COMPLEX REASONING TASKS HAVE DIRECTIONAL STRUCTURE

Anonymous authors
Paper under double-blind review

ABSTRACT

While many datasets are believed to lie near low-dimensional manifolds, many different geometric structures can explain the same observations. What differentiates one manifold from another is its geometry: which inputs are close together and which are far apart. In language, this locality is typically defined by word co-occurrences. As linguist J. R. Firth put it, "You shall know a word by the company it keeps." In vision, locality is often defined via data augmentations: images that differ by a random crop or a small jitter should be similar. However, how to define this locality for downstream decision-making tasks is unclear.

Decision-making, is directional: one state can be close to another in the direction of task progression, but not vice versa. Thus, we argue that to represent sequential tasks, a manifold requires a notion of direction in addition to static representations. In this paper, we show how to equip contrastive reinforcement learning with bits about time, regularizing the learned representations. We show that this regularization improves performance on complex reasoning tasks, reducing the number of steps required to solve the Rubik's Cube by 26% and improving learning efficiency.

1 INTRODUCTION

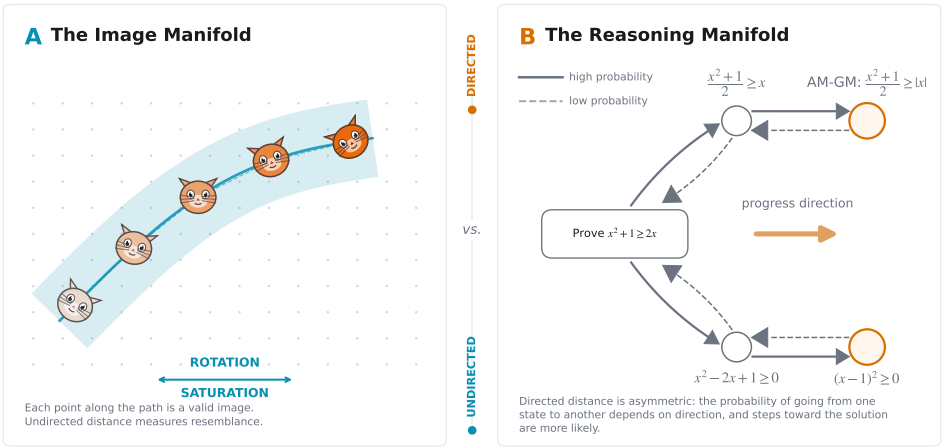

Figure 1: **Image manifolds vs. reasoning manifolds.** (*Left*) On an image manifold, nearby points are related by the undirected distance. (*Right*) On a reasoning manifold, closeness is directed: the distance when moving from a state towards a solution should be smaller than in the other way.

The manifold hypothesis states that real-world high-dimensional data often lies near a low-dimensional manifold (Tenenbaum et al., 2000; Cayton, 2005; Roweis & Saul, 2000) (Figure 1, left). In perceptual domains, this has an intuitive interpretation: although an image may be represented by many pixels, its meaningful variations can often be described by a small number of latent factors, such as color, shape, pose, or texture. Learning such a manifold is useful not only for compressing data, but also for recovering a geometry in which distances and paths reflect meaningful relationships between inputs.

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

Sequential tasks require a different kind of geometry (Figure 1, right). Recent progress in machine learning has increasingly focused on temporal domains, such as combinatorial puzzles, mathematical reasoning, and manipulation (Qu et al., 2025; DeepSeek-AI, 2025; Czechowski et al., 2024). In these tasks, what matters is not whether two states look similar, but whether one can be transformed into another through a sequence of valid steps. Moreover, the temporal relationship between states can be directional: in datasets whose trajectories consistently progress toward a goal, a transition from one state to another may be common while the reverse transition is unlikely to appear. Such relationships cannot, in general, be fully represented by a symmetric distance. We therefore argue that representations for sequential tasks should encode not only which states are related, but also the direction in which they evolve.

To approach this problem, we build on prior work using temporal contrastive learning for control and planning (Ziarko et al., 2025; Eysenbach et al., 2022). We hypothesize that these approaches struggle to find efficient solutions because they fail to capture directionality: for example, they require 460 steps on average to solve the Rubik's Cube, while optimal solutions require at most 26 steps. We introduce a new temporal contrastive learning approach that embeds directionality directly into the representations through an objective that, in special cases, can be interpreted as a latent predictive objective. We introduce a measure of dataset directionality and, using controlled environments, show that on highly directional datasets, a critic capable of modeling directionality is crucial for achieving optimal performance. We perform numerical experiments on five complex reasoning tasks. We show that our method improves data efficiency and is competitive in terms of final performance, and on the Rubik's Cube it reduces the average solution length from 460 to 340 steps, a 26% improvement.

2 Related Work

2.1 Contrastive Learning

Contrastive learning has long been used to learn representations for structured datasets, by pulling related things closer in the latent space while pushing unrelated ones apart. In this work, we use the InfoNCE objective (Sohn, 2016), which is able to capture classification-relevant structure. (Chen et al., 2020; Radford et al., 2021; He et al., 2020) Our work extends prior research on applying contrastive learning to temporal domains. (Oord et al., 2018; Sermanet et al., 2018; Eysenbach et al., 2022; Assran et al., 2023). The closest related work is CRTR (Ziarko et al., 2025), which demonstrates that contrastive representations can reflect the structure of complex reasoning tasks. We extend this line of work by proposing a contrastive learning approach that captures the directionality of a dataset.

2.2 Self-supervised Sequence Modelling Methods

The predominant approach for modelling sequential data is generative modelling (OpenAI et al., 2024; Devlin et al., 2019; Nie et al., 2025). While this paradigm has demonstrated impressive capabilities, recent work has argued for predicting the next latent state, which may enable more efficient architectures. Latent prediction objectives have been shown to be effective in Reinforcement Learning (RL) (Bagatella et al., 2025), language modeling (Huang et al., 2025) and video modeling (Li et al., 2025). Prior work has also shown that it is possible to change the contrastive objective so that it is possible to infer intermediate points between the start and the goal of a trajectory. (Eysenbach et al., 2025) We instead propose a change to the contrastive objective that allows representing the direction of the temporal evolution of a dataset.

2.3 Restricting the Hypothesis Space

Restricting the hypothesis space of neural networks has been shown to improve learning efficiency. (Hinton et al., 2012; Ng, 2004) One common approach is to introduce an auxiliary regularising loss. In RL such auxiliary objectives have been shown to significantly improve sample efficiency (Jaderberg et al., 2016). The line of work most relevant to this paper regularises the learning through representation learning objectives. In particular, several methods introduce latent predictive losses on intermediate representations in addition to the reinforcement learning signal. (Schwarzer et al., 2021; Voelcker et al., 2024; Tang et al., 2023; Ni et al., 2024) The mechanism of our method is similar to these approaches, and as shown in Section 6 it also improves learning efficiency, however, its main objective is to model directionality.

3 PRELIMINARIES

3.1 COMPLEX REASONING PROBLEMS

In this paper we focus on solving complex reasoning problems. The problems considered are: Sokoban, Rubik's Cube, N-Puzzle, Lights Out, and Digit Jumper. We formulate them as a goal-conditioned Markov Decision Process $(\mathcal{S}, \mathcal{A}, P, p_0, \mathcal{G})$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, P is the transition function, p_0 is the initial state distribution and $\mathcal{G} \subseteq \mathcal{S}$ is the goal space. We assume a finite action space and a known, deterministic transition function P. These assumptions are needed, because they allow us to solve the problems by using only a critic, without the need for a separate planner module (for instance a policy function).

We further assume that given a state s, there is a single goal state $g_s \in \mathcal{G}$ that is reachable from s. This assumption is weaker than it seems, since in principle it is always possible to introduce an artificial goal state shared between all the states.

We treat the problem as an offline RL problem, where a dataset $\mathcal{D} = \{\tau_i\}_i^N$, τ_i is given. We assume that for each trajectory $\tau_i = (s_1, \ldots, s_m)$, $s_m \in \mathcal{G}$. We will call such a dataset an offline reasoning dataset.

Let's denote the set of states reachable from s by $\mathcal{R}(s) = \{s' \in \mathcal{S} : \exists t \in \mathbb{N}, a_1, \ldots, a_t \in \mathcal{A} \text{ such that } s' = P(P(\ldots P(s, a_0), \ldots a_{t-1}), a_t)\}$. We will call a transition from s to s' irreversible if $s \notin \mathcal{R}(s')$.

Let's denote $d_*(s, s')$ to be the minimal number of actions needed to reach s' from s. We will call an action a optimal in a state s, if $d_*(s, g_s) = d_*(P(s, a), g_s) + 1$. We denote the set of actions optimal in a state s by $\mathcal{A}^*(s)$.

3.2 CONTRASTIVE REINFORCEMENT LEARNING

Contrastive Reinforcement Learning (CRL) (Eysenbach et al., 2022) is a method for learning representations of datasets with a temporal component. Its main property is that upon convergence, the distances in the latent space mirror the distances in the state space. CRTR (Ziarko et al., 2025) is a formulation of CRL that enables using these representatons for solving complex reasoning problems, and we are using the conventions proposed there. We assume that a discount factor $\gamma \in (0, 1)$ is given.

