Reinforcement Learning Project Proposal

Joseph (Jack) Bosco jab2516
Akshara Pramod ap4613

Date : 11th March, 2025

A Study on Explainability in RL Models

Idea

This project aims to improve explainability in reinforcement learning by incorporating Variational Autoencoders (VAEs) to create meaningful latent representations of the CarRacing-v0 environment. Instead of directly training an RL agent on high-dimensional raw pixel inputs, we propose learning a compressed latent space using a VAE as it would capture essential driving features. The RL model will then learn policies in this reduced representation space, making training more efficient and interpretable. To further enhance explainability, we will apply t-SNE to visualize decision-making in the latent space. We also plan to leverage KL divergence in VAE loss to extract more meaningful and well-formed latent space.

Motivation

For AI models to be safely deployed, especially in environments where human safety or well-being is involved, stakeholders must be able to interpret and trust the decisions made by these models. Deep RL models trained on high-dimensional image data often act as black boxes, making it difficult to interpret why certain decisions are made. This lack of transparency limits trust in RL-based autonomous decision-making, particularly in safety-critical domains like self-driving cars.

Proposal

We aim to follow methods set out in [1] using variational autoencoders (VAE) to learn explicit latent space representations for high-dimensional input space. First we pretrain the autoencoder on a bunch of episodes from an actual human playing the game. The objective is to reconstruct the game frame-by-frame through a low-dimensional bottleneck. After pretraining we append a new neural network to the bottleneck in the architecture which experiences reward from the environment and back-propagates the reward to the encoder. This will be the RL component. For explainability we can use VAE to generate interpretable representations from the state space [7]. More interestingly, we can generate novel images using the geometric average latent space coordinate of a certain feature. This latent space arithmetic is known to reveal model biases and is a useful tool for interpreting how the RL agent responds to certain scenarios.

Dataset: CarRacing-v2 (OpenAI Gymnasium Box2D)

The CarRacing-v2 dataset is a continuous high-dimensional control environment from OpenAI Gymnasium's Box2D suite. It involves controlling a car on procedurally generated racetracks, requiring precise navigation and long-term strategy to optimize lap times. The state space consists of 96x96 RGB images, making it a computer vision-based RL problem. The agent controls acceleration, braking, and steering in a continuous action space. The reward is equal to 1000-the time it takes to complete a lap.

1 Methods

1.1 Data Collection and Preprocessing

We collected over 2 000 top-down RGB frames ($96 \times 96 \times 3$) from human-driven episodes of CarRacing-v2. To increase data diversity, we crop off the static toolbar (pixels 0-83) and apply horizontal flips only above that line, effectively doubling samples for left- and right-turn frames [3, 5]. All images are saved as PNG and converted to floating-point tensors in [0,1].

1.2 Dataset Splitting

We treat the full set of augmented frames as a single map-style `Dataset` and split it randomly into 80% train, 10% validation, and 10% test subsets using `torch.utils.data.random_split` [6]. This ensures non-overlapping, stratified usage for pretraining and held-out evaluation.

1.3 Variational Autoencoder Pretraining

Our pretraining objective is to learn a compact latent representation of frames by reconstructing inputs through a low-dimensional bottleneck (Fig. 1). We employ a Variational Autoencoder (VAE) whose encoder outputs mean and log-variance for a diagonal Gaussian in $\mathbb{R}^k$ [4]. The reparameterization trick makes the stochastic sampling differentiable [4]. Each convolutional layer is followed by Batch Normalization and ReLU activations to stabilize and accelerate training [3].

The VAE loss is the negative Evidence Lower Bound (ELBO):

$$\mathscr{L} = \underbrace{\mathbb{E}_{q(z|x)}\left[-\log p(x \mid z)\right]}_{\text{BCE reconstruction}} + \underbrace{D_{KL}\big(q(z \mid x) \,\|\, p(z)\big)}_{\text{KL divergence}}.$$

We compute reconstruction via binary cross-entropy summed over pixels, and the KL term analytically for two Gaussians.

1.4 Implementation Details

We implement the VAE in PyTorch1.x, optimize with Adam (learning rate 1e-3, $\beta_1 = 0.9, \beta_2 = 0.999$), and train for 200 epochs with batch size 64. Model weights are checkpointed every 20 epochs. Early validation reconstructions are displayed periodically to monitor overfitting.

1.5 Rationale for Pretraining

By learning an unsupervised world model of the CarRacing frames, we extract features that capture essential track geometry and car orientation [2]. These compact latent features greatly reduce the state dimensionality for the downstream RL agent, improving sample efficiency and interpretability [7].

References

[1] Christopher Gebauer and Maren Bennewitz. *The Pitfall of More Powerful Autoencoders in Lidar-Based Navigation*. arXiv:2102.02127 [cs]. Mar. 2021. DOI: 10.48550/arXiv.2102.02127. URL: http://arxiv.org/abs/2102.02127 (visited on 03/11/2025).

[2] David Ha and Jürgen Schmidhuber. “World Models”. In: (Mar. 2018). DOI: 10.5281/zenodo.1207631. URL: https://zenodo.org/records/1207631 (visited on 04/29/2025).

Encoder	Decoder
Input: $96 \times 96 \times 3$ RGB image	**Input**: latent sample $\in \mathbb{R}^k$
Conv. $32 \times 3 \times 3$, stride 2, ReLU, BN	Dense, 256, ReLU
Max pool. 2×2	Dense, $3 \times 3 \times 128$, ReLU reshaped
Conv. $32 \times 3 \times 3$, stride 2, ReLU, BN	Trans. Conv., $128 \times 3 \times 3$, stride 2, ReLU
Conv. $64 \times 3 \times 3$, stride 1, ReLU, BN	Trans. Conv., $64 \times 3 \times 3$, stride 2, ReLU
Avg. pool. 2×2	Trans. Conv., $32 \times 3 \times 3$, stride 2, ReLU
Conv. $128 \times 3 \times 3$, stride 2, ReLU, BN, flatten	Trans. Conv., $32 \times 3 \times 3$, stride 2, ReLU
Dense, 256, ReLU	Trans. Conv., $16 \times 3 \times 3$, stride 2, ReLU
Dense, 2k	Conv. $3 \times 3 \times 3$, stride 1
Output: Diag. Gaussian	**Output**: Ind. Bernoulli

Table 1: Revised network architecture for a 96x96 RGB input.

[3] Sergey Ioffe and Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. arXiv:1502.03167 [cs]. Mar. 2015. DOI: 10.48550/arXiv.1502.03167. URL: http://arxiv.org/abs/1502.03167 (visited on 04/29/2025).

[4] Diederik P. Kingma and Max Welling. *Auto-Encoding Variational Bayes*. arXiv:1312.6114 [stat]. Dec. 2013. DOI: 10.48550/arXiv.1312.6114. URL: http://arxiv.org/abs/1312.6114 (visited on 04/29/2025).

[5] Oleg Klimov. *Gymnasium Documentation*. en. URL: https://gymnasium.farama.org/environments/box2d/car_racing.html (visited on 04/29/2025).

[6] *torch.utils.data — PyTorch 2.7 documentation*. URL: https://pytorch.org/docs/stable/data.html (visited on 04/29/2025).

[7] Tom White. *Sampling Generative Networks*. arXiv:1609.04468 [cs]. Dec. 2016. DOI: 10.48550/arXiv.1609.04468. URL: http://arxiv.org/abs/1609.04468 (visited on 03/11/2025).