Replay buffer engineering for noise-aware quantum circuit optimization

Akash Kundu[1,2*] **Sebastian Feld**[1,2]
[1]Delft University of Technology, Delft, The Netherlands
[2]Quantum Computing Division, QuTech, The Netherlands
{A.kundu, S.Feld}@tudelft.nl



Code on GitHub

Abstract

Deep reinforcement learning for quantum circuit optimization faces three bottlenecks: replay buffers that overlook temporal difference (TD) target reliability, curriculum-based architecture search requiring a full quantum-classical evaluation after every edit, and the discard of noiseless trajectories when retraining under hardware noise. We address these limitations by treating replay as a central algorithmic lever. We introduce ReaPER+, an annealed replay rule that transitions from TD-error prioritization to reliability-aware sampling as value estimates mature. ReaPER+ achieves up to 4x higher sample efficiency than fixed PER, ReaPER, and uniform replay, while matching prior on-policy solution quality with up to 32x fewer interactions At 12 qubits, fixed ReaPER reaches the lowest energy error in the fewest steps, while PER and uniform replay find more compact circuits at higher error. On tasks scaling to 20 qubits, ReaPER+ retains its advantage, demonstrating that reliability-aware annealing extends beyond small-system benchmarks. LunarLander-v3 confirms that the ReaPER+ is domain-agnostic, it improves success rates by up to 26.8% over PER and 21.8% over fixed ReaPER, with a 3% AUC gain over both. We further introduce OptCRLQAS, which amortizes quantum-classical evaluations across multiple architectural edits, reducing training wall-clock time by up to 67.5% on 12-qubit without degrading solution quality. Finally, lightweight replay buffer transfer warm-starts noisy optimization from noiseless trajectories, without weight transfer or ϵ-greedy pretraining, reducing steps to chemical accuracy by 85-90% and final energy error by up to 90% relative to from-scratch learning. Transfer gains increase with system size. Together, these results establish experience storage, sampling, and transfer as decisive levers for sample efficient, noise-aware quantum circuit optimization.

1 Introduction

Optimization is one of the most important application areas of quantum computing [1, 2], both as a target problem class and as an internal algorithmic task in the design and execution of quantum algorithms. On current quantum hardware, quantum computation is realized through sequences of elementary logic gates arranged into quantum circuits [3, 4, 5], and their depth, gate count, and hardware compatibility of these circuits directly determines what can be achieved in practice [6, 7].

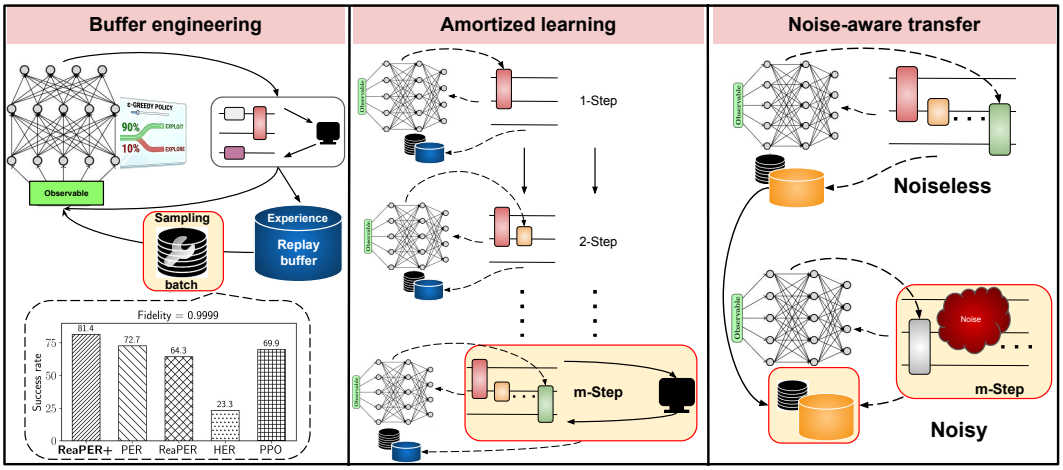
This makes circuit optimization a central challenge across both near-term and fault-tolerant regimes. In the near term [8, 9], limited coherence, connectivity, and high gate noise place strong pressure

*Corresponding author. A.Kundu@tudelft.nl

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

on circuit depth, two-qubit gate count, and hardware compatibility. In the longer term [10, 11, 12], fault-tolerant quantum computing will relax some hardware constraints but will not remove the need for efficient circuit synthesis, compilation, and task-adapted circuit design. Across both settings, the ability to construct compact and effective quantum circuits remains a key determinant of performance, resource cost, and practical utility.

Figure 1: **Overview of replay buffer engineering for quantum optimization.** (Left) *Buffer engineering* improves experience reuse through replay design and sampling. (Middle) *Amortized learning* reduces the cost of curriculum RL-based quantum architecture search by performing expensive quantum-classical updates only every m steps. (Right) *Noise-aware transfer* warm-starts the RL-training in noisy environment by reusing trajectories collected in noiseless training.



Reinforcement learning (RL) [13] has emerged as a promising framework for this problem because it treats circuit design as a sequential decision process, enabling agents to construct gate sequences or parameterized circuit architectures step by step [14, 15]. RL has been successfully applied to quantum compilation [16, 17] and variational quantum circuit design [14, 18]. However, its practical use remains limited by a basic inefficiency: *past experience is often poorly reused*. In most existing workflows [14, 19, 20, 21], each new setting is treated as a fresh training problem, so trajectories collected in a noiseless simulator are discarded when the task is revisited under hardware noise which is an especially costly choice in quantum optimization [22], where the noiseless-to-noisy gap represents a shift between fundamentally distinct computational substrates that is physically motivated, practically unavoidable, and grows in severity with system size, making experience reuse both more challenging and more valuable than in classical transfer settings. In the standard variational framework [23], a parameterized quantum circuit $U(\boldsymbol{\theta}, \mathcal{A})$ is defined by a discrete architecture $\mathcal{A}$ and continuous parameters $\boldsymbol{\theta}$, preparing a state $|\psi(\boldsymbol{\theta}, \mathcal{A})\rangle = U(\boldsymbol{\theta}, \mathcal{A})\,|0\rangle^{\otimes n}$, with cost $C(\boldsymbol{\theta}, \mathcal{A}) = \langle\psi(\boldsymbol{\theta}, \mathcal{A})|\, H\, |\psi(\boldsymbol{\theta}, \mathcal{A})\rangle$ estimated on the QPU and a classical optimizer updating $\boldsymbol{\theta}$ toward targets $d(U(\boldsymbol{\theta}, \mathcal{A}), U_{\text{tar}}) \leq \varepsilon_{\text{comp}}$ or $C(\boldsymbol{\theta}, \mathcal{A}) - E_0 \leq \varepsilon_{\text{chem}}$. Each RL step may trigger a full quantum-classical evaluation, so wall-clock time grows rapidly with system size. While TensorRL-QAS [24] narrows the search space via tensor network warm-starts, it still incurs substantial per-step cost and relies on standard replay buffers without explicit experience transfer; similarly, transfer in RL-optimized error correcting codes [25] does not treat replay buffer design as the primary transfer vehicle. As a result, the practicality of RL-based quantum optimization depends not only on the agent architecture, but critically on *how experience is stored, prioritized, and transferred across the noiseless-to-noisy scenario* which is a direction that is not explored yet in the scope of quantum optimization.

We address these bottlenecks by treating the replay buffer as a primary algorithmic lever for RL-based quantum optimization, motivated by evidence from classical RL that buffer composition and sampling strongly affect stability and sample efficiency [26, 27, 28, 29, 30]. Figure 1 summarizes our framework, combining annealed replay, amortized curriculum learning, and lightweight noiseless-to-noisy transfer. Concretely, we make three contributions:

1. **Annealed replay for quantum optimization.** Under a fixed DQN [31]/DDQN [32] setup, we introduce *ReaPER+*, an annealed replay rule transitioning from TD error prioritization [33] to reliability-aware replay [34]. It achieves up to $4\times$ greater sample efficiency over the strongest off-policy baselines (HER [35], PER [33], and ReaPER [34]) and matches on-policy (PPO) solution quality at $32\times$ fewer interactions; a LunarLander-v3 validation (Appendix F) confirms it is not domain-specific. Under a stated variance-alignment condition, ReaPER+ reduces the asymptotic noise floor versus PER (Theorem D.1, Appendix D), a conditional, not unconditional, result. In Appendix D.2, Algorithm 1 gives the exact implementation, noting three deviations from this idealized form: a reward-based priority warm-start, an approximate online reliability update, and the importance-sampling weighting used in the loss.
2. **Amortized curriculum learning.** *OptCRLQAS*, an amortized variant of CRLQAS [19], reuses quantum-classical evaluations across multiple architectural edits, cutting average wall-clock time per episode by 67.5% ($3\times$ faster) on 12-qubit H_2O without degrading final energy error or gate count.
3. **Replay-based transfer to noise.** A lightweight transfer scheme reuses noiseless trajectories to warm-start noisy-setting learning, without network-weight transfer or long ϵ-greedy pretraining. On 6-qubit BEH_2 and 8-/12-qubit H_2O tasks, this buffer-only transfer cuts steps to chemical accuracy by up to 85-90% and final energy error by up to 90% over from-scratch noisy baselines, with the advantage growing with system size.

2 Related work

Quantum circuit optimization [1] has been approached via adaptive ansatz construction [36], differentiable architecture search [37, 38], evolutionary heuristics [39, 40], Bayesian optimization [41], Monte Carlo tree search [42], neural predictors [43], and sampling-based strategies [44], demonstrating automatic circuit structure discovery but revealing persistent challenges in scaling and noise robustness. Within this landscape, reinforcement learning has emerged as a flexible framework for both quantum compiling [16] and quantum architecture search [14, 45, 46], with value-based and policy-gradient methods applied to variational ground state preparation [14, 19, 24], entangled-state generation [46, 47], and hardware-aware circuit design [48, 49]. However, existing RL-based frameworks treat each new setting as a fresh training instance, when a task moves from a noiseless simulator to noisy hardware, the agent is retrained from scratch and accumulated experience is discarded, incurring substantial GPU and CPU cost per retraining cycle.

Experience replay is a central component of off-policy deep RL [31], with uniform replay, hindsight experience replay [35], prioritized experience replay [33], and reliability-aware variants [34] providing increasingly structured trade-offs between coverage, informativeness, and target reliability. In classical RL, recent offline-to-online methods [27, 28, 29, 30] demonstrate that buffer composition and sampling rules are critical for stable, sample-efficient fine-tuning. In quantum optimization [19, 25], however, replay buffers have largely been treated as fixed implementation choices rather than as a primary algorithmic design lever. This is the gap our work directly addresses by treating the replay buffer as the central object of design for quantum circuit optimization.

3 Methods

We introduce a replay buffer engineering algorithm for quantum circuit optimization (Figure 1) with three components: annealed replay (*ReaPER+*), amortized curriculum learning (*OptCRLQAS*), and lightweight noiseless-to-noisy buffer transfer. All experiments fix a common DQN/DDQN agent with identical state representation, action space, reward, and training protocol where only the replay mechanism is varied.

ReaPER+. Among existing strategies, PER and ReaPER emerge as complementary baselines. PER (see Equation 12) prioritizes transitions with large TD errors, providing aggressive early exploration but potentially amplifying noisy targets. ReaPER (see Equation 14) discounts transitions whose downstream TD errors indicate unreliable targets, better matching the long-horizon structure of episodic quantum optimization but converging more slowly in the early phase. This complementary behavior motivates a hybrid that transitions from PER-like exploration to ReaPER-like refinement over the course of training.

To exploit the best of both regimes, we introduce *ReaPER+*, a replay strategy that transitions smoothly from PER-like prioritization to ReaPER-like prioritization over the course of training.

Specifically, at training step τ we define

$$\Psi_t^{(+,\tau)} = \mathcal{R}_t^{\omega_\tau}(\delta_t^+)^\alpha, \qquad \mu_t^{(+,\tau)} = \frac{\Psi_t^{(+,\tau)}}{\sum_i \Psi_i^{(+,\tau)}}, \tag{1}$$

where $\mathcal{R}_t$ is the reliability score at step t (described more elaborately in Appendix C), the annealing exponent $\omega_\tau \in [0, 1]$ is non-decreasing in τ. $\omega_\tau = 0$ recovers PER and $\omega_\tau = 1$ recovers ReaPER exactly. In practice, we use the linear schedule

$$\omega_\tau = \omega_{\min} + (\omega_{\max} - \omega_{\min}) \min\left(\frac{\tau}{T_{\text{ann}}}, 1\right), \tag{2}$$

which is monotone and controlled by a single interpretable timescale T_{ann}, avoiding the rapid early transitions of cosine or exponential schedules when reliability estimates are still poorly calibrated. We set $\omega_{\min} = 0.1 > 0$ so that a residual TD error signal is retained at initialization, where $\mathcal{R}_t$ carries little meaning under a random Q-function, and $\omega_{\max} = 0.7 < 1$ to prevent premature over-commitment to reliability scores before Q-function convergence values $\omega_{max} \geq 0.9$ produced slower convergence in both settings; see Appendix N.2. T_{ann} is set to half the total training budget (5×10^5 for compilation, 5×10^4 for LunarLander-v3; Table 20), so the transition completes by mid-training when value estimates are sufficiently stable. Early in training sampling is thus driven by TD error; later, the influence of $\mathcal{R}_t$ grows, biasing replay toward transitions that are both informative and reliable. ReaPER+ therefore preserves the sample-efficiency advantages of PER at the beginning of learning while inheriting the stability of ReaPER once value estimates mature, a design choice directly motivated by the empirical finding that PER excels early while ReaPER is preferable once estimates stabilize this is confirmed by our systematic benchmark of uniform replay, HER [35], PER [33], and ReaPER [34] under a fixed agent across all tasks. Under a stated variance-alignment condition on the reliability score, ReaPER+ reduces the asymptotic noise floor compared to PER in Theorem D.1 (Appendix D); we present this as a conditional theoretical result rather than an unconditional guarantee.

Lightweight buffer transfer. In addition to replay buffer engineering, we study a *replay buffer transfer* scheme for noise-aware quantum optimization. The idea is to first collect trajectories in a source environment, typically the noiseless version of a given compiling or QAS task, and then use the resulting replay memory to initialize training in the corresponding noisy target environment. Let $\mathcal{B}_{\text{src}}$ denote the replay buffer obtained after source training and $\mathcal{B}_{\text{tgt}}^{(0)}$ the initial replay buffer of the target task. We initialize the target buffer directly as

$$\mathcal{B}_{\text{tgt}}^{(0)} \leftarrow \mathcal{B}_{\text{src}}, \tag{3}$$

that is, all stored transitions $(S_t, A_t, R_t, S_{t+1}, d_t)$ are copied from the source buffer to the target buffer without modification, relabeling, or filtering. This is valid because the noiseless and noisy environments share identical state and action spaces: the RL-state and the gate action set $\mathcal{A}$ are unchanged by the introduction of noise, which only affects the transition dynamics and reward statistics. Buffer-only transfer of this kind, where source experience warm-starts a target replay buffer without sharing network weights, has recently proved effective in classical deep RL. Zhou et al. [27] show that seeding the online replay buffer with offline transitions suffices for sample-efficient fine-tuning without retaining the offline dataset or transferring network parameters. Lee et al. [28] study balanced mixing of offline and online replay in the offline-to-online setting, finding that the buffer composition critically determines early learning speed. Our scheme follows the same instance-transfer paradigm [50]: experience is transferred at the trajectory level rather than the parameter level. We deliberately decouple buffer and weight transfer, as network weights trained under noiseless dynamics may introduce systematic bias when the target environment includes depolarizing noise which is a risk that does not arise with buffer-only transfer, where source transitions are corrected online as the agent accumulates noisy target experience.

Consequently, trajectories that are informative in the noiseless setting can still provide a strong initialization for learning under noise by improving early buffer coverage and accelerating the discovery of high-quality circuits. In Appendix E we provide a formal argument for why replay buffer transfer is natural in quantum optimization settings under the assumption that source and target tasks share approximately similar state and action spaces.