For an anchor state s_t, such that $\tau \sim \mathcal{D}$, $s_t \sim \tau$, the positive examples are $s_{t+\Delta}$ where $\Delta \sim \text{Geom}(1-\gamma)$. Negative pairs are sampled uniformly from the dataset $\mathcal{D}$. The objective is to learn to classify whether a state is a positive or a negative example: $\mathcal{L}_{CRL} = \mathbb{E}_{(s_t,a_t)\sim\mathcal{D},\Delta\sim\text{Geom}(1-\gamma),s_i\sim\mathcal{D}}\left[-\log \frac{e^{d(s_t,s_{t+\Delta})}}{\sum_{s_i} e^{d(s_t,s_i)}}\right]$. Minimizing this objective results in a score function $d(s, s') = \frac{p_{t+}(s'|s_t)}{p(s')}$, where p_{t+} is the discounted state occupancy measure.

We will call this type of critic a symmetric critic, because $d(s, s') = ||\phi(s) - \phi(s')||_2^2 = d(s', s)$. Assuming for simplicity that $p(s)$ is roughly equal to $p(s')$, if in a given dataset $p_{t+}(s'|s) \neq p_{t+}(s|s')$, then the distance learned by a symmetric critic will not be able to capture the geometry correctly.

In the original CRL paper, the critic is parametrized differently, as there are two neural networks, $\phi, \phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^k$ and $\psi, \psi : \mathcal{G} \to \mathbb{R}^k$, and the critic is parametrized as $d((s, a), s') = ||\phi(s, a) - \psi(s')||_2^2$. Such a critic will be called an asymmetric critic.

4 WHEN DOES UNDIRECTED GEOMETRY FAIL?

Offline reasoning datasets vary in how strongly their trajectories consistently progress toward a solution and the extent to which they are composed of optimal actions. In this section, we formalize this distinction through the *directionality* of a dataset and demonstrate that a symmetric contrastive critic might fail to capture it. We consider two didactic experiments. First, we construct an ordered-subgoal task and study how dataset directionality affects the performance. Then, using a tree-like

task as a simplified model of the geometry of reasoning, we investigate the failure that arises when such a geometry is learned with a symmetric contrastive critic.

4.1 Directionality Definition

Directionality measures how strongly a dataset favors actions that make progress toward the goal. We define dataset directionality as the average, across states, probability that the action observed in the dataset is optimal.

Definition 1 (Dataset Directionality)**.** For a dataset D, we define the dataset directionality $\mathcal{D}(D) = \frac{1}{|\mathcal{S}_D|} \sum_{s \in \mathcal{S}_D} P(A_t \in \mathcal{A}^*(s) \mid S_t = s)$, where $\mathcal{A}^*(s)$ denotes the set of optimal actions in s.

Directionality is a property of the dataset, while symmetry is a property of the critic function. We show that sufficiently directional datasets necessarily contain asymmetric temporal relationships.

Theorem 2 (Directionality Implies Asymmetry)**.** *Denote the averaged probability mass of irreversible transitions by $\delta = \frac{1}{|S_D|} \sum_{s,s' \in D, s \notin \mathcal{R}(s')} P(s'|s)$. If $\mathcal{D}(D) > \frac{1+\delta}{2}$ then there exists a pair of states s, s', such that $g_s = g_{s'}$, for which $P(s \mid s') \neq P(s' \mid s)$, which are reachable from each other.*

The proof is provided in Appendix B.

The theorem therefore links dataset directionality to representational asymmetry: above the threshold $\frac{1+\delta}{2}$, for some reversible state pair going in one direction is more probable than going in the other. Importantly, the converse does not hold: $P(s \mid s') \neq P(s' \mid s)$ does not necessarily imply high directionality. Directionality is a more specific property, capturing dataset's bias toward progress to a goal.

In the next section, we test how the performance of a symmetric critic changes with dataset directionality.

4.2 Symmetric Critics Fail to Capture Highly Directional Data

We consider a didactic maze environment where the goal is for the agent to stand on three different subgoals in a fixed order (See App. 10). Each trajectory uses a different maze layout. We generate datasets with varying levels of directionality. For a given $\epsilon \in 0, 0.2, 0.4, 0.6, 0.8$ we generate a trajectory by taking an optimal action with probability $1 - \epsilon$ and a random action with probability ϵ.

For each dataset, we train two contrastive critics: a symmetric critic and a directional critic that can distinguish forward from backward relationships. We introduce the directional parametrization in detail in Section 5. At evaluation, we solve the task greedily by choosing the action whose successor state the critic predicts to be closest to the completed-goal state.

Figure 2 shows that the two critics perform similarly on less directional datasets. As dataset directionality increases, however, the directional critic achieves substantially higher success rates. Moreover, the performance gap between the two critics increases with the measured directionality of the dataset.

We conclude that in certain environments, as the dataset becomes more directional, making the critic directional is necessary in order to achieve an optimal performance.

4.3 A Geometric View of Directionality: Trees

Many reasoning problems can be abstracted as a tree-like hierarchy of states, where trajectories progress from many initial states (leaves) toward a common goal (the root). In this setting, a dataset with directionality one always moves upward in the tree, with each transition reaching a state one level shallower than the previous one.

We construct a tree where each node is represented by a binary encoding of its position with 1 denoting going left from the current node and 0 going right (See App. 11). We train the contrastive critic on reversed trajectories generated by starting from the root and going down using a random child of the current node. At evaluation, the agent starts at a leaf and acts greedily toward the root under the learned distance.

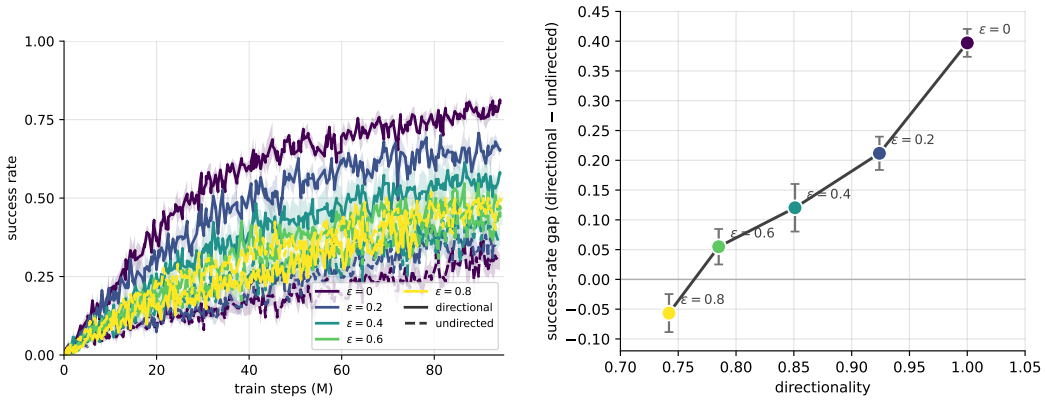

Figure 2: **Directional vs. undirected critic parametrization on Colorful Maze.** As dataset directionality increases, a directional critic parametrization outperforms an undirected one. Averaged over 5 seeds with standard error.

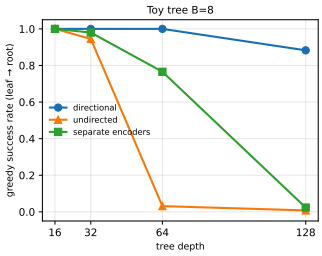

Figure 3: **Directional critics scale to deeper trees than undirected critics.** Success rate of finding a path from a leaf to the root using the learned distance, for different contrastive critic parameterizations.

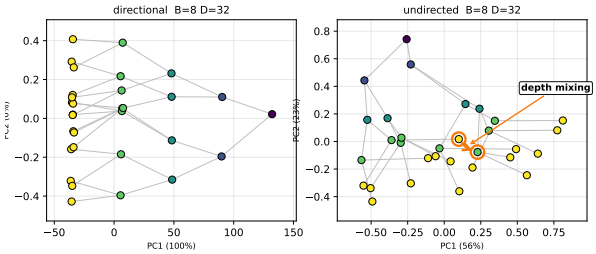

Figure 4: **Undirected critic's representations exhibit mixing of levels between parents and children.** PCA of the critic's representations for a small subtree for the undirected and directional critic.

In Figure 3 we show the success rate of the critic as a function of the tree depth. We find that an undirected critic starts failing to solve the task as the tree grows deeper. Figure 4 shows that this failure might be caused by mixing states from different depths of the tree.

While these didactic examples illustrate why undirected representations are limited, they do not demonstrate how directedness should be built into the representation geometry. In the next section, we will introduce a technique for elegantly introducing directedness.

5 Choosing an Asymmetric Parametrization

Section 4 shows that a symmetric critic might fail to exploit the directionality of a dataset. A straightforward solution is to use separate encoders for the state and goal, as in standard CRL. Such a critic is no longer constrained to satisfy $d(s, g) = d(g, s)$, and can therefore represent the asymmetric temporal relationships induced by highly directional datasets. We will show that this approach doesn't necessarily lead to an improvement in performance in Section 5.1. Therefore, in Section 5.2, we introduce Predictive Temporal Contrastive Representations (PTCR), a critic parametrization to capture directionality.

5.1 Asymmetry Alone Is Not Sufficient

A straightforward way to model asymmetry is to use separate state and goal encoders, as in standard contrastive RL. However, these encoders are completely unconstrained relative to one another. CRL constrains their representations to be approximately correct on the training distribution, but as tasks become more complex, this distribution covers a smaller part of the environment. In line with thise, we find, that while in the shallower tree environment, the performance is better when compared to the symmetric baseline, as the depth gets larger, it declines quicker than the directional approach (See Fig. 3). The asymmetric critic also severly underperforms the symmetric one on other environments, as shown in Appendix H. We therefore seek a more structured asymmetric parametrization, which retains a shared representation space while modeling temporal directionality.

5.2 A Parametrization To Capture the Arrow of Time

In this section, we introduce a new learning algorithm that extends CRTR to acquire representations aware of the arrow of time.

