

Learning Causal State Representations of Partially Observable Environments

Amy Zhang^{1,2,3}, Zachary C. Lipton⁴, Luis Pineda³, Kamyar Azizzadenesheli⁵,
Anima Anandkumar⁶, Laurent Itti⁷, Joelle Pineau^{1,2,3}, and Tommaso
Furlanello^{7,8}

¹McGill University

²Mila

³Facebook AI Research

⁴Carnegie Mellon University

⁵University of California Irvine

⁶California Institute of Technology

⁷University of Southern California

⁸Glia Intelligence

Abstract

Intelligent agents can cope with sensory-rich environments by learning task-agnostic state abstractions. In this paper, we propose mechanisms to approximate *causal states*, which optimally compress the joint history of actions and observations in partially-observable Markov decision processes. Our proposed algorithm extracts causal state representations from RNNs that are trained to predict subsequent observations given the history. We demonstrate that these learned task-agnostic state abstractions can be used to efficiently learn policies for reinforcement learning problems with rich observation spaces. We evaluate agents using multiple partially observable navigation tasks with both discrete (GridWorld) and continuous (VizDoom, ALE) observation processes that cannot be solved by traditional memory-limited methods. Our experiments demonstrate systematic improvement of the DQN and tabular models using approximate causal state representations with respect to recurrent-DQN baselines trained with raw inputs.

1 Introduction

Decision-making and control often require that an agent interact with partially-observed environments whose causal mechanisms are unknown. To enable efficient planning, one might hope to construct latent representations of histories of (action, observation) tuples. At present, this practice is dominated by two points of view: (i) a standard approach to *partially observable Markov decision processes (POMDPs)*, where one starts from a generative model of the latent transitions and emission dynamics, using observations to infer beliefs over the unobserved states [1, 5]; and (ii) *predictive state representations (PSRs)*, where one starts from the history of the process and constructs states through modeling of the trajectories of observations [27, 33]. Both directions have drawbacks: the belief state approach requires access to a model that is equivalent to the real generator, while PSRs are constrained by their high-dimensional nature that often makes planning unfeasible.

Corresponding authors: amyzhang@fb.com, tommaso@glia.ai

We propose a principled approach for learning state representations, that generalizes PSRs to non-linear predictive models and allows for a formal comparison between generator- and history-based state abstractions. We exploit the idea of *causal states* [10, 31, 32, 8], i.e., the coarsest partition of histories into classes that are maximally predictive of the future. By learning this mapping from histories to clusters, causal states constitute a discrete causal graph of the observation process.

Our method exploits the existence of minimal discrete representations to derive approximately optimal representations from recurrent neural networks (RNN) trained to model the environment. At the core lies the idea of discretizing the high-dimensional continuous states generated by RNNs into a finite set of clusters. When the clusters are used as input for predictive models, they achieve the same predictive power as the original RNN. Even if this representation is not minimal, each cluster is guaranteed to each map to a single causal state [31].

We use this approach to extend, theoretically contextualize, and integrate two recent algorithms [25, 7] that learn tabular RL policies from discrete representations extracted with neural networks. We evaluate our algorithm for approximate causal states reconstruction on a modification of the original VizDoom environment used in [25] as well as multiple GridWorld navigation tasks with partially observable states. Our DQN models and their tabularized counterparts systematically outperform (by both return and stability) their recurrent-DQN baselines.

2 State Representation for Decision Processes

Consider the nonlinear stochastic process emerging from the interaction between an agent’s policy which chooses discrete actions A_t taking values a_t from the alphabet \mathbf{A} —and the environmental response O_{t+1} taking values o_{t+1} from the alphabet \mathbf{O} . Let $Y_t = (O_{t+1}, A_t)$ be the joint observation-action variable with realizations $y_t = (o_{t+1}, a_t)$ with $y_t \in \mathbf{O} \times \mathbf{A}$,¹ respectively. The future dynamics $\mathbb{P}(\vec{Y}) = \mathbb{P}(\vec{O}, \vec{A})$ depend jointly on the stationary *policy* $\mathbb{P}(\vec{A}|\vec{Y})$ that maps the joint histories \vec{Y} into future actions \vec{A} and the *environment channel*, $\mathbb{P}(\vec{O}|\vec{A}, \vec{Y})$ that maps the joint histories \vec{Y} and future actions \vec{A} into future observations \vec{O} . An agent’s preferences over the future dynamics \vec{Y} are defined via its reward function $R : \mathbf{O}, \mathbf{A} \mapsto \mathbb{R}$. The *optimal policy* of the agent $\pi^*(\vec{o}, \vec{a})$ maximizes the expected reward $\mathbb{E}[R(\mathbb{P}(\vec{Y}|\mathbb{P}(\vec{A}|\vec{Y}) = \pi^*))]$. We restrict our attention to environment channels and agent policies that generate a stochastic process $\mathbb{P}(\vec{Y}|\vec{Y})$ that is *ergodic stationary*, i.e. processes for which the probability of every bi-sequence $(a_t, o_{t+1}, \dots, a_{t+L}, o_{t+1+L})$ of finite length $L \in \mathbb{Z}^+$ is time-invariant, which can be reliably estimated from empirical data.

The formalism of POMDPs supposes a hidden Markov process $P(\vec{S}|S_t, A_t)$, with realizations $s_t \in \mathbf{S}$ where \mathbf{S} is discrete, and observations emitted through the action-conditional probability $P(O_{t+1}|S_t, A_t)$ [23]. This causal relationship between the observed process and the hidden states implies that the mutual information $I[O_{t+1}; S_t, A_t]$ between the *generator state* S_t and current action A_t (jointly) and the next observation O_{t+1} is at least as great as that achieved by any competing representation of the history Ψ_t . This next-step sufficiency is extended to the infinite due to the recursive nature of generator and belief state computations $I[\vec{O}; S_t, A_t] \geq I[\vec{O}; \Psi_t, A_t] \geq I[\vec{O}; \vec{Y}, A_t]$.

¹For a fixed random variable Y_t , we indicate its non-inclusive past with $\vec{Y} = \dots Y_{-3}, Y_{-2}, Y_{-1}$ and its inclusive future with $\vec{Y} = Y_0, Y_1, Y_2, \dots$, dropping the subscript t from the notation for convenience when the context is clear. The set of all bi-infinite sequences \vec{Y} with alphabet \mathbf{Y} is indicated as $\vec{\mathbf{Y}}$. By \vec{Y}_L and \vec{Y}_L , we indicate the finite sequences Y_{-L}, \dots, Y_0 and Y_1, \dots, Y_L .

