Learning Markov State Abstractions for Deep Reinforcement Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

The fundamental assumption of reinforcement learning in Markov decision processes (MDPs) is that the relevant decision process is, in fact, Markov. However, when MDPs have rich observations, agents typically learn by way of an abstract state representation, and such representations are not guaranteed to preserve the Markov property. We introduce a novel set of conditions and prove that they are sufficient for learning a Markov abstract state representation. We then describe a practical training procedure that combines inverse model estimation and temporal contrastive learning to learn an abstraction that approximately satisfies these conditions. Our novel training objective is compatible with both online and offline training: it does not require a reward signal, but agents can capitalize on reward information when available. We empirically evaluate our approach on a visual gridworld domain and a set of continuous control benchmarks. Our approach learns representations that capture the underlying structure of the domain and lead to improved sample efficiency over state-of-the-art deep reinforcement learning with visual features—often matching or exceeding the performance achieved with hand-designed compact state information.

1 Introduction

Reinforcement learning (RL) in Markov decision processes with rich observations requires a suitable state representation. Typically, such representations are learned implicitly as a byproduct of doing deep RL. However, in domains where precise and succinct "expert" state information is available, an agent trained on such expert state features usually outperforms an agent trained on rich observations. Recent work has sought to close this "representation gap" by incorporating a wide range of representation-learning objectives, including pixel reconstruction (Yarats et al., 2019), pixel prediction (Kaiser et al., 2020), abstract transition model estimation (Gelada et al., 2019), inverse model estimation (Pathak et al., 2017), bisimulation metrics (Zhang et al., 2021), contrastive learning (Laskin et al., 2020b), and data augmentation (Laskin et al., 2020a), as well as combinations of these objectives (Shelhamer et al., 2016; Ha & Schmidhuber, 2018). These approaches collectively help the agent learn representations with various desirable properties.

Perhaps the most obvious property to incentivize in a state representation is the Markov property, which holds if and only if the representation contains enough information to accurately characterize the rewards and transition dynamics of the decision process. Markov decision processes (MDPs) have this property by definition, and many reinforcement learning algorithms rely on having Markov states. Conversely, agents with non-Markov abstract state representations may struggle to learn.

If reward information is available, an agent can use it to help ensure that the learned representations faithfully simulate the (Markov) dynamics of the environment. Similarly, if the agent can reconstruct or predict raw environment observations from its learned representations, then all available informa-

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

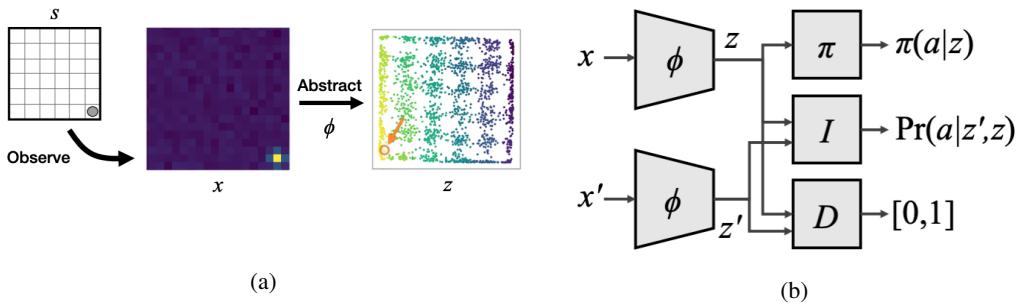


Figure 1: (a) A 6×6 visual gridworld domain where an abstraction function ϕ maps each high-dimensional observed state x to a lower-dimensional abstract state z (orange circle). (b) The Markov abstraction training architecture. A shared encoder ϕ maps ground states x, x' to abstract states z, z', which are inputs to an inverse dynamics model I and a constrastive model D that discriminates between real and fake state transitions. The agent's policy depends only on the abstract state.

tion is preserved (along with the Markov property as well). However, when rewards are sparse or non-existent, or observations are sufficiently complex, these approaches are impractical.

We consider an alternative approach to learning Markov abstract state representations. We begin by introducing a novel set of conditions that are sufficient for an abstraction to retain the Markov property. We next show that these conditions are approximately satisfied using a specific combination of two existing representation learning objectives, neither of which requires reward information or observation reconstruction/prediction: inverse model estimation, which predicts the action distribution that explains two consecutive states, and temporal contrastive learning, which determines whether two given states were in fact consecutive.

We evaluate this combined learning objective for both online and offline representation learning. We show empirically that the offline approach learns an abstract representation of a visual gridworld that captures the underlying structure of the domain and subsequently enables efficient learning. We then evaluate our objective in the online training setting by incorporating it as an auxiliary objective function during RL, where it leads to improved learning performance over the state of the art on a set of continuous control benchmarks.

2 Background

A Markov decision process M consists of sets of states X and actions A, reward function $R : X \times A \times X \to \mathbb{R}$, transition dynamics $T : X \times A \to \Pr(X)$, and discount factor γ. For our theoretical results, we assume M is a Block MDP (Du et al., 2019) whose behavior is governed by a much smaller (but unobserved) set of states S, and where X is a rich observation generated by a noisy sensor function $\sigma : S \to \Pr(X)$. Block MDPs conveniently avoid potential issues arising from partial observability by assuming that each observation uniquely identifies the unobserved state that generated it. In other words, there exists a perfect "inverse sensor" function $\sigma^{-1}(x) \mapsto s$, which means the observations themselves form a Markov state representation, as we define below. Note that S, σ, and σ^{-1} are all unknown to the agent.

Definition 1 (Markov Property). *A decision process $M = (X, A, R, T, \gamma)$ and its state representation X are* Markov *if and only if, for all $a \in A$, $x \in X$, $k \geq 1$: $T^{(k)}\left(x_{t+1} | \{a_{t-i}, x_{t-i}\}_{i=0}^{k}\right) = T(x_{t+1}|a_t, x_t)$ and $R^{(k)}\left(x_{t+1}, \{a_{t-i}, x_{t-i}\}_{i=0}^{k}\right) = R(x_{t+1}, a_t, x_t)$.*

The superscript (k) denotes that the function is being conditioned on k additional steps of history.

