Learning What Matters: Endogenous Representation through Advantage Estimation and Sparse Attention

Hsiao-Ru Pan [1], **Anson Lei** [1, 2], **Ingmar Posner** [2], **Bernhard Schölkopf** [1]
{hpan,bs}@tuebingen.mpg.de, {anson,ingmar}@robots.ox.ac.uk

[1]**Max Planck Institute for Intelligent Systems**
[2]**Applied Artificial Intelligence Lab, Oxford University**

Abstract

Recent works hypothesize that directly learning the advantage function can induce *endogenous* representations that only depend on variables that are causally related to the agent's action. However, previous empirical evidence is limited to small linear environments. We investigate this hypothesis in a more complex image-based environment, showing that advantage function learning leads to improved generalization and robustness. In addition, by using a sparsity-regularized Transformer as the function approximator, we qualitatively demonstrate that the learned advantage function attends only to the decision-relevant part of the observations, leading to improved interpretability.

1 Introduction

As reinforcement learning (RL) (Sutton et al., 1998) agents are deployed in increasingly complex, real-world environments, developing agents that can act in a robust and generalizable manner in the face of changes in the environment, such as visual distractors, presents a critical challenge. The present work combines recent advances in advantage function learning (Pan et al., 2022) and sparse transformers (Lei et al., 2024) to train robust agents in complex image-based domains by attending only to relevant parts of the image.

Recently, the notion of causality has garnered significant interest within the context of RL (Zeng et al., 2024) as it offers a principled framework for understanding generalization in learning systems. In this light, recent works (Pan et al., 2022) elicit the role of the *advantage function* in causal reinforcement learning. In particular, Pan & Schölkopf (2023) shows that, under the ExoMDP framework (Trimponias & Dietterich, 2023), the advantage function inherently only depends on endogenous variables, i.e., variables that are within the agent's control. In other words, the advantage function, and by extension any policy that derives from it, is invariant to distractor variables. The authors investigated the properties of learned representations in simple linear environments, showing that learning advantage functions successfully *induces* endogenous representations that isolate the relevant part of the state space. This provides a promising first step towards learning robust and generalizable representations in complex high-dimensional environments.

In this work, we investigate advantage-function-induced representations in image-based environments. In this context, extracting the dependencies of a learned advantage function is challenging, as the decision-relevant part of the state can be rendered in different parts of the image over time. One possible solution is to use the attention mechanisms (Bahdanau et al., 2014) in the Transformer architecture (Vaswani et al., 2017) as a proxy for the learned dependency structure. However, empirical evidence shows that, across various settings such as natural language (Jain & Wallace, 2019) and world models (Lei et al., 2024), Transformers often attend to spurious features and do not faithfully reflect the causal structure of the problem at hand. To remedy this, Lei et al. (2024) develop a sparsity-regularization scheme to induce attention patterns that capture the underlying dependencies.

Taken together, advantage function learning and sparse attention play complementary roles: the advantage function provides a learning target that depends only on decision-relevant variables, while sparse attention provides a way to interpret the learned dependencies. Empirically, we evaluate our method in CoinRun (Cobbe et al., 2020), an image-based environment with procedurally generated levels to evaluate generalization, showing that our method outperforms a baseline which learns the Q function via SARSA (Rummery & Niranjan, 1994) in terms of generalization and that our model can extract interpretable dependency structures that attend only to task-relevant parts of the observation.

2 Background

We consider the Exogenous MDP (ExoMDP) setting, which provides a more fine-grained view of the decision-making process by separating variables into those that are causally related to the agent's actions (endogenous variables) and those that are not (exogenous variables). Figure 1 shows a comparison between the causal graphs of a normal MDP and an ExoMDP. More specifically, in an ExoMDP, the state space, transition probability, and the reward function can be factorized into, $\mathcal{S} = \mathcal{S}^e \times \mathcal{S}^x$, $p(s_{t+1}|s_t, a_t) = p(s^e_{t+1}|s^e_t, a_t)p(s^x_{t+1}|s^x_t)$, and $r(s, a) = r_e(s^e, a) + r_x(s^x)$, respectively (with $s^e \in \mathcal{S}^e$ and $s^x \in \mathcal{S}^x$).