Eysenbach et al. (2025) showed that when temporal contrastive learning is trained with the scoring function $d(s_t, s_{t+\Delta}) = ||A\phi(s_t) - \phi(s_{t+\Delta})||_2^2$, the A matrix can be used to predict the distribution of directions in which the trajectory representations evolve from the current latent state.

Our goal is to extend this idea into a method that learns representations that enable solving complex reasoning tasks by introducing the awareness of the arrow of time. Rather than modeling only the overall direction of the trajectories, we aim to capture their gradual evolution in the latent space. We define an encoder $\phi : \mathcal{S} \to \mathbb{R}^k$ and latent predictor $f_\theta : \mathbb{R}^k \to \mathbb{R}^k$, both parametrized by neural networks. To encode the temporal evolution, we directly incorporate time into the scoring function by repeatedly applying f_θ for each elapsed time step. We train the networks using the contrastive objective defined in Section 3.2, with the scoring function $d(s_t, s_{t+\Delta}) = ||f_\theta^\Delta(\phi(s_t)) - \phi(s_{t+\Delta})||_2^2$,

where Δ is the number of elapsed time steps between the two states. We optimize the parameters of ϕ and f_θ jointly, using gradient descent.

The connection between this objective and the latent predictive objective is explained in Appendix C Intuitively, this connection shows that different latent states evolve in similar ways, as they are mapped through the same latent predictor, and therefore this is well suited to capture directedness, a property where the trajectories tend to progress in a specific direction.

5.3 The Practical Setup

In practice, we take f_θ to be linear and write $f_\theta^\Delta = QA^\Delta Q^\top$, where A is a ***fixed*** block-diagonal rotation matrix and Q is a ***trainable*** orthogonal change of basis. This construction is motivated by Rotary Positional Embeddings (RoPE) (Su et al., 2023) and by Koopman operator theory, which suggests that nonlinear dynamics can admit linear representations (Brunton et al., 2021; Koopman, 1931). The technical details are provided in Appendix D.

6 Does fixing the geometry help control?

In this section we demonstrate that the representation directionality is important for learning to solve complex reasoning tasks. We do it through considering the following research questions:

- Does learning a directional critic result in better performance and data efficiency on complex reasoning tasks? (Sections 6.2 and 6.3)
- How does the directionality of a dataset affect the performance of a critic? (Section 6.4)
- What are the properties of the representations learned by PTCR? (Section 6.5)

We will start with explaining the experimental setup.

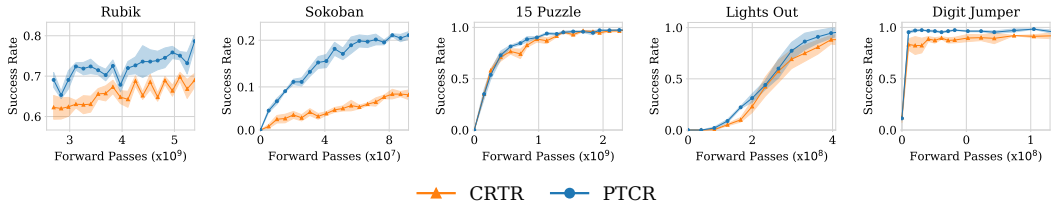

Figure 5: **PCTR is more data efficient than CRTR.** X axis shows the training steps, Y axis shows the percentage of problems solved within a predefined length. Averaged over 5 seeds with standard error plotted.

6.1 Experimental Setup

Environments We focus our evaluation on a suite of challenging complex reasoning environments, including Sokoban (Dor & Zwick, 1999), the Rubik's Cube, N-Puzzle (Johnson & Story, 1879), Lights Out (Anderson & Feil, 1998), and Digit Jumper (Bagatella et al., 2021). Some of these problems are NP-hard (Demaine et al., 2018; Culberson, 1998; Ratner & Warmuth, 1986) and they have been widely adopted as benchmarks for reinforcement learning and planning (Agostinelli et al., 2019; Racanière et al., 2017; Zawalski et al., 2024; Ziarko et al., 2025). For detailed environment descriptions, see Appendix A.

Baselines We compare our method with the following baselines: CRTR (Ziarko et al., 2025) is based on contrastive reinforcement learning with in-trajectory negatives, DeepCubeA (Agostinelli et al., 2019) learns a value function via iterative one-step lookahead, Supervised (Czechowski et al., 2024) learns a value function by predicting the distance from each state to the solution. For the fairness of the comparison, PTCR uses the same architecture as CRTR and the supervised baseline. Appendix E contains the details of the training procedure.

Evaluation Algorithm We evaluate the methods by iteratively applying each possible action in a given state. We only select the one deemed closest to the goal. To avoid cycles, we maintain a set of visited states. For supervised and DeepCubeA baselines, the network returns the distance to the solution, while for CRTR and PTCR we compute the distance in the latent space using the score function. For PTCR, during training we calculate the score function as $d(s_t, s_{t+\Delta}) = ||A^{\Delta}\phi(s_t) - \phi(s_{t+\Delta})||_2^2$. However, during evaluation we use $d(s_t, s_{t+\Delta}) = ||A\phi(s_t) - \phi(s_{t+\Delta})||_2^2$, independent of Δ. This choice retains the directional information encoded by A. Moreover, because Δ is sampled geometrically during the training, $\Delta = 1$ is the most frequent value, likely making A^1 align more closely with the correct direction than alternatives. While this is a heuristic, rather than a theoretically principled inference rule, we find it effective in practice. Appendix F contains additional details of the evaluation procedure.

Code for reproducing experiments is available at: `https://anonymous.4open.science/r/PTCR-7C8E/README.md`.

6.2 Performance on Complex Reasoning Tasks

In this section we evaluate whether our proposed modification yields an effective method for solving complex reasoning tasks. We run the solving on 1000 problem instances for each of the environments.

We report cumulative solution length distributions, where the y-axis denotes the fraction of problems solved with the solution length $\leq n$ and n is shown the x-axis. If the task was not solved within 1000 steps, the solution length is set to infinity. Figure 6 presents these results for each of the environments. We average the results over 5 seeds with standard error plotted.

We observe, that in each of the tasks considered, our method matches or outperforms the baselines. In particular, in the case of the Rubik's Cube, the average solution length is 340 in comparision to 460 achieved by CRTR (counting only the problems that were solved), corresponding to a 26% reduction. This is a fair comparison, as instances solved by CRTR are a subset of those solved by PTCR.

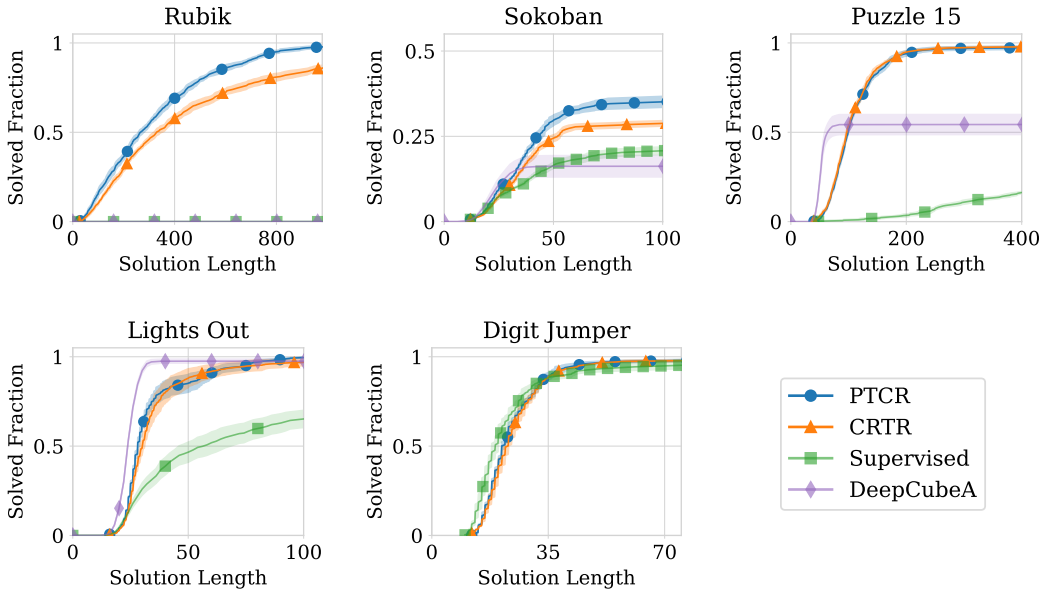

Figure 6: **PTCR performs well in all the tasks.** Cumulative fraction of instances solved with a solution length $\leq n$, by environment.

These results indicate that our method yields competitive performance on complex reasoning tasks.

6.3 Data Efficiency

In this section we evaluate whether our proposed objective improves the data efficiency of the training, when compared to CRTR. In Figure 5, we evaluate the performance on 20 intermediate checkpoints from the training of CRTR and PTCR. On the Rubik's Cube we sample starting from mid-training, as early checkpoints struggle to solve the task. We sample the checkpoints uniformly from the training duration. We evaluate each checkpoint on 100 problem instances. For each checkpoint we report the percentage of problems solved within a predefined length (1000 for Sokoban, 1000 for the Rubik's Cube, 500 for 15-Puzzle, 75 for Lights Out, 50 for Digit Jumper), averaged over 5 seeds with standard error.

In all the tasks, PTCR is more data efficient than CRTR, with the biggest difference visible in Sokoban and Digit Jumper. Appendix G reports the same through-training plots under other solution length cutoffs.