2.1 Belief and Predictive State Representations

A typical approach to planning in POMDPs assumes the agent has access to $P(\vec{S}|S_t, A_t)$ and $P(O_{t+1}|S_t, A_t)$ and uses it to construct the belief states $b_t = P(S_t|\overleftarrow{y})$ from the finite realizations \overleftarrow{y} . Belief states are computed recursively using Bayes formula from an initial belief $b_0 = P(S_0)$ and give rise to the belief process $P(\vec{B}|\vec{B}, \vec{Y})$. The belief process is a sufficient statistic of the *generator state* when $I[\vec{O}; s_t, a_t] = I[\vec{O}; b_t, a_t]$, and is said to be *asymptotically synchronized* when $\lim_{L \rightarrow \infty} H[S_t|\overleftarrow{Y}_L] = H[S_t|b_t] = 0$, where H is the conditional-entropy function. When $I[O_{t+1}; S_t, A_t] > I[O_{t+1}; \overleftarrow{Y}, A_t]$, the generator states contain more information about the future observable than the complete history of observations \overleftarrow{Y} , implying absence of *asymptotical synchronization* [9] and that belief states are only sufficient statistics of the history \overleftarrow{Y} such that $I[\vec{O}; S_t, A_t] > I[\vec{O}; b_t, A_t] = I[\vec{O}; \overleftarrow{Y}, A_t]$.

The PSR approach relaxes the assumption of having any knowledge about the underlying generator and constructs the representation using the outputs of the predictive model $\mathcal{M}_L = \{\mathbb{P}(\vec{O}_L|\vec{Y}, \vec{A}_L = q_1), \dots, \mathbb{P}(\vec{O}_L|\vec{Y}, \vec{A}_L = q_n)\}$ of the next L observations, conditioned on the next L actions \vec{A}_L (the test) sampled from the set of feasible L -length action sequences $\mathbf{Q}_L = \{q_1, \dots, q_n\}$. By $\vec{\mathcal{M}}$, we indicate the collections of predictive models for all $L \in \mathbb{Z}^+$. Each model \mathcal{M}_L is a sufficient statistic of the L -length future observations \vec{O}_L , and the complete collection $\vec{\mathcal{M}}$ is a sufficient statistic of the infinite future observations \vec{O} , i.e. $I[\vec{O}; \vec{\mathcal{M}}, A_t] = I[\vec{O}; \overleftarrow{Y}, A_t]$. Typically, PSRs are constructed for decision processes using a linear model that enables approximate solutions by assuming that the infinite-dimensional *system dynamics matrix* has finite rank [33].

2.2 Causal States Representations

We propose to use the *causal states representation* that expands PSRs to the general case of non-linear predictive models and allows the definition of a formal equivalence between the eventual generator states and the causal states reconstructed from history. As in the PSR framework, causal states depend on a predictive model of the observation process.

Definition 1 [10, 31] *The **causal states** of a stochastic process are partitions $\sigma \in \mathbb{S}$ of the space of feasible pasts \overleftarrow{Y} induced by the causal equivalence \sim_ϵ :*

$$\overleftarrow{y} \sim_\epsilon \overleftarrow{y}' \iff \mathbb{P}(\vec{Y}|\overleftarrow{Y} = \overleftarrow{y}) = \mathbb{P}(\vec{Y}|\overleftarrow{Y} = \overleftarrow{y}'). \quad (1)$$

Which implies:

$$\mathbb{P}(\vec{Y}|S_t = \sigma_i) = \mathbb{P}(\vec{Y}|\overleftarrow{Y} = \overleftarrow{y}) \quad \forall \quad \overleftarrow{y} \in \sigma_i, \quad (2)$$

where S_t is the variable denoting causal state at time t , overwriting the definition in Sec. 2 of the unknown ground truth state. Since all histories belonging to the same equivalence class predict the same (conditional) future, the corresponding causal state can be used to fully summarize the information content of those histories. It can be demonstrated [31] that the partition induced by \sim_ϵ is the coarsest possible and generates the minimal sufficient representation across the model class. Sampling of new symbols in the sequence induces the creation of new histories and consequently new causal states. Because of this mapping from histories to states, the resulting hidden Markov model is *unifilar*.

Definition 2 [31] *A **unifilar hidden Markov model** is a HMM whose state transition probability $\mathbb{P}(S_{t+1}|S_t)$ is deterministic if conditioned on the output symbol, i.e $H[S_{t+1}|Y_{t+1}, S_t] = 0$.*

With explicit reference to the joint input-output history, the state transition dynamics are governed by input-conditional transition matrices $\mathbf{T}^{o|a} \in \mathcal{T}$ with elements:

$$\mathcal{T}_{ij}^{o|a} = \mathbb{P}(S_{t+1} = \sigma_j, O_{t+1} = o | S_t = \sigma_i, A_t = a). \quad (3)$$

Since the causal states are defined over histories of joint symbols, the causal state model is unifilar with respect to the joint variable A_t, O_{t+1} , i.e. the transitions between states are deterministic once the next action and observable have been sampled or $H[S_{t+1}|A_t, O_{t+1}, S_t] = 0$. The unifilar property implies that the recurrent dynamics of the causal states are fully specified by the state-action-conditional symbol emission probability $\mathbb{P}(Y_{t+1}|S_t, A_t)$ and the action-symbol-conditional causal state emission probability $\mathbb{P}(S_{t+1}|Y_{t+1}, A_t, S_t)$. As a consequence, knowledge of the current causal states S_t and of the future action-observation sequence $\overrightarrow{O, A}$ induces a deterministic sequence of future causal states \overrightarrow{S} , $H(\overrightarrow{S}|\overrightarrow{O, A}, S_t) = 0$.

2.3 Stochastic Processes with Finite Causal States

When the joint process admits a finite causal state representation it is called a *finitary* stochastic process which have multiple theoretical implications. In discrete stochastic processes with finite actions, finite-symbol alphabets, and finite memory of length k the causal states are always finite, with a worst case scenario in which each sub-sequence of length k belongs to a distinct causal state forming a k -length Markov model [31]. When the causal states are finite, they are also unique up to isomorphisms [31] and always generate a stationary stochastic process. If the underlying generator is non-unifilar, the causal states have the same information content of the potentially non-synchronizing belief states of the generator, and the belief states defined over the causal states always asymptotically synchronize to the actual causal states [9].

We focus on partially observable environments with discrete causal states and either continuous or discrete observations. For continuous observations it is not possible to derive generic conditions that imply discrete or finite causal states. Therefore, the existence of discrete latent states has to be directly assumed or derived from alternative assumptions like the existence of finite latent discrete variables underlying each continuous observation. When a memory-less map from continuous observation to latent discrete variables exists, the causal states of the revealed continuous variable process coincide with those defined over the underlying discrete variables.

3 Methods

In the previous sections, we introduced a class of stochastic processes with discrete or continuous outputs that are optimally compressed by a finite-state hidden-Markov representation, called the causal state model of the process.