OptCRLQAS. Finally, to scale while operating with CRLQAS we realized the time per episode scales rapidly with qubit count, and training the RL-agent with CRLQAS beyond 10-qubit can take a substantial amount of GPU compute. To address this issue we introduce *OptCRLQAS*, an efficiency-oriented variant of curriculum reinforcement learning-based quantum architecture search. In standard CRLQAS, each environment step triggers a full quantum-classical evaluation of the current circuit, including variational parameter optimization and cost-function estimation, so for an episode of length T, the total evaluation cost scales as $T \cdot C_{\text{eval}}$, where C_{eval} denotes the cost of a single quantum-classical call. OptCRLQAS reduces this by *amortizing* each evaluation over m consecutive architectural edits: rather than invoking a new variational optimization at every step, the agent accumulates m local gate modifications before triggering a single evaluation, spreading the cost C_{eval} across m steps. Formally, let $u_\tau \in \{0, 1\}$ denote the update indicator,

$$u_\tau = \begin{cases} 1, & \text{if } \tau \bmod m = 0 \text{ or the episode terminates,} \\ 0, & \text{otherwise,} \end{cases} \tag{4}$$

so that a full architecture evaluation is performed only when $u_\tau = 1$. Hence, for an episode of length T, the number of expensive quantum-classical evaluations is reduced from T in CRLQAS to approximately $\lceil T/m \rceil$ in OptCRLQAS, yielding an expected reduction by a factor of about m when episode lengths are sufficiently large.

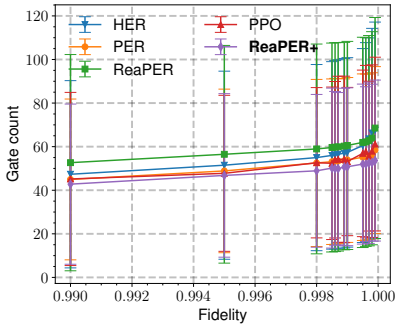
Beyond the computational saving, accumulating m edits before evaluating also improves the learning signal: single-gate modifications are often too small to produce distinguishable rewards when variational parameters $\boldsymbol{\theta}$ can compensate local changes, leading to weakly separated TD targets and slow value propagation. Judging a block of m edits jointly increases reward contrast and allows useful gate combinations to be credited as a unit. A more detailed discussion of both effects is provided in Section 4.2. Throughout the paper, *OptCRLQAS* refers to $m = 10$ unless stated otherwise; a quality-vs-runtime sweep over $m \in \{3, 5, 7, 10\}$ in Appendix J.3 confirms this efficiency gain holds across the full range without systematically degrading solution quality.

4 Results

We evaluate replay buffer design in quantum compilation, where the agent's task is to synthesize a target unitary encoded in an RL-state of size 2×2^N by sequentially appending gates from a fixed action space defined in Section C.2.1. In every training episode the environment resets to the $|0\rangle$ state and a fresh Haar-random 1-qubit target unitary is sampled; the agent acts until the target fidelity tolerance is met or the maximum episode length L is reached. All agents are trained for

Figure 2: **1-qubit compiling of Haar-random target unitaries with** `RX`, `RY`, `RZ`$(\pm\pi/128)$ **gates.** (Left) Success probability and mean fidelity at tolerances 0.99, 0.999, and 0.9999, where ReaPER+ performs best overall. HER [35] uses uniform (non-prioritized) sampling over its goal-relabeled buffer and serves as a uniform-replay reference point. (Right) mean circuit length with $\pm 1\sigma$ error bars versus tolerance; although all methods require deeper circuits at higher accuracy and exhibit a similar growth rate with tightening tolerance, ReaPER+ maintains a consistently lower circuit-length offset, giving the best accuracy-length tradeoff across all tolerance levels.

Tol	Method	Success (%) ↑	Avg. fidelity ↑
0.99	**ReaPER+ (Ours)**	**89.30**	**0.94**
	PER [33]	85.81	0.94
	ReaPER [34]	85.15	0.93
	HER [35]	76.49	0.89
	PPO [51]	75.40	0.88
0.999	**ReaPER+ (Ours)**	**85.30**	**0.95**
	PER	82.35	0.94
	ReaPER	80.97	0.93
	HER	74.44	0.90
	PPO	71.60	0.87
0.9999	**ReaPER+ (Ours)**	**81.40**	**0.95**
	PER	72.66	0.94
	ReaPER	64.35	0.93
	HER	63.30	0.90
	PPO	69.90	0.87



5×10^4 episodes and evaluated on 10^5 independently sampled Haar-random target unitaries. Full hyperparameter details are given in Appendix R. Moreover, the compute details are provided in Appendix S.

Experimental setup and reproducibility. All experiments follow a common pipeline: an RL agent (DQN [31]/DDQN [32], Section 3) proposes a gate action, the resulting circuit is evaluated by a quantum simulator. The parameter of quantum circuit optimized using `COBYLA` [52] and then quantum simulator does cost function estimation. Finally the resulting cost/fidelity signal is returned to the agent as a reward. Quantum compiling experiments (Section 4.1) use `Qiskit` [53], while quantum architecture search experiments (Section 4.2) use `Qulacs` [54] for faster state vector simulation at larger qubit counts. Noise is injected at the simulator level via depolarizing and amplitude damping channels (Section 4.3). All experiments were run on local and HPC machines, with per-experiment runtimes not exceeding 48 hours; hardware configurations and wall-clock time for every experiment category are summarized in Appendix S. Complete hyperparameters are listed in Appendix R, molecule geometries and Hamiltonians in Appendix O, and the full noise-modeling stack in Section 4.3 and Appendix E. Our complete training code, environment wrappers, and evaluation scripts will be publicly released to enable full reproduction of every result in this paper.

4.1 Quantum compiling

1-qubit compiling. We evaluate two settings: (i) the small-rotation basis $\texttt{RX}, \texttt{RY}, \texttt{RZ}(\pm\pi/128)$ and (ii) the discrete HRC gate set [55], following Ref. [16]. Figure 2 reports success probability and fidelity over tolerances 0.99-0.9999 (40 seeds). ReaPER+ achieves the highest success at every tolerance (89.30%, 85.30%, 81.40%), outperforming PER (85.81%, 82.35%, 72.66%) and fixed ReaPER (85.15%, 80.97%, 64.35%), while showing the slowest circuit-length growth, yielding the best success-fidelity-length tradeoff overall. Unlike PER, ReaPER, and ReaPER+, which differ in how transitions are *prioritized*, HER instead samples uniformly from a goal-relabeled buffer, so it serves here as a uniform-replay reference point rather than a fourth prioritized variant. To assess the reliability of this circuit-length advantage, we additionally conduct a sample significance analysis in Appendix G. While the resulting 95% confidence intervals are wide due to the small sample size, the analysis confirms a consistent downward shift in gate count for ReaPER+ relative to PER and fixed ReaPER across all three tolerances. For the HRC basis, Table 7 shows that ReaPER+ reaches 100% success with mean fidelity 0.995 and shortest circuits (14.30 ± 7.89 gates) at 1.56×10^6 steps, $\sim$24% faster than fixed ReaPER, $\sim$ 26% faster than PER, and $\sim$72% faster than our tuned HER baseline (k=5 relabelings, 100-episode warm-up), which despite being stronger than the original [16] still plateaus at 95% success. All subsequent "HER" references denote this tuned implementation.

2-qubit compiling. Using target unitaries sampled via Algorithm 2, we test whether agents can approximate a $\texttt{ZZ}(\pi)$ gate at fidelity threshold 0.9914, matching the benchmark of Ref. [16]. Table 1 shows ReaPER+ reaches fidelity 0.9920 in only 2.5×10^4 episodes, a $4\times$ reduction over fixed ReaPER, PER, and HER (all at 10^5 episodes), and a $\mathbf{32}\times$ reduction over PPO [16] (8×10^5 episodes). Although fixed ReaPER attains a marginally higher best fidelity (0.9931) given its longer budget, ReaPER+ reaches a comparable fidelity regime far more efficiently, the operationally relevant metric when quantum-classical simulation time is limited.

Table 1: **ReaPER+ outperforms all baselines on $ZZ(\pi)$ approximation using 2-qubit gates.** ReaPER+ matches or exceeds fidelity at $32\times$ fewer episodes than PPO [16]. "NA" means the with HER buffer we could not achieve a fidelity below the pre-defined fidelity threshold 0.9914 with fixed computational budget.

Method	Episodes ↓	Min. gates ↓	Best fidelity ↑
ReaPER+ (Ours)	$\mathbf{2.5 \times 10^4}$	**123**	0.9920
ReaPER [34]	10^5	126	**0.9931**
PPO (Moro et al. [16])	8×10^5	122	0.9914
PER [33]	10^5	127	0.9918
HER [35]	10^5	NA	< 0.9914

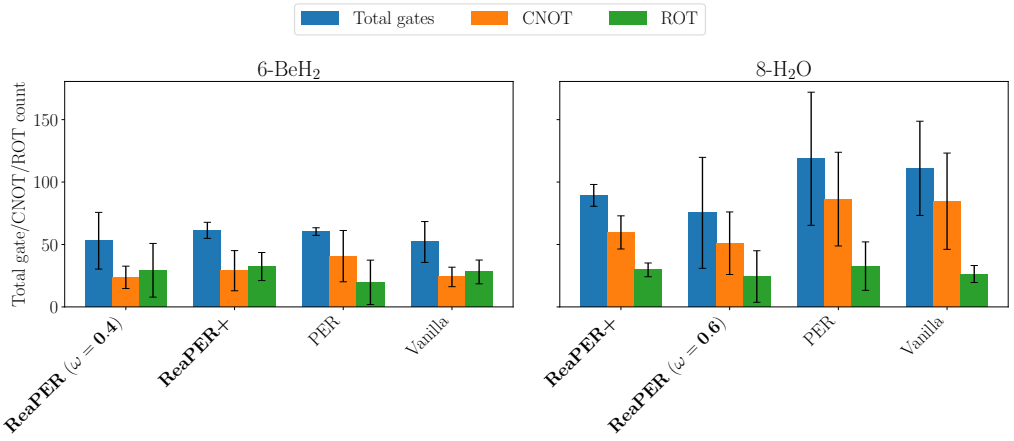
4.2 Quantum architecture search

We evaluate replay buffers in QAS, where the agent seeks a parameterized circuit that prepares the ground state of a target molecular Hamiltonian. Following CRLQAS [19, 14], we use the gate set in

Table 2: **OptCRLQAS + ReaPER+ vs. non-RL baselines.** We report mean ± standard deviation over 5 random seeds for our method. Our method achieves the lowest energy error on 2 of 3 systems while using competitive or fewer gates and CNOTs. The 5-qubit Heisenberg model is described in Appendix O.2. For 5-Heisenberg we utilize an action space consisting of $\{\texttt{XX}(\theta_{xx}), \texttt{YY}(\theta_{yy}), \texttt{ZZ}(\theta_{zz}), \texttt{RX}(\theta_x), \texttt{RY}(\theta_y), \texttt{RZ}(\theta_z)\}$ gates. Baseline results were not re-run (NR) as per-seed statistics/code were unavailable, except SA-QAS, which we re-ran ourselves (R).

Problem	Method	Final error (Ha)	Total gates	CNOT
5-Heisenberg	**OptCRLQAS + ReaPER+ (ours)**	$\mathbf{(5.9 \pm 0.7) \times 10^{-4}}$	41.0 ± 7.5	NA
	DQAS [37] (NR)	1.1×10^{-1}	35	NA
	GQAS [56] (NR)	7.1×10^{-4}	35	NA
	TF-QAS [57] (NR)	1.2×10^{-3}	35	NA
6-$\texttt{BEH}_2$	**OptCRLQAS + ReaPER+ (ours)**	$\mathbf{(6.2 \pm 3.3) \times 10^{-5}}$	$\mathbf{52.8 \pm 14.4}$	$\mathbf{24 \pm 6.1}$
	TF-QAS [57] (NR)	1.8×10^{-3}	57	NA
	SA-QAS [58] (R)	5.6×10^{-3}	73	45
8-$\texttt{H}_2\texttt{O}$	**OptCRLQAS + ReaPER+ (ours)**	$(2.2 \pm 1.4) \times 10^{-4}$	$\mathbf{89.8 \pm 8.6}$	$\mathbf{61.2 \pm 12.6}$
	quantumDARTS [38] (NR)	$\mathbf{1.7 \times 10^{-4}}$	219	68
	SA-QAS [58] (R)	2.6×10^{-3}	95	69

Figure 3: **Replay buffer design controls circuit compactness in QAS.** For 6-$\texttt{BEH}_2$ and 8-$\texttt{H}_2\texttt{O}$, ReaPER+ variants yield the lowest total, CNOT, and rotation gate counts compared to PER and uniform replay (mean ± std over seeds). ω=0 recovers PER; ω=1 gives fully reliability-adjusted replay. Note that the error bars are 1-sigma.

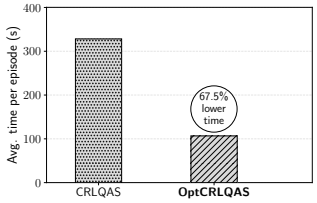


Equation 20 and reward in Equation 23, benchmarking molecular ground state preparation tasks of increasing scale (molecule geometries and hyperparameters in Appendices O and P, respectively).

Figure 3 compares replay strategies on the smaller-scale systems. ReaPER+ achieves the lowest energy error among prioritized methods with competitive circuit compactness; fixed ReaPER (ω=0.4 for $\texttt{BEH}_2$, ω=0.6 for $\texttt{H}_2\texttt{O}$) produces the most compact circuits, reflecting the longer-horizon credit assignment at larger scale (full ω sensitivity in Appendix N). Uniform replay yields superficially shorter circuits but at substantially higher energy error, indicating early trapping in local minima. As shown in Table 2, OptCRLQAS with ReaPER+ (denoted as "OptCRLQAS + ReaPER+") achieves the lowest energy error on the 5- and 6-qubit QAS problems, and is competitive with quantumDARTS [38] on the 8-qubit problem, outperforming non-RL baselines such as DQAS [37], GQAS [56], TF-QAS [57], SA-QAS [58], and quantumDARTS [38] while using competitive or fewer gates and CNOTs overall. Results for our method are reported as mean ± standard deviation over 5 random seeds. For the 5-qubit problem, we target the ground state energy of the Heisenberg model (Hamiltonian given in Appendix O.2). Since this problem is well-suited to the gateset $\{\texttt{RXX}(\theta_{xx}), \texttt{RYY}(\theta_{yy}), \texttt{RZZ}(\theta_{zz}), \texttt{RX}(\theta_x), \texttt{RY}(\theta_y), \texttt{RZ}(\theta_z)\}$, we employ an alternative RL state encoding described in Appendix Q.

At 12-qubit scale, standard CRLQAS is prohibitively expensive since every step triggers a full variational optimization. OptCRLQAS amortizes this cost over m steps, reducing evaluations per episode from T to $\lceil T/m \rceil$ and cutting average wall-clock time by 67.5% ($\sim 3\times$) without degrading solution quality (Figure 4). A quality-vs-runtime sweep over the amortization interval m in Appendix J.3 confirms this efficiency gain holds across $m \in \{3, 5, 7, 10\}$ without systematically degrading solution quality. Batching m edits also yields a more separable learning signal by crediting meaningful architectural blocks rather than nearly indistinguishable single-step edits.

Figure 4: **Efficiency and performance on 12-qubit H_2O. (Left)** OptCRLQAS reduces wall-clock time per episode by 67.5% over CRLQAS [19]. **(Right)** Results are averaged over multiple seeds (mean $\pm$ std. dev.). While fixed ReaPER attains the lowest mean energy error and fewest steps, ReaPER+ achieves a competitive energy error while using substantially fewer gates and CNOTs than all other baselines, resolving the single-seed anomaly reported in the original submission.



Method	Error ↓	Gates ↓	CNOT ↓	Steps ↓
ReaPER	$(\mathbf{2.00 \pm 0.26}) \times \mathbf{10^{-2}}$	221 ± 57	125 ± 29	$(\mathbf{2.5 \pm 2.3}) \times \mathbf{10^4}$
ReaPER+	$(2.13 \pm 0.21) \times 10^{-2}$	$\mathbf{128 \pm 29}$	$\mathbf{75 \pm 35}$	$(3.7 \pm 2.0) \times 10^4$
PER	$(2.43 \pm 0.12) \times 10^{-2}$	165 ± 45	95 ± 28	$(5.7 \pm 3.2) \times 10^4$
Vanilla	$(2.45 \pm 0.20) \times 10^{-2}$	151 ± 35	91 ± 41	$(3.2 \pm 5.2) \times 10^4$

With OptCRLQAS enabling 12-qubit training, ReaPER attains the lowest mean energy error $(2.00 \pm 0.26) \times 10^{-2}$ Ha in the fewest steps ($(2.5 \pm 2.3) \times 10^4$), averaged over multiple seeds. ReaPER+ reaches a competitive energy error ($(2.13 \pm 0.21) \times 10^{-2}$ Ha) while using substantially fewer gates (128 ± 29 vs.221 ± 57) and CNOTs (75 ± 35 vs. 125 ± 29) than fixed ReaPER, whereas PER reaches a similar error floor at roughly $2.3\times$ the step cost and vanilla replay converges to the worst solution among the four methods. We further validate that the $T_{ann} = T_{total}/2$ scaling rule remains reliable at 12-qubit scale in a targeted ablation in Appendix N.3, which confirms that linking T_{ann} to the training horizon minimizes total gate count without sacrificing energy accuracy. Appendix J further shows OptCRLQAS can reduce quantum simulation time by up to 89% and classical optimization time by up to 85% in matched comparisons.

