Hybrid Quantum-Classical Reinforcement Learning in Latent Observation Spaces

Dániel T.R. Nagy[1,2,3], Csaba Czabán[1,3], Bence Bakó[1,2,3],
Péter Hága[3], Zsófia Kallus[3] and Zoltán Zimborás[1,2]

[1] *Eötvös Loránd University, Budapest, Hungary*
[2] *HUN-REN Wigner Research Centre for Physics, Budapest, Hungary*
[3] *Ericsson Research, Budapest, Hungary*

1 Introduction

This work examines the application of quantum reinforcement learning (QRL) [2] in a hybrid quantum-classical system, where the quantum agent is represented as a parametrized quantum circuit (PQC), optimized via gradient descent by a classical optimizer. Many reinforcement learning environments involve high-dimensional observation spaces, such as visual data, resulting in feature vectors with dimensions in the thousands. Encoding such large feature vectors into the limited, noisy qubits available in current quantum devices is infeasible.

To address this, we employ classical autoencoders (AEs) [1] to reduce the dimensionality of the original feature spaces, encoding the resulting latent features into quantum states. While similar approaches have been tested in fully classical settings [3], to our knowledge, this is a novel application for quantum agents.

2 Methods

We use a hybrid quantum-classical system for training QRL agents in classical RL environments. This system integrates a classical AE for dimension reduction, a PQC-based quantum agent, and a classical optimizer that jointly optimizes the parameters of both the quantum and classical components. The joint training algorithm ensures that the AE learns a problem-specific feature compression that is best suited for the given QPU architecture.

The PQC-based quantum agent π_θ uses a QNN circuit architecture defined by the parametric unitary $U(\{\boldsymbol{\theta}_l\}; \mathbf{x})$ with trainable parameters $\{\boldsymbol{\theta}_l\}$, for each QNN layer l. The encoder $\mathcal{E}$ and decoder $\mathcal{D}$ have trainable parameters $\boldsymbol{\theta}_\mathcal{E}$ and $\boldsymbol{\theta}_\mathcal{D}$, respectively. The combined loss function for joint training is:

$$\mathcal{L} = \mathcal{L}^{\text{PPO}} + c_{ae}\mathcal{L}^{\text{AE}},$$

where $\mathcal{L}^{\text{PPO}}$ is the Proximal Policy Optimization (PPO) loss [4], and $\mathcal{L}^{\text{AE}}$ is the AE loss (mean-squared-error). Crucially, all parameters are updated using gradients from the combined loss:

$$\begin{aligned} \boldsymbol{\theta}_l &\leftarrow \boldsymbol{\theta}_l - \alpha \nabla_{\boldsymbol{\theta}_l} \mathcal{L} \\ \boldsymbol{\theta}_\mathcal{E} &\leftarrow \boldsymbol{\theta}_\mathcal{E} - \alpha \nabla_{\boldsymbol{\theta}_\mathcal{E}} \mathcal{L} \\ \boldsymbol{\theta}_\mathcal{D} &\leftarrow \boldsymbol{\theta}_\mathcal{D} - \alpha \nabla_{\boldsymbol{\theta}_\mathcal{D}} \mathcal{L}. \end{aligned} \tag{1}$$

In inference mode, only the trained encoder compresses features into latent vectors for the quantum policy to generate actions. Optionally, we can start the training loop with a pre-trained AE (hot-starting) or cold-starting using random AE weights. Throughout this work, we use quantum agents implemented via a classical critic network and a QNN actor as in [7]. We use strongly entangling [5] QNN layers with Z-measurements for qubit-based agents, while standard CV-QNN layers [6] with homodyne measurements for the photonic agent implementation.

3 Results

We ran a series of numerical experiments to test the feasibility of our approach. We used the `CartPole-v1`, `Acrobot-v1` and `Maze-v0` environments. The first two are part of the OpenAI Gym library, while `Maze-v0` is a custom environment, where the agent has to navigate the player from a random cell to a given goal cell, based on a 48x48 grayscale image observation (see Fig. 1). For all three environments we compared three techniques with a classical baseline: a randomly initialized AE trained together with the agent (Cold AE) a pre-trained AE trained together with the agent (Hot AE†) and a pre-trained AE with freezed weights, where only the agent is traned (Hot AE*). See Fig. 2 for results. To implement the numerical experiments, we used PennyLane [8] for simulating qubit-based QNNs and computing gradients, while for the photonic experiments, we used the Piquasso [9] framework as it supports an efficient TensorFlow backend.

