

Adaptive Multi-Objective Reinforcement Learning with Classical and Variational Quantum Deep Q-Networks

Kamineni Pravalika
University of New Haven

Greeshma Chanduri
University of New Haven

Md Moinuddin Bhuiyan
University of New Haven

ABSTRACT

This paper presents an adaptive multi-objective reinforcement learning (MORL) framework integrating classical Deep Q-Networks (DQN) and Variational Quantum Circuit DQN (VQC-DQN). A modified CartPole environment provides two reward signals combined through an adaptive weighting mechanism that updates every 10 episodes and converges in 14 of 15 runs; a corrected replay buffer rescalarizes stored transitions with the current weight instead of a stale one. Across five weight configurations, classical DQN achieves the highest whole-training balance reward, while a parameter-matched Tiny-Classical baseline (37 parameters) attains the best normalized hypervolume (0.412 vs. 0.376 for VQC-DQN and 0.262 for classical); VQC-DQN outperforms Tiny-Classical on raw balance in 3 of 5 weight settings. Retraining VQC-DQN for 200 episodes instead of 150 increases near-convergence balance by 32–163%. At 200 episodes, VQC-DQN has higher point-estimate near-convergence balance than classical DQN across the tested weight settings; however, this comparison is suggestive rather than statistically established because of unequal seed counts, a single 20-episode evaluation window, and unavailable per-seed variance for this metric. A qubit-count sweep under a matched gradient-step budget finds no evidence of barren plateaus within the tested 2–10 qubit range (gradient norm increases, not decreases, with qubit count), while a Deep Sea Treasure benchmark shows classical DQN fails under extreme weights whereas VQC-DQN consistently reaches a treasure but converges to the shallowest policy. These results suggest that classical-versus-quantum comparisons are budget-dependent rather than absolute, pending variance-controlled, multi-seed confirmation.

General Terms

Algorithms, Experimentation, Performance, Design.

Keywords

Adaptive reward weighting, deep Q-network, Q-learning, multi-objective reinforcement learning, Pareto optimization, matched training budget, PennyLane, variational quantum circuits, quantum reinforcement learning, quantum machine learning, CartPole, Deep Sea Treasure, replay buffer, barren plateaus, hypervolume, ablation study.

1. INTRODUCTION

Reinforcement learning (RL) has established itself as a foundational paradigm for sequential decision-making, underpinning landmark results in game playing, robotics, and autonomous control [1–3]. In practice, agents often must balance conflicting objectives simultaneously, a requirement single-objective RL cannot address. Multi-objective RL (MORL) formalizes this by replacing the scalar reward with a reward vector, requiring agents to discover Pareto-efficient policies [4, 5].

Existing MORL methods predominantly rely on static scalarization: a fixed weight vector linearly combines reward

components before RL is applied. Static weights require expert tuning and produce only one trade-off point per run. Recent preference-conditioned methods such as PD-MORL

[6] learn universal policies over the preference space, but still need explicit preference input and use classical agents only.

Concurrently, quantum machine learning has proposed variational quantum circuits (VQCs) as compact function approximators for RL agents [7, 8]. VQC-based DQNs have demonstrated feasibility on small benchmarks [9], yet their behavior under multi-objective settings and adaptive reward weighting remains largely unexplored.

This paper addresses both gaps with a controlled, multi-seed evaluation of adaptive reward weighting across classical and quantum agents. Contributions are:

1. An adaptive reward-weighting mechanism, empirically verified to converge in 14 of 15 runs via explicit tail-stability logging rather than assumed convergence, together with a replay-buffer correction that rescalarizes stored transitions with the current weight instead of a stale one.
2. A controlled test of the classical/VQC episode-count mismatch alongside a parameter-matched “Tiny-Classical” baseline (37 vs. 38 parameters), showing that both training budget and architecture size materially affect the classical-vs-quantum comparison, and that raw balance performance and Pareto trade-off efficiency can favor different architectures.
3. A matched-gradient-step qubit-count sweep (2–10 qubits) and a second, independent MORL benchmark (Deep Sea Treasure), which find no evidence of barren plateaus in the tested range and reveal qualitatively different failure modes between classical and quantum agents.

The remainder is organized as follows: Section 2 reviews related work, Section 3 describes the methodology, Section 4 presents experimental results, and Section 5 concludes.

2. LITERATURE REVIEW

2.1 Deep Reinforcement Learning and MORL

Sutton and Barto [1] provide the foundational treatment of temporal-difference learning and Q-learning; Mnih et al. [2] demonstrated that neural networks can approximate action-value functions at scale with experience replay and target networks. Haarnoja et al. [10] extended deep RL with entropy regularization, and Dulac-Arnold et al. [11] cataloged real-world RL deployment challenges, including the need for multi-objective formulations in safety-critical control.

Rojers et al. [4] provided the first comprehensive survey of

MORL, classifying methods by scalarization strategy and Pareto coverage. Van Moffaert and Nowe [12] developed Pareto-dominance multi-policy learning and introduced the Deep Sea Treasure benchmark used as the second environment (Section 3.8.4), building on Vamplew et al. [13]. Hayes et al. [5] established MORL evaluation standards including hypervolume and sparsity, which are adopted directly in this work (Section 3.7). Basaklar et al. [6] advanced the state of the art with PD-MORL, which outperforms fixed-weight scalarization but requires explicit preference input and classical agents only; this work instead online-adapts weights across both classical and quantum architectures, with a replay buffer that tracks the current weight (Section 3.4).