We now propose a new approach to approximately reconstruct these causal states from empirical data.

3.1 Empirical Estimation of Causal States

Existing methods either directly partition past sequences of length L into a finite number of causal states via conditional hypothesis tests [32] or use Bayesian inference over a set of existing candidate states [36]. Either method can be adapted to model a joint-process and consequently obtain the next-step conditional output by marginalizing out the action A_t , but do not extend to the real-valued measurement case described without strong assumptions on the shape of the conditional density function.

In this work, we exploit the definition that minimal-sufficient statistics can be computed from any other non-minimal sufficient statistic. We obtain the (approximately) minimal representations of the underlying process $\overleftarrow{S}, \overrightarrow{A}$ by discretizing a sufficient model of the measurement process $\overleftarrow{O}, \overrightarrow{A}$. This approach exploits learning a hierarchy of optimal predictive models with progressively stricter bounds on their representations' dimensionality. We start with infinite dimensional continuous representations learned with a deep neural network, and then partition the continuous representations into a finite set of clusters.

3.2 Learning Sufficient Statistics of History with Recurrent Networks

Recurrent neural networks (RNNs) are unifilar hidden Markov models with continuous states, where the transitions and state output probabilities are parameterized by differentiable functions. We use them to obtain recursively-computable high-dimensional sufficient statistics of the action-measurement joint process. This representation is learned via a recurrent encoder $f : \overleftarrow{\mathbf{O}}, \mathbf{A} \mapsto \hat{\mathbf{S}}$, and a next step prediction network $\eta : \hat{\mathbf{S}} \times \mathbf{A} \mapsto \hat{\mathbf{O}}$. The overall neural-network architecture resembles world models [14], except we auto-encode the observation o_t only with image inputs, and only use it for next step prediction in the other cases. Furthermore, we use an explicit embedding layers for a_t that is concatenated with the output of the recurrent encoder before predicting o_{t+1} .

We note that when $\eta(\hat{s}_t, a_t)$ is maximally predictive of the subsequent observations (the future $\overrightarrow{\mathbf{O}}$), \hat{s}_t constitutes a sufficient statistic of the latent states $\overrightarrow{\mathbf{S}}$. In practice, we estimate the continuous representations using the empirical realizations $\overleftarrow{\delta}, \overleftarrow{a}$ ² to learn a neural network $\Psi(\overleftarrow{\delta}, \overleftarrow{a}, a_t) = (\eta_{w_\eta} \circ f_{w_f})(\overleftarrow{\delta}, \overleftarrow{a}, a_t)$ that approximates end to end the maps f and η by minimizing the temporal loss through the following optimization problems:

$$\min_{w_f, w_\eta} \sum_t^T \mathcal{L}_r(\mathbb{P}(O_{t+1} | \overleftarrow{\delta}, \overleftarrow{a}, a_t), \Psi(\overleftarrow{\delta}, \overleftarrow{a}, a_t)) \quad (4)$$

After solving Eqs. 4 we can use the neural networks parameterized by the optimal parameters w_η^*, w_f^* to derive sufficient continuous representations to create discrete states that are refinement of the causal states.

3.3 Discretization of the RNN Hidden States

Together with the unifilar and Markovian nature of transitions in RNNs, the sufficiency of \hat{s}_t implies that there exists a function $\mathcal{D}^s : \mathbb{R}^k \mapsto \mathbf{S}$ that allows us to describe the causal states s as a partition of the learned latent state \hat{s} [31, 9].

We set up a second optimization problem using the trained neural network and the empirical realizations of the process \overleftarrow{o} to estimate the discretizer $\bar{d}^s : \hat{\mathbf{S}} \mapsto \bar{\mathbf{S}}$ with $|\bar{\mathbf{S}}| = |\mathbf{S}|$ and the new prediction network $\bar{\eta} : \bar{\mathbf{S}} \times A \mapsto \mathbf{O}$ that maps the estimated discrete states into the next observable. We match the predictive behavior between the old network Ψ and the new networks $\Lambda(\overleftarrow{\delta}, \overleftarrow{a}, a_t) = (\bar{\eta}_{w_{\bar{\eta}}} \circ \bar{d}_{w_{\bar{d}}}^s \circ f_{w_f^*})(\overleftarrow{\delta}, \overleftarrow{a}, a_t)$ that use discretized states \bar{s} and corresponding prediction function $\bar{\eta}$ by minimizing the knowledge distillation [19] loss:

$$\min_{w_{\bar{\eta}}, w_{\bar{d}}} \sum_t^T \mathcal{L}_d(\Psi(\overleftarrow{\delta}, \overleftarrow{a}, a_t), \Lambda(\overleftarrow{\delta}, \overleftarrow{a}, a_t)). \quad (5)$$

Minimizing Eq. 5 guarantees a sufficient discrete representation. To summarize, we first minimize \mathcal{L}_r to obtain a neural model able to generate continuous sufficient statistics of the future observables of the process and subsequently minimize \mathcal{L}_d to obtain a sufficient

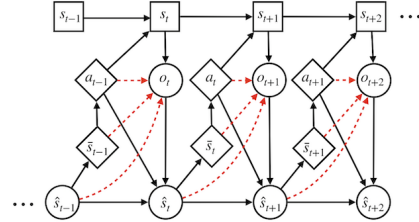


Figure 1: Graphical model generating the joint action-measurement stochastic process. Black-arrows indicate causal relationships between random variables and red-arrows indicate the predictive relationship between the combinations of action a_t , internal state \bar{s} (\hat{s}) and the next measurement o_{t+1} . Circular boxes indicate continuous variables.

²With a small abuse of notation, we use the same convention of $\overleftarrow{\mathbf{Y}}$, where we shift measurements by one time step such that the joint process $\overleftarrow{\mathbf{O}}, \overleftarrow{\mathbf{A}}$ has elements (O_{t+1}, A_t) .

representation of the dynamical system that is a refinement of the original causal states. Fig. 1 shows the stochastic process representing the environment and our learned states \hat{S} and \bar{S} and their interactions.

3.4 Implementation Details

For the GridWorlds and Toy-Processes experiments, the base world model architecture is composed of a three-layer perceptron (MLP) encoding the observation o_t and a single layer linear embedding for the action a_{t-1} . The outputs of the respective layers are concatenated and fed to a Gated Recurrent Unit (GRU) [6], and the output of the recurrent network \hat{s}_t is concatenated with the embedding of a_t and fed to a second MLP that outputs predictions for o_{t+1} . All the embeddings have 64 neurons except in layout 4 where we use 256 dimensional embeddings. The discretization network is composed of a Quantized-Bottleneck-Network [25] with ternary tangent neurons that auto-encodes the continuous representation \hat{s}_t generating the discrete variables \bar{s}_t , and a MLP decoder that uses the estimated discrete states for predicting the next observation o_{t+1} .