6.4 The Dependence Between Directionality and Improvement in Performance

In sections 6.3 and 6.2 it is clearly visible that on some of the environments our method helps more than on others. We estimate the directionality of the datasets used for these five environments to help explain this variance. The exact estimation process is described in App. I. This estimation procedure results in values of directionality of 0.973, 0.965, 0.934, 0.829 and 0.648 for correspondingly Sokoban, Rubik's Cube, Lights Out, Digit Jumper and 15-puzzle.

This partially aligns with the empirical results: for 15-puzzle there is no improvement in final performance or training efficiency, for Digit Jumper there is some improvement and the improvement for Sokoban and the Rubik's Cube is significant. We investigate what happens in Lights Out.

First of all, if we have a complex reasoning problem, there always exists a representation ϕ of the problem in the standard euclidean space, such that for each s, $d_*(s, g_s) = || \phi(s) - \phi(g_s) ||_2^2$. It sufices to map g_s to 0 and each s to $d_*(s, g_s)$. It is learning this distance from the data that is challenging. In Appendix K we show that for lights out, it is not only that CRTR achieves a good performance in Lights Out, but rather that it exhibits a grokking-like (Power et al., 2022) behavior, where it is possible to decode the optimal path to the solution from the internal network representations by using

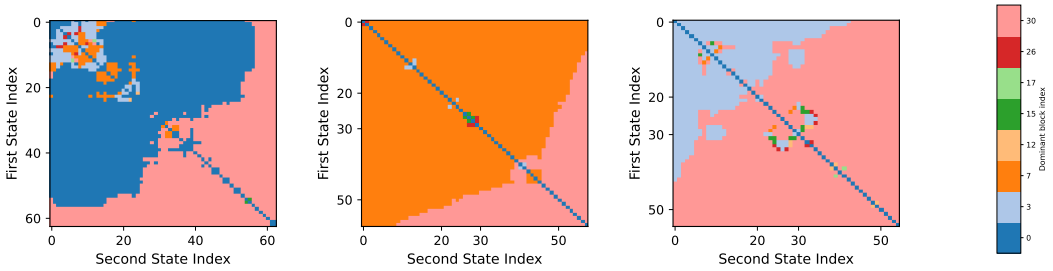

Figure 8: **The latent space of PTCR seems to display signs of specialization.** Sokoban: top-contributing blocks for each pair of statesfor three trajectories.

linear probing. This is possible because Lights Out has a particular structure — the optimal solution can be found through simple matrix operations, which neural networks have been shown to be able to learn in previous work. (Cohen et al., 2025)

6.5 Analyzing the Learnt Representations

Comparing singular values. To verify the hypothesis that PTCR encourages representations that focus more on the temporal aspect of the task, we apply SVD to representations learned by both CRTR and PTCR on the Rubik's Cube. We plot the singular values from this decomposition in Figure 7. PTCR utilizes much fewer dimensions (40 vs 64) than CRTR. We hypothesize that thanks to the awareness of the time direction, PTCR is able to learn a more temporally structured representation of the Rubik's Cube, allowing it to be more compressed.

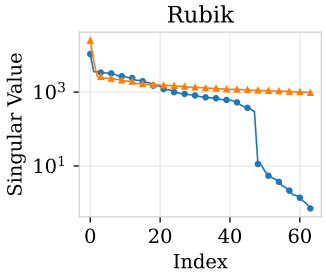

Figure 7: **PTCR utilizes fewer dimensions than CRTR on the Rubik's Cube.** Singular-value spectrum on Rubik's Cube for CRTR and PTCR.

Block specialization. We can write the score function as a sum of scores from individual blocks that rotate by different angles (See Section 5.3). $||\phi(s) - A\phi(s')||_2^2 = \sum_{i=1}^{\frac{k}{2}} ||A_{2i:2(i+1),2i:2(i+1)}\phi(s)_{2i:2(i+1)} - \phi(s')_{2i:2(i+1)}||_2^2$ In Figure 8, we sample three trajectories from the test set of the Sokoban dataset and compute the distance between every two states of the trajectory.

In the plots, on the x-axis, we have the index of the first state from a given trajectory, and on the y-axis, we have the index of the second state. As described above, we think of the score function as the sum of scores from each of the two-dimensional blocks. We choose the color of each of the points based on the block number that contributes the most to this sum. We can observe that when we calculate the distance between a state and another state that comes from the latter part of the trajectory, the block that contributes the most turns out to be the same. We hypothesize that this might be a preliminary sign that different blocks might have different purposes when an asymmetry in the latent space is introduced. Here, we discard A during distance calculation, in order to look exclusively at the static ϕ structure.

7 Discussion

In this paper, we developed a contrastive RL method that explicitly accounts for the arrow of time by way of a latent predictive objective. Our approach was motivated by the observation that CRTR, an existing contrastive RL model, learns task structure but fails to fully exploit it in prediction. We were able to demonstrate a number of improvements over CRTR with improved learning efficiency and performance.

Limitations. We focused on using contrastive RL for fully-observed, deterministic environments. However, real-world domains are often stochastic (e.g., robotics, autonomous driving), making it important to test the efficacy of our method in stochastic environments. It will also be important to extend our method to account for partial observability.

AI USE STATEMENT

This work utilized AI to aid with the spelling and grammar checking of the manuscript. Code written for running experiments and generating figures was partially written using AI, but the Pull Requests were manually reviewed and edited to ensure correctness. Authors take full responsibility for the content of the manuscript.

ETHICS STATEMENT

This work studies representation learning and planning on synthetic combinatorial puzzle environments and does not involve human subjects, sensitive personal data, or dual-use applications beyond standard ML research risks.

REPRODUCIBILITY STATEMENT

We provide experimental details, architecture choices, and training procedures in Section 6 and Appendix E. Proofs appear in Appendix B. We also share the code for running the main experiments in Section 6. The computational resources utilized are specified in Appendix J.

REFERENCES

Forest Agostinelli, Stephen McAleer, Alexander Shmakov, and Pierre Baldi. Solving the rubik's cube with deep reinforcement learning and search. *Nat. Mach. Intell.*, 1(8):356–363, 2019. doi: 10.1038/S42256-019-0070-Z. URL https://doi.org/10.1038/s42256-019-0070-z.

Marlow Anderson and Todd Feil. Turning lights out with linear algebra. *Mathematics magazine*, 71 (4):300–303, 1998.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15619–15629, 2023.

Marco Bagatella, Miroslav Olšák, Michal Rolínek, and Georg Martius. Planning from pixels in environments with combinatorially hard search spaces. *Advances in Neural Information Processing Systems*, 34:24707–24718, 2021.

Marco Bagatella, Matteo Pirotta, Ahmed Touati, Alessandro Lazaric, and Andrea Tirinzoni. Td-jepa: Latent-predictive representations for zero-shot reinforcement learning, 2025. URL https://arxiv.org/abs/2510.00739.

Margherita Barile. Lights out puzzle. https://mathworld.wolfram.com/LightsOutPuzzle.html, n.d. From MathWorld—A Wolfram Resource, created by Eric W. Weisstein.

Steven L. Brunton, Marko Budišić, Eurika Kaiser, and J. Nathan Kutz. Modern koopman theory for dynamical systems, 2021. URL https://arxiv.org/abs/2102.12086.

Lawrence Cayton. Algorithms for manifold learning. 2005. URL https://api.semanticscholar.org/CorpusID:408872.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, Proceedings of Machine Learning Research, pp. 1597–1607. PMLR, 2020. URL http://proceedings.mlr.press/v119/chen20j.html.

Andrew Cohen, Andrey Gromov, Kaiyu Yang, and Yuandong Tian. Spectral journey: How transformers predict the shortest path. *CoRR*, abs/2502.08794, 2025. doi: 10.48550/ARXIV.2502.08794. URL https://doi.org/10.48550/arXiv.2502.08794.

Drissa Coulibaly. Lights out: An application of linear algebra over a finite field, 2023. URL https://people.math.ethz.ch/~acannas/Student_Papers/BSc_Theses/2022_bsc_coulibaly_lights_out.pdf.

J. Culberson. Sokoban is pspace-complete. In *Proceedings of the International Conference on Fun with Algorithms*, pp. 65–76, Elba, Italy, June 1998. Carelton Scientific.

Konrad Czechowski, Tomasz Odrzygóźdź, Marek Zbysiński, Michał Zawalski, Krzysztof Olejnik, Yuhuai Wu, Łukasz Kuciński, and Piotr Miłoś. Subgoal search for complex reasoning tasks, 2024. URL https://arxiv.org/abs/2108.11204.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948, 2025. doi: 10.48550/ARXIV.2501.12948. URL https://doi.org/10.48550/arXiv.2501.12948.

Erik D. Demaine, Sarah Eisenstat, and Mikhail Rudoy. Solving the Rubik's Cube Optimally is NP-complete. In Rolf Niedermeier and Brigitte Vallée (eds.), *35th Symposium on Theoretical Aspects of Computer Science (STACS 2018)*, volume 96 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 24:1–24:13, Dagstuhl, Germany, 2018. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. ISBN 978-3-95977-062-0. doi: 10.4230/LIPIcs.STACS.2018.24. URL https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.STACS.2018.24.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pp. 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/V1/N19-1423. URL https://doi.org/10.18653/v1/n19-1423.

Dorit Dor and Uri Zwick. Sokoban and other motion planning problems. *Comput. Geom.*, 13(4): 215–228, 1999. doi: 10.1016/S0925-7721(99)00017-6. URL https://doi.org/10.1016/S0925-7721(99)00017-6.