2.2 Quantum Reinforcement Learning

Biamonte et al. [14] established early theoretical foundations for quantum machine learning, and Cerezo et al. [15] characterized optimization difficulties, including barren plateaus, that arise as variational circuits grow; this is probed in Section 3.8.3 and 4.8. Chen et al. [7] provided the first VQC-DQN with experience replay and target networks, the direct precedent for the present architecture; Lockwood and Si

[8] validated quantum RL feasibility in discrete environments, and Skolik et al. [9] found limited advantage from shallow circuits in a systematic classical-vs-quantum comparison, which Section 4.10 revisits under a matched episode budget. Jerbi et al. [16] proposed complementary policy-gradient quantum policies; Abdelkhalek et al. [17] and Wang et al. [18] report inconsistent quantum deep RL performance, while Schuld and Petruccione [19] and Huang et al. [20] provide broader context.

2.3 Research Gap

Quantum RL studies to date remain single-objective [7, 9], and MORL studies use classical agents exclusively [4, 5, 6]. No prior work combines adaptive reward weighting with VQC-DQN in a multi-objective setting, evaluates it against a parameter-matched classical baseline and an episode-matched budget, and subjects it to target-network, noise, qubit-count, and cross-benchmark ablations. This paper takes a step toward that gap, scoping its claims to what the reported sample sizes support (Section 4.13).

3. METHODOLOGY

3.1 Problem Formulation

The task is modeled as a multi-objective MDP $\langle S, A, T, R, \gamma \rangle$ where $R: S \times A \rightarrow \mathbb{R}^2$ returns a two-dimensional reward vector at each step. For Q-learning, objectives are scalarized:

$$R_t = w_1(t) \cdot R_{\text{bal},t} + w_2(t) \cdot R_{\text{eng},t}, \quad w_1 + w_2 = 1, \quad w_i \geq 0 \quad (1)$$

Weights $w_1(t)$, $w_2(t)$ are time-varying and adapted online by the mechanism in Section 3.3.

3.2 Modified CartPole Environment

The standard CartPole-v1 environment is extended via a wrapper class *MORLCartPole* that replaces the scalar reward with a two-component vector. At each timestep:

- Balance reward: $R_{\text{bal}} = 1.0$ if the episode continues (pole upright, cart in bounds), else 0.0.
- Energy penalty: $R_{\text{eng}} = -(|\dot{x}| + |\dot{\theta}|X)$, where $\dot{x}$ is cart velocity and $\dot{\theta}X$ is pole angular velocity, penalizing kinetic effort.

The four-dimensional observation $(x, \dot{x}, \theta, \dot{\theta}X)$ and binary action space $\{0,1\}$ are unchanged from the standard

environment. CartPole-v1 was selected following Chen et al. [7] and Skolik et al. [9], who established it as a standard quantum-RL feasibility benchmark.

3.3 Adaptive Reward Weighting

Every 10 episodes, the adaptive weight module computes normalized performance contributions from the mean per-episode balance and energy totals accumulated since the previous update, and updates the weight vector via exponential smoothing with factor $\beta = 0.9$:

$$\hat{w}_1 = \max(0, 1 - R_{\text{eng}} / (|R_{\text{bal}}| + |R_{\text{eng}}| + \epsilon))$$

$$(2) \quad \hat{w}_2 = \max(0, 1 + R_{\text{eng}} / (|R_{\text{bal}}| + |R_{\text{eng}}| + \epsilon))$$

$$(3) \quad w \leftarrow \beta \cdot w + (1 - \beta) \cdot \hat{w}, \quad w \leftarrow w /$$

$$\|w\|_1 \quad (4)$$

This rule increases the weight of the objective with relatively lower normalized performance. Section 4.5 logs $w_1(t)$ for every run and tests whether the tail of training has settled: 14 of 15 runs have a second-half maximum weight jump below

0.02 and a tail standard deviation below 0.01; the exception is reported, not discarded. This establishes numerical convergence only, not superiority over a static schedule (Section 4.13).

3.4 Training Algorithm and a Replay-Buffer Correction

Algorithm 1 details the training loop used for every result in Section IV. Replay transitions store the raw two-dimensional reward vector $(R_{\text{bal}}, R_{\text{eng}})$ and are rescaled with the current weight w at sample time (line 10), rather than pre-scalarized at collection time. Because $w(t)$ changes every 10 episodes, scalarizing at collection would let old transitions bias updates toward a stale trade-off; storing raw vectors avoids this. θ denotes the full trainable parameter set of whichever architecture is in use.

Algorithm 1 Adaptive MORL Training (DQN / VQC-DQN)

Require: N agents, E episodes, W_0 , $\beta=0.9$, update interval=10, target_network $\in \{\text{off, on}\}$ (ablation switch)
Ensure: per-run trade-off point set P

```

1: for each agent  $n = 1..N$  do
2:   Init model  $Q_{\theta}$ , buffer  $D$  (cap. 10000),  $w \leftarrow W_0[n]$ ,  $\epsilon \leftarrow 1.0$ 
3:   for  $e = 1..E$  do
4:      $s \leftarrow \text{ENV.RESET}()$ 
5:     while not done do
6:        $a \sim \epsilon\text{-greedy}(Q_{\theta}, s)$ 
7:        $(s', R_{\text{bal}}, R_{\text{eng}}) \leftarrow \text{ENV.STEP}(a)$ 
8:       Push  $(s, a, [R_{\text{bal}}, R_{\text{eng}}], s', \text{done})$  to  $D$ ;  $s \leftarrow s'$ 
9:       If  $|D| \geq 64$ : sample batch, rescalarize each with current  $w$  (Eq. 1), update  $Q_{\theta}$  via MSE TD-loss (target net. if on)
10:      end while
11:       $\epsilon \leftarrow \max(0.05, \epsilon \times 0.995)$ 
12:      if  $e \bmod 10 = 0$  then
13:        Update  $w$  via Eqs. (2)–(4); log  $w_1$  for convergence check
14:      end if
15:    end for
16:   Record avg.  $(R_{\text{bal}}, R_{\text{eng}})$  over final episodes  $\rightarrow P$ 
17: end for
18: return  $P$ 
```

Algorithm 1: Adaptive MORL training loop, including the replay-buffer correction (line 8–9).