Both networks are trained with the RMSprop algorithm using cross-entropy loss for discrete observations and reconstruction loss for the continuous setting. The world-models is trained through supervised learning of the temporal loss, while the discretization network is trained via knowledge distillation using the soft outputs of the GRU decoder as targets. We ran downstream evaluation of our learned representation with value iteration and compare with baselines. To approximate the value function we use a 2-layer fully-connected DQN architecture separated by ReLU with a 64-dimensional hidden layer. The R-DQN baselines uses the same architectures but they are trained end-end via reward maximization. We also present earlier results where the discretization step is implemented with K-mean clustering and for policies learned with traditional tabular Q-learning.

For the VizDoom and ALE experiments we built upon the base VaST architecture [7] that uses binary Bernoulli variables and Gumbel Softmax [22] for their discrete bottle-neck. The original architecture for MDPs is composed by a variational encoder that embeds the observation o_t into k binary variables which are combined with action a_t to predict the next latent discrete state. We extend it with a recurrent encoder that gives access to the agent to the history of observations \overleftarrow{o} . Since we build over the original implementation we use prioritized sweeping with tabular Q-learning for the downstream policy. The world models is initialized from 10k random rollouts and is trained using the reconstruction loss.

4 Experiments

We employ three partially observable environments to learn approximate causal states through self-supervised learning and use these representations as input for reinforcement learning tasks defined over the domains.

4.1 Gridworlds

We create partially observable gridworld environments where the task is for the agent to first obtain the key, then pass through the door to obtain the final reward. The final state is unseen (i.e. the agent cannot pass through the door to reach it) *if* the agent does not have the key. The agent only knows it has the key if it remembers entering the state with the key, so without infinite memory this task is partially observable. At each time step the agent receives -0.1 reward, 0.5 reward for picking up the key, and a final reward of 1 for passing through the door.

We conduct experiments with three layouts with an increasing number of states (see figures in the supplementary materials). The first layout is a 1-dimensional corridor while the other three are two-dimensional mazes. The minimal memory requirement for solving the environment is given by the shortest path from the key to the final destination.

Table 1: Results for Gridworlds. Reward obtained with tabular Q-learning, DQN, and DRQN with $\gamma = 0.99$. Models trained on 1000 episodes and evaluated on 100. Numbers are mean, standard deviation across 10 random seeds. First section is our method using k-mean clustering for discretization, second is baselines on current observation, history of observations, and S . Final section is using ground truth states.

Method	Layout 1 low-disc	Layout 1 low-cont	Layout 1 ego-cont	Layout 2 low-disc	Layout 2 low-cont	Layout 2 ego-cont
Tab., \bar{S}	0.43, 0.	0.42, 0.	0.437, 0.	0.01, 0.	0.09, 0.	0.036, 0.
DQN, \bar{S}	0.50, 0.015	0.42, 0.10	0.49, 0.032	-0.17, 0.76	0.026, 0.26	0.12, 0.064
DQN, \hat{S}	0.5, 0.	0.5, 0.	0.49, 0.029	0.30, 0.	0.30, 0.	0.30, 0.01
DQN, Y	-9.46, 0.20	-8.55, 0.79	-8.64, 1.01	-9.48, 0.14	-8.59, 0.78	-9.83, 0.25
DQN, \hat{Y}	-0.91, 3.0	0.49, 0.03	-0.34, 2.31	0.23, 0.16	0.084, 0.18	-0.07, 0.24
DRQN, Y	-9.75, 0.22	-6.95, 3.66	-9.27, 0.68	-5.63, 3.74	-3.72, 4.31	-9.97, 0.06
Tab., Y	-9.40, 0.	-	-	-9.11, 0.	-	-
Tab., S_{gt}	0.45, 0.	0.42, 0.	0.43, 0.	0.23, 0.	0.23, 0.	0.23, 0.
DQN, S_{gt}	0.44, 0.019	0.44, 0.027	0.44, 0.023	0.30, 0.01	-0.76, 3.07	0.23, 0.10

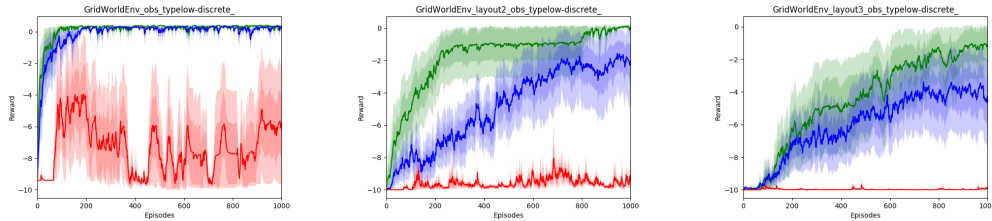


Figure 2: Training curves for DQN policies—discretization with gradient descent and bottleneck networks (*left*) Layout 1 using discrete inputs. (*center*) Layout 2 using discrete inputs. (*right*) Layout 3 using discrete inputs. Averages over 10 runs with different random seeds with two standard deviations shaded. Y-axis is mean reward per step. Green is the World model, Blue is causal states, red is DRQN.

We use three types of observation processes for the agent that give different priors to the agent’s behavior but share the same underlying causal states. **low-discrete** is the discrete observation of the agent’s absolute position in the grid. **low-cont** is the continuous observation (x, y) of the agent’s absolute position in the grid. **ego-cont** is ego-centric continuous observation (up, down, left, right) of the distance from walls in the 4 cardinal directions.

dqnDQN on \hat{S} is able to achieve optimal policy across all 10 random seeds with very low or zero standard deviation, showing the stability of our learned \hat{S} . We expect \hat{S} to perform as well as or better than \bar{S} , as \bar{S} is distilled from \hat{S} and therefore contains the same information. Using only current observation learns using a recurrent DQN (DRQN) [16].

4.2 Doom

We modify the t-maze VizDoom [24] environment of [7] to make it partially observable. We randomize the goal location between the two ending corners and signal its location with a stochastic signal in the observation space. The agent must remember where the goal is in order to navigate to it. We convey the signal to the agent through a fourth channel (after RGB) that intermittently contains information about where the goal is. The frequency at which the information is displayed is a tunable factor $f = 5$ for these experiments. Example trajectories to different goals that the learned agent takes are shown in Fig. 3 on the left.

