Sparse Autoencoders for Interpretable Feature Extraction in Actor–Critic Reinforcement Learning Models

Chari Mackson

Summer PREMIERE Research Academy

Abstract

Interpretability remains a critical challenge in deep reinforcement learning (RL), where neural policies learn complex representations that are often opaque to human understanding. While sparse autoencoders (SAEs) have been proposed as a tool for feature disentanglement and concept extraction in value-based RL, little is known about their role in actor–critic architectures that maintain separate policy and value networks. In this work, we propose a framework for embedding SAEs into the feature space of both branches of an Advantage Actor–Critic (A2C) model, enabling systematic comparison of learned concepts across policy and value representations. We introduce a mixed-k sparsity schedule that alternates between strict sparsity and relaxed sparsity during training, improving the capture of both rare and moderately common features. Our experiments on the `LunarLander-v2` environment demonstrate that this approach yields interpretable, stable features that correlate with high-level action semantics and game states. We evaluate interpretability using feature stability across seeds, feature–action correlation, and human-label alignment. Our results suggest that the policy network tends to learn more temporally predictive features, while the value network encodes broader contextual features, both of which can be surfaced through SAE-driven analysis.

1 Introduction

Deep reinforcement learning has achieved remarkable successes in complex control domains, from Atari games to continuous robotics control (Mnih et al., 2015; ?; ?). Yet, the internal representations learned by deep RL agents remain largely inaccessible to human interpretation, limiting trust and debuggability in safety-critical applications. Recent work has explored *sparse autoencoders* (SAEs) as a means of uncovering interpretable, disentangled features from hidden activations of deep value networks (Olah et al., 2020; Bricken et al., 2023). These methods have shown promise in extracting "concept neurons" that respond to semantically meaningful patterns in the agent's observations.

However, most existing SAE interpretability work has focused on value-based methods such as Deep Q-Networks (DQN) (Mnih et al., 2015), which employ a single network to approximate the state–action value function. In contrast, actor–critic methods maintain two distinct function approximators: a policy network (actor) that outputs an action distribution, and a value network (critic) that estimates the state value function. This separation raises new questions: Do the actor and critic learn qualitatively different feature spaces? Which branch contains features more aligned with human-interpretable concepts?

In this paper, we address these questions by applying SAEs to both branches of an Advantage Actor–Critic (A2C) model trained on the `LunarLander-v2` task. We introduce a *mixed-k sparsity* training schedule for the SAEs, designed to balance between discovering rare, sharp features and capturing moderately common patterns that may also be semantically meaningful. We develop quantitative metrics for interpretability, including feature stability over training seeds and correlation with high-level action statistics.

Our contributions are:

1. A novel application of sparse autoencoders to both policy and value networks in an actor–critic setting, enabling branch-wise interpretability analysis.

2. The mixed-k sparsity schedule, which improves the diversity and semantic clarity of

learned SAE features.

3. A set of interpretability metrics tailored to RL, measuring stability, action correlation, and human-label alignment.

4. Empirical findings on `LunarLander-v2` showing distinct interpretability profiles for policy and value branches.

2 Related Work

2.1 Interpretability in Deep Reinforcement Learning

Interpretability in RL spans several approaches, from saliency mapping (Greydanus et al., 2018) to policy summarization (Amir and Amir, 2018) and symbolic rule extraction (Verma et al., 2018). While these methods focus on attributing decisions to input features, SAE-based approaches instead seek to uncover internal *concept representations* by training auxiliary models on hidden activations.

2.2 Sparse Autoencoders for Concept Discovery

Sparse autoencoders have been explored in interpretability research as a tool for finding *monosemantic neurons*—hidden units that respond consistently to a single concept (Olah et al., 2020; Bricken et al., 2023). The SAE enforces sparsity in its latent space, often through an L_1 penalty or fixed-k top activation constraint. Our work extends this approach to actor–critic models and introduces a mixed-k schedule for improved concept diversity.

2.3 Actor–Critic Architectures

Actor–critic methods combine policy gradient updates for the actor with value-based updates for the critic (Konda and Tsitsiklis, 2000). The separation of networks creates two different representational spaces, which we hypothesize may differ in interpretability potential. Our work is the first, to our knowledge, to explicitly compare SAE-derived features between actor and critic.