3.5 Model Architectures

3.5.1 Classical Deep Q-Network

A three-layer fully connected network, $4 \rightarrow 64 \rightarrow 64 \rightarrow 2$ neurons with ReLU, totaling 4,610 trainable parameters. Adam

optimizer ($\text{lr}=10^{-3}$), MSE TD loss, experience replay ($|D|=10,000$, batch 64), and ϵ -greedy decay ($\epsilon_0=1.0$, $\epsilon_{\min}=0.05$, $\text{decay}=0.995/\text{episode}$).

3.5.2 Tiny-Classical Baseline

To separate raw parameter count from any quantum-specific effect, a second classical network is sized to match VQC-DQN: $4 \rightarrow 5 \rightarrow 2$ neurons, 37 trainable parameters (vs. VQC-DQN's 38). All other hyperparameters match the full classical DQN.

3.5.3 Variational Quantum Circuit DQN

A 4-qubit, 2-layer circuit: a classical pre-layer $\text{Linear}(4 \rightarrow 4) + \tanh$ maps the state to rotation angles; each qubit is encoded via $R_y(x_i)$; two variational layers apply $R_y(\theta_{1,i})$ followed by a CNOT chain; Pauli-Z expectation values $\langle Z_i \rangle$ are read out through $\text{Linear}(4 \rightarrow 2)$ to produce Q-values. Trainable parameters: $\theta \in \mathbb{R}^{2 \times 4}$ (circuit) plus classical layers ($\text{fc}_{\text{in}}: 20$, $\text{fc}_{\text{out}}: 10$), totaling 38. Gradients are computed via the analytic parameter-shift rule and verified against finite differences (relative error $< 10^{-9}$), using PennyLane.

3.6 Hyperparameter Configuration

Table 1 lists all hyperparameters used to produce the results in Section IV. Episode counts differ by architecture (200 for classical/Tiny-Classical, 150 for VQC-DQN) because exact statevector simulation is markedly more compute-intensive per gradient step than the classical forward/backward pass, so equal episode counts are not equal computational cost; Section 4.10 directly tests the effect of this episode-count mismatch by retraining VQC-DQN at 200 episodes. Seed counts are likewise asymmetric: 3 seeds for classical/Tiny-Classical, 5 for VQC-DQN.

Table 1. Hyperparameter configuration for all three architectures, used throughout Section IV.

Parameter	Classical	Tiny-C.	VQC-DQN
Episodes E	200	200	150 ^b
Learning rate	10^{-3}	10^{-3}	10^{-3}
Discount γ	0.99	0.99	0.99
Replay buffer	10,000	10,000	10,000
Batch size	64	64	64
ϵ decay/ep.	0.995	0.995	0.995
Update interval	10 ep.	10 ep.	10 ep.
Smoothing β	0.9	0.9	0.9
Hidden units	64, 64	5	—
Qubits / layers	—	—	4 / 2
Noise (main run)	—	—	noiseless
Target network	off ^a	—	off ^a
Total parameters	4,610	37	38
Seeds / config	3	3	5

^aTarget network is ablated in Section 4.6. ^bSection 4.10 reruns VQC-DQN at 200 episodes (same 5 seeds) to isolate the effect of episode count.

3.7 Evaluation Metrics

Following Hayes et al. [5], four quantities are computed from the scalar reward sequence $\{R_e\}$: (1) mean reward $\bar{R}$; (2) variance σ_Z ; (3) instability index $I = \sigma_Z / (\bar{R} + \epsilon)$; and (4) convergence episode, the first episode where $\text{MA}_{10} \geq 0.8 \cdot \text{MA}_{10}^{\max}$. Table 2 reports (3) and (4) alongside the raw balance reward, averaged over each run's full training history. Section 4.10 additionally uses a near-convergence balance

metric, the mean of history_bal over each run's final 20 episodes, reported as a point estimate without per-seed variance (Section 4.13).

Normalized hypervolume and sparsity are additionally reported, with both objectives min-max normalized jointly across architectures per Hayes et al. [5]. These are computed from each run's whole-training-history average reward rather than a frozen final policy, and each run yields one trade-off point rather than a non-dominated solution set (Section 4.13).

3.8 Ablation and Robustness Studies

3.8.1 Target-Network Ablation

Both classical and VQC-DQN are trained with and without a target network (synced every 50 gradient steps) at $[0.5, 0.5]$, single seed, holding other hyperparameters fixed.

3.8.2 Noise-Model Ablation

The main VQC-DQN results use noiseless statevector simulation. A depolarizing-noise variant is added: after every single-qubit rotation, a random Pauli error (X, Y, or Z) is applied per qubit with probability $p=0.02$ (single seed), and mean reward and instability are compared against the noiseless run.