The Markov property means that each state X it is a sufficient statistic for predicting the next state and expected reward, for any action the agent might select. Technically, each state must also be sufficient for determining the set of actions available to the agent in that state, but here we assume, as is common, that every action is available in every state.

The behavior of an RL agent is typically determined by a (Markov) policy $\pi : X \to \Pr(A)$, and each policy induces value function $V^\pi : X \to \mathbb{R}$, which is defined as the expected sum of future

discounted rewards starting from a given state and following the policy π thereafter. The agent's objective is to learn an optimal policy π^* that maximizes value at every state. Note that the assumption that the optimal policy is stationary and Markov—that it only depends on state—is appropriate only if the decision process itself is Markov; almost all RL algorithms simply assume this to be true.

2.1 State Abstraction

To support decision making when X is too high-dimensional for tractable learning, we turn to state abstraction. Our objective is to find an abstraction function $\phi : X \to Z$ mapping each x to an abstract state representation $z = \phi(x)$, with the hope that learning is tractable in Z (see Figure 1a). We refer to X as *ground states* (and M as the ground MDP), to reflect that they are grounded in the true environment. We interchangeably refer to $z \in Z$ as an abstract state or an abstract representation.

An abstraction $\phi : X \to Z$, when applied to an MDP M, induces a new abstract decision process $M_\phi = (Z, A, T^\pi_{\phi,t}, R^\pi_{\phi,t}, \gamma)$, whose dynamics may depend on the current timestep t or the agent's behavior policy π, and, crucially, which is not necessarily Markov. Consider the following example.

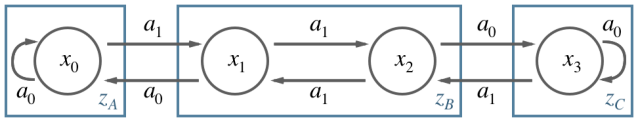


Figure 2: An MDP and a non-Markov abstraction.

The figure above depicts a four-state, two-action MDP, and an abstraction ϕ where $\phi(x_1) = \phi(x_2) = z_B$. It is common to view state abstraction as aggregating or partitioning ground states into abstract states in this way (Li et al., 2006). Generally this involves choosing a fixed weighting scheme $w(x)$ to express how much each ground state x contributes to its abstract state z, where the weights sum to 1 for the set of x in each z. We can then define the abstract transition dynamics T_ϕ as a w-weighted sum of the ground dynamics (and similarly for reward): $T_\phi(z'|a, z) = \sum_{x' \in z'} \sum_{x \in z} T(x'|a, x)w(x)$. A natural choice for $w(x)$ is to use the ground-state visitation frequencies. For example, if the agent selects actions uniformly at random, this leads to $w(x_0) = w(x_3) = 1$ and $w(x_1) = w(x_2) = 0.5$.

In this formulation, the abstract decision process is assumed to be Markov by construction, and T_ϕ and R_ϕ are assumed not to depend on the policy or timestep. But this is an oversimplification. The abstract transition dynamics are *not* Markov; they change depending on how much history is conditioned on: $T_\phi(z_A|a_t = a_0, z_t = z_B) = 0.5$, whereas $\Pr(z_A|a_t = a_0, z_t = z_B, a_{t-1} = a_1, z_{t-1} = z_A) = 1$. By contrast, the ground MDP's dynamics are fully deterministic (and Markov). Clearly, if we define the abstract MDP in this way, it may not match the behavior of the original MDP.[1] Even worse, if the agent's policy changes—such as during learning—this discrepancy can cause RL algorithms with bounded sample complexity in the ground MDP to make an arbitrarily large number of mistakes in the abstract MDP (Abel et al., 2018).

The solution is to allow $w(x)$ to vary such that it always reflects the correct ground-state frequencies. In essence, we will replace $w(x)$ with a belief distribution $B_\phi(x|z)$ (defined in Sec. 4) that can be conditioned on any amount of history, and which is policy-dependent and possibly non-stationary.[2]

Definition 2 (Markov Abstraction). *Given an MDP $M = (X, A, R, T, \gamma)$, initial state distribution P_0, and policy class Π_C, a state abstraction $\phi : X \to Z$ is* Markov *if and only if for any policy $\pi \in \Pi_C$, ϕ induces a belief distribution B_ϕ such that for all $x \in X$, $z \in Z$, $a \in A$, and $k \geq 1$:* $B^{(k)}_\phi \left(x|z_t, \{a_{t-i}, z_{t-i}\}_{i=1}^k\right) = B_\phi \left(x|z_t\right)$.

Since our goal is to support effective decision making in M_ϕ, we are mainly concerned with the policy class Π_ϕ, the set of policies with the same behavior for all ground states in each abstract state:

$$\Pi_\phi := \left\{\pi : \left(\phi(x_1) = \phi(x_2)\right) \implies \left(\pi(a|x_1) = \pi(a|x_2)\right), \forall\, a \in A; x_1, x_2 \in X\right\}. \qquad (1)$$

[1] Abel et al. (2018) presented a three-state chain MDP where they made a similar observation.

[2] This formulation involving belief distributions is similar to a POMDP-based approach by Bai et al. (2016). However, while we use B_ϕ for our theoretical results, our method does not require estimating or maintaining a belief distribution in order to learn a Markov abstraction, and the abstract process we construct is still an MDP.

We describe precisely how a policy and an abstraction induce a belief distribution in Section 4. Then we introduce our main theoretical result, which is to derive conditions under which ϕ and M_ϕ are Markov. First, we discuss related work on learning abstract state representations.

3 Related Work

The idea of learning Markov state abstractions is related to the concepts of bisimulation (Dean & Givan, 1997), the strictest types of state aggregation discussed in Li et al. (2006), where ground states are equivalent if they have exactly the same expected reward and transition dynamics. Preserving the Markov property is a prerequisite for a bisimulation abstraction, since the abstraction must also preserve the (Markov) ground-state transition dynamics. Bisimulation-based abstraction is appealing because, by definition, it leads to high-fidelity representations. But bisimulation is also very restrictive, because it requires ϕ to be Markov for *any* policy (rather than just those in Π_ϕ).