Scalability analysis. Beyond the 12-qubit ceiling of curriculum RL, combining ReaPER+ and amortized evaluation with TensorRL-QAS [24] extends scaling to 20 qubits: with a smaller amortization step (m=2 at 15/17 qubits, m=4 at 20), wall-clock time per episode falls from 120.7s at 12 qubits to 11.6s at 15 qubits and only 100.4s at 20, while the agent adds just 20–40 compact edits atop the fixed MPS circuit. We report this as a structural proof-of-concept rather than a chemically accurate result, since errors here remain an order of magnitude above our chemical-accuracy threshold; full details in Appendix K.

4.3 Noise-aware learning through buffer transfer

We evaluate the transfer scheme of Section 3, where a buffer $\mathcal{B}_{\text{src}}$ collected in a noiseless source environment initializes training in a related noisy target. This is, to our knowledge, the first demonstration that a lightweight, weight-free replay buffer alone transfers noiseless experience to realistic depolarizing noise settings, with advantages growing with system size up to 12 qubits. All experiments use OptCRLQAS with uniform replay to isolate transfer's effect. We study (i) *noiseless-to-noiseless* and (ii) *noiseless-to-noisy* transfer (depolarizing noise, strengths p_1/p_2) on molecular ground state benchmarks of increasing scale. The source buffer comes from a vanilla agent trained for a fixed 12 GPU-hour budget; to exploit the warm start, we reduce initial exploration from ϵ=1.0 to 0.55 and tighten the curriculum. We quantify transfer via the weighted score

$$S = w_1\,\Delta\text{steps} + w_2\,\Delta\text{ROT} + w_3\,\Delta\texttt{CNOT} + w_4\,\Delta\text{err}. \tag{5}$$

Weighted scores are commonly used in benchmark studies [59]. In Equation 5 weights $0.4, 0.1, 0.2, 0.3$, measures relative improvement over the no-transfer baseline.

Across all systems and noise settings, buffer transfer consistently reduces steps to chemical accuracy and improves final energy error. At the smallest scale (Figure 5), step reductions of 47-58% and energy gains up to 90.2% yield composite scores of 19.2-35.8%. At 8 qubits (Figure 6), step reductions reach 84.8% under combined noise. Crucially, at 12 qubits (Figure 7), *the advantage*

Figure 5: **Weighted transfer matrix for BEH_2 under noiseless and noisy transfer.** Buffer transfer reduces steps to chemical accuracy by 47-58% and improves final energy by up to 90.2%, yielding composite scores of 19.2-35.8%; the strongest score (35.8%) comes from the largest energy improvement at p_2=0.001.

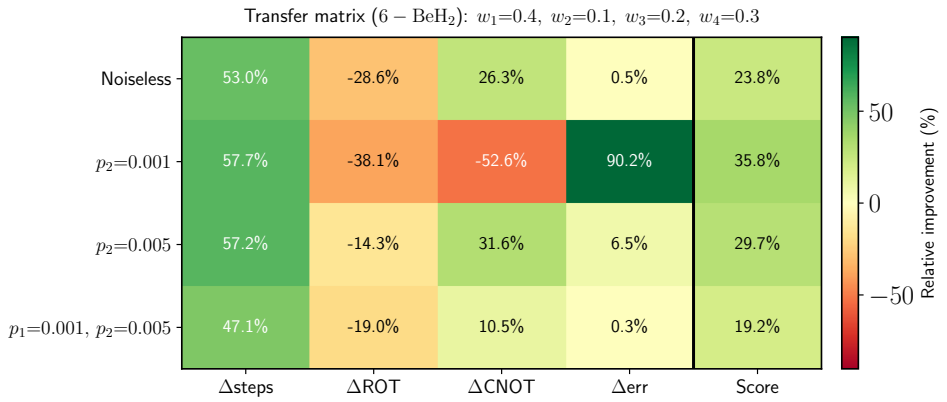


Figure 6: **Weighted transfer matrix for H_2O under noiseless and noisy transfer.** Step reductions range from 49.8% to 84.8%, with energy improvements up to 46.7% under combined noise (p_1=0.001, p_2=0.005), yielding the highest score of 28.7%.

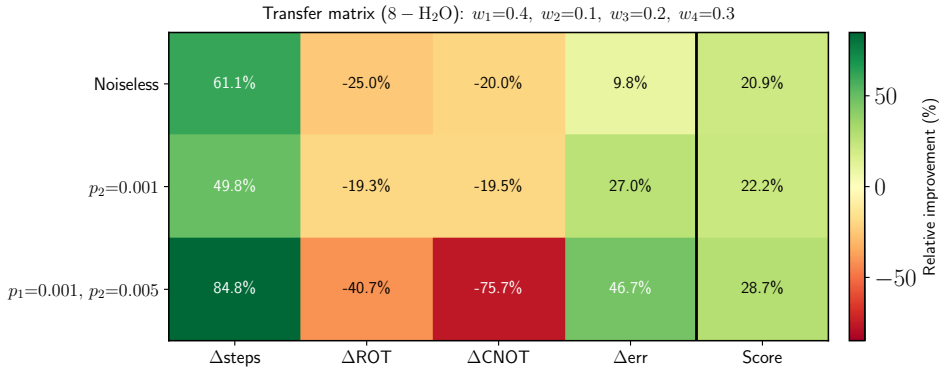
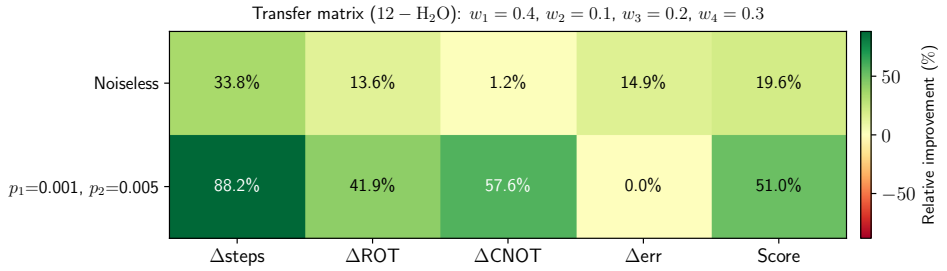


Figure 7: **Weighted transfer matrix for 12-qubit H_2O under noiseless and noisy transfer.** Transfer reduces steps to match non-transfer accuracy by 88.2% and CNOT count by 57.6%, under combined depolarizing noise (p_1=0.001, p_2=0.005).



strengthens with system size: transfer reduces steps by 88.2% (without error improvement), achieving the highest composite score of 51.0% across all benchmarks. Lightweight buffer transfer thus becomes *more* beneficial as the system grows, with *no architectural changes, no reward relabeling, and no network weight sharing*.

Transfer under amplitude damping noise. To test generalization beyond depolarizing noise, we evaluate transfer under amplitude damping (p=0.001) on 6-qubit BEH_2, comparing a from-scratch baseline against transfer at ϵ=1.00 and ϵ=0.55. Transfer reduces error from 8.45×10^{-5} Ha to 5.77×

Table 3: **Transfer under amplitude damping noise (6-qubit BEH_2, $p = 0.001$).** Buffer transfer improves both energy error and CNOT count relative to the from-scratch baseline, with the largest gains at reduced initial exploration ($\epsilon = 0.55$).

Setting	ϵ	Error (Ha)	CNOT	ROT
No transfer	1.00	8.45×10^{-5}	29	16
Transfer	1.00	5.81×10^{-5}	**26**	15
Transfer	0.55	$\mathbf{5.77 \times 10^{-5}}$	27	**10**

10^{-5} Ha at ϵ=0.55 (31.7% improvement) and CNOT count by 11.3%, confirming the mechanism is effective under non-unital noise and not an artifact of the depolarizing model used elsewhere.

Score-weight sensitivity. Equation 5 uses a single fixed weighting throughout. Recomputing S under six alternative weightings at 6- and 12-qubit scale shows in Appendix L that the transfer benefit becomes markedly less weight-sensitive as system size grows; at 12 qubits, transfer consistently improves speed and CNOT count but not final energy accuracy, a distinction we make explicit rather than implying a uniform accuracy benefit.

Buffer initialization and exploration interact. In Appendix M a 2×2 ablation study varying buffer initialization (empty vs. transferred, for three noise strengths) and ϵ on 6- and 8-qubit tasks shows the preferred ϵ flips depending on transfer source; under the strongest 6-qubit noise (p_2=0.005), ϵ=1.0 gives an order-of-magnitude error improvement over ϵ=0.55. Buffer transfer's benefit is thus conditional on its interaction with exploration rate, not uniformly additive.

Why the transfer works. Noiseless and noisy tasks share identical state/action spaces, so their optimal value functions differ only by a bounded perturbation. Classical transfer analyses [60] show high-value source trajectories remain informative under modest dynamics shifts; Bellman-alignment theory [61] formalizes this via one-step Bellman mismatch, motivating noiseless-collected transitions as approximately consistent Bellman targets for the noisy task.

4.4 ReaPER+ generalizes beyond quantum domain

To confirm the annealing mechanism is not quantum-specific, we benchmark all three replay strategies on **LunarLander-v3** [62] with identical DQN agents and schedule ($\omega_{\min}$=0.1 $\rightarrow$ $\omega_{\max}$=0.7, T_{ann}=5×10^4 steps). Appendix F we show that over 6 seeds, ReaPER+ achieves the highest mean success rate (0.638±0.037 vs. 0.524±0.102 for ReaPER and 0.503±0.056 for PER) and a 3% AUC gain over both baselines; steps to first solve are comparable given high seed variance, confirming the PER→ReaPER annealing principle generalizes across reward regimes.

5 Conclusion

We introduced a *previously unexplored* replay buffer engineering framework for quantum circuit optimization, comprising three complementary components. ReaPER+ improves sample efficiency and circuit compactness through annealed PER→ReaPER prioritization, achieving up to 4× greater sample efficiency than the strongest off-policy baselines while matching on-policy solution quality with up to 32× fewer interactions. OptCRLQAS reduces wall-clock time per episode by up to 67.5% through the amortization of expensive quantum-classical evaluations and, when combined with ReaPER+ and tensor network warm-starts, extends curriculum-based architecture search to 20 qubits as a structural proof-of-concept. Lightweight buffer transfer reduces the steps to chemical accuracy by up to 88% and the final energy error by up to 90% relative to from-scratch noisy baselines, without network-weight sharing or reward relabeling. These gains strengthen with system size and generalize beyond depolarizing noise to amplitude damping channels.

Across quantum compiling and QAS benchmarks, our results show that *how experience is stored, sampled, and transferred* is a primary algorithmic lever for sample-efficient, noise-aware quantum circuit optimization. A classical validation on LunarLander-v3 further confirms that ReaPER+'s annealing principle is domain-agnostic: with identical agents and schedules, it improves asymptotic success rate by up to 26.8% over PER and 21.8% over fixed ReaPER, while providing a 3% AUC gain over both baselines (see Appendix F). Limitations (including future directions) and broader impact of our work are discussed in Appendix A and Appendix B respectively.

Acknowledgment SF and AK gratefully acknowledge financial support by QDNL via the National Growth Fund KAT-1 program. We also acknowledge the use of computational resources of the DelftBlue supercomputer, provided by Delft High Performance Computing Centre (`https://www.tudelft.nl/dhpc`). The authors are very grateful to the anonymous reviewers and AC of NeurIPS conference for engaging in discussion and helping us improving the article.

References

[1] Amira Abbas, Andris Ambainis, Brandon Augustino, Andreas Bärtschi, Harry Buhrman, Carleton Coffrin, Giorgio Cortiana, Vedran Dunjko, Daniel J Egger, Bruce G Elmegreen, et al. Challenges and opportunities in quantum optimization. *Nature Reviews Physics*, 6(12):718–735, 2024.

[2] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014.

[3] Michel H Devoret and Robert J Schoelkopf. Superconducting circuits for quantum information: an outlook. *Science*, 339(6124):1169–1174, 2013.

[4] Xanthe Croot, Kasra Nowrouzi, Christopher Spitzer, Carmen G Almudever, Alexandre Blais, Malcolm Carroll, Jerry Chow, Daniel Friedman, Masao Tokunari, Edoardo Charbon, et al. Enabling technologies for scalable superconducting quantum computing. *arXiv preprint arXiv:2512.15001*, 2025.

[5] Morten Kjaergaard, Mollie E Schwartz, Jochen Braumüller, Philip Krantz, Joel I-J Wang, Simon Gustavsson, and William D Oliver. Superconducting qubits: Current state of play. *Annual Review of Condensed Matter Physics*, 11(1):369–395, 2020.

[6] Abhinav Kandala, Antonio Mezzacapo, Kristan Temme, Maika Takita, Markus Brink, Jerry M Chow, and Jay M Gambetta. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. *nature*, 549(7671):242–246, 2017.

[7] Youngseok Kim, Andrew Eddins, Sajant Anand, Ken Xuan Wei, Ewout Van Den Berg, Sami Rosenblatt, Hasan Nayfeh, Yantao Wu, Michael Zaletel, Kristan Temme, et al. Evidence for the utility of quantum computing before fault tolerance. *Nature*, 618(7965):500–505, 2023.

[8] John Preskill. Quantum computing in the NISQ era and beyond. *Quantum*, 2:79, 2018.

[9] Kishor Bharti, Alba Cervera-Lierta, Thi Ha Kyaw, Tobias Haug, Sumner Alperin-Lea, Abhinav Anand, Matthias Degroote, Hermanni Heimonen, Jakob S Kottmann, Tim Menke, et al. Noisy intermediate-scale quantum algorithms. *Reviews of Modern Physics*, 94(1):015004, 2022.

[10] Austin G Fowler, Matteo Mariantoni, John M Martinis, and Andrew N Cleland. Surface codes: Towards practical large-scale quantum computation. *Physical Review A—Atomic, Molecular, and Optical Physics*, 86(3):032324, 2012.

[11] Tobias V Forster, Nils Quetschlich, and Robert Wille. Quantum circuit optimization for the fault-tolerance era: Do we have to start from scratch? In *2025 IEEE International Conference on Quantum Computing and Engineering (QCE)*, volume 1, pages 584–590. IEEE, 2025.

[12] Zoltán Zimborás, Bálint Koczor, Zoë Holmes, Elsi-Mari Borrelli, András Gilyén, Hsin-Yuan Huang, Zhenyu Cai, Antonio Acín, Leandro Aolita, Leonardo Banchi, et al. Myths around quantum computation before full fault tolerance: What no-go theorems rule out and what they don't. *arXiv preprint arXiv:2501.05694*, 2025.

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

[14] Mateusz Ostaszewski, Lea M Trenkwalder, Wojciech Masarczyk, Eleanor Scerri, and Vedran Dunjko. Reinforcement learning for optimization of variational quantum circuit architectures. *Advances in neural information processing systems*, 34:18182–18194, 2021.

[15] Marin Bukov and Florian Marquardt. Reinforcement learning for quantum technology. *arXiv preprint arXiv:2601.18953*, 2026.

[16] Lorenzo Moro, Matteo GA Paris, Marcello Restelli, and Enrico Prati. Quantum compiling by deep reinforcement learning. *Communications Physics*, 4(1):178, 2021.

[17] ZT Wang, Qiuhao Chen, Yuxuan Du, ZH Yang, Xiaoxia Cai, Kaixuan Huang, Jingning Zhang, Kai Xu, Jun Du, Yinan Li, et al. Quantum compiling with reinforcement learning on a superconducting processor. *arXiv preprint arXiv:2406.12195*, 2024.

[18] Akash Kundu. Reinforcement learning-assisted quantum architecture search for variational quantum algorithms. *arXiv preprint arXiv:2402.13754*, 2024.