Benjamin Eysenbach, Tianjun Zhang, Sergey Levine, and Ruslan Salakhutdinov. Contrastive learning as goal-conditioned reinforcement learning. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/e7663e974c4ee7a2b475a4775201ce1f-Abstract-Conference.html.

Benjamin Eysenbach, Vivek Myers, Ruslan Salakhutdinov, and Sergey Levine. Inference via interpolation: Contrastive representations provably enable planning and inference, 2025. URL https://arxiv.org/abs/2403.04082.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, 2015. URL https://arxiv.org/abs/1502.01852.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pp. 9726–9735. Computer Vision Foundation / IEEE, 2020. doi: 10.1109/CVPR42600.2020.00975. URL https://doi.org/10.1109/CVPR42600.2020.00975.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors, 2012. URL https://arxiv.org/abs/1207.0580.

Hai Huang, Yann LeCun, and Randall Balestriero. Llm-jepa: Large language models meet joint embedding predictive architectures, 2025. URL https://arxiv.org/abs/2509.14252.

Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks, 2016. URL https://arxiv.org/abs/1611.05397.

Wm. Woolsey Johnson and William E. Story. Notes on the "15" puzzle. *American Journal of Mathematics*, 2(4):397–404, 1879. ISSN 00029327, 10806377. URL http://www.jstor.org/stable/2369492.

B. O. Koopman. Hamiltonian systems and transformation in hilbert space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, 1931. doi: 10.1073/pnas.17.5.315. URL https://www.pnas.org/doi/abs/10.1073/pnas.17.5.315.

Xianhang Li, Chen Huang, Chun-Liang Li, Eran Malach, Josh Susskind, Vimal Thilak, and Etai Littwin. Rethinking jepa: Compute-efficient video ssl with frozen teachers, 2025. URL https://arxiv.org/abs/2509.24317.

Andrew Ng. Feature selection, l 1 vs. l 2 regularization, and rotational invariance. *Proceedings of the Twenty-First International Conference on Machine Learning*, 09 2004. doi: 10.1145/1015330.1015435.

Tianwei Ni, Benjamin Eysenbach, Erfan Seyedsalehi, Michel Ma, Clement Gehring, Aditya Mahajan, and Pierre-Luc Bacon. Bridging state and history representations: Understanding self-predictive RL. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=ms0VgzSGF2.

Shen Nie, Fengqi Zhu, Chao Du, Tianyu Pang, Qian Liu, Guangtao Zeng, Min Lin, and Chongxuan Li. Scaling up masked diffusion models on text, 2025. URL https://arxiv.org/abs/2410.18514.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo

Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on small algorithmic datasets. *CoRR*, abs/2201.02177, 2022. URL https://arxiv.org/abs/2201.02177.

Yuxiao Qu, Anikait Singh, Yoonho Lee, Amrith Setlur, Ruslan Salakhutdinov, Chelsea Finn, and Aviral Kumar. RLAD: training llms to discover abstractions for solving reasoning problems. *CoRR*, abs/2510.02263, 2025. doi: 10.48550/ARXIV.2510.02263. URL https://doi.org/10.48550/arXiv.2510.02263.

Sébastien Racanière, Theophane Weber, David P. Reichert, Lars Buesing, Arthur Guez, Danilo Jimenez Rezende, Adrià Puigdomènech Badia, Oriol Vinyals, Nicolas Heess, Yujia Li, Razvan Pascanu, Peter W. Battaglia, Demis Hassabis, David Silver, and Daan Wierstra. Imagination-augmented agents for deep reinforcement learning. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 5690–5701, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/9e82757e9a1c12cb710ad680db11f6f1-Abstract.html.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, Proceedings of Machine Learning Research, pp. 8748–8763. PMLR, 2021. URL http://proceedings.mlr.press/v139/radford21a.html.

Daniel Ratner and Manfred K. Warmuth. Finding a shortest solution for the N × N extension of the 15-puzzle is intractable. In Tom Kehler (ed.), *Proceedings of the 5th National Conference on Artificial Intelligence. Philadelphia, PA, USA, August 11-15, 1986. Volume 1: Science*, pp. 168–172. Morgan Kaufmann, 1986. URL http://www.aaai.org/Library/AAAI/1986/aaai86-027.php.

Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000. doi: 10.1126/science.290.5500.2323. URL https://www.science.org/doi/abs/10.1126/science.290.5500.2323.

Max Schwarzer, Ankesh Anand, Rishab Goel, R. Devon Hjelm, Aaron C. Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=uCQfPZwRaUu.

Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine, and Google Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 1134–1141. IEEE, 2018.

Kihyuk Sohn. Improved Deep Metric Learning With Multi-Class N-Pair Loss Objective. In *Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2023. URL https://arxiv.org/abs/2104.09864.

Yunhao Tang, Zhaohan Daniel Guo, Pierre Harvey Richemond, Bernardo Ávila Pires, Yash Chandak, Rémi Munos, Mark Rowland, Mohammad Gheshlaghi Azar, Charline Le Lan, Clare Lyle, András György, Shantanu Thakoor, Will Dabney, Bilal Piot, Daniele Calandriello, and Michal Valko. Understanding self-predictive learning for reinforcement learning. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, Proceedings of Machine Learning Research, pp. 33632–33656. PMLR, 2023. URL https://proceedings.mlr.press/v202/tang23d.html.

Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000. doi: 10.1126/science.290.5500.2319. URL https://www.science.org/doi/abs/10.1126/science.290.5500.2319.

Claas Voelcker, Tyler Kastner, Igor Gilitschenski, and Amir-massoud Farahmand. When does self-prediction help? understanding auxiliary tasks in reinforcement learning. *RLJ*, 4:1567–1597, 2024. URL https://rlj.cs.umass.edu/2024/papers/Paper197.html.

Michal Zawalski, Michal Tyrolski, Konrad Czechowski, Tomasz Odrzygozdz, Damian Stachura, Piotr Piekos, Yuhuai Wu, Łukasz Kucinski, and Piotr Milos. Fast and precise: Adjusting planning horizon with adaptive subgoal search, 2024. URL https://arxiv.org/abs/2206.00702.

Alicja Ziarko, Michal Bortkiewicz, Michal Zawalski, Benjamin Eysenbach, and Piotr Milos. Contrastive representations for temporal reasoning. *CoRR*, abs/2508.13113, 2025. doi: 10.48550/ARXIV.2508.13113. URL https://doi.org/10.48550/arXiv.2508.13113.

A Evaluation Environments

A.1 Environment Descriptions

- In Sokoban, the goal is to move boxes to the goal positions. Its main difficulty is that there exist dead ends — positions from which it's impossible to win.
- In the Rubik's Cube, the goal is to get from a randomly scrambled cube to the solved state by rotating the faces. The main difficulty is that the number of possible states is $4.3252 \cdot 10^{19}$.
- In 15-Puzzle, the goal is to get from a randomly scrambled puzzle to the solved state by moving the tiles. It's a permutation puzzle like the Rubik's Cube, but its mixing time is slower.
- In Lights Out, the goal is to turn all the lights off. Its main difference from other puzzles is that the actions are commutative.
- In Digit Jumper, the goal is to jump to the goal position by jumping from a state by n steps into a chosen direction, where n depends on the number the agent is currently at. This differs from other tasks because the actions move the agent by different numbers of steps.

A.2 Lights Out Visualization

In Figure 9 we visualize the Lights Out environment, showing an example of how actions affect the state of the environment.

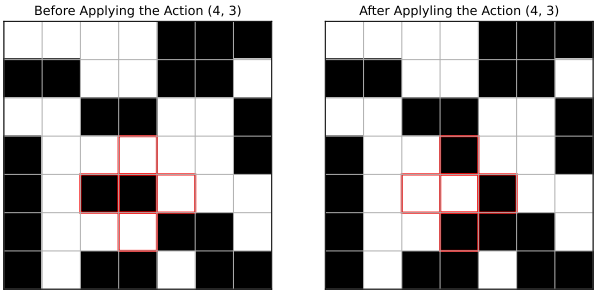

Figure 9: In Lights Out the goal is to turn all the lights off. At each step, the agent clicks on a button, which toggles the light on the button and the lights in the 4 adjacent cells.

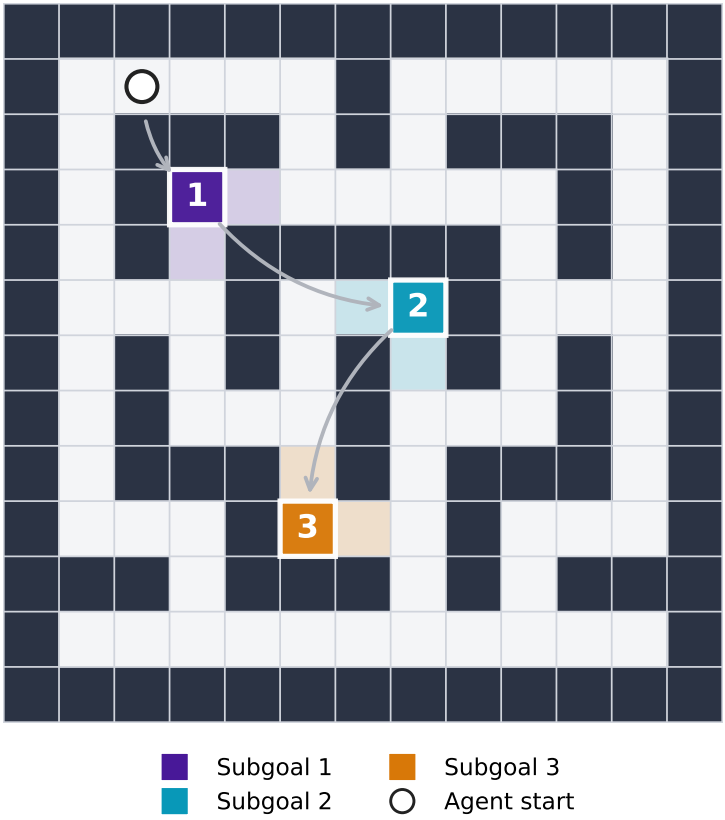

Figure 10: Colorful Maze: an ordered-subgoal grid environment. The agent starts in a random location and must visit subgoals $1 \rightarrow 2 \rightarrow 3$ in this order. The layout and subgoal locations are chosen randomly for each episode.