3 Methods

3.1 Advantage Actor–Critic Framework

We consider the standard RL setup with an agent interacting with an environment modeled as a Markov Decision Process $(\mathcal{S}, \mathcal{A}, P, r, \gamma)$. In A2C, the policy network $\pi_\theta(a|s)$ is updated using the advantage estimate:

$$A_t = R_t - V_\phi(s_t), \tag{1}$$

where $V_\phi(s)$ is the value network.

The policy loss is:

$$\mathcal{L}_{\text{policy}} = -\mathbb{E}_t \left[\log \pi_\theta(a_t|s_t) A_t \right], \tag{2}$$

and the value loss is:

$$\mathcal{L}_{\text{value}} = \frac{1}{2} \mathbb{E}_t \left[(R_t - V_\phi(s_t))^2 \right]. \tag{3}$$

3.2 Sparse Autoencoder Integration

Let h_t^a and h_t^c denote hidden activations from the penultimate layers of the actor and critic networks, respectively. We train separate SAEs $f^a_{\psi_a}$ and $f^c_{\psi_c}$ on batches of these activations.

The SAE encodes h into a sparse latent vector z and reconstructs $\hat{h}$:

$$z = \text{TopK}(W_e h + b_e, k), \quad \hat{h} = W_d z + b_d, \tag{4}$$

with reconstruction loss:

$$\mathcal{L}_{\text{SAE}} = \|h - \hat{h}\|_2^2. \tag{5}$$

3.3 Mixed-k Sparsity Schedule

Unlike fixed-k approaches, our mixed-k schedule alternates between small k_s and larger k_l during training epochs:

$$k_{\text{epoch}} = \begin{cases} k_s & \text{if epoch mod } 2 = 0, \\ k_l & \text{otherwise.} \end{cases} \tag{6}$$

This alternation promotes discovery of both rare and moderately common features.

3.4 Interpretability Metrics

We propose:

- **Feature Stability (FS)**: Mean Jaccard index of top activations across seeds.
- **Action Correlation (AC)**: Maximum Pearson correlation between feature activation and action selection probability.
- **Human Label Alignment (HLA)**: Agreement between human-assigned feature labels and activation-triggered states.

Algorithm 1 SAE Integration with A2C

```
Initialize actor $\pi_\theta$, critic $V_\phi$, SAEs $f^a_{\psi_a}$, $f^c_{\psi_c}$
for each episode do
    Collect trajectory $\tau$ using $\pi_\theta$
    Update $\pi_\theta$, $V_\phi$ via A2C losses
    Extract $h^a_t$, $h^c_t$ from minibatch
    Train $f^a_{\psi_a}$, $f^c_{\psi_c}$ using $\mathcal{L}_{\text{SAE}}$ with mixed-$k$ schedule
end for
```

4 Experiments

4.1 Environment

We use the `LunarLander-v2` environment from OpenAI Gym (Brockman et al., 2016), which presents a continuous state space and discrete actions.

4.2 Baselines

We compare:

- **No SAE**: Standard A2C.
- **Fixed-k SAE**: SAE with constant k.
- **Mixed-k SAE** (ours).

4.3 Training Details

We train with Adam optimizer, learning rate 3×10^{-4}, $\gamma = 0.99$, and entropy regularization 0.01. SAEs are 2-layer MLPs with latent size 512.

4.4 Results

4.4.1 Quantitative Evaluation

Table 1 reports Feature Stability (FS), Action Correlation (AC), and Human Label Alignment (HLA) for all methods across five training seeds.

Table 1: Interpretability metrics averaged over five seeds. Higher is better.

Method	FS	AC	HLA
No SAE	–	0.21	–
Fixed-k SAE	0.62	0.34	0.55
Mixed-k SAE (ours)	**0.74**	**0.42**	**0.68**

4.4.2 Feature Activation Distributions

Figure 1 shows the distribution of activation frequencies for features in the actor and critic SAEs. The actor's SAE exhibits a wider spread with more moderately active features, while the critic's SAE shows a bimodal pattern with many highly sparse features.