Subsequent work on approximate MDP homomorphisms (Ravindran & Barto, 2004) and bisimulation metrics (Ferns et al., 2004, 2011) relaxed these strict assumptions and allowed ground states to have varying degrees of "bisimilarity." While such approaches have historically been computationally expensive and difficult to scale, recent work has started to change that (Castro, 2020; Lehnert & Littman, 2020; Van der Pol et al., 2020; Biza et al., 2021). Two recent algorithms in particular, DeepMDP (Gelada et al., 2019) and Deep Bisimulation for Control (DBC) (Zhang et al., 2021), learn approximate bisimulation abstractions by training the abstraction end-to-end with an abstract transition model and reward function. This is a rather straightforward way to learn Markov abstract state representations since it effectively encodes Def. 1 as a loss function.

One drawback of bisimulation-based methods is that learning an accurate model can be challenging and typically requires restrictive modeling assumptions, such as deterministic, linear, or Gaussian transition dynamics. Bisimulation methods may also struggle if rewards are sparse or if the abstraction must be learned without access to rewards. Jointly training an abstraction ϕ with only the transition model $\widehat{T}(\phi(x), a) \approx \phi(x')$ can easily lead to a trivial abstraction like $\phi(x) \mapsto 0$ for all x, since ϕ produces both the inputs and outputs for the model. Our approach to learning a Markov abstraction avoids this type of representation collapse without learning a forward model, and is less restrictive than bisimulation, since it is compatible with both reward-free and reward-informed settings.

Pixel prediction (Watter et al., 2015; Song et al., 2016; Kaiser et al., 2020) mitigates representation collapse by forcing the abstract state to be sufficient not just for predicting future *abstract* states, but also future *ground states*. Unfortunately, in stochastic domains, this comes with the challenging task of density estimation over the ground state space, and as a result, performance is about on-par with end-to-end deep RL (Van Hasselt et al., 2019). Moreover, both pixel prediction and the related task of pixel reconstruction (Mattner et al., 2012; Finn et al., 2016; Higgins et al., 2017; Corneil et al., 2018; Ha & Schmidhuber, 2018; Yarats et al., 2019; Hafner et al., 2020; Lee et al., 2020) are misaligned with the fundamental goal of state abstraction. These approaches train models to perfectly reproduce the relevant ground state, ergo the abstract state must effectively throw away no information. By contrast, the objective of state abstraction is to throw away *as much information as possible*, while preserving only what is necessary for decision making. Provided the abstraction is Markov and faithfully simulates the ground-state dynamics, we can safely discard the rest of the observation.

As an alternative to (or in addition to) learning a forward model, it is sometimes beneficial to learn an inverse model. An inverse model $I(a|x', x)$ predicts the distribution over actions that could have resulted in a transition between a given pair of states. Inverse models have been used for improving generalization from simulation to real-world problems (Christiano et al., 2016), enabling effective robot motion planning (Agrawal et al., 2016), defining intrinsic reward bonuses for exploration (Pathak et al., 2017; Choi et al., 2019), and (alongside pixel reconstruction) decoupling representation learning from rewards (Zhang et al., 2018). But while inverse models often help with representation learning, we show in Section 4.2 that they are insufficient for ensuring a Markov abstraction.

Since the main barrier to effective next-state prediction is learning an accurate forward model, a compelling alternative is contrastive learning (Gutmann & Hyvärinen, 2010), which sidesteps the prediction problem and instead simply aims to decide whether a particular state, or sequence of states, came from one distribution or another. Contrastive loss objectives typically aim to distinguish either sequential states from non-sequential ones (Shelhamer et al., 2016; Anand et al., 2019; Stooke et al., 2020), real states from predicted ones (Van den Oord et al., 2018), or determine whether

two augmented views came from the same or different observations (Laskin et al., 2020b). But while contrastive methods learn representations that in some cases lead to empirically substantial improvements in learning performance, none has explicitly addressed the question of whether the resulting state abstractions actually preserve the Markov property.

One contrastive approach which turns out to be closely related to Markov abstraction is Misra et al.'s (2020) HOMER algorithm, and the corresponding notion of *kinematic inseparability* (KI) abstractions. Two states x'_1 and x'_2 are defined to be kinematically inseparable if $\Pr(x, a|x'_1) = \Pr(x, a|x'_2)$ and $T(x''|a, x'_1) = T(x''|a, x'_2)$ (which the authors call "backwards" and "forwards" KI, respectively). The idea behind KI abstractions is that unless two states can be distinguished from each other—by either their backward or forward dynamics—they ought to be treated as the same abstract state. In HOMER, abstract states are necessarily discrete, whereas our approach works for either discrete or continuous latent state spaces. The KI conditions are also slightly stronger than the ones we describe in Section 4, although when we convert our conditions to a training objective in Section 5, we actually recover an alternate form of the KI conditions, which helps to prevent representation collapse. (For a more detailed discussion about KI and Markov abstractions, see Appendix D.)

Since our approach does not require any reward information and is agnostic as to the underlying RL algorithm, it would naturally complement exploration methods designed for sparse reward problems (Pathak et al., 2017; Burda et al., 2018). Exploration helps to ensure that the experiences used to learn an abstract representation cover as much of the ground MDP's state space as possible. In algorithms like HOMER (above) and the more recent Proto-RL (Yarats et al., 2021), the exploration and representation learning objectives are intertwined, whereas our approach is, in principle, compatible with any exploration algorithm. Here we focus solely on the problem of learning Markov state abstractions and view exploration as an exciting direction for future work.

In summary, we are the first to show that, without forward model estimation, pixel prediction/reconstruction, or dependence on reward, the specific combination of inverse model estimation and contrastive learning that we introduce in the next section is sufficient to learn a Markov abstraction.