3.8.3 Qubit-Count Sweep (Matched Compute Budget)

VQC-DQN is retrained at 2, 4, 6, 8, and 10 qubits (2 variational layers, $\text{weight}=[0.5, 0.5]$, single seed). Every configuration trains for a fixed budget of 2,600 global gradient steps rather than a fixed episode count, so this sweep is matched in compute rather than in episodes. The L2 norm of the circuit's parameter-shift gradient is logged at every step as one barren-plateau diagnostic: a shrinking gradient norm with qubit count would be consistent with a barren plateau, while a flat or growing norm would not. This is a single-run check over a narrow range and is not the standard test (gradient variance across many initializations); results are scoped accordingly (Section 4.13).

3.8.4 Second Benchmark: Deep Sea Treasure

To test generalization beyond CartPole, both architectures are evaluated on Deep Sea Treasure (DST) [12, 13], a discrete-grid MORL benchmark with two objectives (treasure value to maximize, time penalty to minimize) and ten treasures at increasing depth. The submarine occupies a 10×11 grid with 4 actions; an episode ends on reaching a treasure or after 50 steps. All 5 standard weight configurations are run for both architectures, single seed, and the fraction of the final 50 episodes in which the agent reached no treasure at all (no-treasure rate) is reported, so that a low mean treasure value can be distinguished from occasional near-misses.

4. RESULTS AND DISCUSSION

4.1 Quantitative Performance

Table 2 reports per-agent balance reward, instability index, and convergence episode for all three architectures across all five weights, aggregated as mean $\pm$ standard deviation across seeds (3 for classical/Tiny-Classical, 5 for VQC-DQN; Section 3.6), averaged over each run's full training history.

Table 2. Adaptive training metrics, all three architectures, mean $\pm$ std across seeds, averaged over full training history.

Arch.	Weights	Balance	Instab.	Conv. Ep.
Classical	[1,0]	63.92 $\pm$ 33.98	25.10	98.0
Classical	[0.8,0.2]	64.21 $\pm$ 34.67	29.95	107.0

Classical	[0.5,0.5]	71.29±11.29	33.51	129.3
Classical	[0.2,0.8]	68.77±5.14	38.74	127.7
Classical	[0,1]	80.16±3.50	56.80	151.7
Tiny-C.	[1,0]	32.12±8.67	8.46	66.0
Tiny-C.	[0.8,0.2]	30.12±4.02	7.84	147.7
Tiny-C.	[0.5,0.5]	48.91±14.00	19.89	170.7
Tiny-C.	[0.2,0.8]	54.44±21.58	32.83	180.7
Tiny-C.	[0,1]	38.92±12.88	48.53	162.3
VQC-DQN	[1,0]	27.24±7.44	16.43	102.6
VQC-DQN	[0.8,0.2]	39.81±9.98	23.91	132.0
VQC-DQN	[0.5,0.5]	59.09±6.56	33.90	139.2
VQC-DQN	[0.2,0.8]	49.65±3.92	52.76	142.6
VQC-DQN	[0,1]	48.46±9.08	82.89	134.4

Instability rises as more weight shifts to the energy objective in all three architectures (classical: 25.10→56.80; Tiny-C.: 8.46→48.53; VQC-DQN: 16.43→82.89), suggesting this reflects the objective structure rather than the model class. On this whole-history metric, classical DQN’s minimum balance reward (63.92, at [1,0]) already exceeds every Tiny-Classical and VQC-DQN configuration’s maximum (54.44 and 59.09 respectively), so classical DQN leads on raw, whole-training balance reward; Section 4.10 revisits this under a near-convergence metric at a matched episode count.

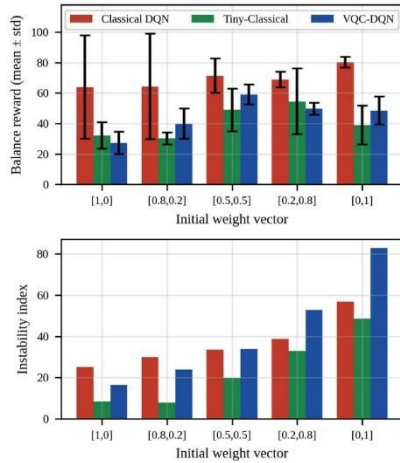


Figure 1: Balance reward (mean ± std across seeds, top) and instability index (bottom) for each architecture and initial weight vector.

Figure 1 shows that the three architectures respond differently to the weight vector. Classical DQN’s balance reward rises toward the energy-dominated end (63.92 at [1,0] to 80.16 at [0,1]) while its seed-to-seed spread collapses from about 34 to 3.5, so its apparent weakness at the balance-dominated weights reflects high variance across seeds rather than a uniformly low score. VQC-DQN peaks at the balanced setting (59.09 at [0.5,0.5]) and Tiny-Classical at [0.2,0.8] (54.44), so only classical DQN is monotonic in the weight. The instability index spans a factor of 2.3 across weights for classical DQN (25.10 to 56.80), 5.7 for Tiny-Classical (8.46 to 48.53), and 5.0 for VQC-DQN (16.43 to 82.89). The mean convergence episode is earliest for classical DQN (98.0 to 151.7), latest for Tiny-Classical (up to 180.7 at [0.2,0.8]), and intermediate for VQC-DQN (102.6 to 142.6).

4.2 Isolating Capacity from “Quantumness”: the Tiny-Classical Baseline

Averaged across all five weights (Table 2), mean balance reward is 69.7 for the full classical network, 40.9 for Tiny-Classical, and 44.9 for VQC-DQN. The 38-param VQC-DQN circuit exceeds the 37-param Tiny-Classical baseline on 3 of 5 weight configurations ([0.8,0.2]: +32%;

[0.5,0.5]: +21%; [0,1]: +25%), with Tiny-Classical ahead only at [1,0] (+18%) and [0.2,0.8] (+10%). Dropping from 4,610 to 37 classical parameters costs 41% of average balance reward, yet the 38-parameter quantum circuit recovers slightly past that loss. These results indicate that architecture-specific differences can persist beyond raw parameter count; however, they do not isolate a uniquely quantum contribution. As Section 4.4 shows, this ranking does not carry over to trade-off efficiency, where Tiny-Classical still leads.