[19] Yash J Patel, Akash Kundu, Mateusz Ostaszewski, Xavier Bonet-Monroig, Vedran Dunjko, and Onur Danaci. Curriculum reinforcement learning for quantum architecture search under hardware errors. In *International Conference on Learning Representations*, volume 2024, pages 32298–32329, 2024.

[20] Ryan Sweke, Markus S Kesselring, Evert PL van Nieuwenburg, and Jens Eisert. Reinforcement learning decoders for fault-tolerant quantum computation. *Machine Learning: Science and Technology*, 2(2):025005, 2021.

[21] Kevin Reuer, Jonas Landgraf, Thomas Fösel, James O'Sullivan, Liberto Beltrán, Abdulkadir Akin, Graham J Norris, Ants Remm, Michael Kerschbaum, Jean-Claude Besse, et al. Realizing a deep reinforcement learning agent for real-time quantum feedback. *Nature Communications*, 14(1):7138, 2023.

[22] Tyler LeBlond, Christopher Dean, George Watkins, and Ryan Bennink. Realistic cost to execute practical quantum circuits using direct clifford+ t lattice surgery compilation. *ACM Transactions on Quantum Computing*, 5(4):1–28, 2024.

[23] Marco Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, et al. Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644, 2021.

[24] Akash Kundu and Stefano Mangini. TensorRL-QAS: Reinforcement learning with tensor networks for improved quantum architecture search. *Advances in Neural Information Processing Systems*, 38:120896–120930, 2026.

[25] Remmy Zen, Jan Olle, Luis Colmenarez, Matteo Puviani, Markus Müller, and Florian Marquardt. Quantum circuit discovery for fault-tolerant logical state preparation with reinforcement learning. *Physical Review X*, 15(4):041012, 2025.

[26] Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, et al. Deep q-learning from demonstrations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[27] Zhiyuan Zhou, Andy Peng, Qiyang Li, Sergey Levine, and Aviral Kumar. Efficient online reinforcement learning fine-tuning need not retain offline data. In *The Thirteenth International Conference on Learning Representations*, 2025.

[28] Seunghyun Lee, Younggyo Seo, Kimin Lee, Pieter Abbeel, and Jinwoo Shin. Offline-to-online reinforcement learning via balanced replay and pessimistic q-ensemble. In *5th Annual Conference on Robot Learning*, 2021.

[29] Siyuan Guo, Lixin Zou, Hechang Chen, Bohao Qu, Haotian Chi, Philip S. Yu, and Yi Chang. Sample efficient offline-to-online reinforcement learning. *IEEE Transactions on Knowledge and Data Engineering*, 36(3):1299–1310, 2024.

[30] Chihyeon Song, Jaewoo Lee, and Jinkyoo Park. Adaptive replay buffer for offline-to-online reinforcement learning. *arXiv preprint arXiv:2512.10510*, 2025.

[31] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[32] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

[33] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

[34] Leonard S Pleiss, Tobias Sutter, and Maximilian Schiffer. Reliability-adjusted prioritized experience replay. In *International Conference on Learning Representations*, volume 2026, pages 59206–59232, 2026.

[35] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

[36] Harper R Grimsley, Sophia E Economou, Edwin Barnes, and Nicholas J Mayhall. An adaptive variational algorithm for exact molecular simulations on a quantum computer. *Nature communications*, 10(1):3007, 2019.

[37] Shi-Xin Zhang, Chang-Yu Hsieh, Shengyu Zhang, and Hong Yao. Differentiable quantum architecture search. *Quantum Science & Technology*, 7(4):045023, 2022.

[38] Wenjie Wu, Ge Yan, Xudong Lu, Kaisen Pan, and Junchi Yan. QuantumDARTS: differentiable quantum architecture search for variational quantum algorithms. In *International conference on machine learning*, pages 37745–37764. PMLR, 2023.

[39] Riccardo Rasconi and Angelo Oddi. An innovative genetic algorithm for the quantum circuit compilation problem. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 7707–7714, 2019.

[40] Leo Sünkel, Darya Martyniuk, Denny Mattern, Johannes Jung, and Adrian Paschke. Ga4qco: genetic algorithm for quantum circuit optimization. *arXiv preprint arXiv:2302.01303*, 2023.

[41] Kim Nicoli, Christopher J Anders, Lena Funcke, Tobias Hartung, Karl Jansen, Stefan Kühn, Klaus-Robert Müller, Paolo Stornati, Pan Kessel, and Shinichi Nakajima. Physics-informed bayesian optimization of variational quantum circuits. *Advances in Neural Information Processing Systems*, 36:18341–18376, 2023.

[42] Peiyong Wang, Muhammad Usman, Udaya Parampalli, Lloyd CL Hollenberg, and Casey R Myers. Automated quantum circuit design with nested monte carlo tree search. *IEEE Transactions on Quantum Engineering*, 4:1–20, 2023.

[43] Shi-Xin Zhang, Chang-Yu Hsieh, Shengyu Zhang, and Hong Yao. Neural predictor based quantum architecture search. *Machine Learning: Science and Technology*, 2(4):045027, 2021.

[44] Yuxuan Du, Tao Huang, Shan You, Min-Hsiu Hsieh, and Dacheng Tao. Quantum circuit architecture search for variational quantum algorithms. *npj Quantum Information*, 8(1):62, 2022.

[45] Thomas Fösel, Murphy Yuezhen Niu, Florian Marquardt, and Li Li. Quantum circuit optimization with deep reinforcement learning. *arXiv preprint arXiv:2103.07585*, 2021.

[46] En-Jui Kuo, Yao-Lung L Fang, and Samuel Yen-Chi Chen. Quantum architecture search via deep reinforcement learning. *arXiv preprint arXiv:2104.07715*, 2021.

[47] Akash Kundu, Aritra Sarkar, and Abhishek Sadhu. KANQAS: Kolmogorov-arnold network for quantum architecture search. *EPJ Quantum Technology*, 11(1):76, 2024.

[48] David Kremer, Victor Villar, Hanhee Paik, Ivan Duran, Ismael Faro, and Juan Cruz-Benito. Practical and efficient quantum circuit synthesis and transpiling with reinforcement learning. *arXiv preprint arXiv:2405.13196*, 2024.

[49] Akash Kundu and Leopoldo Sarra. Reinforcement learning with learned gadgets to tackle hard quantum problems on real hardware. *Communications Physics*, 9(1):44, 2026.

[50] Heng You, Tianpei Yang, Yan Zheng, Jianye Hao, E Taylor, et al. Cross-domain adaptive transfer reinforcement learning based on state-action correspondence. In *Uncertainty in Artificial Intelligence*, pages 2299–2309. PMLR, 2022.

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

[52] Michael JD Powell. A direct search optimization method that models the objective and constraint functions by linear interpolation. In *Advances in optimization and numerical analysis*, pages 51–67. Springer, 1994.

[53] Ali Javadi-Abhari, Matthew Treinish, Kevin Krsulich, Christopher J Wood, Jake Lishman, Julien Gacon, Simon Martiel, Paul D Nation, Lev S Bishop, Andrew W Cross, et al. Quantum computing with qiskit. *arXiv preprint arXiv:2405.08810*, 2024.

[54] Yasunari Suzuki, Yoshiaki Kawase, Yuya Masumura, Yuria Hiraga, Masahiro Nakadai, Jiabao Chen, Ken M Nakanishi, Kosuke Mitarai, Ryosuke Imai, Shiro Tamiya, et al. Qulacs: a fast and versatile quantum circuit simulator for research purpose. *Quantum*, 5:559, 2021.

[55] Aram W Harrow, Benjamin Recht, and Isaac L Chuang. Efficient discrete approximations of quantum gates. *Journal of Mathematical Physics*, 43(9):4445–4451, 2002.

[56] Zhimin He, Jiachun Wei, Chuangtao Chen, Zhiming Huang, Haozhen Situ, and Lvzhou Li. Gradient-based optimization for quantum architecture search. *Neural Networks*, 179:106508, 2024.

[57] Zhimin He, Maijie Deng, Shenggen Zheng, Lvzhou Li, and Haozhen Situ. Training-free quantum architecture search. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 12430–12438, 2024.

[58] Xudong Lu, Kaisen Pan, Ge Yan, Jiaming Shan, Wenjie Wu, and Junchi Yan. Qas-bench: rethinking quantum architecture search and a benchmark. In *International conference on machine learning*, pages 22880–22898. PMLR, 2023.

[59] Azhar Ikhtiarudin, Aditi Das, Param Thakkar, and Akash Kundu. BenchRL-QAS: Benchmarking reinforcement learning algorithms for quantum architecture search. In *Proceedings of the AAAI Symposium Series*, volume 7, pages 358–367, 2025.

[60] Matthew E Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(7), 2009.

[61] Jinhang Chai, Enpei Zhang, Elynn Chen, and Yujun Yan. Optimistic transfer under task shift via bellman alignment. *arXiv preprint arXiv:2601.21924*, 2026.

[62] Mark Towers, Ariel Kwiatkowski, Jordan Terry, John U Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, et al. Gymnasium: A standard interface for reinforcement learning environments. *arXiv preprint arXiv:2407.17032*, 2024.

[63] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. Pmlr, 2018.

[64] Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of artificial intelligence research*, 47:253–279, 2013.

[65] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[66] Renhao Wang, Kevin Frans, Pieter Abbeel, Sergey Levine, and Alexei A Efros. Prioritized generative replay. In *The Thirteenth International Conference on Learning Representations*, 2025.

[67] Francisco JR Ruiz, Tuomas Laakkonen, Johannes Bausch, Matej Balog, Mohammadamin Barekatain, Francisco JH Heras, Alexander Novikov, Nathan Fitzpatrick, Bernardino Romera-Paredes, John Van De Wetering, et al. Quantum circuit optimization with alphatensor. *Nature Machine Intelligence*, 7(3):374–385, 2025.

[68] Abhijit Gosavi. Boundedness of iterates in q-learning. *Systems & control letters*, 55(4):347–349, 2006.

[69] Jianqing Fan, Zhaoran Wang, Yuchen Xie, and Zhuoran Yang. A theoretical analysis of deep q-learning. In *Learning for dynamics and control*, pages 486–489. PMLR, 2020.

[70] John N Tsitsiklis. Asynchronous stochastic approximation and q-learning. *Machine learning*, 16(3):185–202, 1994.

[71] Qiming Sun, Timothy C Berkelbach, Nick S Blunt, George H Booth, Sheng Guo, Zhendong Li, Junzi Liu, James D McClain, Elvira R Sayfutyarova, Sandeep Sharma, et al. Pyscf: the python-based simulations of chemistry framework. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 8(1):e1340, 2018.

[72] Pascual Jordan and Eugene Wigner. Über das paulische äquivalenzverbot. *Zeitschrift für Physik*, 47(9):631–651, 1928.

[73] Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. *Advances in neural information processing systems*, 29, 2016.

A Limitations and future work

Limitations. (1) All quantum experiments use a fixed DQN/DDQN backbone; whether ReaPER+'s annealing advantage persists under off-policy actor-critic agents [63] remains an open question; pure on-policy methods fall outside this scope as they do not employ replay buffers. (2) The replay buffer transfer scheme assumes shared state and action spaces between source and target tasks, which may not hold when porting across substantially different hardware topologies or gate sets. (3) The classical ML benchmarks are restricted to `LunarLander-v3`, a relatively low-dimensional discrete-action environment. Evaluating ReaPER+ on high-dimensional benchmarks such as the Atari 2600 suite [64] (e.g., *Pong*, *Breakout*, *Seaquest*) or continuous control domains such as MuJoCo [65] remains an important open direction. Finally, (4) all experiments are conducted on classical simulators; execution on real quantum hardware, where device-specific noise, qubit connectivity constraints, and gate calibration drift introduce challenges beyond those captured by depolarizing noise models, remains an important direction for future work, depending on hardware access.

Future work. Immediate extensions include: (1) combining OptCRLQAS with tensor network warm-starts [24] to further reduce search cost at 20-qubit scale; (2) hardware-aware replay, where buffered circuits are ranked using limited QPU evaluations before transfer to realistic noise models; and (3) prioritized generative replay [66] to synthesize high-value transitions for richer buffer initialization under depolarizing noise.

B Broader Impact

This work advances quantum circuit optimization by introducing a previously unexplored replay buffer engineering framework for deep reinforcement learning (RL). By improving the sample efficiency, computational efficiency, and noise awareness of RL agents for circuit synthesis and quantum architecture search, the proposed methods may lower the practical cost of optimizing circuits for near-term quantum hardware. Beyond quantum circuit optimization, the central idea—that the design of experience storage, sampling, and transfer can substantially affect learning performance—may be relevant to a broader range of sequential decision-making and combinatorial optimization problems.

Positive impacts By reducing circuit depth and gate count under realistic noise conditions, our approach can help make meaningful quantum computations more feasible on noisy intermediate-scale quantum devices. This may support progress in near-term applications such as quantum chemistry, materials modeling, combinatorial optimization, and quantum simulation, where reducing resource requirements is essential for obtaining useful results before fault-tolerant hardware becomes widely available.

The replay buffer engineering principles developed here may also extend beyond quantum computing. In particular, annealed replay prioritization, computational reuse through replay, and lightweight experience transfer could be applicable to RL problems in which environment interactions or reward evaluations are expensive. Potential examples include robotic control with costly real-world data collection, scientific experiment design, molecular and materials discovery, autonomous laboratory systems, scheduling and resource allocation, network control, and energy-system optimization. In such settings, improving the usefulness of previously collected trajectories may reduce the number of expensive interactions, simulations, or experiments needed to learn effective policies.

The framework may further be relevant to transfer and continual-learning settings, where an agent must adapt to related tasks, changed operating conditions, or noisy deployment environments. For example, buffer transfer could potentially support adaptation across related physical systems, device calibrations, manufacturing conditions, or optimization instances without requiring full retraining or direct sharing of model parameters.

Negative impacts Training RL agents with large prioritized replay buffers on high-fidelity quantum simulators can demand substantial GPU, memory, and storage resources. This may widen the gap between well-funded and under-resourced research groups, particularly when experiments

require extensive hyperparameter sweeps, repeated random seeds, or large-scale simulation. The energy consumption associated with prolonged GPU training and simulator-based evaluations also raises environmental concerns.

Moreover, methods that improve optimization efficiency can be applied broadly, including in domains where automated decision-making may have uneven social, economic, or environmental consequences. For example, applications in scheduling, resource allocation, or autonomous optimization should be evaluated carefully for fairness, safety, robustness, and the consequences of errors under distribution shift. In quantum computing specifically, improved circuit optimization could contribute to both beneficial scientific applications and, in the longer term, capabilities with dual-use implications. These concerns are not unique to the present work, but they motivate transparent reporting of computational cost, reproducible evaluation protocols, and domain-specific safeguards when deploying RL-based optimization systems.

We encourage future work to develop lightweight replay buffer variants, report energy and hardware costs, investigate methods that remain effective under constrained computational budgets, and adopt green-computing practices and infrastructure. We also encourage evaluation of replay buffer engineering in real-world high-cost decision-making domains, with appropriate safety, fairness, and human-oversight considerations.

C Preliminaries

Here we discuss the preliminaries based on which the methods (in Section 3) are based on. This discussion includes a thorough description of the state-of-the-art replay buffer. It also includes the encoding methods through which we represent the action space, reward functions and the RL-state.

C.1 Replay buffers in deep reinforcement learning

Experience replay is a central component of off-policy deep RL because it determines which past transitions are revisited during optimization. Uniform replay [31], hindsight experience replay [35], prioritized experience replay [33], and more recent reliability-aware variants [34] provide increasingly structured ways to trade off coverage, informativeness, and target reliability. In quantum optimization, however, replay buffers have largely been treated as fixed implementation choices rather than as a primary object of algorithmic design.