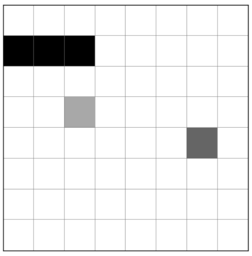
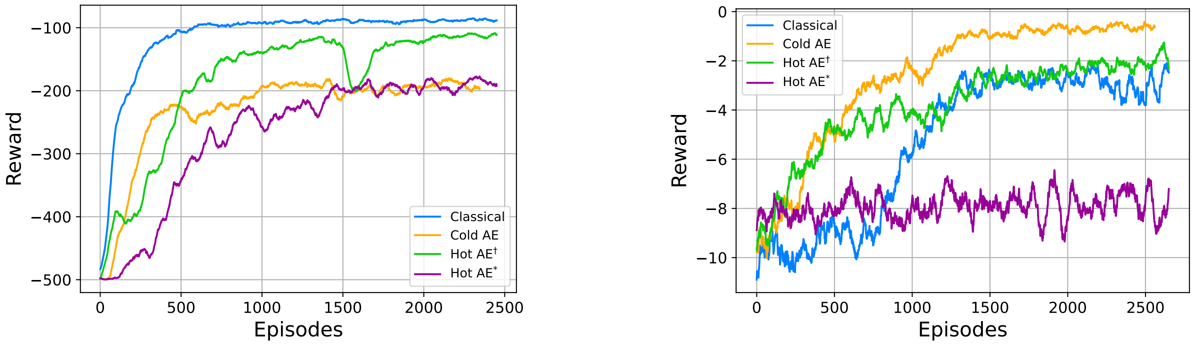


Figure 1: **The `Maze-v0` environment.** The agent has to move the player to the goal (light grey cell) while avoiding obstacles and going off-grid. The player starts at a random cell every time. At each timestep the agent has to choose between 4 possible actions: `up`, `down`, `left`, `right`.



(a) (b)

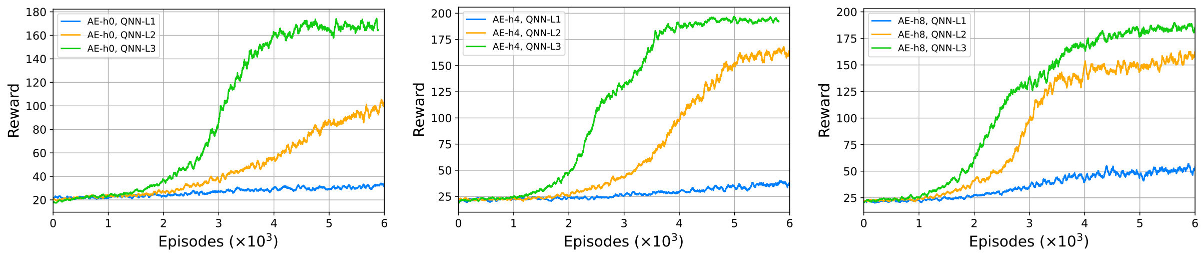
Figure 2: **Preliminary results.** Different techniques of training were tested on the `Acrobot-v1` environment with photonic quantum agents (a) and the `Maze-v0` environment using a qubit-based QNN agent (b); and compared to classical agents. Figures compare three techniques with a classical baseline: a randomly initialized AE trained together with the agent (Cold AE) a pre-trained AE trained together with the agent (Hot AE†) and a pre-trained AE with freezed weights, where only the agent is traned (Hot AE*). Results indicate that the joint training of the AE with the agent is essential for the convergence of reward curves.

3.1 Tradeoff exploration

An interesting aspect of our research was exploring the tradeoff between the size of the AE and the number of QNN layers.

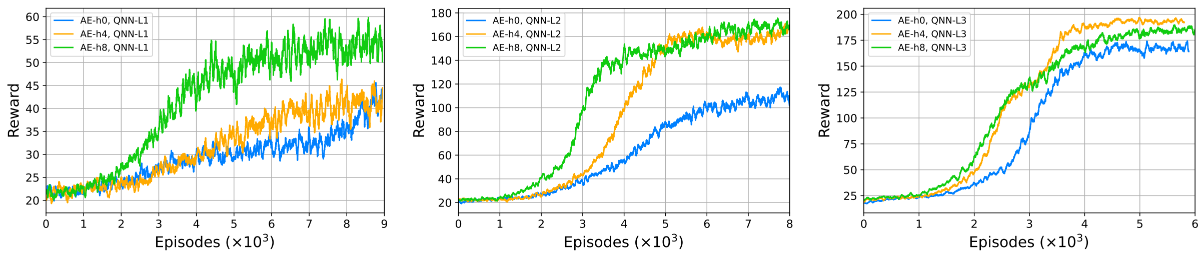
In the `CartPole-v1` environment, our experiments revealed that increasing the size of the AE allows for a reduction in the number of QNN layers required. However, there are limits to this tradeoff. Specifically, for extremely small QNN architectures, such as those with only one layer, a larger AE alone is not sufficient to achieve satisfactory performance. Instead, a minimum of three QNN layers is necessary for the `CartPole-v1` problem to ensure adequate performance.