Results Fig. 3 (right) shows the speedup in learning from explicitly learning to cluster

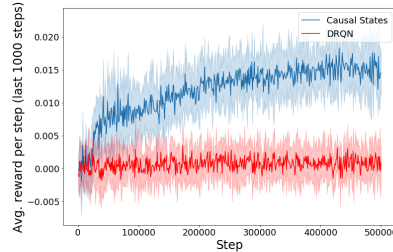
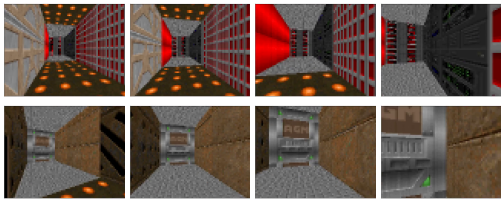


Figure 3: (left) Example trajectories in VizDoom. First goal (above), second goal (below). (right) Doom T-Maze POMDP: Averaged over 10 runs with different random seeds with one standard deviation shaded. Y-axis is mean reward per step. Blue is causal states, red is DRQN.

sequences of observations into causal states. Additional results with varying frequencies in Supplementary Material.

4.3 Atari Pong

We evaluate our model on the Atari Learning Environment’s Pong [3], an environment where causal states are less intuitive. Our algorithm learns a set of discrete causal states suitable for learning successful policies (Fig. 4) We use the same preprocessing of observations in [29] except that the agent receives a single frame as observation at each time step. The environment is now partially observable as the agent must learn to retain information about velocity of the paddles and ball from the hidden state of the RNN. We see a decrease in performance with DRQN, due to instability in training.

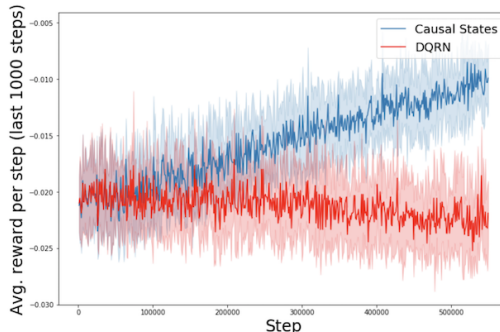


Figure 4: Averaged over 10 runs with different random seeds with one standard deviation shaded.

4.4 Toy Stochastic-Processes

We apply our method to input-output stochastic processes that can be modeled as HMMs with memory length k and alphabet size $|\mathbf{Y}|$. We complicate the continuous observation process by mapping the environment’s outputs into a) multivariate gaussians and b) images sampled from the MNIST dataset. The goal is to maximize the occurrence of $o_t = 1$, $0 \leq t < \infty$. The environment returns a reward of 1 at time step t if $o_t = 1$, and 0 otherwise. Each episode lasts 100 time steps. Results in the appendix.

5 Related Literature

The relationship between PSR and causal states has been previously suggested in the computation mechanics literature [31, 32, 2]. [20] also derive parallels between PSR, POMDP, and automata through the construction of equivalence classes that groups states with common action conditional future observations. Causal states, and the related information-theoretic notion of complexity, minimality, and sufficiency are used to derive task-agnostic policies with an intrinsic exploration-exploitation trade off in [34, 35].

In [4] the authors learn PSRs in Reproducing Kernel Hilbert Space extending the approach to continuous, potentially infinite, action-observation processes. More similar to our latent

discrete states with continuous observable, [12] model spatio-temporal processes as being generated by a finite set of discrete causal states in the form of light-cones.

Other methods for learning deep representations for reinforcement learning POMDPs have been recently proposed, starting with adding recurrency to DQN [16] to integrate the history in the estimation of the Q-value as opposed to using only the current observation. However, this method stops short of ensuring sufficiency for next step prediction as it learns a task specific representation. [21, 38] use deep variational methods to learn a probability distribution over states, i.e. belief states, and use the belief states for policy optimization with actor-critic [21] and planning [38]. [13] also use neural methods to learn belief states with next-step prediction [17, 18] learn PSRs with RNNs and spectral methods and use policy gradient with an alternating optimization method for the policy and the state representation to handle continuous action spaces. However, none of these explore the connection to causal states and compression via a discrete representation. [7] do learn discrete representations but not in partially observable environments and with no link to PSRs. Instead, they propose the discrete representation solely for using tabular Q-learning with prioritized sweeping.

The idea of extracting the implicit knowledge of a neural network [37, 11, 28, 15] is not novel and is rooted in early attempts to merge traditional rule-based methods with machine learning. The most recent examples [41, 39, 40] are focused on the ability of character level RNNs to implicitly learn grammars. The only application of these ideas to RL in partially observable environments that we are aware of is in [25] where deep recurrent policies [16] are quantized into Moore machines. The main difference between our approaches is that we reduce models of the environment, not of the policy, to ϵ -machines which are edge-emitting Mealy machines. The connection between ϵ -machines and causal states is further discussed in Supplementary Materials. Because the optimal policy will in general use a subset of the possible input sequences, the minimal sufficient representation of a policy is typically smaller than the causal states of the complete environment.

6 Conclusions

In this paper, we proposed a self-supervised method of learning a minimal sufficient statistic \bar{S} for next step prediction and articulated its connection to the causal states of a controlled process. Our experiments demonstrate the practical utility of this representation, both with value iteration for control and exhaustive planning, in solving stochastic and infinite-memory POMDP environments as well as k -order MDP environments with high dimensional observations, matching the performance achieved with ground truth states.

We encountered multiple problems while training recurrent DQN models end to end, even when we know from our causal states results that the same architecture is able to learn sufficient representations for the task. We plan on extending the Distributed Experience Replay algorithm [26] with extra asynchronous workers dedicated respectively to the construction of continuous and discrete world models.

Acknowledgements

Part of this work was supported by the National Science Foundation (grant number CCF-1317433), C-BRIC (one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA), and the Intel Corporation. A. Anandkumar is supported in part by Bren endowed chair, Darpa PAI, Raytheon, and Microsoft, Google and Adobe faculty fellowships. K. Azizzadenesheli is supported in part by NSF Career Award CCF-1254106 and AFOSR YIP FA9550-15-1-0221, work done while he was visiting Caltech. The authors affirm that the views expressed herein are solely their own, and do not represent the views of the United States government or any agency thereof.