To formalize the replay mechanisms studied in this work, we consider an episodic reinforcement-learning setting. An episode is a sequence $D = \{C_t\}_{t=1}^{n}$ of transitions, where each transition $C_t = (S_t, A_t, R_t, S_{t+1}, d_t)$ records the state $S_t \in \mathcal{S}$, action $A_t \in \mathcal{A}$, scalar reward R_t, next state S_{t+1}, and terminal flag $d_t \in \{0, 1\}$, which equals 1 if the episode ends at step t and 0 otherwise. The agent maintains an online Q-network Q_θ with parameters θ and a periodically updated target network $Q_{\bar{\theta}}$ with parameters $\bar{\theta}$, and $\gamma \in (0, 1]$ denotes the discount factor. For each stored transition, we define four quantities that replay strategies use to assign sampling priorities. The TD target Y_t is the bootstrapped return estimate used to update Q_θ:

$$Y_t = R_t + \gamma(1 - d_t) \max_{a'} Q_{\bar{\theta}}(S_{t+1}, a'). \tag{6}$$

The TD error δ_t measures how far the current value estimate deviates from this target:

$$\delta_t = Y_t - Q_\theta(S_t, A_t), \tag{7}$$

and its magnitude $\delta_t^+ = |\delta_t|$ serves as a proxy for how much a transition still has to teach the agent. The true value error e_t measures the deviation of the current estimate from the optimal Q-function $Q^\star$:

$$e_t = Q_\theta(S_t, A_t) - Q^\star(S_t, A_t). \tag{8}$$

Finally, the target bias ε_t captures the error introduced by the target network itself:

$$\varepsilon_t = Q_{\bar{\theta}}(S_t, A_t) - Q^\star(S_t, A_t). \tag{9}$$

Replay strategies differ in how they assign sampling preference to stored transitions. For each transition C_i in the replay buffer, let Ψ_i denote its unnormalized replay priority and $\mu_i = \Psi_i / \sum_{j=1}^{N} \Psi_j$ the corresponding sampling probability. Different replay rules are then specified by different choices of Ψ_i.

Uniform replay (Vanilla). A uniform replay buffer samples all stored transitions with equal probability. If the current buffer contains $\{C_i\}_{i=1}^{N}$, then

$$\Psi_i^{\mathrm{uni}} = 1, \qquad \mu_i^{\mathrm{uni}} = \frac{1}{N}. \tag{10}$$

Hindsight experience replay (HER). HER augments the replay buffer by relabeling goals from achieved future states within the same trajectory [35]. The key idea is that even a failed trajectory, one that did not reach the intended goal g, can be treated as a successful one with respect to a different goal g', typically the state actually reached. Concretely, given a goal-conditioned transition $(S_t, A_t, R_t, S_{t+1}, d_t, g)$, HER adds an additional relabeled copy to the buffer,

$$\big(S_t, A_t, R_t'(g'), S_{t+1}, d_t'(g'), g'\big), \tag{11}$$

where g' is the relabeled goal and $R_t'(g')$, $d_t'(g')$ are the reward and terminal flag recomputed with respect to g'. Unless stated otherwise, HER samples uniformly from the augmented buffer. HER is utilized in the quantum compilation task in Ref. [16].

Prioritized experience replay (PER). PER samples transitions according to their absolute TD error [33]:

$$\Psi_i^{\mathrm{PER}} = (\delta_i^+)^\alpha, \qquad \mu_i^{\mathrm{PER}} = \frac{\Psi_i^{\mathrm{PER}}}{\sum_{j=1}^{N} \Psi_j^{\mathrm{PER}}}, \tag{12}$$

where $\alpha > 0$ controls the strength of prioritization.

Reliability-adjusted prioritized experience replay (ReaPER). ReaPER was recently introduced in ref. [34]. It discounts transitions whose targets are unreliable due to large downstream TD errors. For an episode $D = \{C_t\}_{t=1}^{n}$, the reliability score is

$$\mathcal{R}_t = 1 - \frac{\sum_{i=t+1}^{n} \delta_i^+}{\sum_{i=1}^{n} \delta_i^+}, \tag{13}$$

and the corresponding priorities are

$$\Psi_t^{\mathrm{ReaPER}} = \mathcal{R}_t^\omega (\delta_t^+)^\alpha, \qquad \mu_t^{\mathrm{ReaPER}} = \frac{\Psi_t^{\mathrm{ReaPER}}}{\sum_{i=1}^{n} \Psi_i^{\mathrm{ReaPER}}}, \tag{14}$$

where $\omega \in [0, 1]$ controls the strength of reliability weighting ($\omega = 0$ recovers PER and $\omega = 1$ corresponds to fully reliability-adjusted replay). Thus, relative to PER, high-error and low-reliability transitions are down-weighted.

C.2 Reinforcement learning for quantum optimization

Reinforcement learning has been applied to both quantum compilation and parameterized quantum circuit design. Existing work includes RL-based gate-sequence synthesis [67, 45, 48], architecture search for molecular ground state preparation [14], and hardware-aware variational circuit construction [19], demonstrating that RL can discover nontrivial circuit structures across a range of quantum tasks. At the same time, prior methods typically operate under substantial sample and compute demands, especially when each environment step requires an expensive variational optimization or when the target setting includes hardware noise. To tackle this, we formulate both quantum compilation and quantum architecture search as Markov decision processes in which an agent sequentially constructs a quantum circuit. At time step t, the agent observes $S_t \in \mathcal{S}$, selects $A_t \in \mathcal{A}$, receives reward R_t, and transitions to S_{t+1}. Episodes terminate when a target threshold is reached or when a maximum circuit length is exceeded.

C.2.1 RL-state and action for quantum compilation

In quantum compilation, the agent aims to approximate a target unitary U_{tar} by appending gates from a finite base $\mathcal{G}$. Following Ref. [16], the system is initialized in the all-zero state $|0\rangle^{\otimes n}$, and the circuit unitary after t steps is $U_t = \prod_{j=1}^{t} A_j$, so that the prepared state at step t is $U_t |0\rangle^{\otimes n}$. The observation is based on the residual unitary

$$O_t = U_t^\dagger U_{\mathrm{tar}}, \tag{15}$$

whose real and imaginary entries are provided to the agent.

1-qubit compiling For 1-qubit compiling, the chosen gateset base is

$$\mathcal{G}_{\text{rot}}^{(1)} = \left\{ R_i(\theta_j) \;\middle|\; i \in \{x, y, z\},\ \theta_j \in \left\{ +\frac{\pi}{128}, -\frac{\pi}{128} \right\} \right\}, \tag{16}$$

where $R_i(\theta_j)$ denotes a single-qubit rotation by angle θ_j about axis i. The rotation angle is thus restricted to a binary set of small fixed increments $\pm\pi/128$, making each action a minimal discrete rotation rather than a continuous parameter choice.

In an alternate setting, we also consider the Harrow-Recht-Chuang (HRC) [55] efficient universal basis, $V_1 = \frac{1}{\sqrt{5}} \begin{pmatrix} 1 & 2i \\ 2i & 1 \end{pmatrix}$, $V_2 = \frac{1}{\sqrt{5}} \begin{pmatrix} 1 & 2 \\ -2 & 1 \end{pmatrix}$, $V_3 = \frac{1}{\sqrt{5}} \begin{pmatrix} 1+2i & 0 \\ 0 & 1-2i \end{pmatrix}$, with action space

$$\mathcal{A}_{\text{HRC}}^{(1)} = \{V_1, V_2, V_3\}. \tag{17}$$

2-qubit compiling. For the 2-qubit setting, we use the action space:

$$\mathcal{A}^{(2)} = \{R_z(\theta_j) \otimes \mathbb{I}, \mathbb{I} \otimes R_z(\theta_j), XX = XX\left(\pm\frac{\pi}{128}\right), YY = YY\left(\pm\frac{\pi}{128}\right)\}, \tag{18}$$

C.2.2 RL-state and action for quantum architecture search

For quantum architecture search (QAS), the agent searches over parameterized quantum circuits to minimize a task-dependent cost, typically the expectation value [23]. Following CRLQAS [19], the state is represented by a tensor-based binary encoding of the circuit:

$$\mathcal{S}_{\text{QAS}} = \left\{ (\mathbf{E}_t, c_t) \;\middle|\; \mathbf{E}_t \in \{0,1\}^{N_{\text{depth}} \times N_{\text{wires}} \times N_{\text{gate_type}}},\ c_t \in \mathbb{R} \right\}, \tag{19}$$

where $\mathbf{E}_t$ encodes the circuit structure and c_t summarizes its current performance by calculating the current cost function i.e. the expectation value of Hamiltonian. Using the gate set $\mathcal{G}_{\text{QAS}} = \{\texttt{RX}(\theta_x), \texttt{RY}(\theta_y), \texttt{RZ}(\theta_z), \texttt{CNOT}\}$, the action space consists of placing a gate at a particular circuit location. A compact description is

$$\mathcal{A}_{\text{QAS}} = \left\{ a = (\ell, i, g) \;\middle|\; \ell \in \{1, \ldots, N_{\text{layers}}\},\ i \in \{1, \ldots, N_{\text{wires}}\}^{k_g},\ g \in \mathcal{G}_{\text{QAS}} \right\}, \tag{20}$$

where ℓ is the layer index, i specifies the qubit(s) the gate acts on, and k_g is the number of qubits required by gate g which isequal to 1 for single-qubit gates $\{\texttt{RX}, \texttt{RY}, \texttt{RZ}\}$ and 2 for the two-qubit gate `CNOT`. Illegal actions are masked out during training.

C.2.3 Reward structure

In both tasks, an episode ends either when a target threshold is met or when the maximum number of steps is reached.

Quantum compilation. Following Ref. [16], we use two reward designs depending on the gate base. For bases small-rotation (i.e $\pm\pi/128$), we use the dense reward

$$r_t^{\text{comp}} = \begin{cases} (L - t) + 1, & \text{if } d(U_t, U_{\text{tar}}) < \varepsilon, \\ -d(U_t, U_{\text{tar}})/L, & \text{otherwise,} \end{cases} \tag{21}$$

where L is the maximum episode length and ε is the target accuracy threshold. For discrete bases such as HRC, we use the sparse reward

$$r_t^{\text{comp}} = \begin{cases} 0, & \text{if } d(U_t, U_{\text{tar}}) < \varepsilon, \\ -1/L, & \text{otherwise.} \end{cases} \tag{22}$$

Quantum architecture search. Here we use the reward used as in [14, 19]. Let C_t be the optimized variational cost at step t, ξ the target threshold, $C_{\min}$ the desired minimum cost, and T_s^e the maximum number of steps in episode e. Then

$$R_t^{\text{QAS}} = \begin{cases} 5, & \text{if } C_t < \xi, \\ -5, & \text{if } t \geq T_s^e \text{ and } C_t \geq \xi, \\ \max\left(\dfrac{C_{t-1} - C_t}{C_{t-1} - C_{\min}}, -1 \right), & \text{otherwise.} \end{cases} \tag{23}$$

This reward provides a positive terminal signal for success, a negative terminal signal for failure, and otherwise a normalized improvement-based intermediate reward.

D Sample complexity and justification for ReaPER+

Here we provide an argument on why the proposed ReaPER+ replay rule can be expected to outperform a fixed replay strategy over the full course of training. Recall that ReaPER+ defines the replay priority of transition t at training step τ as

$$\Psi_t^{(+,\tau)} = \mathcal{R}_t^{\omega_\tau}(\delta_t^+)^\alpha, \qquad \mu_t^{(+,\tau)} = \frac{\Psi_t^{(+,\tau)}}{\sum_i \Psi_i^{(+,\tau)}}, \tag{24}$$

where $\delta_t^+ = |\delta_t|$ is the absolute temporal-difference error, $\mathcal{R}_t \in [0,1]$ is the reliability score defined in Equation 13, $\alpha > 0$ is the prioritization exponent, and $\omega_\tau \in [0,1]$ is a non-decreasing annealing parameter. When $\omega_\tau = 0$, the scheme reduces to PER, whereas $\omega_\tau = 1$ recovers ReaPER.

Proposition 1. *Assume that: (i) larger reliability $\mathcal{R}_t$ corresponds to smaller bias in the TD target of transition t; (ii) early in training, reliability estimates are noisy and only weakly correlated with true target quality; and (iii) later in training, reliability estimates become more informative. Then a schedule ω_τ that starts near 0 and increases toward 1 induces a replay distribution that is better aligned with the needs of the learning process than either fixed PER [33] or ReaPER [34] used throughout training.*

Justification. Consider two transitions i and j. Their relative sampling probability under ReaPER+ is

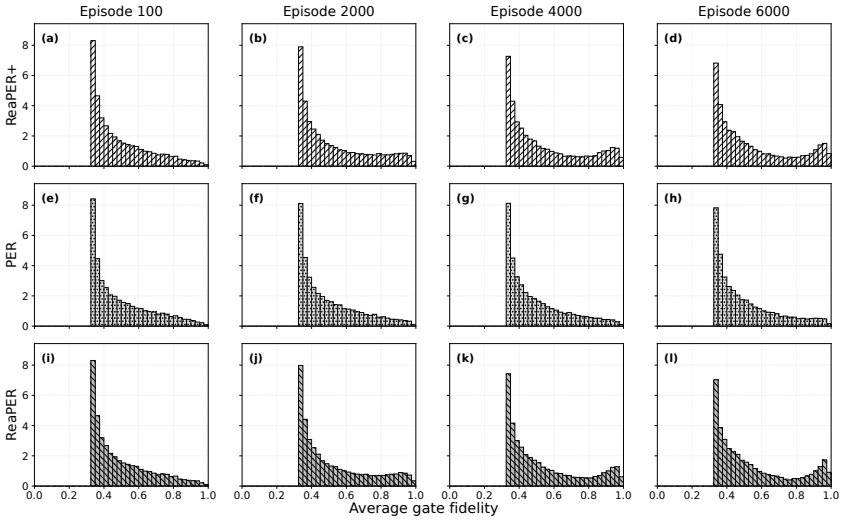
$$\frac{\mu_i^{(\omega)}}{\mu_j^{(\omega)}} = \frac{\mathcal{R}_i^\omega(\delta_i^+)^\alpha}{\mathcal{R}_j^\omega(\delta_j^+)^\alpha} = \left(\frac{\mathcal{R}_i}{\mathcal{R}_j}\right)^\omega \left(\frac{\delta_i^+}{\delta_j^+}\right)^\alpha. \tag{25}$$

If $\mathcal{R}_i > \mathcal{R}_j$, then for any $\omega_2 > \omega_1$ we have

$$\frac{\mu_i^{(\omega_2)}}{\mu_j^{(\omega_2)}} > \frac{\mu_i^{(\omega_1)}}{\mu_j^{(\omega_1)}}. \tag{26}$$

Hence increasing ω monotonically shifts replay mass toward transitions with higher reliability while preserving the PER-style dependence on TD error.

Figure 8: **ReaPER+ progressively concentrates buffer mass toward higher-fidelity transitions** (fidelity ≥ 0.95) while retaining broader early-training coverage, consistent with its annealed transition from PER-like exploration to ReaPER-like reliability-aware sampling. PER maintains broader low-fidelity coverage throughout training, while ReaPER shows intermediate concentration behavior.



This monotonicity explains the role of the annealing schedule. At the beginning of training, the value function is inaccurate and the resulting reliability estimates can be unstable. In this regime, setting $\omega_\tau \approx 0$ prevents the learner from overcommitting to a noisy reliability signal and recovers

the PER [33] behavior that emphasizes large TD error transitions. Later in training, once the critic becomes more accurate, the reliability score carries more information about whether a large TD error corresponds to a genuinely useful correction or to a misleading target. Increasing ω_τ then down-weights high-error but low-reliability transitions and shifts the replay distribution toward more trustworthy updates.

Equivalently, one may examine the log-priority

$$\log \Psi_t^{(+,\tau)} = \omega_\tau \log \mathcal{R}_t + \alpha \log \delta_t^+. \tag{27}$$

The annealing coefficient ω_τ controls the contribution of $\log R_t$ relative to $\log \delta_t^+$. Early in training, the replay rule is dominated by the TD error term; later, the reliability term gradually regularizes prioritization by suppressing transitions whose downstream targets are less trustworthy. Therefore, ReaPER+ inherits the strong early-learning behavior of PER and the late-stage stability of ReaPER.

The argument above does not claim universal dominance for every environment or every annealing schedule. Rather, it shows that under mild assumptions on the quality of reliability estimates across training, ReaPER+ provides a natural continuation method between two useful replay regimes. As shown in Figure 8, during the 1-qubit compiling task, ReaPER+ progressively shifts the fidelity distribution toward higher-fidelity, more reliable replay samples significantly faster than both PER and ReaPER.

D.1 The sample complexity bound for ReaPER+

We establish a formal sample complexity guarantee for ReaPER+ under tabular Q-learning. The main result shows that annealing the replay distribution from PER-like to ReaPER-like sampling provably reduces the asymptotic noise floor of the Q-learning iterates compared to fixed PER.

