathbf{c}})(s^*) = f(g_{\mathbf{c}})(s'^*).$$

Since (s^*, s'^*) maximizes $D_{s,s'}$, it follows that

$$\epsilon(f(g_{\mathbf{c}})) = \max_{s,s'} D_{s,s'}.$$

□