A.3 Colorful Maze Visualization

In Figure 10 we illustrate the Colorful Maze environment used in Section 4. The agent must visit three colored subgoals in a fixed order before the episode succeeds; different episodes use different maze layouts.

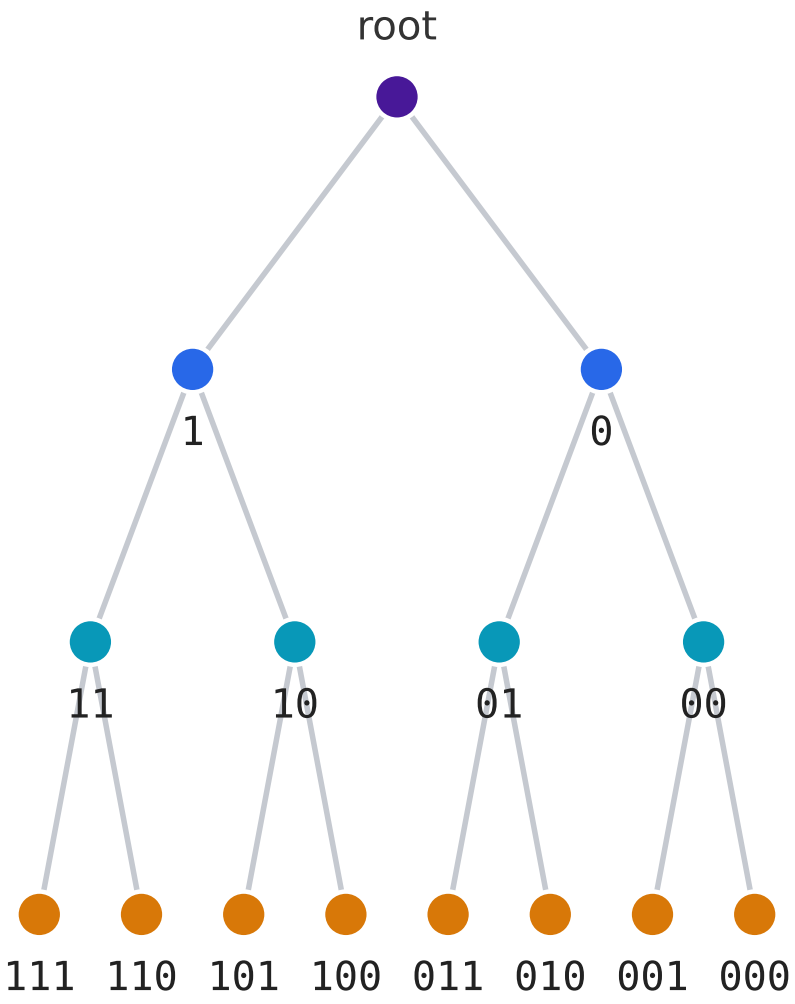

Figure 11: Didactic tree environment with branching factor 2 and depth 3. Each non-root node is labeled by the binary encoding of its path from the root (left 1, right 0). Training trajectories roll out from the root toward a leaf. During evaluation, the agent starts at a leaf and acts greedily toward the root.

A.4 Toy Tree Visualization

In Figure 11 we illustrate the tree environment used in Section 4, for branching factor 2 and depth 3.

B Directionality Proofs

Definition (1, restated)**.** For a dataset D, we define the dataset directionality $\mathcal{D}(D) = \frac{1}{|\mathcal{S}_D|} \sum_{s \in \mathcal{S}_D} P(A_t \in \mathcal{A}^*(s) \mid S_t = s)$, where $\mathcal{A}^*(s)$ denotes the set of optimal actions in s.

Theorem (2, restated)**.** *Denote the averaged probability mass of irreversible transitions by* $\delta = \frac{1}{|S_D|} \sum_{s,s' \in D, s \notin \mathcal{R}(s')} P_D(s'|s)$*. If* $\mathcal{D}(D) > \frac{1+\delta}{2}$ *then there exists a pair of states* s, s'*, such that* $g_s = g_{s'}$*, for which* $P(s \mid s') \neq P(s' \mid s)$*, which are reachable from each other.*

B.1 Proof of Theorem 2

Proof. Assume that $\mathcal{D}(D) > \frac{1+\delta}{2}$ and for the sake of a proof by contradiction assume that for every pair of states s', s $P(s \mid s') = P(s' \mid s)$, whenever the dynamics is reversible.

Take any two neighbouring states from a trajectory in the dataset, between which the transition is reversible. They share the goal state. Assume that the transition s to s' is optimal for reaching their common goal. Then, $d_*(s', g) = d_*(s, g) - 1 \leftrightarrow d_*(s, g) = d_*(s', g) + 1$, therefore the reverse transition is not optimal. Hence, every optimal reversible transition can be paired with a non-optimal reversed transition. By the initial assumption $P(s'|s) = P(s|s')$. Therefore, among the reversible transitions, the probability mass of optimal transitions can be at most half of the total reversible probability mass.

The total probability mass of irreversible transitions is δ, so the probability mass of reversible transitions is $1 - \delta$. Therefore, the reversible transitions can contribute at most $\frac{1-\delta}{2}$ to directionality. In the best case possible, every irreversible transition is optimal. Then we get $\mathcal{D}(D) \leq \frac{1-\delta}{2} + \delta = \frac{1+\delta}{2}$, which contradicts the initial assumption, and therefore there are states s, s' for which $P(s'|s) \neq P(s|s')$. □

C Latent Predictive Objective

The latent predictive objective has been defined in different ways. We build on the formulation used by Ni et al. (2024). They define the latent predictive objective as: $||f_\theta(\phi(s_t, a)) - SG(\psi(s_{t+1}))||_2^2$, where $f_\theta : \mathbb{R}^k \to \mathbb{R}^k$ is a neural network and SG is an optional stop-gradient operator. Since in our case $\psi = \phi$ we will simplify this objective to the form: $||f_\theta(\phi(s_t)) - SG(\phi(s_{t+1}))||_2^2$.

Here, we explain the connection between PTCR and the latent predictive objective. We will show that the following statement holds:

Theorem 3 (informal)**.** *If we train ϕ jointly with f_θ using our proposed objective, f_θ in expectation maps the current latent state to the same next latent state as the optimal latent predictive function f for the fixed encoder ϕ.*

Intuitively, this should hold, because the contrastive loss is lower when the anchor representation is moved toward the expected latent position of the future state. Since a state may have multiple possible futures, this expected latent position is also the optimal prediction under the latent predictive objective.

We begin with showing what the optimal value for the latent predictive objective for an uncontrolled Markov Process is, assuming that the encoder ϕ is fixed.

Theorem 4. *Let X, Y be random variables with the support $\phi(\mathcal{D})$. The optimal value for the latent predictive objective $\mathbb{E}_{x\sim X, y\sim Y|X}||f(x) - y||_2$ is achieved for $f : x \mapsto \mathbb{E}_{Y'\sim P(\cdot|x)}[Y']$.*

Proof. Denote $\mu = \mathbb{E}_{y'\sim P(\cdot|x)}[y']$.

Then: $||f(x) - y||_2^2 = ||f(x) - \mu + \mu - y||_2^2 = ||f(x) - \mu||_2^2 + ||\mu - y||_2^2 + 2\langle f(x) - \mu, \mu - y\rangle$. Applying the conditional expectation to both sides we get: $\mathbb{E}[||f(x) - y||_2^2|X] = \mathbb{E}[||f(x) - \mu||_2^2|X] + \mathbb{E}[||\mu - y||_2^2|X] + 2\mathbb{E}[\langle f(x) - \mu, \mu - y\rangle|X]$.

$f(x) - \mu$ doesn't depend on Y so it is constant with respect to the expectation. $2\mathbb{E}[\langle f(x) - \mu, \mu - y\rangle|X] = 2\langle f(x) - \mu, \mathbb{E}[y' - \mu|X]\rangle = 2\langle f(x) - \mu, 0\rangle = 0$. Hence: $\mathbb{E}[||f(x) - y||_2^2|X] = \mathbb{E}[||f(x) - \mu||_2^2|X] + \mathbb{E}[||\mu - y||_2^2|X]$. The second term doesn't depend on f, therefore the optimal value of $f(x)$ is μ. □

Following Eysenbach et al. (2025), we will make three assumptions about the learned representations. We will restate them here for completeness.

Assumption 5. The representation's norm is bounded by a constant c, that is $||\phi(s)||_2 \leq c$ for all $s \in \mathcal{S}$.

This assumption is not overly restrictive, in practice the represenations' norms have to be bounded, because computers have bounded memory.

We make two further assumptions:

Assumption 6. Regularized, temporal contrastive learning acquires representations whose marginal distribution representations $p(\psi) = \int p(x) \mathbb{1}(\psi(x) = \psi)\, dx$ is an isotropic Gaussian distribution: $p(\psi) = \mathcal{N}(\psi; \mu, \sigma = cI)$.