Theorem D.1 (Conditional Variance Reduction)**.** *Consider tabular Q-learning on a fixed dataset $\mathcal{D} = \{C_i\}_{i=1}^N$ under the ReaPER$^+$ annealed sampling distribution with linear schedule $\omega_t = \min\big(\omega_{\min} + \frac{\omega_{\max}-\omega_{\min}}{T_{\text{ann}}} t,\, \omega_{\max}\big)$. Under Assumptions (1)-(4), Assumption 1, and the MLR condition of Lemma D.4, the expected supremum error after T updates satisfies*

$$\begin{aligned}\mathbb{E}[\|Q_T - Q^\star\|_\infty] \le & \underbrace{(1-\alpha(1-\gamma))^T \|Q_0 - Q^\star\|_\infty}_{\textit{initialization decay}} \\ & + \underbrace{\frac{\alpha}{1-\gamma}\Big(\frac{T_{\text{ann}}}{T}\,\sigma_{\text{PER}} + \frac{T-T_{\text{ann}}}{T}\,\sigma_{\text{ReaPER}(\omega_{\max})}\Big)}_{\textit{noise floor}} \\ & + \underbrace{O(\zeta(T_{\text{ann}})) + O(B)}_{\textit{irreducible error}}.\end{aligned} \tag{28}$$

where $B \le \frac{2\gamma R_{\max}}{1-\gamma}$ is the bootstrap bias constant and $\zeta(T_{\text{ann}})$ captures the reliability score estimation error during the annealing transition. When reliability scores $\mathcal{R}_i$ successfully down-weight high-variance transitions, $\sigma_{\text{ReaPER}(\omega_{\max})} \le \sigma_{\text{PER}}$, so 2nd term in Equation 28 for ReaPER+ is strictly smaller than that of fixed PER, yielding a lower asymptotic noise floor.

Scope of the result. This guarantee is conditional on Assumption 1 and the MLR property holding for the empirical joint distribution of $(\mathcal{R}_i, \delta_i)$; it should be read as an idealized intuition motivating the annealed schedule rather than an unconditional guarantee for arbitrary replay buffers. Verifying the MLR condition directly requires estimating a conditional likelihood ratio over the full joint distribution of $(\mathcal{R}_i, \delta_i)$, which is computationally prohibitive for buffers containing millions of transitions, and we do not attempt this verification here. Even the weaker, directly testable implication of Assumption 1, that $\mathbb{E}[\delta_i^2 \mid \mathcal{R}_i \ge r]$ is decreasing in r, would require dedicated logging and binning infrastructure not currently part of our training pipeline. We therefore present Theorem D.1 as a conditional theoretical result whose premises are plausible given the design of the reliability score in Equation 13, but whose empirical validation on logged buffers remains an open direction for future work.

The three terms in Equation (28) have the following interpretation. The 1st term decays geometrically to zero and is independent of the replay strategy. The 2nd term is the dominant term at convergence and is directly controlled by the sampling distribution that is where ReaPER+ improves over PER. Finally, the 3rd term collects irreducible errors from bootstrap bias and reliability estimation, both of which are bounded constants independent of T.

The remainder of this section provides a complete proof of Theorem D.1. We first state the setting and assumptions, then establish the bias bound in Lemma D.2, then the one-step contraction in Lemma D.3, followed by the variance bound in Lemma D.4, and finally the unrolled recursion with annealing in Lemma D.5.

The agent maintains an online Q-network Q_θ and a target network $Q_{\bar{\theta}}$, following the notation of Section C. The TD target and TD error for transition $C_i = (S_i, A_i, R_i, S_{i+1}, d_i)$ are

$$Y_i = R_i + \gamma(1 - d_i) \max_{a'} Q_{\bar{\theta}}(S_{i+1}, a'), \qquad \delta_i = Y_i - Q_\theta(S_i, A_i), \tag{29}$$

The reliability score $\mathcal{R}_i \in [r_{\min}, 1]$ is defined as in Equation (13), and the ReaPER+ sampling probability follows the annealed priority

$$\Psi_i^{(t)} = \mathcal{R}_i^{\omega_t} (\delta_i^+)^\alpha, \qquad \mu_i^{(t)} = \frac{\Psi_i^{(t)}}{\sum_{j=1}^N \Psi_j^{(t)}}, \tag{30}$$

recovering PER at $\omega_t = 0$ and full ReaPER at $\omega_t = \omega_{\max}$. The importance-sampling corrected update for the Q-value is

$$Q_{t+1}(S_i, A_i) = Q_t(S_i, A_i) + \alpha\, w_i^{(t)} \big(Y_i - Q_t(S_i, A_i)\big), \tag{31}$$

where $w_i^{(t)} = \frac{1}{N \mu_i^{(t)}}$ is the importance-sampling weight.

Analysis-implementation weight mismatch. The analysis above uses fully bias-corrected importance-sampling weights $w_i^{(t)} = 1/(N\mu_i^{(t)})$. Our deep RL implementation instead follows standard PER practice [33], using β-annealed weights $w_i^{(t)} = (N\mu_i^{(t)})^{-\beta_t}$ with β_t annealed from $\beta_0 < 1$ toward 1 over training, since applying full correction from initialization is known to destabilize early optimization when $\mu_i^{(t)}$ estimates are still unreliable. Theorem D.1 therefore provides qualitative support for the direction of the annealing schedule rather than an exact guarantee for the specific weighting scheme used in the deployed algorithm.

Throughout, we denote by $R_{\max} = \max_{s,a} |r(s,a)|/(1-\gamma)$ the supremum of the discounted return, by $\Delta_{\max} = \sup_{i,t} |\delta_i^{(t)}|$ the uniform upper bound on TD errors, and by $r_{\min} > 0$ the minimum reliability score in the buffer. To realize the Theorem D.1 we require the following conditions. (1) *Boundedness:* the Q-function satisfies $|Q_t(s,a)| \le R_{\max}$ at all times, which is the natural bound implied by discounted cumulative rewards [68]; (2) *Bounded TD error:* the temporal-difference errors satisfy $|\delta_i| \le \Delta_{\max}$, ensuring the variance analysis in Lemma D.4 is finite; (3) *Step size:* the learning rate satisfies $\alpha \in \big(0, \frac{1}{1-\gamma}\big)$, which guarantees the contraction factor $(1-\alpha(1-\gamma)) \in (0,1)$ in Lemma D.3; in practice we further restrict to $\alpha < \frac{1}{2\gamma}$ to keep the bias constant B finite [69]; (4) *Reliability bounded:* the reliability scores satisfy $\mathcal{R}_i \in [r_{\min}, 1]$, preventing any transition from receiving zero sampling probability and ensuring the importance-sampling weights $w_i^{(t)}$ remain well-defined and finite.

Assumption 1 (Reliability-Variance Alignment)**.** *For all thresholds $r \in [r_{\min}, 1]$,*

$$\mathbb{E}[\delta_i^2 \mid \mathcal{R}_i \ge r] \;\le\; \mathbb{E}[\delta_i^2 \mid \mathcal{R}_i < r]. \tag{32}$$

That is, transitions with higher reliability scores have, on average, no larger squared TD error than transitions with lower reliability scores. This is the minimal condition under which reliability-weighted sampling can be expected to reduce variance rather than merely redistribute it.

Lemma D.2 (Bias bound, in expectation)**.** *Under Assumption (1), the bias introduced by the fixed target network $Q_{\bar{\theta}}$ satisfies*

$$B \;\coloneqq\; \mathbb{E}_{i\sim P}\Big[\big|w_i^{(t)}\big(Y_i - Q_t(S_i,A_i)\big) - \big((TQ_t)(S_i,A_i) - Q_t(S_i,A_i)\big)\big|\Big] \;\le\; \frac{2\gamma R_{\max}}{1-\gamma}. \tag{33}$$

Proof. By definition of Y_i and the Bellman operator T, the reward terms cancel and

$$Y_i - (TQ_t)(S_i, A_i) = \gamma\Big(\max_{a'} Q_{\bar{\theta}}(S_{i+1}, a') - \mathbb{E}_{S_{i+1}}\big[\max_{a'} Q_t(S_{i+1}, a')\big]\Big),$$

so $|Y_i - (TQ_t)(S_i, A_i)| \le \gamma\|Q_{\bar{\theta}} - Q_t\|_\infty \le \gamma\Big(\|Q_{\bar{\theta}} - Q^\star\|_\infty + \|Q_t - Q^\star\|_\infty\Big) \le \frac{2\gamma R_{\max}}{1-\gamma}$ pointwise for every i, by the same argument as before. Taking expectation over $i \sim P$ preserves this bound by Jensen's inequality, giving $B \le \mathbb{E}_{i\sim P}\big[\gamma\|Q_{\bar{\theta}} - Q_t\|_\infty\big] \le \frac{2\gamma R_{\max}}{1-\gamma}$. This expectation-based statement is also the form actually used in Lemma D.3 below, which only ever invokes B inside an expectation over Q_{t+1}, so this correction additionally removes an internal inconsistency between Lemmas D.2 and D.3. □

Lemma D.3 (One-step contraction)**.** *Under Assumptions (1)–(3) and Lemma D.2,*

$$\|\mathbb{E}[Q_{t+1} \mid Q_t] - Q^\star\|_\infty \le (1 - \alpha(1-\gamma))\|Q_t - Q^\star\|_\infty + \alpha B. \tag{34}$$

Proof. By the importance-sampling update rule and linearity of expectation,

$$\mathbb{E}[Q_{t+1} \mid Q_t] = Q_t + \alpha\,\mathbb{E}\big[w_i^{(t)}(Y_i - Q_t) \mid Q_t\big]. \tag{35}$$

Writing $w_i^{(t)}(Y_i - Q_t) = (TQ_t - Q_t) + \big[w_i^{(t)}(Y_i - Q_t) - (TQ_t - Q_t)\big]$ and taking expectation,

$$\mathbb{E}[Q_{t+1} \mid Q_t] = Q_t + \alpha(TQ_t - Q_t) + \alpha\,\mathbb{E}\big[w_i^{(t)}(Y_i - Q_t) - (TQ_t - Q_t) \mid Q_t\big], \tag{36}$$

where the last term is bounded in absolute value by B by Lemma D.2. Subtracting $Q^\star$ and using the identity $Q_t + \alpha(TQ_t - Q_t) - Q^\star = (1-\alpha)(Q_t - Q^\star) + \alpha(TQ_t - Q^\star)$,

$$\mathbb{E}[Q_{t+1} \mid Q_t] - Q^\star = (1-\alpha)(Q_t - Q^\star) + \alpha(TQ_t - Q^\star) + \alpha\,\eta_t, \tag{37}$$

where $\|\eta_t\|_\infty \le B$. Taking $\|\cdot\|_\infty$, applying the triangle inequality, and using the γ-contraction $\|TQ_t - Q^\star\|_\infty \le \gamma\|Q_t - Q^\star\|_\infty$ [70] together with $TQ^\star = Q^\star$,

$$\|\mathbb{E}[Q_{t+1} \mid Q_t] - Q^\star\|_\infty \le \big((1-\alpha) + \alpha\gamma\big)\|Q_t - Q^\star\|_\infty + \alpha B = (1 - \alpha(1-\gamma))\|Q_t - Q^\star\|_\infty + \alpha B, \tag{38}$$

where $(1 - \alpha(1-\gamma)) \in (0, 1)$ by Assumption (3). □

Lemma D.4 (Variance bound, conditional)**.** *Under Assumptions (2) and (4), the variance of the importance-sampling weighted TD update satisfies*

$$\sigma_t^2 \;\coloneqq\; \mathbb{E}\big[(w_i^{(t)}\delta_i)^2\big] = \frac{1}{N^2}\sum_{i=1}^{N} \frac{\delta_i^2}{\mu_i^{(t)}}. \tag{39}$$

Suppose further that Assumption 1 holds and that the sampling distribution satisfies the Monotone Likelihood Ratio (MLR) property with respect to the reliability scores:

$$\frac{P(\mathcal{R}_i = r \mid \delta_i^2 \text{ is small})}{P(\mathcal{R}_i = r \mid \delta_i^2 \text{ is large})} \quad \text{is strictly increasing in } r. \tag{40}$$

Then σ_t^2 is non-increasing in ω_t, with maximum at $\omega_t = 0$ (PER) and minimum at $\omega_t = \omega_{\max}$ (full ReaPER). Without the MLR condition, increasing ω_t need not reduce σ_t^2, since $\sum_i 1/\mu_i^{(t)}$ is minimized by the uniform *distribution; reliability-weighting only reduces variance relative to PER when it concentrates mass specifically on the low-error transitions identified by Assumption 1.*

Proof. Since index i is drawn according to $\mu^{(t)}$, the law of total expectation and the substitution $w_i^{(t)} = \frac{1}{N\mu_i^{(t)}}$ give

$$\mathbb{E}\big[(w_i^{(t)}\delta_i)^2\big] = \sum_{i=1}^{N} \mu_i^{(t)} \left(\frac{\delta_i}{N\mu_i^{(t)}}\right)^2 = \frac{1}{N^2}\sum_{i=1}^{N} \frac{\delta_i^2}{\mu_i^{(t)}}. \tag{41}$$

The weights are well-defined by Assumption (4). For monotonicity, note that under $\Psi_i^{(t)} = \mathcal{R}_i^{\omega_t}(\delta_i^+)^\alpha$, increasing ω_t reweights $\mu^{(t)}$ toward transitions with higher $\mathcal{R}_i$. The MLR condition

guarantees this reweighting is *targeted*: it implies that, for any threshold, the conditional odds of high $\mathcal{R}_i$ given small δ_i^2 dominate the conditional odds given large δ_i^2, so raising ω_t provably shifts probability mass from high-δ_i^2 transitions to low-δ_i^2 transitions rather than redistributing mass arbitrarily. Combined with Assumption 1, this mass shift strictly decreases $\sum_i \delta_i^2/\mu_i^{(t)}$ in expectation, since probability is moved away from terms with large $\delta_i^2/\mu_i^{(t)}$ toward terms with small δ_i^2. Hence σ_t^2 is non-increasing in ω_t, with the extremes at $\omega_t = 0$ and $\omega_t = \omega_{\max}$ following by substitution. □

Lemma D.5 (Unrolled recursion with annealing)**.** *Under Lemmas D.3 and D.4, iterating the one-step recursion over T steps and applying the annealing partition yields*

$$\begin{aligned}\mathbb{E}[\|Q_T - Q^\star\|_\infty] \le (1-\alpha(1-\gamma))^T \|Q_0 - Q^\star\|_\infty \\ + \frac{\alpha}{1-\gamma}\Big(\frac{T_{\text{ann}}}{T}\,\sigma_{\text{PER}} + \frac{T-T_{\text{ann}}}{T}\,\sigma_{\text{ReaPER}(\omega_{\max})}\Big) \\ + O(\zeta(T_{\text{ann}})) + O(B).\end{aligned} \tag{42}$$

Proof. Combining Lemmas D.3 and D.4 via standard stochastic approximation arguments [70],

$$\mathbb{E}[\|Q_{t+1} - Q^\star\|_\infty] \le (1-\alpha(1-\gamma))\,\mathbb{E}[\|Q_t - Q^\star\|_\infty] + \alpha\sqrt{\sigma_t^2} + \alpha B. \tag{43}$$

Let $\rho \coloneqq 1-\alpha(1-\gamma) \in (0,1)$. Iterating this recursion over T steps,

$$\mathbb{E}[\|Q_T - Q^\star\|_\infty] \le \rho^T \|Q_0 - Q^\star\|_\infty + \alpha \sum_{t=0}^{T-1} \rho^{T-1-t}\big(\sqrt{\sigma_t^2} + B\big). \tag{44}$$

Bounding the geometric sum $\sum_{t=0}^{T-1} \rho^{T-1-t} \le \frac{1}{1-\rho} = \frac{1}{\alpha(1-\gamma)}$ and replacing the geometrically-weighted average by a uniform Cesàro average at the cost of a constant factor,

$$\mathbb{E}[\|Q_T - Q^\star\|_\infty] \le \rho^T \|Q_0 - Q^\star\|_\infty + \frac{\alpha}{1-\gamma}\cdot\frac{1}{T}\sum_{t=1}^{T}\sqrt{\sigma_t^2} + O(B). \tag{45}$$

The linear schedule partitions training into a PER-like phase ($t \le T_{\text{ann}}$, variance σ_{PER}) and a ReaPER-like phase ($t > T_{\text{ann}}$, variance $\sigma_{\text{ReaPER}(\omega_{\max})}$) by Lemma D.4. The time-averaged noise therefore satisfies

$$\frac{1}{T}\sum_{t=1}^{T}\sqrt{\sigma_t^2} \le \frac{T_{\text{ann}}}{T}\,\sigma_{\text{PER}} + \frac{T-T_{\text{ann}}}{T}\,\sigma_{\text{ReaPER}(\omega_{\max})} + O(\zeta(T_{\text{ann}})), \tag{46}$$

where $\zeta(T_{\text{ann}})$ accounts for the estimation error of $\mathcal{R}_i$ during the transition region. Substituting into the unrolled bound yields the statement. □

Theorem D.1 follows immediately by combining Lemmas D.2–D.5. The annealing schedule, controlled by $T_{\text{ann}} = T/2$ in all experiments (Section 3), ensures PER-like exploration robustness early in training and variance reduction once the reliability estimates $\mathcal{R}_i$ have stabilized, directly explaining the empirical gains reported in Tables 1, 2 and Figure 2.