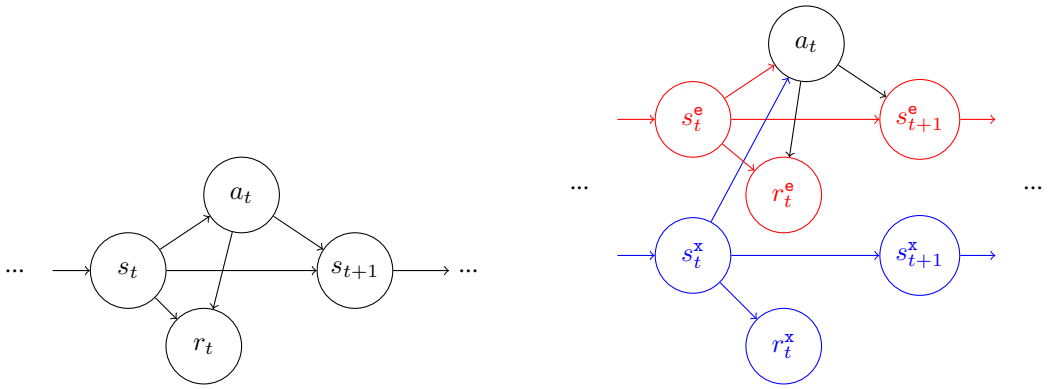

Figure 1: Graphical models of MDPs (left) and ExoMDPs (right). Colors and superscripts indicate whether the variable or relationship is **e**ndogenous (red) or e**x**ogenous (blue).

2.1 Direct Advantage Estimation

Pan et al. (2022) showed that the advantage function can be seen as quantifying causal effects of actions on the return and proposed Direct Advantage Estimation (DAE), an on-policy method that can estimate the advantage function directly. Specifically, the learning process is formulated as minimizing the following constrained least-squares objective:

$$\mathcal{L}(\hat{A}, \hat{V}) = \mathbb{E}_\pi \left[\left(\sum_{t=0}^{n-1} \gamma^t \left(r_t - \hat{A}_t \right) + \gamma^n V_{\text{target}}(s_n) - \hat{V}(s_0) \right)^2 \right] \quad \text{s.t.} \sum_a \hat{A}(s, a)\pi(a|s) = 0, \tag{1}$$

where V_{target} is the bootstrapping target, and $(\hat{A}, \hat{V})$ are the variables to be minimized. Minimizing this objective can be seen as a multi-step estimate of (A^π, V^π). Pan & Schölkopf (2023) subsequently showed that, unlike other value functions such as V or Q, the advantage function is inherently endogenous (i.e., it only depends on the endogenous part of an ExoMDP), and hypothesized that learning the advantage function using DAE can lead to representations that are also endogenous. This is empirically verified in a toy linear ExoMDP by showing that the representations learned by DAE are less dependent on exogenous variables compared to representations induced via learning the Q function. However, scaling this approach up to more complex image-based environments remains an open challenge.

2.2 Sparse Transformer

Transformers (Vaswani et al., 2017) have emerged as a prominent architecture across many deep learning applications. While earlier works have shown that aggregating attention patterns across transformer layers via Attention Rollout (Abnar & Zuidema, 2020) can, to some extent, extract learned dependencies between variables, Lei et al. (2024) demonstrated that standard transformers often attend to spurious features, and propose a sparsity regularization scheme to induce attention patterns that better represent the underlying local dependency structure. Specifically, given the key, query and value embeddings $\{k_i^l, q_i^l, v_i^l\}$ for each variable i at layer l, a binary adjacency matrix is sampled according to

$$\Lambda_{ij}^l \sim \text{Bern}(\sigma(q_i^{l^T} k_j^l)), \tag{2}$$

where $\Lambda_{ij} = 0$ indicates that there is no attention edge from the variable i to the variable j. These adjacency matrices are then element-wise multiplied to the standard attention weights and act as learnable gates that selectively block the flow of information between variables. By regularizing [1] the expected number of edges between variables, sparse transformers effectively prune out unnecessary dependencies. Across multiple transformer layers, these adjacency matrices can be aggregated via

$$\bar{\Lambda} = \prod_l (\Lambda^l + \mathbb{I}), \tag{3}$$

where $\mathbb{I}$ is the identity matrix and Λ^l is the sampled adjacency matrix at layer l. $\bar{\Lambda}_{ij}$ is the number of paths from variable i to j across the entire transformer. Note that $\bar{\Lambda}_{ij} = 0$ implies that the prediction for variable j is completely independent of the variable i.

2.3 CoinRun

The CoinRun environment is a 2D platformer from the ProcGen suite (Cobbe et al., 2020; 2019) that uses procedural generation to generate different levels (layouts, background images, appearances of entities, etc.) This provides a natural way to examine the generalization capability of an RL agent by varying the number of training levels seen by the agents. We can then measure their generalization performance by evaluating them at test levels that are not seen during training. In CoinRun, the agent gets a reward of 10 if a coin is successfully collected; otherwise, the reward is 0. An episode ends if the agent collects a coin or touches a trap or an enemy.