Assumption 7. $e^{||\phi(s_t) - f_\theta^k \phi(s_{t+k})||_2} = \frac{p_{t+}(s_{t+k}|s_t, c)}{p(s_t, c)C}$

Theorem 8. *Under the assumption above, the distribution of representations of the future state is given by* $\phi_{t+k} \sim \mathcal{N}(\mu = \frac{c}{c+1} f_\theta^k \phi(s_t), \Sigma = \frac{c}{c+1} I)$.

Proof. Proof can be found in Eysenbach et al. (2025) and translates to our case, where A is replaced with f_θ^k and x_{t+} is replaced with $s_{t+\Delta}$. □

Corollary 9. *For* $c \to \infty$*, in expectation* f_θ *maps* x *to the same value as* f.

D Practical Parametrization of f_θ

As described in Section 5.3, we parametrize the latent predictor as $f_\theta^\Delta = QA^\Delta Q^\top$, where A is fixed and only Q is trained. This appendix details the construction of A, how Q is optimized, and the choice of base angle.

Fixed rotation matrix A. To make powers A^Δ efficient to compute, we take A to be block-diagonal, with each 2×2 block a planar rotation. Empirically, we find that fixing the angles is more stable than learning them. Specifically, we use angles $\alpha, 2\alpha, \ldots, \frac{k}{2}\alpha$, where α is a hyperparameter and k is the latent dimension:

$$A = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) & 0 & \cdots & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \cos(\frac{k\alpha}{2}) & -\sin(\frac{k\alpha}{2}) \\ 0 & 0 & \cdots & \sin(\frac{k\alpha}{2}) & \cos(\frac{k\alpha}{2}) \end{pmatrix}.$$

This construction resembles Rotary Positional Embeddings (RoPE) (Su et al., 2023). Following the same intuition, for shorter-horizon tasks (Rubik's Cube, Lights Out) we use a larger base angle $\alpha = \frac{\pi}{100}$, and for longer-horizon tasks (Sokoban, Digit Jumper, 15-Puzzle) we use $\alpha = \frac{\pi}{200}$.

Trainable orthogonal basis Q. The only learned parameters of f_θ are those that define Q. We initialize Q with Kaiming uniform initialization (He et al., 2015) and keep it orthogonal throughout training via the Cayley transform, which maps an unconstrained matrix to an orthogonal matrix. We optimize these parameters jointly with the encoder ϕ by gradient descent on the contrastive objective in Section 5.2.

Although restricting f_θ to this linear family reduces expressivity relative to a generic neural network, Koopman operator theory suggests that suitably chosen representations can still exhibit linear dynamics (Brunton et al., 2021; Koopman, 1931).

E Training Details

Datasets We generate the datasets following the same procedure as in previous work (Ziarko et al., 2025; Czechowski et al., 2024; Zawalski et al., 2024). For Rubik's Cube, N-Puzzle, Digit Jumper and Lights Out we start from the solved state, perform a set number of random actions (respectively 20, 150, 100 and 49) while avoiding loops. For Sokoban we use a suboptimal MCTS-based agent to generate the dataset.

Architectures We use the same base architectures as CRTR (Ziarko et al., 2025) for all the environments. The predictive function uses the parametrization $f_\theta^\Delta = QA^\Delta Q^\top$ detailed in Appendix D.

Training Procedure We follow the same training procedure as CRTR (Ziarko et al., 2025).

F Evaluation Details

Instances for evaluation We sample the instances for evaluation as follows:

- For Rubik's Cube, we shuffle the cube 1000 times.
- For Sokoban, we use a dataset introduce by Czechowski et al. (2024).
- For Lights Out, we sample a random instance.
- For 15 Puzzle, we sample a random solvable instance.
- For Digit Jumper, we sample a random solvable instance.

G Sensitivity to Maximal Solution Length

Figure 5 reports success rate under a fixed maximal solution length for each environment (the fraction of problems solved with solution length strictly below the cutoff, see Section F): 1000 for Sokoban and the Rubik's Cube, 500 for N-Puzzle, 75 for Lights Out, and 50 for Digit Jumper. Figure 12 repeats the through-training comparison for several budgets for each of the environments in increasing order. On Sokoban and Digit Jumper the curves saturate once the budget exceeds the typical solution length, so larger cutoffs leave the relative gap essentially unchanged. On the Rubik's Cube, absolute success rates rise with budget for both methods, while PTCR remains ahead. On Lights Out and N-Puzzle the two methods stay close across budgets.

H Separate Goal Encoders vs Shared Critic

Section 5 argues that fully separate state and goal encoders are not a good way of introducing directionality into a critic as they are too unconstrained.

Figure 13 reports success rates through training on the five domains, at the same maximal length limits as Figure 5, but on shorter training runs.

The separate encoder performs significantly worse than shared encoders on Sokoban, N-Puzzle, and Lights Out, but it matches them on Digit Jumper (the easiest environment considered). All of the encoders remain at zero on the Rubik's Cube under the paper hardness (1000 shuffles). Figure 14 repeats Rubik at an easier 10-shuffle evaluation, where both shared critics solve a large fraction of instances while the separate encoder stays near zero.

I Estimating Dataset Directionality

This appendix details how we estimate the directionality scores reported in Section 4 and Section 6.3. Recall from Section 4.1 that

$$\mathcal{D}(D) = \frac{1}{|\mathcal{S}_D|} \sum_{s \in \mathcal{S}_D} P\big(A_t \in \mathcal{A}^*(s) \mid S_t = s\big),$$

where $\mathcal{S}_D$ is the set of states in dataset D and $\mathcal{A}^*(s)$ is the set of optimal actions at s. We estimate this quantity for the goals sampled during the training, rather than the final goals from the trajectories. This make the computations tractable in the case of Rubik's Cube/15-puzzle. We therefore replace $\mathcal{A}^*(s)$ by $\mathcal{A}^*(s, g)$, the set of first actions of shortest paths from s to g.

We sample state and goal pairs as during CRL training, as detailed in Section 3.2. We ignore pairs with $s = g$.

For each sampled pair s, g, we can get the action chosen by checking the state after s.

We calculate the set of optimal actions for each pair s, g as follows:

- **Colorful Maze, Digit Jumper, Sokoban.** We run BFS from s until we find g.
- **Rubik's Cube and 15-puzzle.** We use a bidirectional, depth-limited BFS.
- **Lights Out.** We use the Gaussian elimination modulo 2, as described in Section 6.4.

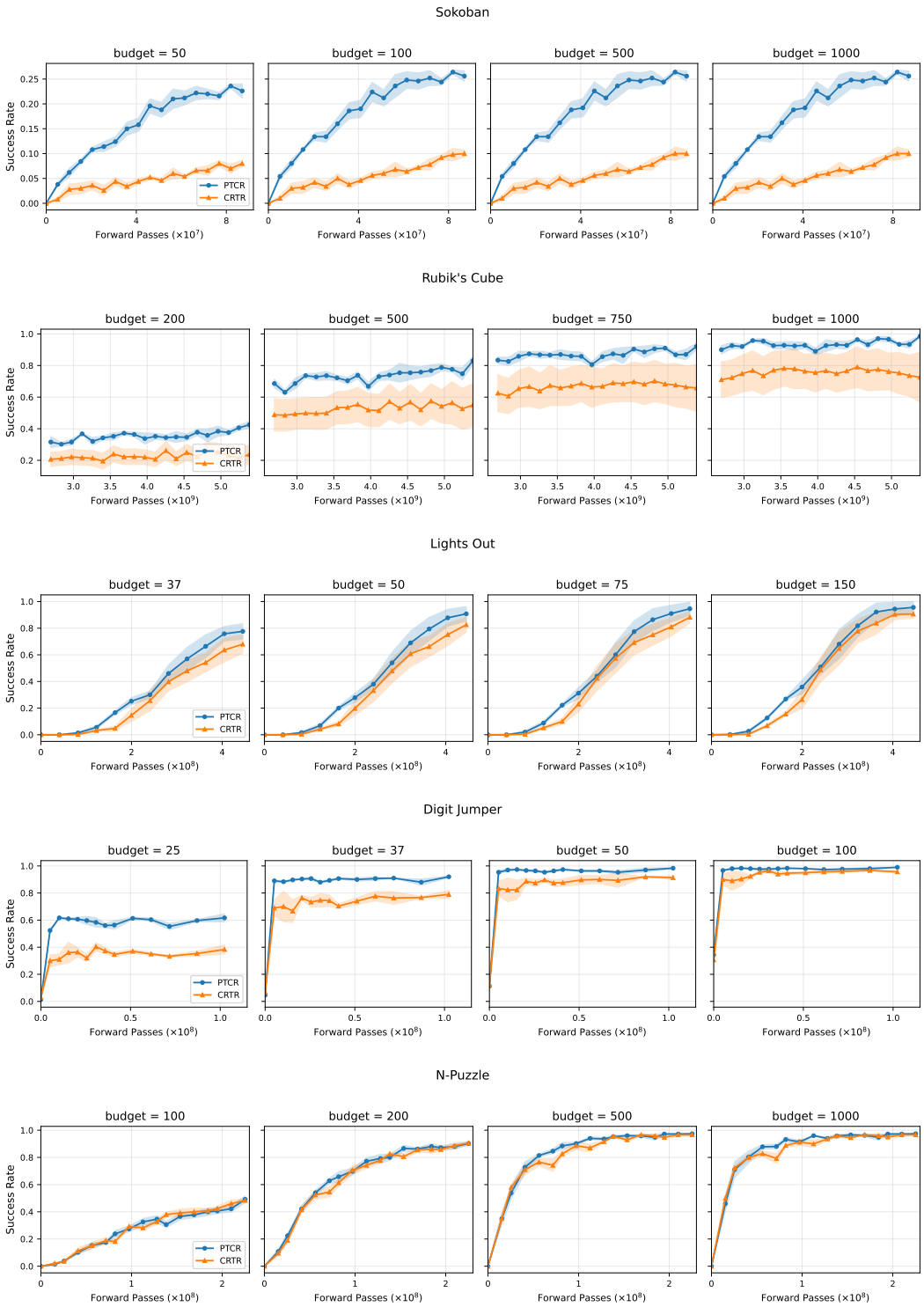

Figure 12: **Data efficiency of PTCR is robust to length-budget.** Each row is one environment; columns are maximal length limits. Shaded regions are the standard error over seeds.