D.2 ReaPER+ implementation details

Algorithm 1 documents the exact replay buffer and training-step logic used in our implementation, as a companion to the idealized formulation in Section 3 and Appendix D. Three implementation choices deviate from the idealized formulation and are noted inline in the algorithm: (i) new transitions are seeded with a reward-based priority proxy $\max(|r_t|, 0.1)$ rather than a true TD error, with exact reliability computed only once the episode completes; (ii) priority updates between episode boundaries approximate the downstream TD sum as $\Sigma_e - |\delta_i|$ rather than recomputing the exact suffix sum of Equation 12; and (iii) the importance-sampling weight is applied to both the prediction and the target before computing the SmoothL1 loss, which effectively strengthens the weighting relative to the standard convention of weighting the loss term alone.

Algorithm 1: ReaPER+ replay buffer and training step implementation

Input: SumTree buffer $\mathcal{T}$ (capacity N); exponent α; fixed/annealed ω; IS-exponent β (start β_0, horizon F_β); $\epsilon = 10^{-6}$; networks $Q_\theta, Q_{\bar{\theta}}$; sync interval K

Procedure `Push`$(s_t, a_t, r_t, s_{t+1}, d_t$, *episode* $e)$
- $\hat{\delta}_t \leftarrow \max(|r_t|, 0.1)$; `// reward-based warm-start, not a true TD error`
- Insert at max tree priority; append index to episode e's list;
- $\Sigma_e \leftarrow \Sigma_e + |\hat{\delta}_t|$;
- **if** $d_t = 1$ **then**
  - `ComputeEpisodePriorities`(e);
- **end**

end

Procedure `ComputeEpisodePriorities`(e)
- **if** $\Sigma_e \leq \epsilon$ **then**
  - $p_i \leftarrow (|\hat{\delta}_i| + \epsilon)^\alpha$ for all $i \in e$;
- **else**
  - **foreach** *position* k *in* e **do**
    - $d_k \leftarrow \sum_{j>k} |\hat{\delta}_j|$;
    - $\mathcal{R}_k \leftarrow \texttt{clip}(1 - d_k/\Sigma_e,\ 0,\ 1)$;
    - $p_k \leftarrow \mathcal{R}_k^\omega (|\hat{\delta}_k| + \epsilon)^\alpha$;
  - **end**
- **end**

end

Procedure `Sample`(B)
- $\beta \leftarrow \min(1,\ \beta_0 + \text{frame} \cdot (1 - \beta_0)/F_\beta)$;
- Stratify $[0, \mathcal{T}.\text{total}]$ into B segments; draw and retrieve one transition per segment;
- $\mu_i \leftarrow p_i / \mathcal{T}.\text{total}$;
- $w_i \leftarrow (N\mu_i)^{-\beta}$, normalized by $\max_j w_j$;
- **return** *batch, indices* $I = \{idx_i\}$, *weights* $W = \{w_i\}$;

end

Procedure `UpdatePriorities`$(I, \{\delta_i\})$
- **foreach** (idx_i, δ_i) **do**
  - $e \leftarrow$ episode of idx_i;
  - $\Sigma_e \leftarrow \Sigma_e + |\delta_i| - |\hat{\delta}_i^{\text{old}}|$;
  - Overwrite stored TD error at idx_i with δ_i;
  - **if** $\Sigma_e \leq \epsilon$ **then**
    - $\mathcal{R}_i \leftarrow 1$;
  - **else**
    - $d_i \leftarrow \max(0,\ \Sigma_e - |\delta_i|)$; `// approximation, not exact suffix-sum recompute`
    - $\mathcal{R}_i \leftarrow \texttt{clip}(1 - d_i/\Sigma_e,\ 0,\ 1)$;
  - **end**
  - $p_i \leftarrow \mathcal{R}_i^\omega (|\delta_i| + \epsilon)^\alpha$;
- **end**

end

Procedure `TrainStep`(B)
- **if** $|\mathcal{T}| < B$ **then**
  - **return** *None*;
- **end**
- **if** *step* $\bmod K = 0$ **then**
  - $\bar{\theta} \leftarrow \theta$;
- **end**
- $(s, a, r, s', d, I, W) \leftarrow$ `Sample`(B);
- $a^\star \leftarrow \arg\max_{a'} Q_\theta(s')$;
- $Y \leftarrow r + \gamma_{\text{layer}}(1 - d) Q_{\bar{\theta}}(s')[a^\star]$;
- $\delta \leftarrow |Y - Q_\theta(s)[a]|$;
- `UpdatePriorities`(I, δ);
- $\mathcal{L} \leftarrow \text{SmoothL1}(W \odot Q_\theta(s)[a],\ W \odot Y)$; `// weights scale both sides, see note`
- $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}$;
- Anneal ϵ_{greedy};

end

E Replay buffer transfer under source-target task similarity

We justify why replay buffer transfer from a noiseless source environment to a corresponding noisy target environment can accelerate learning in quantum optimization. Let $\mathcal{M}_{\mathrm{src}}$ and $\mathcal{M}_{\mathrm{tgt}}$ denote two Markov decision processes with the same state and action spaces, where the target task differs from the source task by noise in the transition dynamics, reward evaluations, or both. Let $\mathcal{B}_{\mathrm{src}}$ be a replay buffer collected in the source task, and let the target replay buffer be initialized as

$$\mathcal{B}_{\mathrm{tgt}}^{(0)} \leftarrow \mathcal{T}(\mathcal{B}_{\mathrm{src}}), \tag{47}$$

where $\mathcal{T}$ is a transfer operator that copies compatible transitions from source to target. The following proposition provides a structured justification for this scheme under source-target task similarity. It is not intended as a formal convergence theorem under general function approximation — rigorous sample-complexity bounds in that setting remain an open problem — but rather as a precise statement of the conditions under which transfer is expected to help, with a constructive argument for each mechanism.

Proposition 2. *Assume the source and target tasks share the same state and action spaces $\mathcal{S}$ and $\mathcal{A}$ (shared structure); that the target task is a bounded perturbation of the source task in the sense that there exist constants $\varepsilon_r, \varepsilon_p \geq 0$ such that for all $(s,a) \in \mathcal{S} \times \mathcal{A}$,*

$$|r^{\mathrm{tgt}}(s,a) - r^{\mathrm{src}}(s,a)| \leq \varepsilon_r, \tag{48}$$

$$\left\| \mathbb{E}_{P^{\mathrm{tgt}}}\left[\max_{a'} Q(s',a')\right] - \mathbb{E}_{P^{\mathrm{src}}}\left[\max_{a'} Q(s',a')\right] \right\| \leq \varepsilon_p, \tag{49}$$

(bounded task shift); that the Q-function is uniformly bounded, $\|Q\|_\infty \leq Q_{\max} < \infty$, which holds under bounded rewards $|r| \leq R_{\max}$ and $\gamma \in (0,1)$ since $Q_{\max} \leq R_{\max}/(1-\gamma)$ (bounded Q-function); and that high-value source trajectories remain informative, though not necessarily optimal, in the target task, with source transitions concentrated in high-reward regions of $\mathcal{S} \times \mathcal{A}$ overlapping with high-value regions of the target task (informative source trajectories). Then the Bellman targets of corresponding source and target transitions differ by a bounded amount,

$$|Y^{\mathrm{tgt}}(s,a) - Y^{\mathrm{src}}(s,a)| \leq \varepsilon_r + \gamma\varepsilon_p, \tag{50}$$

and initializing the target replay buffer with transferred source transitions improves early target learning relative to training from an empty buffer via three mechanisms: (a) *reduced Bellman target mismatch,* (b) *improved initial buffer coverage over promising regions of circuit space, and* (c) *self-correction as on-target experience progressively replaces stale source transitions.*

Justification. Let the source and target Bellman targets for a transition (s,a) be defined as in Equation 48-Equation 49. Under the *bounded task shift* assumption, the triangle inequality yields

$$|Y^{\mathrm{tgt}}(s,a) - Y^{\mathrm{src}}(s,a)| \leq \varepsilon_r + \gamma\varepsilon_p, \tag{51}$$

as suggested by classical simulation-lemma arguments and recent Bellman-alignment analyses for transfer RL [61]. Thus a transition informative for the source task remains approximately informative for the target task whenever the source-target shift is moderate. In our quantum optimization setting, both assumptions are naturally satisfied: the *informative source trajectories* assumption holds because noiseless and noisy environments share identical circuit-building structure, so gate sequences that reduce energy error under noiseless simulation identify the same architecturally productive regions of circuit space under depolarizing noise where noise acts as a bounded stochastic perturbation on the cost landscape rather than altering which gate combinations are structurally meaningful. Finally, the *bounded task shift* assumption is satisfied because depolarizing noise with 1-qubit strength p_1 and 2-qubit strength p_2 introduces a perturbation that is bounded yet grows with system size, which is precisely why the transfer advantage grows with qubit count (Figure 5-Figure 7): as the noiseless-to-noisy gap widens, the warm-start from source experience becomes proportionally more valuable relative to from-scratch exploration.

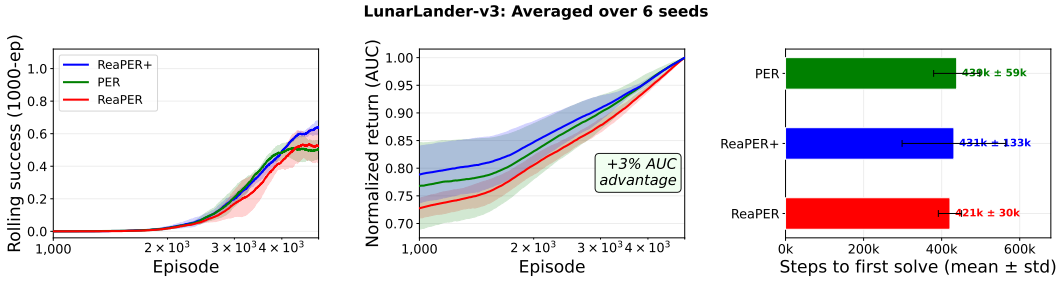
Beyond bounded target mismatch, transferred source transitions provide two further benefits. First, *improved coverage*: at the start of noisy training, an empty buffer contains no information about which gate sequences or architectural motifs are promising, whereas a transferred buffer is already concentrated in regions of state-action space that were useful in the source task. Since noiseless and noisy tasks share the same circuit-building structure, these transitions provide a more informative

initial replay distribution than random target experience alone, allowing the learner to update from semantically meaningful trajectories before independently rediscovering them under noise. Second, *self-correction*: as the agent interacts with the target environment, new target-domain transitions are added to the buffer and progressively shift the replay distribution toward the true noisy task. Transfer therefore does not lock the agent into the source domain; it supplies a warm start that is gradually refined by on-target experience. In prioritized replay schemes this correction is especially effective, since target transitions with larger TD errors or higher reliability automatically gain greater replay probability, accelerating adaptation.

F Classical RL validation: LunarLander-v3

The three contributions presented in the main text ReaPER+, OptCRLQAS, and replay buffer transfer, are motivated by and evaluated on quantum circuit optimization tasks. To verify that the annealing schedule underlying Reaper+ is not specific to the sparse, long-horizon reward structure of quantum environments, we provide a supplementary validation on `LunarLander-v3` [62], a well-established continuous-control benchmark from classical deep RL withdense rewards and a standard solved threshold of $+200$ cumulative return. We stress that this experiment is *not* intended as a primary result; its sole purpose is to confirm that the PER→ReaPER annealing idea generalizes beyond the quantum domain.

Figure 9: **LunarLander-v3 validation of ReaPER+, averaged over 6 seeds.** (*Left*) Rolling success rate (1000-episode window) for ReaPER+, PER, and fixed ReaPER. (*Middle*) Normalized cumulative-return AUC; ReaPER+ retains a $+3\%$ AUC advantage over both baselines. (*Right*) Steps to first solve (mean $\pm$ std): PER $439\text{k} \pm 59\text{k}$, ReaPER+ $431\text{k} \pm 133\text{k}$, ReaPER $421\text{k} \pm 30\text{k}$ – these differences are within one standard deviation given ReaPER+'s high seed-to-seed variance. All methods use identical DQN agents, only the replay mechanism differs. Fills/error bars denote 1-sigma.



We benchmark three replay strategies using identical DQN agents:

- **PER** [33]: prioritization by absolute TD error, α=0.6, β_0=0.4 annealed to 1.0 over 5×10^4 frames.
- **ReaPER** [34]: reliability-adjusted prioritization with fixed ω=0.4, α=0.4.
- **ReaPER+ (ours)**: annealed ω schedule $\omega_{\min}$=0.1 → $\omega_{\max}$=0.7 over $T_{\text{ann}}=5\times10^4$ steps (identical schedule to the quantum compiling experiments), α=0.4.

All agents are trained for 5,000 episodes on `LunarLander-v3` with γ=0.99, ε-greedy exploration (ε_0=1.0 → $\varepsilon_{\min}$=0.05, decay 0.9995 per episode), learning rate 10^{-3}, batch size 64, replay capacity 10^5, and target-network synchronization every 10 episodes. Four gradient updates are performed per episode after a 20-episode warm-up. The same agent architecture and training code used for the quantum experiments are employed without modification; only the environment is swapped. Results are averaged over 6 independent random seeds (extended from 3 seeds in the original submission), reported in Table 4.

Figure 9 summarizes three complementary metrics. *Left*: the 1000-episode rolling success rate (fraction of episodes exceeding the $+200$ solved threshold). ReaPER+ sustains the highest asymptotic success rate (0.638 ± 0.037) compared with fixed ReaPER (0.524 ± 0.102) and PER (0.503 ± 0.056). *Middle*: the normalized area under the reward curve (AUC), computed as the per-episode running mean of the cumulative return, shifted and normalized to $[0, 1]$. ReaPER+ retains a $+3\%$ AUC advantage over both baselines under the expanded seed count, down from the $+9\%$ reported under 3

seeds – indicating that part of the original margin was attributable to seed variance, though a smaller, more reliable advantage persists. *Right*: steps to first solve. Here PER and fixed ReaPER reach first solve in nominally fewer average steps (439k and 421k, respectively) than ReaPER+ (431k), reversing the direction reported under 3 seeds; however, ReaPER+'s standard deviation on this metric ($\pm 133\text{k}$) is 3–4$\times$ larger than either baseline's, so this ordering is not statistically meaningful. We report it transparently rather than omitting it, and treat ReaPER+'s consistently higher success rate as the more robust signal of the annealing schedule's benefit in this domain.

Table 4: **LunarLander-v3 validation over 6 seeds.** Mean $\pm$ standard deviation of success rate and steps to first solve.

Metric	PER	ReaPER	ReaPER+
Success rate	0.503 ± 0.056	0.524 ± 0.102	$\mathbf{0.638 \pm 0.037}$
Steps	438541.7 ± 59143.9	$\mathbf{421304.0 \pm 29782.3}$	431427.3 ± 132823.6

These results are consistent with the mechanism described in Section 3 and Appendix D: early in training, $\omega \approx 0$ recovers PER-style exploration; as value estimates stabilize, rising ω down-weights unreliable high-error transitions and concentrates replay on trustworthy updates. This two-phase behavior is beneficial regardless of whether rewards are sparse and episodic (quantum compilation) or dense and continuous (LunarLander), supporting the view that ReaPER+ is a general-purpose annealing strategy rather than a domain-specific heuristic; the expanded seed count refines the magnitude of this advantage without changing its direction.

Table 5: Hyperparameters for the LunarLander-v3 validation experiment. Shared parameters apply to all three replay strategies; method-specific parameters are listed in the lower block.