Figures 3 (a), (b), and (c) illustrate the performance with a fixed number of QNN layers (1, 2, and 3 respectively) and varying AE sizes. Conversely, Figures 4 (a), (b), and (c) show the results for different numbers of QNN layers with fixed AE sizes. These figures collectively demonstrate that as the size of the AE increases, smaller QNN configurations yield improved performance, underscoring the interplay between AE capacity and QNN depth in achieving optimal solutions.



(a) (b) (c)

Figure 3: **Comparing various numbers of QNN layers and AE sizes for the `CartPole-v1` environment.** Figures (a), (b), and (c) show results with fixed AE sizes and different numbers of QNN layers.



(a) (b) (c)

Figure 4: **Comparing various numbers of QNN layers and AE sizes for the `CartPole-v1` environment.** Figures (a), (b), and (c) show results with fixed numbers of QNN layers (1, 2, and 3 respectively) and different AE sizes.

4 Conclusion

This study shows that hybrid quantum-classical reinforcement learning is feasible by combining classical AEs with QNN-based agents to manage high-dimensional observation spaces. Experiments in different environments demonstrate that jointly training the AE and quantum agent significantly boosts performance. The results show the existence of a tradeoff between AE size and QNN layer count.

References

[1] Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neu- ral networks. **Science 313(5786), 504–507** (2006)

[2] Dunjko, V., Taylor, J.M., Briegel, H.J.: Advances in quantum reinforcement learning. In: **2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 282–287** (2017)

[3] Hoof, H., Chen, N., Karl, M., Smagt, P., Peters, J.: Stable reinforcement learning with autoencoders for tactile and visual data. In: **2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3928–3934** (2016)

[4] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal Policy Optimization Algorithms (2017)**arXiv:1707.06347**

[5] Schuld, M., Bocharov, A., Svore, K.M., Wiebe, N.: Circuit-centric quantum clas- sifiers. **Physical Review A 101(3)**(2020)

[6] Killoran, N., Bromley, T.R., Arrazola, J.M., Schuld, M., Quesada, N., Lloyd, S.: Continuous-variable quantum neural networks. **Phys. Rev. Res. 1, 033063** (2019)

[7] Nagy, D., Tabi, Z., Hága, P., Kallus, Z., Zimborás, Z.: Photonic quantum policy learning in openai gym. **2021 IEEE International Conference on Quantum Computing and Engineering (QCE), pp. 123–129.** IEEE Computer Society, Los Alamitos, CA, USA (2021)

[8] Bergholm, V., Izaac, J., Schuld, M., Gogolin, C., Ahmed, S., Ajith, V., Sohaib Alam, M., Alonso-Linaje, G., AkashNarayanan, B., Asadi, A., Arrazola, J.M., Azad, U., Banning, S., Blank, C., Bromley, T.R., Cordier, B.A., Ceroni, J., Del- gado, A., Di Matteo, O., Dusko, A., Garg, T., Guala, D., Hayes, A., Hill, R., Ijaz, A., Isacsson, T., Ittah, D., Jahangiri, S., Jain, P., Jiang, E., Khandelwal, A., Kottmann, K., Lang, R.A., Lee, C., Loke, T., Lowe, A., McKiernan, K., Meyer, J.J., Montañez-Barrera, J.A., Moyard, R., Niu, Z., O'Riordan, L.J., Oud, S., Panigrahi, A., Park, C.-Y., Polatajko, D., Quesada, N., Roberts, C., Sá, N., Schoch, I., Shi, B., Shu, S., Sim, S., Singh, A., Strandberg, I., Soni, J., Száva, A., Thabet, S., Vargas-Hernández, R.A., Vincent, T., Vitucci, N., Weber, M., Wierichs, D., Wiersema, R., Willmann, M., Wong, V., Zhang, S., Killoran, N.: PennyLane: Automatic differentiation of hybrid quantum-classical computations. **arXiv:1811.04968** (2018)

[9] Kolarovszki, Z., Rybotycki, T., Rakyta, P., Kaposi Á., Póor, B., Jóczik, S., Nagy, D.T.R., Varga, H., El-Safty, K.H., Morse, G., Oszmaniec, M., Kozsik, T., Zimborás, Z.: Piquasso: A Photonic Quantum Computer Simulation Software Platform. **arXiv:2403.04006** (2024)