4 Markov State Abstractions

Recall that for a state representation to be Markov (whether ground or abstract), it must be a sufficient statistic for predicting the next state and expected reward, for any action the agent selects. The ground state representation of an MDP is Markov by definition, but learned state abstractions have no such guarantees. In a moment, we will define the abstract decision process M_ϕ using belief distribution B_ϕ, such that M_ϕ faithfully simulates the ground MDP's dynamics, and then provide conditions under which it is Markov, but first we must make B_ϕ's policy dependence and non-stationarity explicit:

$$B^\pi_{\phi,t}(x|z) := \frac{\mathbb{1}[\phi(x) = z]\, P^\pi_t(x)}{\sum_{\tilde{x}\in z} P^\pi_t(\tilde{x})}, \qquad P^\pi_t(x) := \sum_{a\in A}\sum_{\tilde{x}\in X} T(x|a,\tilde{x})\pi_{t-1}(a|\tilde{x})P^\pi_{t-1}(\tilde{x}), \tag{2}$$

for $t \geq 1$, where $\mathbb{1}[\cdot]$ denotes the indicator function, π is the agent's (possibly non-stationary) behavior policy, and P_0 is an arbitrary initial state distribution. Note that P^π_t and $B^\pi_{\phi,t}$ may still be non-stationary even if π is stationary.[3] Given these definitions, we can define the abstract transitions and rewards for the policy class Π_ϕ (see Eqn. (1)) as follows:[4]

$$T^\pi_{\phi,t}(z'|a,z) = \sum_{x'\in z'}\sum_{x\in z} T(x'|a,x)B^\pi_{\phi,t}(x|z), \tag{3}$$

$$R^\pi_{\phi,t}(z'|a,z) = \sum_{x'\in z'}\sum_{x\in z} \frac{R(x',a,x)T(x'|a,x)}{T^\pi_{\phi,t}(z'|a,z)} B^\pi_{\phi,t}(x|z). \tag{4}$$

In the special case where the belief distribution is stationary and policy-independent (and if we interpret rewards as being defined over state-action pairs), $B^\pi_{\phi,t}(x|z)$ is equivalent to the fixed weighting function $w(x)$ of Li et al. (2006).

[3] This can happen, for example, when the policy induces either a Markov chain that does not have a stationary distribution, or one whose stationary distribution is different from P_0.

[4] For the more general definitions that support arbitrary policies, see Appendix B. This section closely follows Hutter (2016), except here we consider distributions over ground states, rather than full histories.

In order to apply Definition 1 and determine whether M_ϕ is Markov, we generalize $B^\pi_{\phi,t}$ to a k-step belief distribution, for $k \geq 1$, by conditioning (2) on additional history:

$$B^{\pi(k)}_{\phi,t}\left(x_t | z_t, \{a_{t-i}, z_{t-i}\}_{i=1}^k\right) := \frac{\mathbb{1}[\phi(x_t) = z_t] \sum_{x_{t-1} \in X} T(x_t|a_{t-1}, x_{t-1})\, B^{\pi(k-1)}_{\phi,t}\left(x_{t-1} \mid z_{t-1}, \{a_{t-i}, z_{t-i}\}_{i=2}^k\right)}{\sum_{\tilde{x}_t \in z_t} \sum_{\tilde{x}_{t-1} \in z_{t-1}} T(\tilde{x}_t|a_{t-1}, \tilde{x}_{t-1})\, B^{\pi(k-1)}_{\phi,t}\left(\tilde{x}_{t-1} \mid z_{t-1}, \{a_{t-i}, z_{t-i}\}_{i=2}^k\right)}, \quad (5)$$

where $B^{\pi(0)}_{\phi,t} := B^\pi_{\phi,t}$. We then substitute (5) for $B^\pi_{\phi,t}$ in Equations (3) and (4) to produce $T^{\pi(k)}_{\phi,t}$ and $R^{\pi(k)}_{\phi,t}$, the extended-history versions of $T^\pi_{\phi,t}$ and $R^\pi_{\phi,t}$. Next we describe sufficient conditions for ensuring ϕ and M_ϕ are Markov.

4.1 Sufficient Conditions for a Markov Abstraction

The strictly necessary conditions for ensuring an abstraction ϕ is Markov over its policy class Π_ϕ depend on T and R, which are typically unknown and hard to estimate due to X's high-dimensionality. However, we can still find sufficient conditions without explicitly knowing T and R. To do this, we require that two quantities are equivalent in M and M_ϕ: the inverse dynamics model, and a density ratio that we define below. The inverse dynamics model $I^\pi_t(a|x', x)$ is defined in terms of the transition function $T(x'|a, x)$ and expected next-state dynamics $P^\pi_t(x'|x)$ via Bayes' theorem: $I^\pi_t(a|x', x) := \frac{T(x'|a,x)\pi_t(a|x)}{P^\pi_t(x'|x)}$, where $P^\pi_t(x'|x) = \sum_{\tilde{a} \in A} T(x'|\tilde{a}, x)\pi_t(\tilde{a}|x)$. The same is true of the abstract inverse model $I^\pi_{\phi,t}(a|z', z)$.

Theorem 1. *If $\phi : X \to Z$ is an abstraction of MDP $M = (X, A, R, T, \gamma)$ such that for any policy π in the policy class Π_ϕ, the following conditions hold for every timestep t:*

1. ***Inverse Model.*** *The ground and abstract inverse models are equal: $I^\pi_{\phi,t}(a|z', z) = I^\pi_t(a|x', x)$, for all $a \in A$; $z, z' \in Z$; $x, x' \in X$, such that $\phi(x) = z$ and $\phi(x') = z'$.*

2. ***Density Ratio.*** *The ground and abstract next-state density ratios are equal, when conditioned on the same abstract state: $\frac{P^\pi_t(z'|z)}{P^\pi_t(z')} = \frac{P^\pi_t(x'|z)}{P^\pi_t(x')}$, for all $z, z' \in Z$; $x' \in X$, such that $\phi(x') = z'$, where $P^\pi_t(x'|z) = \sum_{\tilde{x} \in X} P^\pi_t(x'|\tilde{x}) B^\pi_{\phi,t}(\tilde{x}|z)$, and $P^\pi_t(z') = \sum_{\tilde{x}' \in X} P^\pi_t(x') B^\pi_{\phi,t}(\tilde{x}'|z')$.*

Then, ϕ is a Markov abstraction.