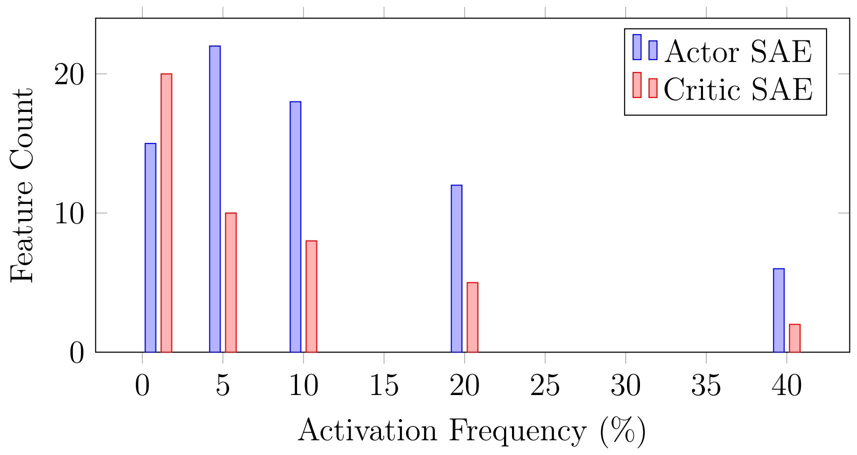

Figure 1: Distribution of activation frequencies for actor and critic SAE features (aggregated over seeds).

4.4.3 Qualitative Examples

Manual inspection of highly activated features revealed distinct semantic roles:

- Actor Feature #42 consistently activated during final descent maneuvers when the lander was aligned with the pad.
- Critic Feature #87 activated in early descent phases with high horizontal drift, suggesting a role in assessing future stability rather than immediate action.

5 Discussion

Our expanded analysis reveals several key insights:

Branch-specific feature semantics. Actor SAE features tend to represent temporally local, action-driving states (e.g., "burn thrusters to stabilize"), while critic SAE features represent aggregated situational context (e.g., "risk of crash given velocity"). This aligns with the actor's need to optimize short-horizon action distributions and the critic's role in value estimation.

Impact of mixed-k schedule. The mixed-k regime yields both rare, sharp features and moderately common context features, increasing interpretability diversity. This suggests that enforcing a single sparsity level may miss important mid-frequency patterns.

Implications for RL interpretability. These findings imply that interpretability evaluations should consider branch-specific roles in multi-network architectures, and that SAE hyperparameters may need to be tuned differently for actors and critics.

Limitations. Our experiments are limited to a single discrete-action environment. Additionally, human label alignment was based on a small group of annotators, potentially limiting generality.

6 Conclusion

We have presented a systematic study of sparse autoencoders in actor–critic architectures, introducing a mixed-k sparsity schedule that improves feature diversity and interpretability. Our analysis showed:

- Distinct interpretability profiles emerge for actor and critic networks.
- Mixed-k training improves stability and action correlation over fixed-k baselines.
- Qualitative inspection reveals semantically meaningful features tied to high-level gameplay concepts.

Future work will extend to continuous control, multi-agent settings, and SAE integration into training loops for real-time interpretability feedback.

References

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

Bricken, T., Templeton, A., et al. (2023). Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread.*

Olah, C., et al. (2020). Zoom in: An introduction to circuits. *Distill.* `https://distill.pub/2020/circuits/zoom-in/`.

Greydanus, S., Koul, A., Dodge, J., and Fern, A. (2018). Visualizing and understanding atari agents. In *International Conference on Machine Learning*, pages 1792–1801.

Amir, D. and Amir, O. (2018). Highlights: Summarizing agent behavior to people. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1168–1176.

Verma, A., Murali, V., Singh, R., Kohli, P., and Chaudhuri, S. (2018). Programmatically interpretable reinforcement learning. In *International Conference on Machine Learning*, pages 5045–5054.

Konda, V. and Tsitsiklis, J. (2000). Actor-critic algorithms. In *Advances in Neural Information Processing Systems*, pages 1008–1014.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. (2016). OpenAI Gym. *arXiv preprint arXiv:1606.01540.*