3 Method

We use a Vision Transformer (ViT) (Dosovitskiy et al., 2020) as the function approximator to jointly learn an advantage function and a value function. Specifically, image observations (64×64) are first segmented into patches (8×8) which are subsequently encoded into patch embeddings via early convolutions (Xiao et al., 2021). These patch tokens serve as the input to the Transformer encoder. To extract the advantage and value functions, we append two learnable query tokens (one for advantage and one for value) to the patch tokens. This is akin to the `[cls]` token, which extracts image information for classification in a standard ViT. The transformer outputs for the query tokens are mapped to the correct dimensions, i.e., one for the value function and the number of actions for the advantage function, via separate MLPs. Figure 2 illustrates the overall pipeline of our approach.

The ViT, convolution embedding, and query tokens are jointly trained to minimize the DAE objective (eq.1) plus an attention penalty term via a PPO style (Schulman et al., 2017) on-policy value-based algorithm. We use the aggregated path matrix (eq.3) to extract and visualize the learned dependencies of the advantage and value functions. In the following section, we empirically show that our approach achieves improved generalization and is able to learn an advantage function that only depends on a small number of decision-relevant image patches.

[1] The hard attention samples are made differentiable via the Gumbel max trick (Jang et al., 2017).

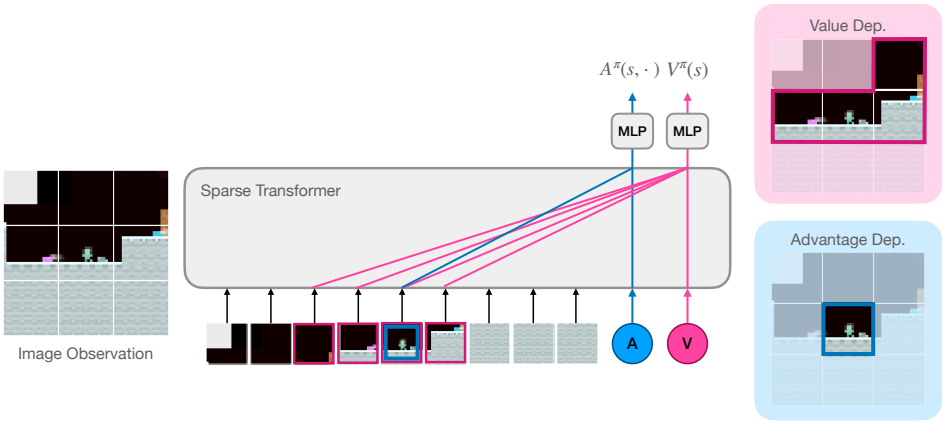

Figure 2: The overall architecture of the proposed method. A sparse transformer operates over patched tokens and query tokens which outputs the learned advantage and value functions. The learned dependencies of the value functions are extracted via the sparse attention patterns. For illustrative purposes, we only show 9 patches.

4 Experiments

We compare the performance of our Direct Advantage Estimation (DAE) method against a baseline that estimates the Q-function. The baseline uses the same architecture as our method (replace the A/V token by a Q token), but replaces DAE with multi-step SARSA to learn the Q-function. To evaluate the generalization performance of the learned policies, we train the policies on a fixed number of procedurally generated CoinRun levels, and test on *unseen* levels. We train for 250 iterations, where each iteration corresponds to 8192 (parallel actors) $\times$ 64 (multi-step) = 524288 frames. We evaluate the agent at the end of each iteration by averaging the scores over 1000 episodes from randomly sampled test levels. Each configuration is repeated for 3 random seeds.

Generalization & Learning Efficiency We present the results in figure 3 (top row), showing that DAE is not only more data efficient in terms of training frames, but also demonstrates better generalization performance across different training levels. Interestingly, we find that while the generalization performance of both methods is similar when the number of training levels is very low (10^2 levels) or very high (10^4 levels), DAE is able to generalize to new levels better when trained on a moderate number of levels (10^3 levels). This corroborates our hypothesis that learning the advantage function leads to more efficient generalization.

Exogenous Reward The CoinRun environment, while capable of generating diverse levels through procedural generation, does not include distractive rewards (e.g., r_x in an ExoMDP). To further test the robustness of both DAE and SARSA, we consider a more challenging version of CoinRun, where we add an additional exogenous reward that depends on the randomly sampled background image of a level. Note that, since the background image does not depend on the agent's actions, this reward function is exogenous by the definition of an ExoMDP. From Figure 3 (bottom row), we find that SARSA struggles to generalize in this case even with 10^4 training levels, while DAE shows similar trends of generalization with respect to the number of training levels. This suggests that estimating the advantage directly via DAE can be beneficial when there are rewards that are not causally related to the agent's actions.