4.3 Trade-off Point Analysis

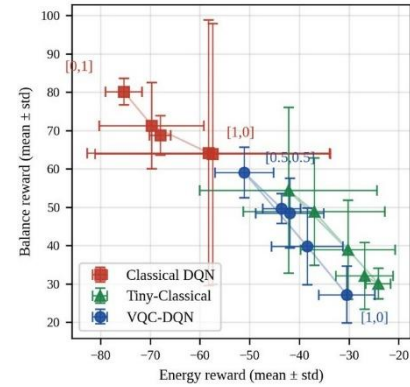


Figure 2: Adaptive trade-off points (mean ± std across seeds) for classical DQN, Tiny-Classical, and VQC-DQN; labels give the initial weight vector.

Figure 2 plots the mean trade-off points. Classical DQN’s points span the highest balance rewards (63.9 to 80.2) at the most negative energy cost (−57.4 to −75.3); every classical configuration exceeds both other architectures’ best point on this whole-history metric. VQC-DQN’s points (27.2 to 59.1 balance, −30.5 to −51.1 energy) overlap substantially with Tiny-Classical’s (30.1 to 54.4 balance, −24.1 to −42.2 energy) rather than trailing them uniformly, consistent with Section

4.2 (see Section 3.7 on how these points should be read).

4.4 Normalized Hypervolume and Sparsity

Table 3. Normalized hypervolume (higher = better trade-off coverage) and sparsity (lower = more even coverage), jointly normalized across all three architectures per Section 3.7.

Architecture	Norm. Hypervol.	Sparsity
Classical DQN	0.2617	59.05
Tiny-Classical	0.4120	68.10
VQC-DQN	0.3762	114.51

Tiny-Classical achieves the highest hypervolume (0.412), VQC-DQN second (0.376), and classical DQN lowest (0.262); the architecture with the highest raw balance reward has the

lowest trade-off efficiency, because its energy cost grows faster than its balance gain. This apparent paradox persists even though, per Section 4.2, VQC-DQN beats Tiny-Classical on raw balance at 3 of 5 weights; VQC-DQN’s higher energy cost relative to Tiny-Classical’s is what keeps it in second place on hypervolume despite its balance lead. Classical DQN retains the best (lowest) sparsity (59.05), with Tiny-Classical (68.10) and VQC-DQN (114.51, least evenly spread) behind it (Figure 3; the top panel shows convergence episodes by weight).

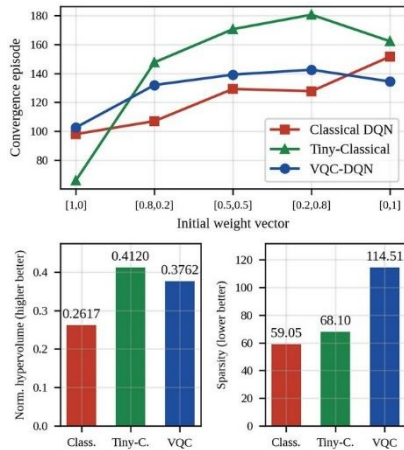


Figure 3: Mean convergence episode by weight vector (top), normalized hypervolume and sparsity by architecture (bottom).

4.5 Adaptive Weight Convergence

Logging $w_i(t)$ for all 15 main-result runs (Section 3.3) and testing whether the second half of training has settled (tail standard deviation < 0.01 and maximum consecutive jump < 0.02) shows 14 of 15 runs settled by the end of training, across all three architectures and all five initial weights. The one exception is VQC-DQN at $[0,1]$ (tail std = 0.0100, max jump = 0.0287), which is still drifting slightly at episode 150; since this is also the configuration with VQC-DQN’s highest instability (82.89, Table 2), the two are plausibly linked, and this run’s trade-off point (Section 4.3) should be read as a snapshot of an incompletely converged policy rather than a stable equilibrium. All other 14 runs converge cleanly. This shows the mechanism converges numerically; it does not show that the resulting policies are better than a static-weight alternative, which was not tested (Section 4.13).

4.6 Target-Network Ablation

Table 4. Target-network ablation, weight=[0.5,0.5], single seed.

Configuration	Instability	Mean Reward
Classical, no target	33.66	23.43
Classical, + target	20.15	20.90
VQC, no target	28.96	24.09
VQC, + target	30.31	22.08

As Table 4 and Figure 4 (top) show, a target network reduces classical DQN instability by 40% (33.66→20.15) at a modest 11% mean-reward cost, but has a small, opposite-direction effect on VQC-DQN (28.96→30.31, +4.7% instability). This is single-seed (Section 4.13); the classical direction is treated as consistent with known target-network decorrelation effects, and the VQC-DQN result as inconclusive rather than evidence of harm.

4.7 Noise-Model Ablation

The main VQC-DQN results use noiseless statevector simulation. In a single-seed depolarizing-noise variant (Figure 4, bottom; $p=0.02$ per single-qubit gate, applied after every rotation), mean reward falls 74% (24.09→6.37) and instability falls 75% (28.96→7.30). This is not a contradiction: instability here is a variance-to-mean ratio, and noise compresses the circuit’s Pauli-Z expectation values toward zero, giving a smaller but more uniform signal despite substantially worse absolute performance.