Corollary 1.1. *If $\phi : X \to Z$ is a Markov abstraction of MDP $M = (X, A, R, T, \gamma)$ over the policy class Π_ϕ, then the abstract decision process $M_\phi = (Z, A, R^\pi_{\phi,t}, T^\pi_{\phi,t}, \gamma)$ is also Markov.*

We defer all proofs to Appendix C.

Theorem 1 describes a pair of conditions under which ϕ results in a Markov state representation. Of course, the conditions themselves do not constitute a training objective—we can only use them to confirm an abstraction is Markov. In Section 5 we adapt these conditions into a practical representation learning objective that is differentiable and suitable for learning ϕ using deep neural networks. First, we show why the Inverse Model condition alone is insufficient.

4.2 An Inverse Model Counterexample

The example MDP in Figure 2 also demonstrates why the Inverse Model condition alone is insufficient to produce a Markov abstraction. Note that any valid transition between two ground states uniquely identifies the selected action. The same is true for abstract states since the only way to reach z_B is via action a_1, and the only way to leave is action a_0. Therefore, the abstraction satisfies the Inverse Model condition for any policy. However, as noted in Section 2.1, conditioning on additional history changes the abstract transition probabilities, and thus the Inverse Model condition is not sufficient for an abstraction to be Markov. In fact, we show in Appendix C.2 that, given the Inverse Model condition, the Density Ratio condition is actually *necessary* for a Markov abstraction.

5 Training a Markov Abstraction

We now present a set of training objectives for approximately satisfying the conditions of Theorem 1. Since the theorem applies for the policy class Π_ϕ induced by the abstraction, we restrict the policy by defining π as a mapping from $Z \to \Pr(A)$, rather than from $X \to \Pr(A)$. In cases where π is defined implicitly via the value function, we ensure that the latter is defined over abstract states.

Inverse Models. To ensure the ground and abstract inverse models are equal, we consider a batch of experiences $\{(x_i, a_i, x_i')\}_{i=1}^N$, encode ground states with ϕ, and jointly train a model $f(a|\phi(x'), \phi(x); \theta_f)$ to predict a distribution over actions, with a_i as the label. This can be achieved by minimizing a cross-entropy loss, for either discrete or continuous action spaces:

$$\mathcal{L}_{Inv} := -\frac{1}{N}\sum_{i=1}^{N} \log f(a = a_i|\phi(x_i'), \phi(x_i); \theta_f).$$

Note that because the policy class is restricted to Π_ϕ, if the policy is stationary and deterministic, then $I^\pi_{\phi,t}(a|z', z) = \pi_\phi(a|z) = \pi(a|x) = I^\pi_t(a|x', x)$ and the Inverse Models condition is satisfied trivially. Thus we expect $\mathcal{L}_{Inv}$ to be most useful for representation learning when the policy has high entropy or is changing rapidly, such as during early training.

Density Ratios. The second condition, namely that $\frac{P^\pi_{\phi,t}(z'|z)}{P^\pi_{\phi,t}(z')} = \frac{P^\pi_t(x'|z)}{P^\pi_t(x')}$, means we can distinguish conditional samples from marginal samples equally well for abstract states or ground states. We can encode this objective as a type of contrastive loss. Suppose we have a dataset consisting of samples $X_c = \{{x_c'}^{(i)}\}_{i=1}^{n_c}$ drawn from the conditional distribution $\Pr(x'|x)$, and samples $X_m = \{{x_m'}^{(j)}\}_{j=1}^{n_m}$ drawn from the marginal $\Pr(x')$. These samples can come from any batch of (x, x') experience pairs, where the conditioning states x are randomly shuffled for the marginal samples. We assign label $y = 1$ to samples from X_c and $y = 0$ to samples from X_m, and our goal is to predict the label associated with each sample. To construct an estimator, we follow the derivation of Tiao (2017) by renaming the two distributions $p(x'|y = 1) := \Pr(x'|z)$ and $p(x'|y = 0) := \Pr(x')$ and rewriting the density ratio $\delta(x') := \frac{\Pr(x'|x)}{\Pr(x')}$ as follows:

$$\delta(x') = \frac{p(x'|y=1)}{p(x'|y=0)} = \frac{p(y=1|x')p(x')}{p(y=1)}\frac{p(y=0)}{p(y=0|x')p(x')} = \frac{n_m}{(n_m+n_c)}\frac{(n_m+n_c)}{n_c}\frac{p(y=1|x')}{p(y=0|x')} = \frac{n_m}{n_c}\frac{p(y=1|x')}{1-p(y=1|x')}. \tag{6}$$

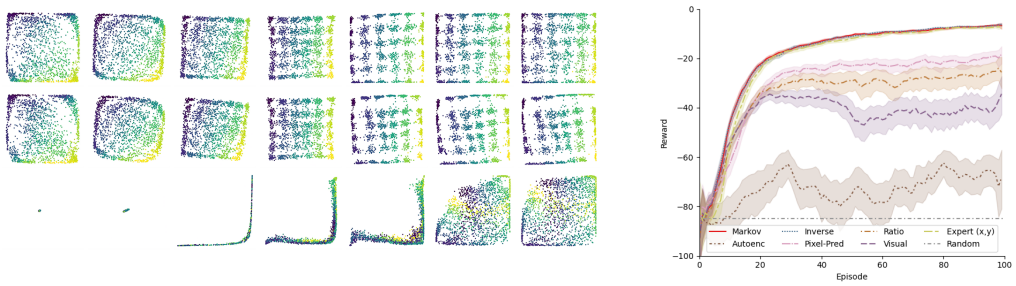
We then train a classifier $g(x', x; \theta_g)$ to estimate $p(y = 1|x')$. To estimate $\delta(x')$, we could substitute g for $p(y = 1|x')$ in (6); however, to satisfy the Density Ratio condition, we need only ensure $\delta_\phi(z') = \mathbb{E}_{B_\phi}[\delta(x')]$. We therefore repeat the derivation for abstract states and obtain $\delta_\phi(z') := \frac{\Pr(z'|z)}{\Pr(z')} = \frac{n_m}{n_c}\frac{q(y=1|z')}{1-q(y=1|z')}$, where q is our renamed distribution. Next, we modify our classifier to train it jointly with the abstraction ϕ, minimizing the cross-entropy between predictions and labels y_i:

$$\mathcal{L}_{Ratio} := -\frac{1}{n_c + n_m}\sum_{i=1}^{n_c+n_m} \log g(y = y_i|\phi(x_i'), \phi(x_i); \theta_g).$$

In doing so, we ensure g approaches q and p simultaneously, which drives $\delta_\phi(z') \to \delta(x')$. Note that this is stronger than the original condition, which only required the ratios to be equal in expectation. As mentioned in Section 3, this stronger objective is in fact related to kinematic inseparability abstractions and helps avoid representation collapse (see discussion below, and also Appendix D).