Dependency Visualization To qualitatively investigate whether the advantage function learned via DAE depends on only decision-relevant parts of the image, we visualize the sparse attention patterns learned by the sparse transformer as described in Section 3. Figure 4 shows an example of how A, V, and Q depend on the observation. Here, we see that the advantage function's dependency

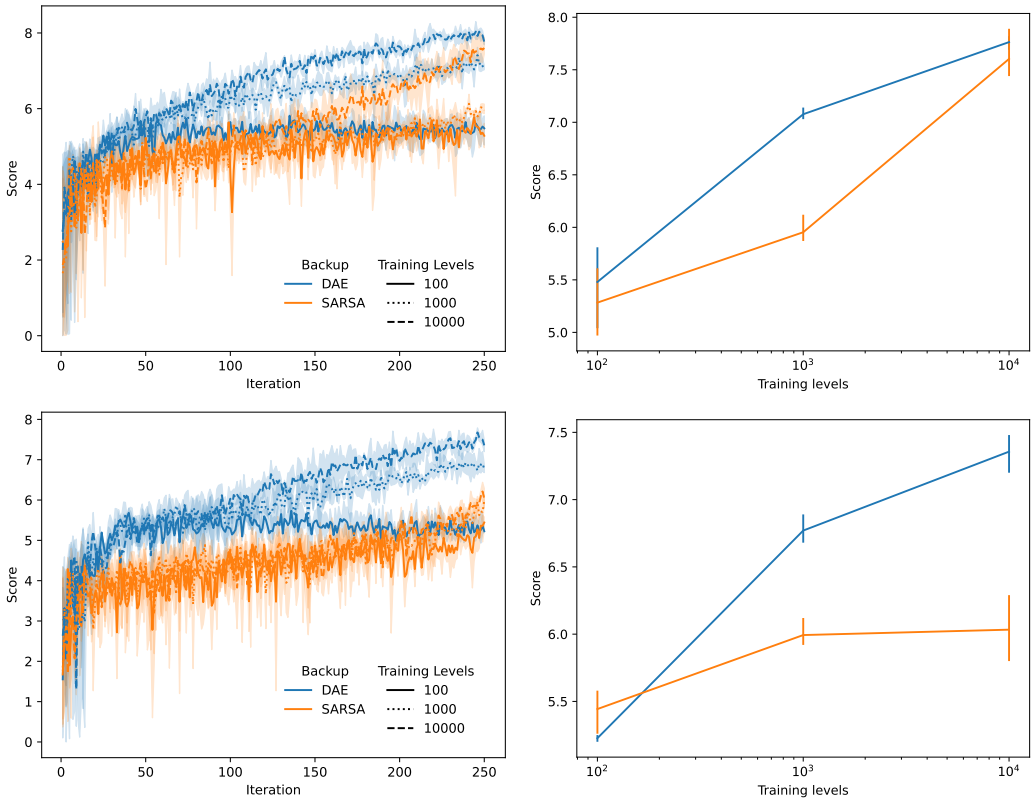

Figure 3: Test performance without exogenous rewards (top) and with exogenous rewards (bottom). Left: Learning efficiency. Right: Final evaluation score. Lines and shadings (error bars) represent the mean and min/max score over 3 seeds.

map is much more sparse compared to the V or the Q function. In this example, the advantage function only depends on the velocity of the character (rendered in the upper-left corner), the trap next to the character, and the boundary of the level on the right (we note that all levels end with a vertical wall on the right). On the other hand, the value function can depend on various details of the observation, including various entities (e.g., the traps and the coin), and the background image (due to the exogenous rewards). The Q function learned by SARSA shows similar patterns to the value function learned by DAE, demonstrating strong dependency on the irrelevant details of the observation. Since the policies were constructed either with Q (for SARSA) or A (for DAE), the advantage function's ability to ignore irrelevant features offers a possible explanation for the better generalization performance of DAE. In Appendix A.4, we present extra examples of learned dependencies and compare the attention patterns induced by our approach against standard Transformers, demonstrating that sparse regularization significantly improves interpretability.