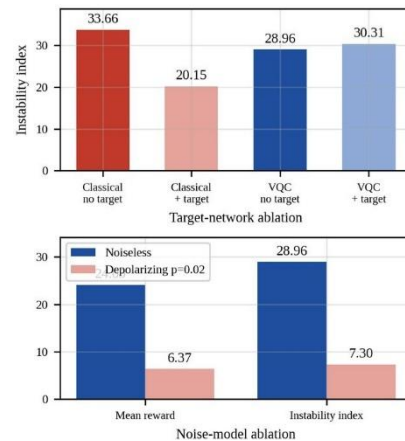


Figure 4: Target-network ablation (top) and noise-model ablation (bottom), single seed, weight=[0.5,0.5].

4.8 Qubit-Count Sweep Under a Matched Compute Budget

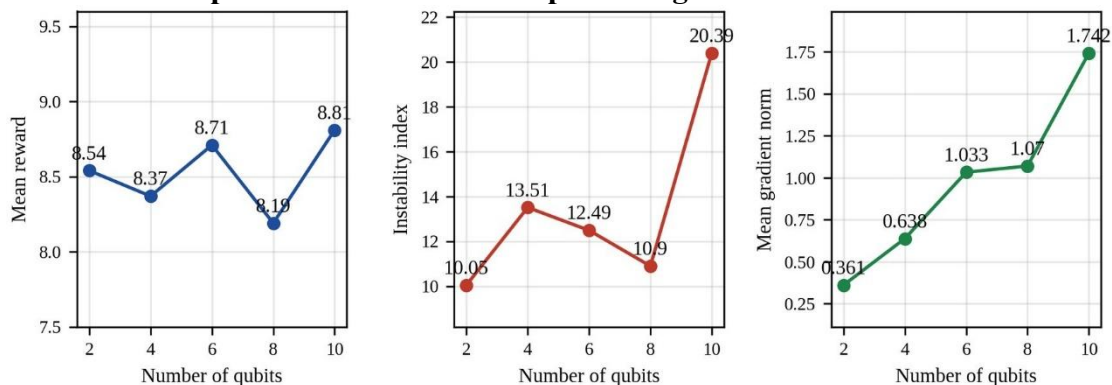


Figure 5: VQC-DQN mean reward, instability index, and mean gradient norm vs. qubit count, matched 2,600-step budget, weight=[0.5,0.5]

Table 5. Qubit-count sweep, weight=[0.5,0.5], single seed, matched 2,600-step budget.

Qubits	Mean Reward	Instab.	Episodes	Grad. Norm
2	8.54	10.05	81	0.361
4	8.37	13.51	83	0.638
6	8.71	12.49	77	1.033
8	8.19	10.90	81	1.070
10	8.81	20.39	68	1.742

Table 5 and Figure 5 show that, under a training budget matched in gradient steps rather than episodes, mean reward is flat across qubit count (8.19–8.81, a 7.6% range, no monotonic trend). The mean gradient norm instead increases monotonically, nearly 5× from n=2 to n=10 (0.361 to 1.742)

— the opposite of what a shrinking-gradient-norm barren-plateau signature would predict, within this range and this single-run diagnostic (Section 3.8.3, 4.13). Instability does spike at n=10 (20.39, roughly double the rest) despite the larger gradient norm, suggesting the 10-qubit circuit’s larger parameter space, rather than vanishing gradients, is associated with instability at high qubit count in this experiment.

4.9 Second Benchmark: Deep Sea Treasure

Table 6. Deep Sea Treasure results, all 5 weight configurations, single seed, mean over final 50 episodes.

Arch.	Weights	Treasure	Time	No-Tr. %
Classical	[1,0]	0.22	−42.72	78%
Classical	[0.8,0.2]	0.48	−34.68	52%
Classical	[0.5,0.5]	85.48	−25.32	6%
Classical	[0.2,0.8]	65.46	−17.70	0%
Classical	[0,1]	5.90	−36.90	52%
VQC	[1,0]	1.04	−1.28	0%
VQC	[0.8,0.2]	1.08	−1.28	0%
VQC	[0.5,0.5]	1.08	−1.22	0%
VQC	[0.2,0.8]	1.06	−1.16	0%
VQC	[0,1]	1.12	−1.34	0%

Table 6 and Figure 6 show what raw treasure values alone could not: the no-treasure rate explains that classical DQN’s near-zero mean treasure at [1,0] and [0.8,0.2] (0.22, 0.48) is a genuine failure mode, reaching no treasure at all on 78% and 52% of episodes. It only reliably solves the task under balanced-to-time-favoring weights ([0.5,0.5]: 6% failure, treasure 85.5).

Table 7. VQC-DQN near-convergence balance reward (mean of final 20 episodes), 150 vs. 200 episodes, 5 seeds. Point estimates only; per-seed variance for this metric is not available (Section 4.13).

Weights	150 ep.	200 ep.	Δ
[1,0]	60.23	158.49	+98.26 (163%)
[0.8,0.2]	87.56	166.20	+78.64 (90%)
[0.5,0.5]	136.13	179.14	+43.01 (32%)
[0.2,0.8]	129.50	215.23	+85.73 (66%)
[0,1]	110.18	160.98	+50.80 (46%)

The effect is large and uniform in direction: 50 more episodes raises VQC-DQN’s near-convergence balance reward by 32–

VQC-DQN never fails to reach a treasure (0% at every weight) but converges to the shallowest, lowest-value treasure (≈ 1.0 – 1.1) regardless of weight. Both failure modes rest on a single seed (Section 4.13) but are real and different: classical DQN is weight-sensitive but brittle at the extremes, while VQC-DQN is robust to outright failure but insensitive to the objective trade-off. VQC-DQN’s episodes are also far shorter (time penalty -1.16 to -1.34 versus -17.70 to -42.72 for classical DQN), consistent with it stopping at the nearest treasure.