References

- [1] Karl J Aastrom. Optimal control of markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, 10(1):174–205, 1965.
- [2] Nix Barnett and James P Crutchfield. Computational mechanics of input–output processes: Structured transformations and the eps-transducer. *Journal of Statistical Physics*, 161(2):404–451, 2015.
- [3] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, jun 2013.
- [4] Byron Boots, Geoffrey Gordon, and Arthur Gretton. Hilbert space embeddings of predictive state representations. *arXiv preprint arXiv:1309.6819*, 2013.
- [5] Anthony R Cassandra, Leslie Pack Kaelbling, and Michael L Littman. Acting optimally in partially observable stochastic domains. In *AAAI*, volume 94, pages 1023–1028, 1994.
- [6] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.
- [7] Dane S. Corneil, Wulfram Gerstner, and Johanni Brea. Efficient model-based deep reinforcement learning with variational state tabulation. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 1057–1066, 2018.
- [8] James P Crutchfield. The origins of computational mechanics: A brief intellectual history and several clarifications. *arXiv preprint arXiv:1710.06832*, 2017.
- [9] James P Crutchfield, Christopher J Ellison, Ryan G James, and John R Mahoney. Synchronization and control in intrinsic and designed computation: An information-theoretic analysis of competing models of stochastic computation. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 20(3):037105, 2010.
- [10] James P Crutchfield and Karl Young. Inferring statistical complexity. *Physical Review Letters*, 63(2):105, 1989.
- [11] LiMin Fu. Rule generation from neural networks. *IEEE Transactions on Systems, Man, and Cybernetics*, 24(8):1114–1124, 1994.
- [12] Georg Goerg and Cosma Shalizi. Mixed licors: A nonparametric algorithm for predictive state reconstruction. In *Artificial Intelligence and Statistics*, pages 289–297, 2013.
- [13] Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Bernardo A. Pires, Toby Pohlen, and Rémi Munos. Neural predictive belief representations. *CoRR*, abs/1811.06407, 2018.
- [14] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.
- [15] Tameru Hailesilassie. Rule extraction algorithm for deep neural networks: A review. *arXiv preprint arXiv:1610.05267*, 2016.
- [16] Matthew Hausknecht and Peter Stone. Deep recurrent q-learning for partially observable mdps. 2015.
- [17] Ahmed Hefny, Carlton Downey, and Geoffrey J Gordon. Supervised learning for dynamical system learning. In *Advances in neural information processing systems*, pages 1963–1971, 2015.

- [18] Ahmed Hefny, Zita Marinho, Wen Sun, Siddhartha Srinivasa, and Geoffrey Gordon. Recurrent predictive state policy networks. *arXiv preprint arXiv:1803.01489*, 2018.
- [19] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.
- [20] Christopher Hundt, Joelle Pineau, and Doina Precup. Representing systems with hidden state. 2006.
- [21] Maximilian Igl, Luisa Zintgraf, Tuan Anh Le, Frank Wood, and Shimon Whiteson. Deep variational reinforcement learning for pomdps. *arXiv preprint arXiv:1806.02426*, 2018.
- [22] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *CoRR*, abs/1611.01144, 2016.
- [23] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artif. Intell.*, 101(1-2):99–134, May 1998.
- [24] Michał Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski. ViZDoom: A Doom-based AI research platform for visual reinforcement learning. In *IEEE Conference on Computational Intelligence and Games*, pages 341–348, Santorini, Greece, Sep 2016. IEEE. The best paper award.
- [25] Anurag Koul, Sam Greydanus, and Alan Fern. Learning finite state representations of recurrent policy networks. *arXiv preprint arXiv:1811.12530*, 2018.
- [26] Thanard Kurutach, Aviv Tamar, Ge Yang, Stuart Russell, and Pieter Abbeel. Learning plannable representations with causal infogan. *arXiv preprint arXiv:1807.09341*, 2018.
- [27] Michael L Littman and Richard S Sutton. Predictive representations of state. In *Advances in neural information processing systems*, pages 1555–1561, 2002.
- [28] Hongjun Lu, Rudy Setiono, and Huan Liu. Effective data mining using neural networks. *IEEE transactions on knowledge and data engineering*, 8(6):957–961, 1996.
- [29] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. In *NIPS Deep Learning Workshop*. 2013.
- [30] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013.
- [31] Cosma Rohilla Shalizi and James P Crutchfield. Computational mechanics: Pattern and prediction, structure and simplicity. *Journal of statistical physics*, 104(3-4):817–879, 2001.
- [32] Cosma Rohilla Shalizi and Kristina Lisa Shalizi. Blind construction of optimal nonlinear recursive predictors for discrete sequences. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 504–511. AUAI Press, 2004.
- [33] Satinder Singh, Michael R James, and Matthew R Rudary. Predictive state representations: A new theory for modeling dynamical systems. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 512–519. AUAI Press, 2004.
- [34] Susanne Still. Information-theoretic approach to interactive learning. *EPL (Europhysics Letters)*, 85(2):28005, 2009.

- [35] Susanne Still and Doina Precup. An information-theoretic approach to curiosity-driven reinforcement learning. *Theory in Biosciences*, 131(3):139–148, 2012.
- [36] Christopher C Strelhoff and James P Crutchfield. Bayesian structural inference for hidden processes. *Physical Review E*, 89(4):042119, 2014.
- [37] Geoffrey G Towell and Jude W Shavlik. Extracting refined rules from knowledge-based neural networks. *Machine learning*, 13(1):71–101, 1993.
- [38] Sebastian Tschiatschek, Kai Arulkumaran, Jan Stühmer, and Katja Hofmann. Variational inference for data-efficient model learning in pomdps. *CoRR*, abs/1805.09281, 2018.
- [39] Qinglong Wang, Kaixuan Zhang, II Ororbia, G Alexander, Xinyu Xing, Xue Liu, and C Lee Giles. An empirical evaluation of recurrent neural network rule extraction. *arXiv preprint arXiv:1709.10380*, 2017.
- [40] Qinglong Wang, Kaixuan Zhang, II Ororbia, G Alexander, Xinyu Xing, Xue Liu, and C Lee Giles. A comparison of rule extraction for different recurrent neural network models and grammatical complexity. *arXiv preprint arXiv:1801.05420*, 2018.
- [41] Gail Weiss, Yoav Goldberg, and Eran Yahav. Extracting automata from recurrent neural networks using queries and counterexamples. *arXiv preprint arXiv:1711.09576*, 2017.
- [42] Amy Zhang, Adam Lerer, Sainbayar Sukhbaatar, Rob Fergus, and Arthur Szlam. Composable planning with attributes. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80, pages 5837–5846. JMLR.org, 2018.

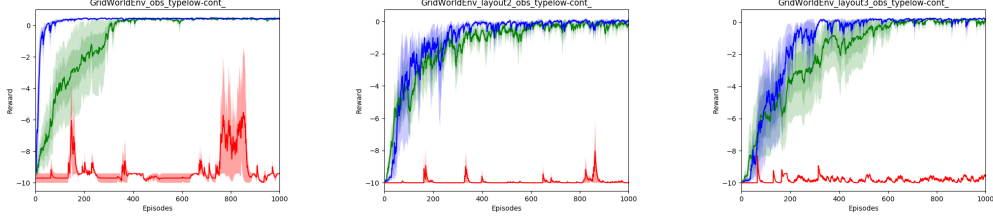


Figure 5: LOW-CONT: Training curves for DQN policies - discretization with gradient descent and bottleneck networks. (left) Layout 1 using x-y coordinates. (center) Layout 2 using x-y coordinates. (right) Layout 3 using x-y coordinates. Averaged over 10 runs with different random seeds with two standard deviations shaded. Y-axis is mean reward per step. Green is the World model, Blue is causal states, red is DRQN.