5 Discussion

In the present work, we compared the generalization performance of DAE and SARSA. We found DAE to (1) converge more efficiently in terms of numbers of training levels, and (2) be less sensitive to exogenous rewards. By utilizing the sparse transformer architecture, we qualitatively examined the dependency maps of the different value functions and found that, as the theory suggests, the advantage function learned by DAE is more robust compared to the Q function learned by SARSA.

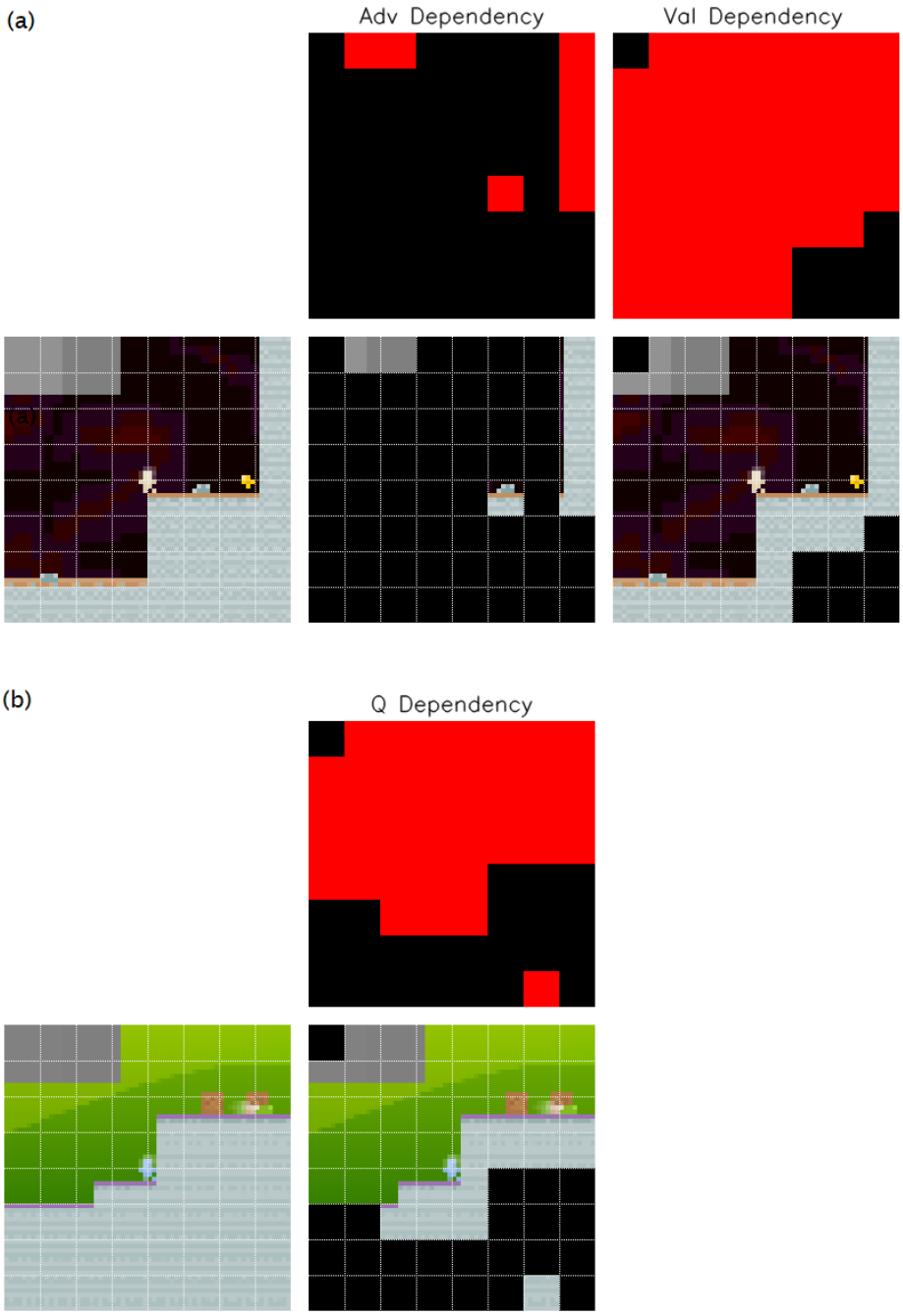

Figure 4: Dependency map with exogenous rewards. The top row shows the adjacency matrix from the corresponding token to the visual patches. The bottom row shows the masked observations. (a) DAE. (b) SARSA.

References

Samira Abnar and Willem Zuidema. Quantifying attention flow in transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4190–4197, 2020.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In *International conference on machine learning*, pp. 1282–1289. PMLR, 2019.