Markov Abstraction Objective. We generate a batch of experiences using any policy that is a member of Π_ϕ, such as a uniform random policy, then train ϕ end-to-end while minimizing a weighted combination of the inverse-model and contrastive losses: $\mathcal{L}_{Markov} := \alpha\mathcal{L}_{Inv} + \beta\mathcal{L}_{Ratio}$.

The inverse and contrastive losses avoid the common pitfall of learning a trivial abstraction like $\phi(x) \mapsto 0$, without requiring reward information or ground state reconstruction/prediction. While such a trivial abstraction would technically satisfy the Density Ratio condition, it does not minimize $\mathcal{L}_{Markov}$, since $\phi(x)$ contains no useful information for predicting actions or distinguishing $x' \sim \Pr(x'|z)$ versus $x' \sim \Pr(x')$. This partially explains why inverse models and temporal contrastive losses have been such effective representation learning objectives (Shelhamer et al., 2016; Pathak et al., 2017; Van den Oord et al., 2018; Anand et al., 2019; Misra et al., 2020; Stooke et al., 2020).



(a) (b)

Figure 3: (a) Visualization of learning progress at selected times (left to right) of a 2-D state abstraction for the 6×6 visual gridworld domain: (top row) $\mathcal{L}_{Markov}$; (middle row) $\mathcal{L}_{Inv}$ only; (bottom row) $\mathcal{L}_{Ratio}$ only. Color denotes ground-truth (x, y) position, which is not shown to the agent. (b) Mean episode reward for the visual gridworld navigation task. Markov abstractions significantly outperform end-to-end training with visual inputs, and match the performance of the expert (x, y) position features. (300 seeds; 5-point moving average; shaded regions denote 95% confidence intervals.)

6 Offline Abstraction Learning for Visual Gridworlds

First, we evaluate our approach for learning an abstraction offline for a visual gridworld domain.[5] Each discrete (x, y) position in the 6×6 gridworld is mapped to a noisy image like the one in Figure 1a. We emphasize that the agent only sees these images; it does not have access to the ground-truth (x, y) position. The agent gathers a batch of experiences in a version of the gridworld with *no rewards or terminal states*, using a uniform random exploration policy over the four directional actions.

These experiences are then used offline to train an abstraction function ϕ_{Markov}, by minimizing $\mathcal{L}_{Markov}$ (with $\alpha = \beta = 1$). We visualize the learned 2-D abstract state space in Figure 3a (top row). We compare against ablations that train with only $\mathcal{L}_{Inv}$ or $\mathcal{L}_{Ratio}$, as well as against two baselines that we train via pixel prediction and reconstruction, respectively. We observe that ϕ_{Markov} and ϕ_{Inv} cluster the noisy observations and recover the 6×6 grid structure, whereas the others do not generally have an obvious interpretation. The ϕ_{Inv} representation usually separates the edges and corner states from the central states, since the self-loop transitions at edge and corner states lead to different action distributions. We also observed that ϕ_{Ratio} and $\phi_{Autoenc}$ frequently failed to converge (see Appendix G for more visualizations).

Next we froze these abstraction functions and used them to map images to abstract states while training DQN (Mnih et al., 2015) on the resulting features. We measured the learning performance of each pre-trained abstraction, as well as that of end-to-end DQN with no pretraining. We plot learning curves in Figure 3b. For reference, we also include learning curves for a uniform random policy and DQN trained on ground-truth (x, y) position with no abstraction.

Markov abstractions match the performance of ground-truth position, and beat every other learned representation except ϕ_{Inv}, which performs similarly, although—as discussed in Section 4.2—the Inverse Model condition alone does not necessarily produce Markov abstractions.

7 Online Abstraction Learning for Continuous Control

Next, we evaluate our approach in an online setting and train an agent on a collection of six image-based, continuous control tasks from the DeepMind Control Suite (Tassa et al., 2020). Our training objective is agnostic about the underlying RL algorithm, so we use as our baseline the state-of-the-art technique that combines Soft Actor-Critic (SAC) (Haarnoja et al., 2018) with random data augmentation (RAD) (Laskin et al., 2020a). We train our model in two phases: first we initialize the replay buffer with experiences from a uniform random policy and pretrain a Markov abstraction on those experiences for 100K steps *with the reward information removed*; then we continue to train $\mathcal{L}_{Markov}$ as an auxiliary task alongside the traditional RAD reinforcement learning objective.

[5] Code included in supplementary materials.