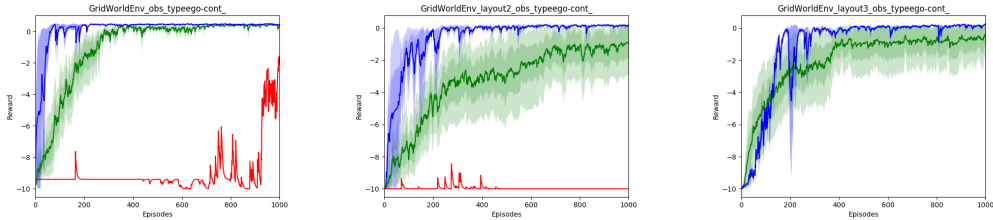


Figure 6: EGO-CONT: Training curves for DQN policies - discretization with gradient descent and bottleneck networks. (left) Layout 1 using egocentric distance from walls in each cardinal direction. (center) Layout 2 using egocentric distance from walls in each cardinal direction. (right) Layout 3 using egocentric distance from walls in each cardinal direction. Averaged over 10 runs with different random seeds with two standard deviations shaded. Y-axis is mean reward per step. Green is the World model, Blue is causal states, red is DRQN.

A Causal States and ϵ -machines

In the computational mechanics literature, causal state models are usually called ϵ -machines and are formally defined as:

Definition 3 The ϵ -*machine* of a stochastic process \vec{Y} is given by the tuple $\epsilon = \langle \mathcal{S}, \mathcal{Y}, \mathcal{T} \rangle$ where \mathcal{S} is the discrete alphabet of causal states, \mathcal{Y} the discrete alphabet of observation and \mathcal{T} is a set of observation conditional state-to-state transition matrices. [10, 31]

To summarize, the ϵ -*machine* of a stochastic process is the minimal unifilar hidden Markov model able to generate its empirical realizations. The hidden states of an ϵ -machine are called the causal states of the process, and correspond to partitions of the process history.

B Additional Visualizations for GridWorlds experiments

In Figures 5 and 6, we present additional DQN results for Layout 1, 2 and 3 using continuous inputs. The models in Figure 5 use continuous x,y coordinates as input while those in Figure 6 use the egocentric distance from walls in each cardinal direction. Figures 7 and 8 depict the configuration of the four layouts. Figure 9 and 10 contain the learning curve for layout 1 and 2 for all input modalities using k-mean clustering for discretization.

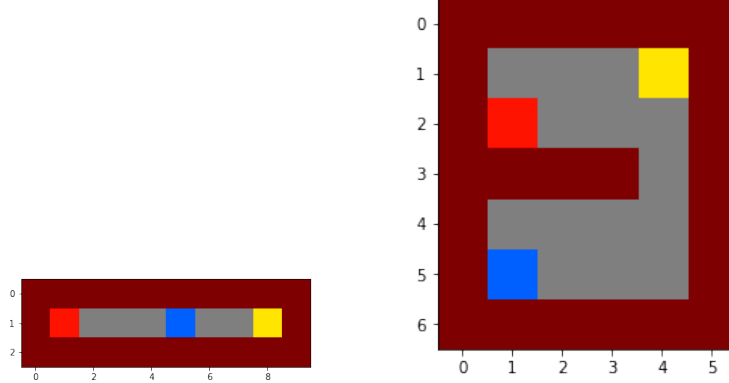


Figure 7: Visual representation of the layouts 1 and 2 used in the gridworld experiments. In *Blue* the starting location, in *Red* the key location in *Yellow* the final goal. *Brown* represents wall and in *Grey* are walkable cells

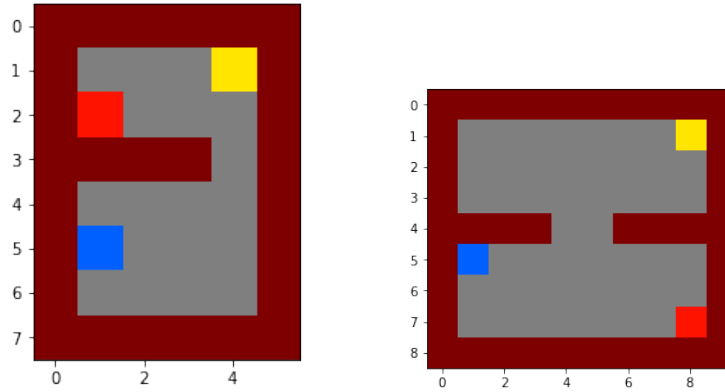


Figure 8: Visual representation of the layouts 3 and 4 used in the gridworld experiments. In *Blue* the starting location, in *Red* the key location in *Yellow* the final goal. *Brown* represents wall and in *Grey* are walkable cells

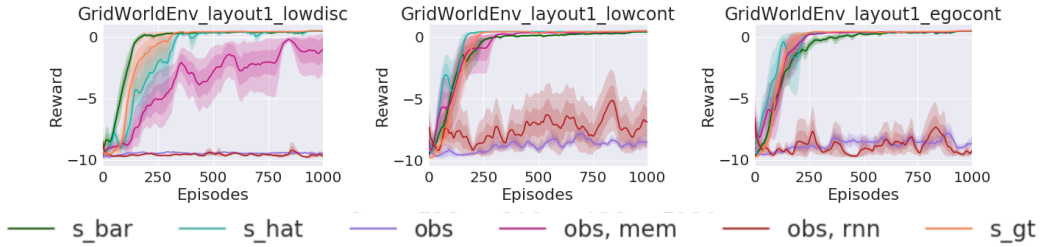


Figure 9: DQN training curves across 10 runs with different random seeds for all input types in Tab. 1 for Layout 1 - Causal States \hat{S} estimated using K-mean clustering. Two levels of shading represent 1 and 2 standard deviations from the mean.

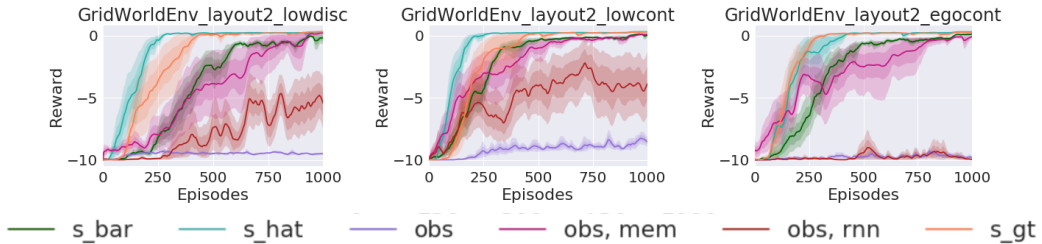


Figure 10: DQN training curves across 10 runs with different random seeds for all input types in Tab. 1 for Layout 2 - Causal States \hat{S} estimated using K-mean clustering. Two levels of shading represent 1 and 2 standard deviations from the mean.