Karl Cobbe, Chris Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. In *International conference on machine learning*, pp. 2048–2056. PMLR, 2020.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Sarthak Jain and Byron C Wallace. Attention is not explanation. *arXiv preprint arXiv:1902.10186*, 2019.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=rkE3y85ee.

Anson Lei, Bernhard Schölkopf, and Ingmar Posner. Spartan: A sparse transformer learning local causation. *arXiv preprint arXiv:2411.06890*, 2024.

Hsiao-Ru Pan and Bernhard Schölkopf. Learning endogenous representation in reinforcement learning via advantage estimation. In *Causal Representation Learning Workshop at NeurIPS 2023*, 2023.

Hsiao-Ru Pan and Bernhard Schölkopf. Skill or luck? return decomposition via advantage functions. *arXiv preprint arXiv:2402.12874*, 2024.

Hsiao-Ru Pan, Nico Gürtler, Alexander Neitz, and Bernhard Schölkopf. Direct advantage estimation. *Advances in Neural Information Processing Systems*, 35:11869–11880, 2022.

Gavin A Rummery and Mahesan Niranjan. *On-line Q-learning using connectionist systems*, volume 37. University of Cambridge, Department of Engineering Cambridge, UK, 1994.

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

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 32–42, 2021.

George Trimponias and Thomas G Dietterich. Reinforcement learning with exogenous states and rewards. *arXiv preprint arXiv:2303.12957*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Tete Xiao, Mannat Singh, Eric Mintun, Trevor Darrell, Piotr Dollár, and Ross Girshick. Early convolutions help transformers see better. *Advances in neural information processing systems*, 34:30392–30400, 2021.

Yan Zeng, Ruichu Cai, Fuchun Sun, Libo Huang, and Zhifeng Hao. A survey on causal reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

Supplementary Materials

The following content was not necessarily subject to peer review.

A Additional Experiment Details

A.1 Environment Setting

We use a simplified version of the CoinRun environment by enabling the velocity information via `paint_vel_info`, furthermore, we constrain the action space to actions that are relevant to CoinRun (i.e., the 9 actions that control the character instead of the full 15 actions). These changes were made to reduce partial observability and ease exploration. To further reduce partial observability, we also stack the last 4 observations.

For the exogenous reward experiments, the exogenous reward is defined as a function of the background index (ranging from 0 to 61). More specifically, we define $r_x = 0.1 \times (\frac{\texttt{bkg\_idx}}{32} - 1)$. To simulate the effect of infinite horizons, the value of the terminal state is defined by $\frac{r_x}{1-\gamma}$ instead of 0.

A.2 Pseudocode

Algorithm 1 A simple on-policy value-based algorithm (PPO style)

Require: `backup`
  Initialize network parameter θ, τ
  for iteration i=1, 2, ... **do**
    if `backup` == SARSA **then**
      $\pi_i(a|\cdot) \leftarrow \texttt{softmax}_a(Q_{\theta_{i-1}}(\cdot, a)/e^\tau)$
    else if `backup` == DAE **then**
      $\pi_i(a|\cdot) \leftarrow \texttt{softmax}_a(A_{\theta_{i-1}}(\cdot, a)/e^\tau)$
    end if
    Collect k-step partial trajectories using π_i with parallelized environments
    $\theta \leftarrow \theta_{i-1}$
    for n=1, 2, ..., gradients **do**
      Sample $\mathcal{B}$ a batch of k-step trajectories
      if `backup` == SARSA **then**
        $\mathcal{L}(\theta) = \mathbb{E}_\mathcal{B}\left[\left(Q_\theta(s_0, a_0) - \left(\sum_{t=0}^{k-1} \gamma^t r_t + \sum_a \pi_i(a|s_k) Q_{\theta_{i-1}}(s_k, a)\right)\right)^2\right]$
        $\pi_\tau(a|s) \leftarrow \texttt{softmax}_a(Q_\theta(s, a)/e^\tau)$
      else if `backup` == DAE **then**
        Enforce constraint by $A_\theta(s, a) \leftarrow A_\theta(s, a) - \sum_a A_\theta(s, a)\pi_i(a|s)$
        $\mathcal{L}(\theta) = \mathbb{E}_\mathcal{B}\left[\left(V_\theta(s_0) - \left(\sum_{t=0}^{k-1} \gamma^t (r_t - A_\theta(s_t, a_t)) + \gamma^k V_{\theta_{i-1}}(s_k)\right)\right)^2\right]$
        $\pi_\tau(a|s) \leftarrow \texttt{softmax}_a(A_\theta(s, a)/e^\tau)$
      end if
      $\mathcal{L}_\text{S} \leftarrow \sum_{i,j,l} \sigma(q_i^{l^T} k_j^l)$
      Gradient update θ with respect to $\mathcal{L}(\theta) + \beta_S \mathcal{L}_\text{S}$
      Gradient update τ with respect to $\tau + \beta_\text{KL} \mathbb{E}_\mathcal{B}\left[\text{KL}(\pi_i || \pi_\tau)\right]$
    end for
    $\theta_i \leftarrow \theta$
  end for