We sample 500 trajectories and 2000 pairs (s_t, g) per dataset. We average the probabilities over the pairs following the definition of dataset directionality. Since it usually is quite unlikely that the same state is sampled twice, the probability most often ends up being in the set $\{0, 1\}$.

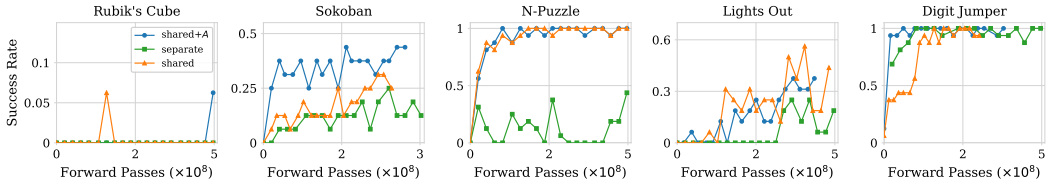

Figure 13: **Separate vs shared encoders through training.** Success rate under the maximal lengths (1000 Rubik/Sokoban, 500 N-Puzzle, 75 Lights Out, 50 Digit Jumper).

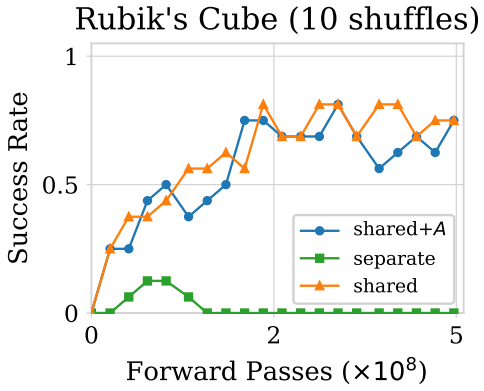

Figure 14: **Rubik's Cube at 10 shuffles.** Same three training runs as Figure 13, evaluated with an easier scramble depth=10.

J Computational Resources

The cost of the evaluation is very small in comparison to the training cost. Below, we list the computational resources consumed by the training experiments in GPU hours on a cluster of A100 GPUs. The CPU hours can be calculated by multiplying the GPU hours by 16.

- DeepCubeA each training took around 48 hours, for 5 environments and seeds: $5 * 5 * 48 = 1200$ GPU hours
- CRTR/PTCR/Supervised each took $5 * 48$ for the Rubik's Cube, $5 * 24$ for Sokoban, $5 * 24$ for Lights Out, $5 * 24$ for 15 Puzzle and $5 * 12$ for Digit Jumper, which adds up to around 2000 GPU hours.

Together with all the failed experiments, the project used up around 25000 GPU hours.

K Lights Out Analysis

Lights Out is a grid-based puzzle game where the goal is to reach a state with all the lights turned off. (See Appendix A). For an $n \times n$ grid with $n = 2^m - 1$, every starting state admits a unique set of buttons, which when pressed solve the problem, regardless of the order. (Barile, n.d.) The structure of Lights Out is algebraic, as it can be solved by performing Gaussian elimination modulo 2. (Coulibaly, 2023)

We train on a dataset generated by starting from the solved state and pressing a sequence of 49 random fields. Since actions are commutative, we expect this to result

We analyze checkpoints throughout the training of CRTR on Lights Out. The training set consists of 10000 randomly sampled boards, while the test set contains 1000 additional boards. For each of the 49 possible actions, we label the state with whether that action is a part of the optimal solution and train a linear probe to classify whether that's the case. We attach the linear probe after the sixth layer (out of eight layers) of the encoder.

Dataset	$\mathcal{D}(D)$
Sokoban	0.973
Rubik's Cube	0.965
Lights Out	0.934
Digit Jumper	0.829
15-puzzle	0.648

Table 1: Directionality estimates for the five environments.

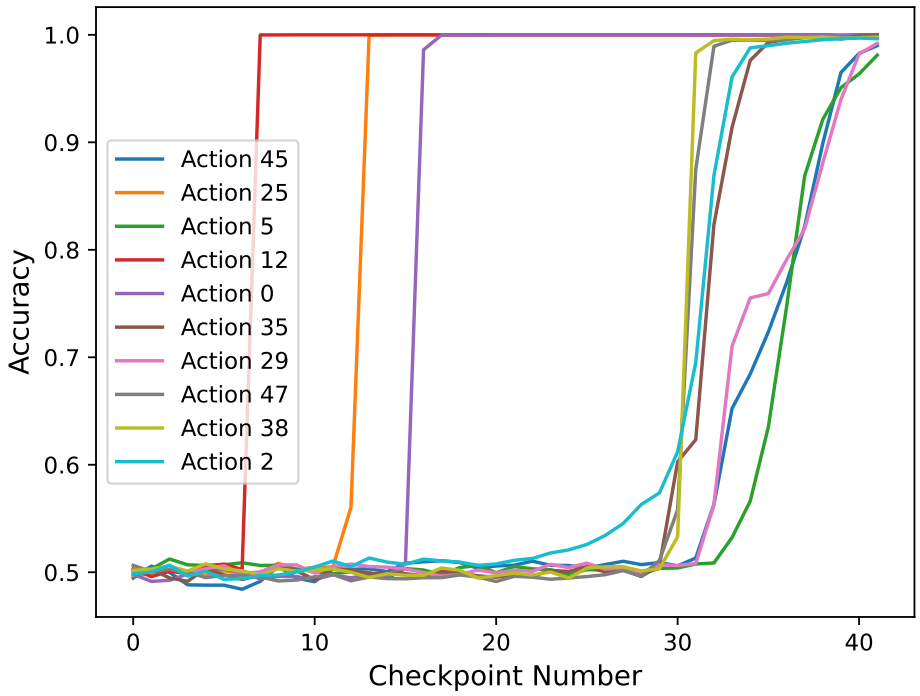

Figure 15: **The actions forming the optimal path to the solution can be recovered from the latent representations of the CRTR network.** Lights Out probes through training: action-classification accuracy for 10 randomly chosen actions over the training.

For every action, there is a point during training at which the probe accuracy abruptly exceeds 95% accuracy (See Fig. 15 for examples). This suggests that the network might internally recover the optimal solution when constructing the representations. These changes seem to be accompanied by a rapid increase in the success rate (See Figure 16).

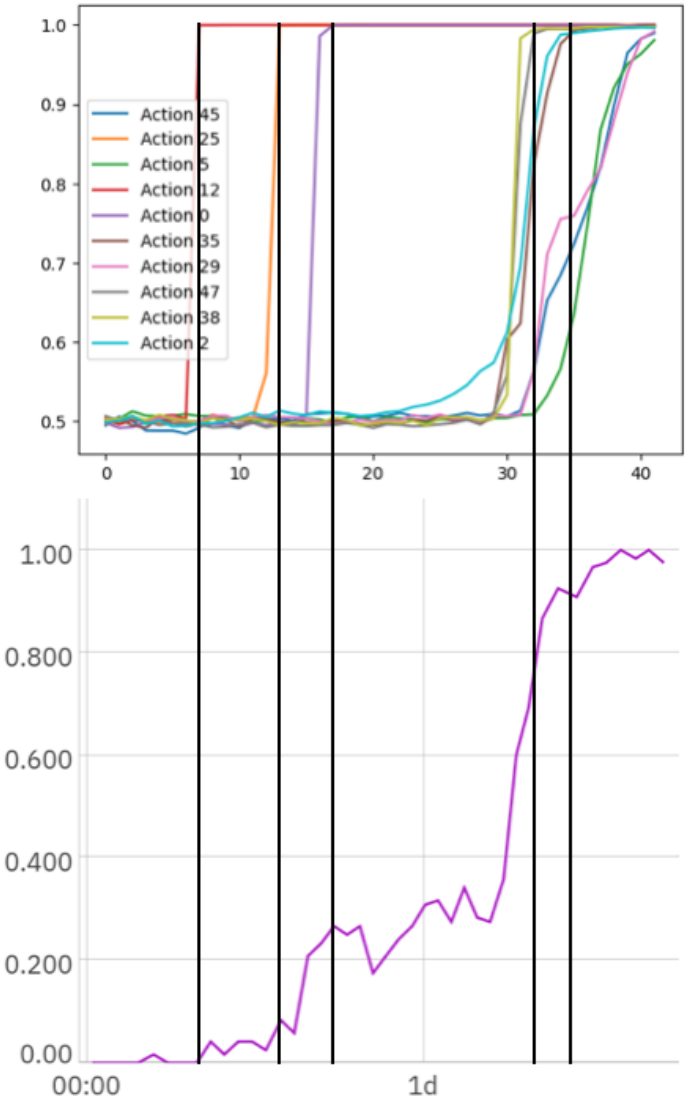

Figure 16: Lights Out: relationship between probe accuracy and solved rate through training.