Table 2: Results for the Toy Processes. Reward obtained with DQN initialized with 10 random seeds. Numbers in each cell correspond to mean and standard deviations. All models are trained on 500 episodes and evaluated on 100. Results within a std of the best are bolded.

Method	Discrete		Gaussian		MNIST	
	$ Y , k = 2$	$ Y , k = 4$	$ Y , k = 2$	$ Y , k = 4$	$ Y , k = 2$	$ Y , k = 4$
DQN on Y	50.1, 1.01	25.1, 1.12	50.6, 1.26	25.0, 1.35	50.1, 1.80	25.0, 1.27
DQN on \hat{Y}	73.7, 0.73	55.5, 1.62	73.3, 1.20	54.9, 1.71	72.3, 1.33	54.2, 1.39
DQN on \hat{S}	72.7, 1.04	54.6, 1.61	73.6, 0.82	55.3, 1.91	72.8, 1.23	50.8, 1.80
DQN on \bar{S}	72.6, 4.10	49.2, 3.29	73.7, 2.18	52.7, 3.07	72.6, 2.50	43.2, 3.02

C Additional Experiments

C.1 Toy Processes

We apply our method to input-output stochastic processes that can be modeled as HMMs with memory length k and alphabet size $|\mathbf{Y}|$. We construct the process such that the probability of the next output o_{t+1} depends only on the value of o_{t-k} by sampling from the multinomial distribution: $\mathbb{P}(O_{t+1} = o' | O_{t-k} = o') = p$ and $\mathbb{P}(O_{t+1} = o' | O_{t-k} \neq o') = 1 - \frac{p}{|O|}$ for all $o' \in \mathbf{O}$.

We introduce a binary action space $\mathcal{A} = \{0, 1\}$ where

$$p(O_{t+1} = i | A_t = 0) = \begin{cases} p & \text{if } o_{t-k} = i, \\ \frac{1-p}{|O|} & \text{otherwise.} \end{cases},$$

$$p(O_{t+1} = i | A_t = 1) = \begin{cases} p & \text{if } o_{t-k} = i - 1, \\ \frac{1-p}{|O|} & \text{otherwise.} \end{cases}.$$

The goal is to maximize the occurrence of $o_t = 1$, $0 \leq t < \infty$. The environment returns a reward of 1 at time step t if $o_t = 1$, and 0 otherwise. Each episode lasts 100 time steps. We again train on sequence data $\{o_1, a_0, o_2, a_1, \dots, o_{T+1}, a_T\}$ generated with a random policy and pass the actions into the RNN with a linear layer and concatenate with the observation embedding.

The multivariate Gaussians measurements are constructed by taking a vector composed by k blocks of size $|\mathbf{O}|$, the i^{th} block has mean $\mu_i = 4$ if $O_t = y_i$ or $\mu_i = 0$ if $O_t \neq o_i$. For MNIST we use a process alphabet of up to size $|O| = 10$, and associate each symbol O to an image category, the measurement x_t are generated by first sampling the realization o_t from the process sampling a random image from the corresponding MNIST category.

For the case of rendering high-dimensional measurements with images, we use the full image dataset for sampling X_t and use *train* and *val* sets for training and *test* for evaluation, showing generalization ability in image classification through this method, with *no explicit training on class label*.

Results: We compare the learned representations with end-to-end DQN [30] trained on length k sequences of actions-outputs observations \hat{Y}_t , single observation Y_t , the continuous sufficient statistics \hat{S} , and the causal states refinements \bar{S} . The difficulty level of these environments can be extended by increasing the class size $|Y|$ and the memory k . Table 2 reports the results for $|Y| = k = 1, 2, 4$. As can be appreciated from the first row, it is impossible to obtain more than random reward of 50 (25) without using memory, and the task can be successfully solved by stacking k frames as input as indicated by the second row, making the task fully-observable. The third and fourth row show the matching performance of our task-agnostic representation for the continuous \hat{s} and the discrete \bar{s} , respectively. We found for the Gaussian and MNIST results that vector quantization worked better than k-means, so the results are reported with VQ.

C.2 Doom at Additional Frequencies

We tune frequency f for the Doom environment and show results in Fig. 11. DRQN learns slightly on $f = 2$ and noticeably drops with longer frequencies, whereas our method (Causal States) exhibits consistently good performance at all frequencies.

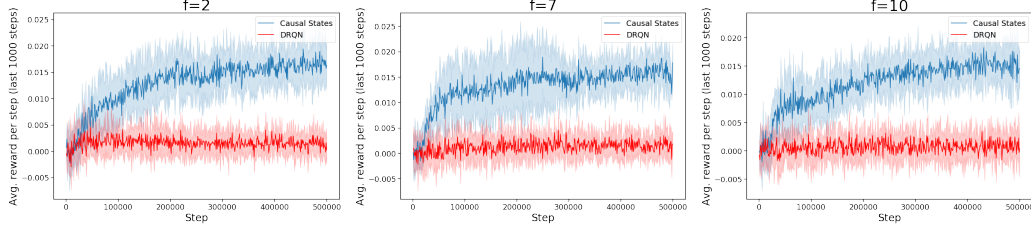


Figure 11: Performance of Causal States and DRQN on frequencies $f \in \{2, 7, 10\}$.

C.3 Planning

With a minimal sufficient unifilar model we can perform efficient planning by representing it as a labeled directed graph $\mathcal{G} = (\mathcal{S}, T_{ij}^{(o|a)})$, with causal states as nodes and action-observation conditional transitions as edges. Because of the unifilar property, once an agent has perfect knowledge of the current causal states, the information in the future action-observation process are sufficient to uniquely determine the future causal states. This property can be exploited by multi-step planning algorithms which need not keeping track of potential stochastic transitions between the underlying states, enabling a variety of methods that are otherwise amenable only to MDPs like Dijkstra’s algorithm. We plan over our learned discrete representation by building a graph $G := (V, E)$ where $V := \{\bar{\mathbf{S}}\}$, $E := \{(\bar{s}_i, \bar{s}_j); s_i, s_j \in \bar{\mathbf{S}}\}$, and $p(s_j | s_i, a) > 0$ for $a \in \mathbf{A}$. We obtain the optimal policy for Layout 1 and 2 obtaining respectively 0.5 and 0.3 reward we derive G and save high reward states seen during the rollouts as goal states. Then one can map the initial and final observations to a node in the graph and run Dijkstra’s algorithm to find the shortest path as proposed in [42]. Unlike value iteration, this requires no learning and no re-sampling of the environment by making use of the graph edges, where Q-learning only uses the nodes.