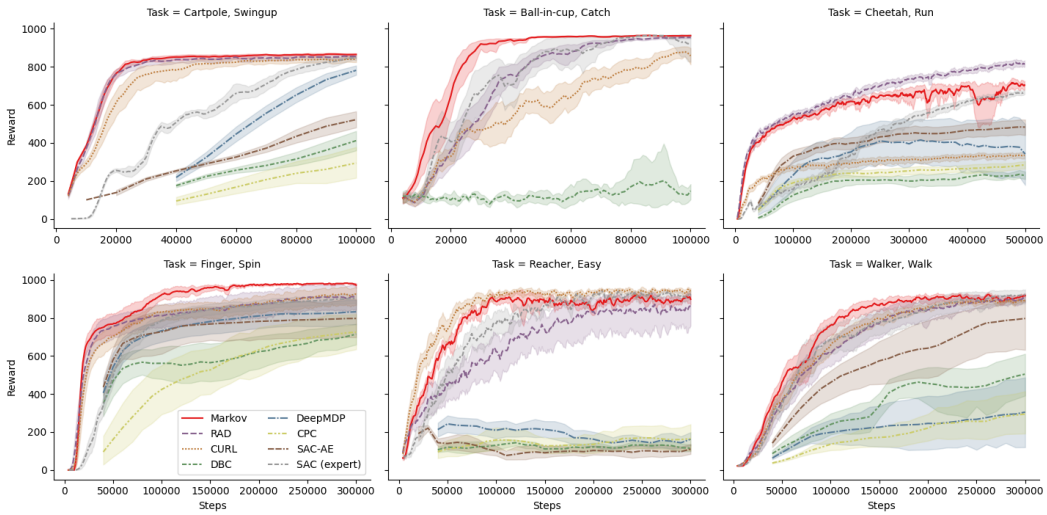


Figure 4: Mean episode reward vs. environment steps for a set of continuous control benchmarks from the DeepMind Control Suite. Adding our Markov training objective leads to state-of-the-art learning performance. (10 seeds each; 5-point moving average; shaded regions denote 90% confidence intervals).

To improve robustness and encourage our method to learn "smooth" representations, we add an additional term to our loss function, $\mathcal{L}_{Smooth} := (\text{ReLU}(\|\phi(x') - \phi(x)\|_2 - d_0))^2$, that penalizes consecutive abstract states for being more than some predefined distance d_0 away from each other. Appendix J describes an additional experiment and discusses why representation smoothness is an important consideration that complements the Markov property.

In addition to RAD, we compare against contrastive methods CURL (Laskin et al., 2020b) and CPC (Van den Oord et al., 2018), bisimulation methods DeepMDP (Gelada et al., 2019) and DBC (Zhang et al., 2021), and pixel-reconstruction method SAC-AE (Yarats et al., 2019). As a reference, we also include non-visual SAC with expert features. The number of RL updates during the learning phase is the same for our method and for RAD, and all methods use the same number of environment steps (though some of the baselines besides RAD use different action-repeat lengths).

We observe that our Markov objective matches or exceeds state-of-the-art performance on five of six domains (see Figure 4). CURL is perhaps most similar to our approach, since it combines contrastive learning with a similar form of data augmentation.[6] These experiments show that even in the online setting, where the agent has access to reward information and a ground state that is assumed to be Markov, adding our training objective to encourage Markov abstractions improves learning performance over state-of-the-art image-based RL.

8 Conclusion

We have developed a principled approach to learning abstract state representations that provably results in Markov abstract states, and which does not require estimating transition dynamics nor observation reconstruction/prediction. We defined what it means for a state abstraction to be Markov while ensuring that the abstract MDP faithfully reflects the dynamics of its ground MDP, and introduced sufficient conditions for achieving such an abstraction. We adapted these conditions into a practical training objective that combines inverse model estimation and temporal contrastive learning.

Our approach learns abstract state representations, with and without reward, that capture the structure of the underlying problem domain and substantially improve learning performance over existing approaches. This work provides theoretical justification for why inverse models and temporal contrastive learning are each effective individually, and shows that their combination offers a path to learning Markov state representations that avoids many of the drawbacks of competing methods.

[6] We ran another experiment with no data augmentation, using a different state-of-the-art continuous control algorithm, RBF-DQN (Asadi et al., 2021), and found similar results there as well (see Appendix I for details).

References

Abel, D., Arumugam, D., Lehnert, L., and Littman, M. L. State abstractions for lifelong reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, 2018.

Agrawal, P., Nair, A., Abbeel, P., Malik, J., and Levine, S. Learning to poke by poking: Experiential learning of intuitive physics. In *Advances in Neural Information Processing Systems*, pp. 5074–5082, 2016.

Anand, A., Racah, E., Ozair, S., Bengio, Y., Côté, M.-A., and Hjelm, R. D. Unsupervised state representation learning in Atari. In *Advances in Neural Information Processing Systems*, pp. 8769–8782, 2019.

Asadi, K., Parikh, N., Parr, R. E., Konidaris, G. D., and Littman, M. L. Deep radial basis value functions for continuous control. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence*, 2021.

Bai, A., Srivastava, S., and Russell, S. J. Markovian state and action abstractions for MDPs via hierarchical MCTS. In *IJCAI*, pp. 3029–3039, 2016.

Biza, O., Platt, R., Van de Meent, J.-W., and Wong, L. L. S. Learning discrete state abstractions with deep variational inference. In *Third Symposium on Advances in Approximate Bayesian Inference*, 2021.

Burda, Y., Edwards, H., Storkey, A. J., and Klimov, O. Exploration by random network distillation, 2018.

Castro, P. S. Scalable methods for computing state similarity in deterministic Markov decision processes. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pp. 10069–10076, 2020.

Choi, J., Guo, Y., Moczulski, M., Oh, J., Wu, N., Norouzi, M., and Lee, H. Contingency-aware exploration in reinforcement learning. In *International Conference on Learning Representations*, 2019.

Christiano, P., Shah, Z., Mordatch, I., Schneider, J., Blackwell, T., Tobin, J., Abbeel, P., and Zaremba, W. Transfer from simulation to real world through learning deep inverse dynamics model. *arXiv*, art. 1610.03518, 2016.

Corneil, D., Gerstner, W., and Brea, J. Efficient model-based deep reinforcement learning with variational state tabulation. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 1049–1058, 2018.

Dean, T. and Givan, R. Model minimization in Markov decision processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 106–111, 1997.

Du, S. S., Krishnamurthy, A., Jiang, N., Agarwal, A., Dudík, M., and Langford, J. Provably efficient RL with rich observations via latent state decoding. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 1665–1674, 2019.

Ferns, N., Panangaden, P., and Precup, D. Metrics for finite Markov decision processes. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, volume 4, pp. 162–169, 2004.

Ferns, N., Panangaden, P., and Precup, D. Bisimulation metrics for continuous Markov decision processes. *SIAM Journal on Computing*, 40(6):1662–1714, 2011.