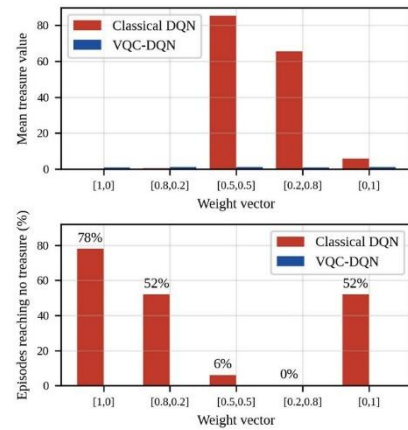


Figure 6: Deep Sea Treasure results: mean treasure value (top) and percentage of final-50 episodes reaching no treasure (bottom), single seed

4.10 Episode-Budget Parity Check

Section 4.1’s comparison trains classical/Tiny-Classical for 200 episodes and VQC-DQN for 150 (Table 1), because exact statevector simulation is far more expensive per step; equal episode counts are therefore not equal computational cost. To test whether this episode-count gap was masking VQC-DQN’s capability, VQC-DQN is retrained at 200 episodes (same 5 seeds) and near-convergence balance reward (mean of the final 20 episodes) is compared at 150 vs. 200 episodes.

163% across every weight, indicating that 150 episodes undertrained it relative to 200. At 200 episodes, VQC-DQN’s near-convergence balance is numerically higher than the corresponding classical DQN point estimate at every tested weight. These differences are reported as point-estimate observations rather than statistically established improvements because Table 7 does not provide per-seed variance, the comparison uses 3 classical versus 5 VQC seeds, and evaluation is based on a single 20-episode window. This is treated as evidence against comparing architectures at unequal episode budgets, not as a confirmed reversal; a multi-seed variance analysis of this metric is the most direct next step (Section 4.13). Figure 7 shows the gain is largest at [1,0] (60.23 to 158.49) and smallest at [0.5,0.5] (136.13 to 179.14).

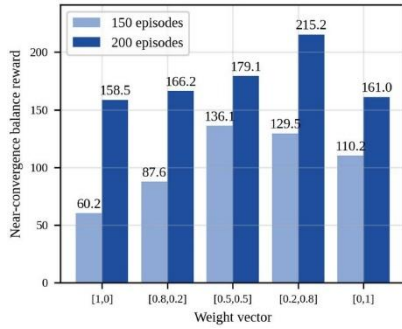


Figure 7: VQC-DQN near-convergence balance reward (final 20 episodes) at 150 vs. 200 training episodes, 5 seeds, point estimates.

4.11 Comprehensive Evaluation

Table 8 consolidates the whole-history results of Tables 2 and 3 into per-architecture summaries, averaged over the five weight vectors. All entries are derived from values already reported in Sections 4.1 and 4.4.

Table 8. Per-architecture summary averaged over the five weight vectors (derived from Tables 2 and 3).

Metric	Classical	Tiny-C.	VQC-DQN
Trainable parameters	4,610	37	38
Avg. balance reward	69.7	40.9	44.9
Avg. energy reward	-65.7	-32.1	-41.1
Balance / energy	1.06	1.27	1.09
Avg. seed std (balance)	17.7	12.2	7.4
Avg. instability	36.8	23.5	42.0
Avg. convergence ep.	122.7	145.5	130.2
Norm. hypervolume	0.2617	0.4120	0.3762
Sparsity	59.05	68.10	114.51
Balance / parameter	0.015	1.105	1.180

Three patterns emerge. First, classical DQN attains the highest average balance reward (69.7) but also the largest average energy cost (-65.7), so its balance gained per unit of energy cost (1.06) is lower than Tiny-Classical's (1.27) and close to VQC-DQN's (1.09); this ratio is the mechanism behind the hypervolume ordering in Table 3. Second, VQC-DQN has the smallest average seed-to-seed spread in balance reward (7.4,

- Narrow-range barren-plateau check. The 2–10 qubit sweep uses a single-run gradient-norm trend, not the standard test (variance across initializations); findings are scoped accordingly.
- Trade-off points from whole-history averages, not frozen policies. Each run yields one point, not a non-dominated solution set.
- Noiseless-by-default simulation, and Deep Sea Treasure is not fully solved by either architecture across all weights.

5. CONCLUSION

This work presents an adaptive MORL framework evaluating classical DQN and VQC-DQN under a controlled, multi-seed protocol, and finds that the classical-vs-quantum comparison depends heavily on training budget and architecture size. At a matched episode count, VQC-DQN is numerically competitive with, or ahead of, classical DQN's near-convergence performance across the tested weights, and a parameter-matched Tiny-Classical baseline shows VQC-DQN exceeding it on raw balance in 3 of 5 configurations; both findings are suggestive rather than statistically confirmed, given limited and unequal seed counts (Section 4.13). A qubit-

versus 17.7 for classical DQN and 12.2 for Tiny-Classical), although it also has the highest average instability index (42.0 vs. 36.8 and 23.5), so its outcomes are consistent across seeds but fluctuate within a training run. Third, per trainable parameter, the 38-parameter circuit and the 37-parameter classical baseline obtain far more balance reward (about 1.18 and 1.1 per parameter) than the 4,610-parameter network (0.015), whereas the absolute reward ordering is reversed. These observations are descriptive and scoped by Section 4.13.