We use softmax policies for exploration and smoothing the policy changes, which were found to be crucial for the DAE objective (Pan & Schölkopf, 2024). The parameter τ is the log of the temperature for the softmax function, which is minimized along with a KL penalty to ensure the policy does not change too much between iterations.

A.3 Hyperparameter & Network Architecture

We follow the original ViT block design, which consists of a multi-head self-attention layer followed by an MLP block. The patchify the image using a three layer 2D convolutional network as suggested by Xiao et al. (2021). We use LayerScale (Touvron et al., 2021) in the ViT blocks, which we found to stabilize training at initialization. The learning rate follows a one cycle schedule with linear warmup for the first 10% gradient steps followed by linear annealing to 0. Value heads are all MLPs with a single hidden layer.

ViT	
Conv. Stem (channels, kernel size, stride)	[256, 4, 4] [512, 2, 2] [512, 1, 1]
ViT blocks	2
Attention heads	4
MLP dimension	2048
Other Hyperparameters	
Parallel actors	8192
Backup length	64
Epochs	2
Batch size	32
Learning rate	1.25×10^{-4}
Learning rate (τ)	10^{-2}
Adam ($\beta_1, \beta_2, \epsilon$)	(0.9, 0.95, 10^{-5})
β_{KL}	5
β_{S}	Linearly increased from 0 to 0.1

Table 1: Network Architecture & Hyperparameters

A.4 Extra Examples

Figure 5, 6, 7 present extra examples of the learned dependency. We observe that the learned sparse advantage function consistently attends to the rendered velocity information (top left corner) and relevant features that need to be avoided, such as enemy sprites, traps, and lava. We also compare against the attention pattern learned by a vanilla Transformer using Attention Rollout (Abnar & Zuidema, 2020)(rendered on the top row of the figures using a log color scale to visualize small but non-zero attention weights), showing that it is difficult to reliably extract the learned dependencies without using sparsity regularization.

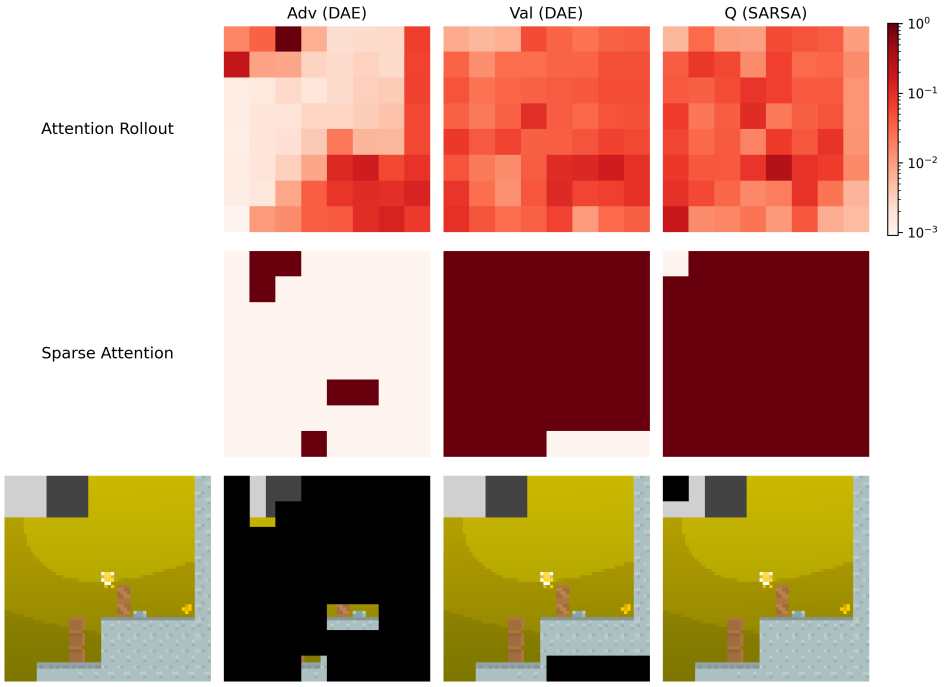

Figure 5: Extra example of learned sparse dependence. Here, the sparse advantage attends to an enemy sprite that is in front of the agent.