Finn, C., Tan, X. Y., Duan, Y., Darrell, T., Levine, S., and Abbeel, P. Deep spatial autoencoders for visuomotor learning. In *IEEE International Conference on Robotics and Automation*, pp. 512–519, 2016.

Gelada, C., Kumar, S., Buckman, J., Nachum, O., and Bellemare, M. G. DeepMDP: Learning continuous latent space models for representation learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 2170–2179, 2019.

Gutmann, M. and Hyvärinen, A. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 297–304, 2010.

Ha, D. and Schmidhuber, J. World models. *arXiv*, art. 1803.10122, 2018.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the International Conference on Machine Learning*, pp. 1861–1870. PMLR, 2018.

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. Dream to control: Learning behaviors by latent imagination. In *Proceedings of the 35th International Conference on Machine Learning*, 2020.

Van Hasselt, H., Hessel, M., and Aslanides, J. When to use parametric models in reinforcement learning? In *Advances in Neural Information Processing Systems*, pp. 14322–14333, 2019.

Higgins, I., Pal, A., Rusu, A., Matthey, L., Burgess, C., Pritzel, A., Botvinick, M., Blundell, C., and Lerchner, A. DARLA: Improving zero-shot transfer in reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pp. 1480–1490, 2017.

Hutter, M. Extreme state aggregation beyond Markov decision processes. *Theoretical Computer Science*, 650: 73–91, 2016.

Jonschkowski, R. and Brock, O. Learning state representations with robotic priors. *Autonomous Robots*, 39(3): 407–428, 2015.

Kaiser, Ł., Babaeizadeh, M., Miłos, P., Osiński, B., Campbell, R. H., Czechowski, K., Erhan, D., Finn, C., Kozakowski, P., Levine, S., Mohiuddin, A., Sepassi, R., Tucker, G., and Michalewski, H. Model based reinforcement learning for Atari. In *International Conference on Learning Representations*, 2020.

Laskin, M., Lee, K., Stooke, A., Pinto, L., Abbeel, P., and Srinivas, A. Reinforcement learning with augmented data. *Advances in Neural Information Processing Systems*, 33, 2020a.

Laskin, M., Srinivas, A., and Abbeel, P. CURL: Contrastive unsupervised representations for reinforcement learning. *Proceedings of the 37th International Conference on Machine Learning*, 119:5639–5650, 2020b.

Lee, A. X., Nagabandi, A., Abbeel, P., and Levine, S. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. In *Proceedings of the 34th Conference on Neural Information Processing Systems*, 2020.

Lehnert, L. and Littman, M. L. Successor features combine elements of model-free and model-based reinforcement learning. *Journal of Machine Learning Research*, 21(196):1–53, 2020.

Li, L., Walsh, T. J., and Littman, M. L. Towards a unified theory of state abstraction for MDPs. In *Proceedings of the Ninth International Symposium on Artificial Intelligence and Mathematics*, pp. 531–539, 2006.

Mattner, J., Lange, S., and Riedmiller, M. Learn to swing up and balance a real pole based on raw visual input data. In *Proceedings of the 19th International Conference on Neural Information Processing*, volume Part V, pp. 126–133. Springer, 2012.

Misra, D., Henaff, M., Krishnamurthy, A., and Langford, J. Kinematic state abstraction and provably efficient rich-observation reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 6961–6971, 2020.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Van den Oord, A., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv*, art. 1807.03748, 2018.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 488–489, 2017.

Van der Pol, E., Kipf, T., Oliehoek, F. A., and Welling, M. Plannable approximations to MDP homomorphisms: Equivariance under actions. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems*, pp. 1431–1439, 2020.

Ravindran, B. and Barto, A. G. Approximate homomorphisms: A framework for non-exact minimization in Markov decision processes. In *Proceedings of the Fifth International Conference on Knowledge Based Computer Systems*, 2004.

Shelhamer, E., Mahmoudieh, P., Argus, M., and Darrell, T. Loss is its own reward: Self-supervision for reinforcement learning. *arXiv*, art. 1612.07307, 2016.

Song, Z., Parr, R., Liao, X., and Carin, L. Linear feature encoding for reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 29, pp. 4224–4232, 2016.

Stooke, A., Lee, K., Abbeel, P., and Laskin, M. Decoupling representation learning from reinforcement learning. *arXiv*, art. 2009.08319, 2020.

Tassa, Y., Tunyasuvunakool, S., Muldal, A., Doron, Y., Liu, S., Bohez, S., Merel, J., Erez, T., Lillicrap, T., and Heess, N. dm_control: Software and tasks for continuous control, 2020.

Tiao, L. A simple illustration of density ratio estimation and KL divergence estimation by probabilistic classification. http://louistiao.me/notes/a-simple-illustration-of-density-ratio-estimation-and-kl-divergence-estimation-by-probabilistic-classification, 2017. [Online; accessed 16-March-2020].

Watter, M., Springenberg, J. T., Boedecker, J., and Riedmiller, M. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in Neural Information Processing Systems*, volume 28, pp. 2746–2754, 2015.

Yarats, D. and Kostrikov, I. Soft actor-critic (SAC) implementation in PyTorch. https://github.com/denisyarats/pytorch_sac, 2020.

Yarats, D., Zhang, A., Kostrikov, I., Amos, B., Pineau, J., and Fergus, R. Improving sample efficiency in model-free reinforcement learning from images. *arXiv*, art. 1910.01741, 2019.

Yarats, D., Fergus, R., Lazaric, A., and Pinto, L. Reinforcement learning with prototypical representations. *arXiv*, art. 2102.11271, 2021.

Zhang, A., Satija, H., and Pineau, J. Decoupling dynamics and reward for transfer learning. *arXiv*, art. 1804.10689, 2018.

Zhang, A., McAllister, R. T., Calandra, R., Gal, Y., and Levine, S. Learning invariant representations for reinforcement learning without reconstruction. In *International Conference on Learning Representations*, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 3.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Appendix A.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 2.
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix C.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Code is included in the supplementary materials.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix E and F.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See captions for Figures 3b and 4.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix E and F.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [No]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]