4.12 Comparative Discussion

The picture that emerges is budget-dependent, though not equally well established across comparisons. Classical DQN leads on whole-history balance reward (Section 4.1), while Tiny-Classical achieves the best trade-off efficiency (Section 4.4). At a matched episode count, VQC-DQN is numerically competitive with or ahead of classical DQN at every weight, suggestively (Section 4.10), and outperforms Tiny-Classical on raw balance at 3 of 5 weights (Section 4.2). The qubit-count sweep finds no evidence of a barren-plateau signature within the tested range, with instability better explained by a larger parameter space (Section 4.8). Adaptive weighting converges in 14 of 15 runs (Section 4.5) but was not compared against a static-weight control (Section 4.13). These findings argue for reporting classical-vs-quantum RL comparisons at matched training budgets with explicit variance reporting, rather than a simple “classical wins” or “quantum wins” conclusion.

4.13 Limitations

- No static-weight control. Adaptive weighting converges numerically but is not compared against a fixed-weight baseline, so an advantage over static scalarization is unestablished.
- Limited, unequal seeds. Ablations use one seed; main experiments use 3 (classical/Tiny-Classical) or 5 (VQC-DQN), limiting comparisons such as Tiny-Classical vs. VQC-DQN (Section 4.2).
- Unreported variance, near-convergence metric. Table 7 (Section 4.10) reports means only; the reported advantage of VQC-DQN at 200 episodes is suggestive, pending per-seed variance and significance testing.

count sweep finds no evidence of a barren-plateau signature within the tested 2–10 qubit range, and the adaptive weighting mechanism converges reliably in

14 of 15 runs, though not shown to outperform static scalarization. Results on Deep Sea Treasure show the two architectures failing in different, complementary ways. These findings argue for reporting classical-vs-quantum RL comparisons at matched training budgets with explicit variance reporting, rather than treating a single mismatched-budget run as conclusive. The most direct next steps are a static-weight control, multi-seed variance analysis of the near-convergence metric, a standard barren-plateau diagnostic, additional MORL benchmarks, and real-hardware validation.

6. REFERENCES

- [1] Sutton, R. S. and Barto, A. G. 2018. Reinforcement Learning: An Introduction, 2nd ed. MIT Press, Cambridge, MA.
- [2] Mnih, V. et al. 2015. Human-level control through deep reinforcement learning. Nature 518, 7540 (Feb. 2015), 529-533.

- [3] Arulkumaran, K., Deisenroth, M. P., Brundage, M., and Bharath, A. A. 2017. Deep reinforcement learning: A brief survey. *IEEE Signal Process. Mag.* 34, 6 (Nov. 2017), 26-38.
- [4] Roijers, D. M., Whiteson, S., and Oliehoek, F. A. 2013. A survey of multi-objective sequential decision-making. *J. Artif. Intell. Res.* 48 (2013), 67-113.
- [5] Hayes, C. F. et al. 2022. A practical guide to multi-objective reinforcement learning and planning. *Auton. Agents Multi-Agent Syst.* 36, 1 (Apr. 2022), Article 26.
- [6] Basaklar, T., Gumussoy, S., and Ogras, U. Y. 2023. PD-MORL: Preference-driven multi-objective reinforcement learning algorithm. In *Proceedings of the International Conference on Learning Representations (ICLR 2023)*.
- [7] Chen, S. Y.-C., Yang, C.-H. H., Qi, J., Chen, P.-Y., Ma, X., and Goan, H.-S. 2020. Variational quantum circuits for deep reinforcement learning. *IEEE Access* 8 (2020), 141007-141024.
- [8] Lockwood, O. and Si, M. 2020. Reinforcement learning with quantum variational circuit. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* 16 (2020), 245-251.
- [9] Skolik, A., Jerbi, S., and Dunjko, V. 2022. Quantum agents in the Gym: A variational quantum algorithm for deep Q-learning. *Quantum* 6 (May 2022), 720.
- [10] Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning (ICML 2018)*, 1861-1870.
- [11] Dulac-Arnold, G. et al. 2019. Challenges of real-world reinforcement learning. *arXiv:1904.12901*.
- [12] Van Moffaert, K. and Nowe, A. 2014. Multi-objective reinforcement learning using sets of Pareto dominating policies. *J. Mach. Learn. Res.* 15 (2014), 3483-3512.
- [13] Vamplew, P., Dazeley, R., Berry, A., Issabekov, R., and Dekker, E. 2011. Empirical evaluation methods for multiobjective reinforcement learning algorithms. *Machine Learning* 84, 1-2 (2011), 51-80.
- [14] Biamonte, J. et al. 2017. Quantum machine learning. *Nature* 549 (Sep. 2017), 195-202.
- [15] Cerezo, M. et al. 2021. Variational quantum algorithms. *Nat. Rev. Phys.* 3 (2021), 625-644.
- [16] Jerbi, S. et al. 2021. Parametrized quantum policies for reinforcement learning. In *Advances in Neural Information Processing Systems* 34 (NeurIPS 2021).
- [17] Abdelkhalek, A. et al. 2023. Quantum deep reinforcement learning. *PRX Quantum* 4, 2 (2023).
- [18] Wang, S. et al. 2023. Exploring quantum advantage in reinforcement learning. *Quantum Inf. Process.* (2023).
- [19] Schuld, M. and Petruccione, F. 2018. *Supervised Learning with Quantum Computers*. Springer, Berlin, Germany.
- [20] Huang, H.-Y. et al. 2022. Quantum advantage in learning from experiments. *Science* 376, 6598 (2022), 1182-1186.