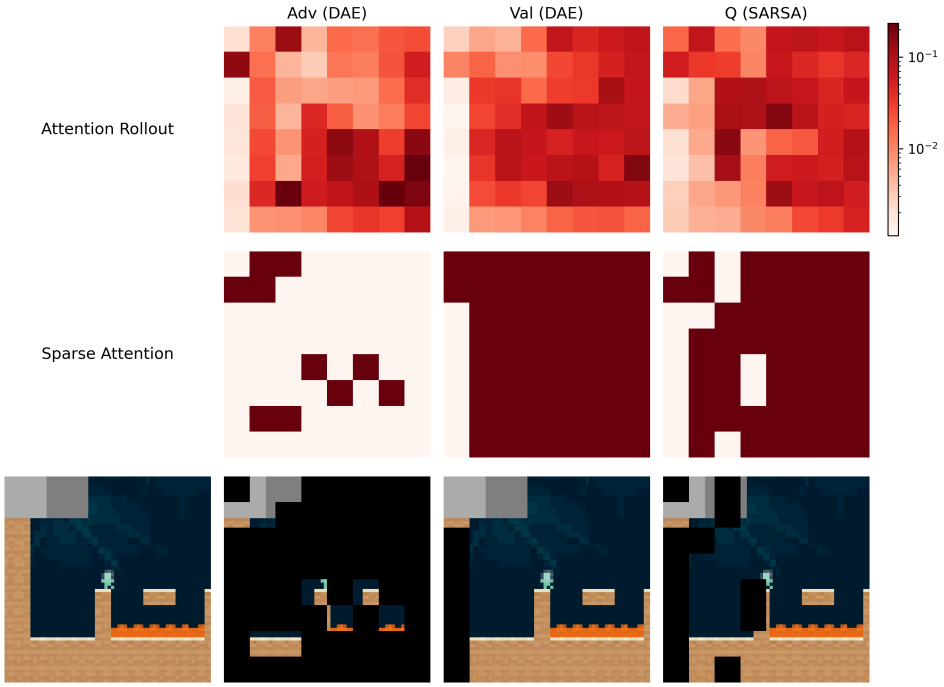

Figure 6: Extra example of learned sparse dependence. Here, the sparse advantage attends to lava covered floor.

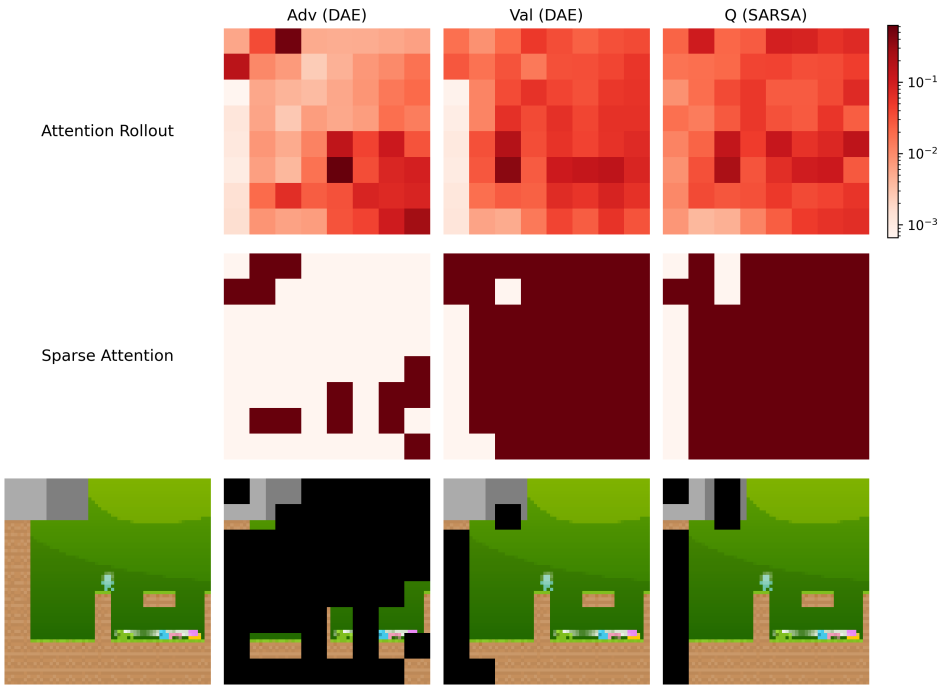

Figure 7: Extra example of learned sparse dependence. Here, the sparse advantage attends to traps to be avoided.