Parameter	**Value**
Shared (all methods)	
Network architecture	2×128 hidden units, SELU
Discount γ	0.99
Learning rate	10^{-3}
Batch size	64
Replay capacity	10^5
ε schedule	$1.0 \to 0.05$, decay 0.9995/episode
Gradient updates/ep.	4 (after 20-episode warm-up)
Training episodes	5000
Solved threshold	+200 cumulative return
PER	
α	0.6
β_0	0.4, annealed to 1.0 over 5×10^4 frames
ReaPER	
α	0.4
ω	0.4 (fixed)
β_0	0.4, annealed to 1.0 over 5×10^4 frames
ReaPER+	
α	0.4
$\omega_{\min}$	0.1
$\omega_{\max}$	0.7
T_{ann}	5×10^4 steps
β_0	0.4, annealed to 1.0 over 5×10^4 frames

G 1-qubit sample significance analysis

The circuit-length comparisons in Figure 2 are aggregated over a small number of independent training seeds ($n = 5$), which limits the precision with which we can estimate the true population mean gate count for each method. To quantify this uncertainty rigorously, rather than relying on point-estimate comparisons alone, we compute the standard error of the mean and construct approximate

95% confidence intervals using the t-distribution with $df = n - 1 = 4$. Table 6 reports the resulting mean gate count, standard deviation, and confidence interval for PER, fixed ReaPER, and ReaPER+ at each tolerance level. As expected given the small sample size, the intervals are wide and overlap substantially across methods, so we do not claim statistical significance in the classical hypothesis-

Table 6: **Sample significance analysis of circuit length.** Mean gate count $\pm$ standard deviation and approximate 95% confidence interval ($df = 4$) for the small-rotation basis compiling task.

Tol	Method	Gate count	95% CI (approx.)
0.99	PER	79.00 ± 48.32	$[19.0, 139.0]$
	ReaPER	90.48 ± 61.78	$[13.8, 167.2]$
	ReaPER+	$\mathbf{75.72 \pm 48.36}$	$[\mathbf{15.7}, \mathbf{135.8}]$
0.999	PER	85.93 ± 53.22	$[19.9, 152.0]$
	ReaPER	89.21 ± 61.98	$[12.3, 166.2]$
	ReaPER+	$\mathbf{70.88 \pm 50.74}$	$[\mathbf{7.9}, \mathbf{133.9}]$
0.9999	PER	128.51 ± 53.13	$[62.6, 194.5]$
	ReaPER	123.94 ± 62.83	$[45.9, 201.9]$
	ReaPER+	$\mathbf{74.09 \pm 41.13}$	$[\mathbf{23.0}, \mathbf{125.2}]$

testing sense. However, the analysis is informative in a different way: it shows that the sample mean for ReaPER+ is consistently lower than PER and fixed ReaPER at all three tolerances, and that this directional shift persists even under the most conservative (widest) interval estimate, which supports the qualitative trend reported in the main text and motivates evaluating ReaPER+ on a larger seed budget in future work.

H 1-qubit compiling with HRC gateset

For a direct comparison with Ref. [16], we evaluate the discrete HRC basis [55] using the action space and sparse reward defined in Equation 17 and Equation 22. Table 7 shows that ReaPER+ reaches 100% success with a mean fidelity of 0.995 and the shortest average circuit length (14.30 ± 7.89 gates), achieving this at 1.56×10^6 steps. This is slightly earlier than fixed ReaPER (1.60×10^6 steps), clearly earlier than PER (2.10×10^6 steps), and substantially earlier than our tuned HER baseline (5.50×10^6 steps), while also producing shorter circuits than all three. In contrast, the

Table 7: **1-qubit compiling of Haar-random targets using the HRC basis.** Success rate, mean fidelity, and circuit length for different replay strategies. All methods except the original HER baseline [16] reach 100% success. ReaPER+ yields the shortest circuits (14.30 ± 7.89 gates) and reaches peak performance $\sim$24%, $\sim$26%, and $\sim$72% faster than fixed ReaPER, PER, and tuned HER, respectively.

Method	**Success (%)** $\uparrow$	**Fidelity** $\uparrow$	**Mean length** $\downarrow$	**At step** $\downarrow$
ReaPER+ (Ours)	$\mathbf{100.0}$	$\mathbf{0.995}$	$\mathbf{14.30 \pm 7.89}$	$\mathbf{1.56 \times 10^6}$
ReaPER [34]	100.0	0.995	20.34 ± 16.16	1.60×10^6
PER [33]	100.0	0.995	19.54 ± 14.61	2.10×10^6
HER (tuned)	100.0	0.992	28.60 ± 21.58	5.50×10^6
HER (Moro et al. [16])	95.0	0.990	35	–

original unoptimized HER implementation from Moro et al. [16] plateaus at 95% accuracy. Note that throughout the remainder of this paper, any reference to the HER baseline indicates our tuned implementation rather than the original formulation of Ref. [16], ensuring all comparisons are made against the strongest possible version of the baseline.

I Target dataset for 2-qubit compiling task

For the compiling experiments with the 1-qubit small-rotation basis and the 2-qubit gate set, we generate target unitaries synthetically by sampling random circuits from the corresponding elementary gate library. Concretely, a target unitary is constructed by first sampling a circuit

length uniformly at random and then composing gates drawn uniformly from the same basis used by the agent during training. This yields a diverse dataset of reachable targets with varying circuit complexity while ensuring that the target distribution is consistent with the underlying compilation task. For the 2-qubit setting, Algorithm 2 details this procedure for the gate set $\{XX(\pm\pi/128),\ YY(\pm\pi/128),\ R_z(\pm\pi/128)\otimes\mathbb{I},\ \mathbb{I}\otimes R_z(\pm\pi/128)\}$; the 1-qubit small-rotation targets are generated analogously from their corresponding 1-qubit rotation basis.

Algorithm 2: 2-qubit target unitary generation.

Input: Gate set $\mathcal{B} = \left\{XX(\pm\frac{\pi}{128}),\ YY(\pm\frac{\pi}{128}),\ R_z(\pm\frac{\pi}{128})\otimes\mathbb{I},\ \mathbb{I}\otimes R_z(\pm\frac{\pi}{128})\right\}$
Output: Target unitary $U_{\text{tar}} \in \mathrm{SU}(4)$

```
Sample circuit length N ~ Uniform({6, 7, ..., 10^4 − 1});
Initialize U_tar ← I_4;
for k = 1 to N do
    Sample gate G_k ~ Uniform(B);
    U_tar ← G_k · U_tar;
end
return U_tar
```

J Benchmarking OptCRLQAS against CRLQAS [19]

Before we provide the comprehensive benchmark of OptCRLQAS against CRLQAS we discuss how the reward function and and RL-state calculated and populated in the intermediate steps in Section J.1 and then the benchmark is provided in Section J.2.

J.1 Reward and RL-state during intermediate steps

In standard CRLQAS, a full quantum-classical evaluation is performed at every environment step, so the variational cost C_t, the RL-state cost component c_t, and the reward R_t^{QAS} (Equation 23) are all updated synchronously. OptCRLQAS breaks this synchrony by restricting full evaluations to every m steps. We describe here how rewards and states are populated during the $m-1$ intermediate steps between evaluations.

Cost function evaluation Let τ_{prev} denote the most recent step at which $u_\tau = 1$, i.e., the last step at which a full quantum-classical evaluation was performed. During all intermediate steps $\tau \in (\tau_{\text{prev}},\ \tau_{\text{prev}} + m)$, the variational cost is held constant at its cached value:

$$C_\tau \leftarrow C_{\tau_{\text{prev}}}, \quad u_\tau = 0. \tag{52}$$

This cached value is used for both reward computation and state construction until the next evaluation is triggered.

State representation The QAS state $S_\tau = (E_\tau, c_\tau)$ (Equation 18) carries the current circuit structure E_τ, which is updated at every step as the agent appends gates, and the scalar cost summary $c_\tau = C_{\tau_{\text{prev}}}$, which remains frozen between evaluations. The agent therefore observes the evolving circuit structure but receives a delayed cost signal, analogous to a control system operating with periodic rather than continuous feedback.

Reward computation The intermediate reward follows Equation 22 with the cached cost substituted for the true cost:

$$R_\tau^{\text{QAS}} = \begin{cases} 5, & \text{if } C_{\tau_{\text{prev}}} < \xi, \\ -5, & \text{if } \tau \geq T_s^e \text{ and } C_{\tau_{\text{prev}}} \geq \xi, \\ \max\left(\dfrac{C_{\tau_{\text{prev}}-m} - C_{\tau_{\text{prev}}}}{C_{\tau_{\text{prev}}-m} - C_{\min}},\ -1\right), & \text{otherwise}, \end{cases} \tag{53}$$

where $C_{\tau_{\text{prev}}-m}$ is the cost from the evaluation immediately preceding the current block. During intermediate steps the improvement-based term evaluates to zero, since $C_{\tau_{\text{prev}}-m}$ and $C_{\tau_{\text{prev}}}$ are both

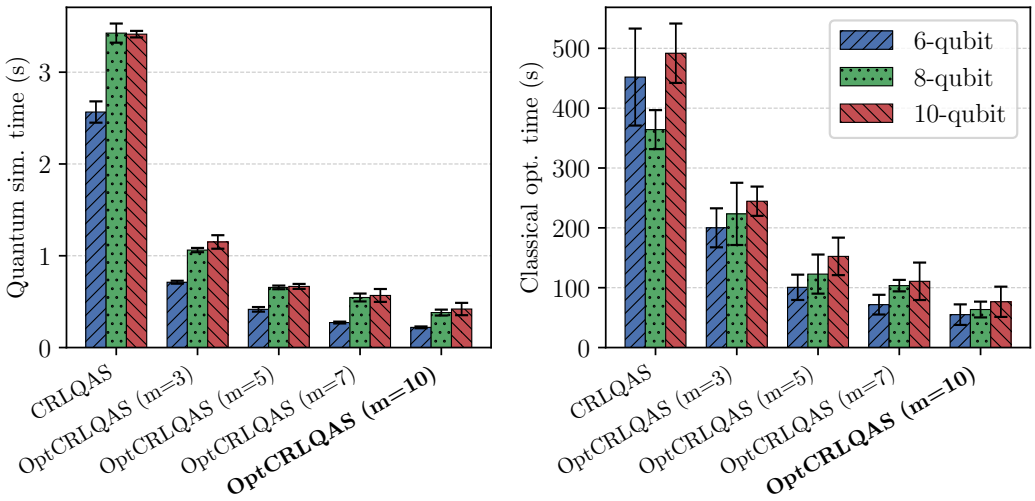
fixed within the block. The agent therefore receives a zero intermediate reward during accumulation steps, with the non-trivial learning signal arriving only at evaluation steps. This is intentional: rather than rewarding each individual gate placement which produces weakly separated TD targets when variational parameters θ can locally compensate single-gate changes — OptCRLQAS credits the outcome of an entire block of m edits jointly. As discussed in Section 4.2, this increases reward contrast and accelerates value propagation, steering the agent toward architecturally meaningful gate combinations rather than single-step local optima.

Terminal conditions Terminal detection ($d_\tau = 1$) is checked at every step regardless of the update schedule. If the episode terminates at an intermediate step (i.e., $u_\tau = 0$ but $\tau \geq T_s^e$), a final evaluation is immediately triggered (u_τ is overridden to 1) so that the terminal reward reflects the true cost of the final circuit rather than a stale cached value. This ensures that the terminal transition carries an accurate TD target and prevents the agent from exploiting the caching mechanism to avoid negative terminal penalties.

J.2 Benchmark

To empirically validate the computational efficiency gained by amortizing the architecture update overhead, we benchmarked OptCRLQAS against the baseline CRLQAS framework across 6-, 8-, and 10-qubit configurations. We tracked both the expectation value evaluation and the classical optimization time over multiple episodes. Figure 10 illustrates the substantial reduction in computational cost when utilizing OptCRLQAS across varying step intervals ($m \in \{3, 5, 7, 10\}$).

Figure 10: **Comparative runtime analysis of the baseline CRLQAS versus OptCRLQAS across 6-, 8-, and 10-qubit tasks**. The left panel shows the quantum energy evaluation time (in seconds), and the right panel displays the classical optimization time (in seconds). Results for OptCRLQAS are reported for different replay buffer sizes ($m \in \{3, 5, 7, 10\}$). Error bars denote the standard deviation measured over 3 to 5 independent episodes. The utilization of OptCRLQAS yields an average runtime reduction of up to 89% for quantum evaluations and 85% for classical optimizations at $m = 10$. Note that the error bars are 1-sigma.



Because CRLQAS triggers a full quantum-classical evaluation at every environment step, its runtime scales poorly as circuit complexity increases. By accumulating local architecture edits and restricting full evaluations to every m steps, OptCRLQAS closely matches the expected runtime reduction factor of $\approx 1/m$. Our empirical results demonstrate an aggressive decline in runtime as m increases. For quantum energy evaluations, OptCRLQAS achieves an average time reduction of 69.2% at $m = 3$, scaling up to an impressive 89.3% reduction at $m = 10$ across all qubit scales. Similarly, the classical optimization time is heavily mitigated, yielding an average decrease of 48.2% at $m = 3$ and up to 84.9% at $m = 10$. This benchmarking clearly establishes that accumulating structural modifications before parameter optimization successfully breaks the runtime bottleneck,

enabling RL-based quantum architecture search to scale to larger qubit regimes without prohibitive GPU compute requirements.

Table 8: **Comparison between vanilla CRLQAS and OptCRLQAS ($m = 10$) on 8-qubit H_2O over three random seeds**. For OptCRLQAS, the total gate count is computed as ROT + CNOT.

Method	Seed	Error (Ha)	Total gates	CNOT	ROT
CRLQAS [19]	1	1.167×10^{-3}	153	129	24
	42	1.171×10^{-3}	80	59	21
	786	1.167×10^{-3}	100	66	34
OptCRLQAS (Ours)	1	1.167×10^{-3}	81	45	36
	42	1.171×10^{-3}	97	23	74
	786	1.168×10^{-3}	87	31	56

To isolate the effect of amortized evaluation beyond wall-clock speedup, we additionally compare vanilla CRLQAS and OptCRLQAS on 8-H_2O under matched training conditions, varying only the update schedule. Table 8 shows that the two methods reach essentially identical final energies: the mean error is 1.168636×10^{-3} Ha for CRLQAS and 1.168738×10^{-3} Ha for OptCRLQAS, a difference of only $\sim 10^{-7}$ Ha. Despite this negligible difference in energy, OptCRLQAS produces substantially more hardware-efficient circuits, reducing the mean total gate count from 111.0 to 88.3 (20.4% fewer gates) and the mean CNOT count from 84.7 to 33.0 (61.0% fewer CNOTs), at the cost of more single-qubit rotations. This is consistent with the intuition behind OptCRLQAS: by evaluating the circuit only after multiple architectural edits, the agent receives a less myopic and more informative reward signal, which can preserve final accuracy while steering learning toward circuits with substantially lower 2-qubit cost.

J.3 Quality-vs-runtime sweep over the amortization interval m

To further characterize the amortization/quality tradeoff, we performed a quality-runtime sweep (fixed 1000-episode budget) over $m \in \{3, 5, 7, 10\}$ across the 6-, 8-, and 10-qubit tasks. OptCRLQAS trades optimization frequency for stability: standard CRLQAS updates the optimizer after every gate insertion, risking overfitting or divergence on partially formed circuits. By amortizing updates over m gates, OptCRLQAS (i) avoids penalizing the policy with unstable intermediate optimizations early in training, and (ii) learns to adjust composite gate blocks rather than isolated gates, reducing variance. Table 9 reports the results.

Table 9: **Quality-vs-runtime sweep over the OptCRLQAS amortization interval m** (fixed 1000-episode budget). OptCRLQAS matches or improves on CRLQAS's minimum error at comparable or lower gate counts at 6 and 8 qubits, and remains competitive at 10 qubits, while substantially reducing both quantum-simulation and classical-optimization time at every m and scale.

Qubits	Method	Min. error (Ha)	Gate count	Q-Sim time (s)	C-Opt time (s)
6	CRLQAS	5.3×10^{-4}	51	~ 2.6	~ 450
	OptCRLQAS (m=3)	5.4×10^{-4}	51	~ 0.7	~ 200
	OptCRLQAS (m=5)	$\mathbf{5.6 \times 10^{-5}}$	52	~ 0.4	~ 100
	OptCRLQAS (m=7)	5.9×10^{-5}	53	~ 0.3	~ 70
	OptCRLQAS (m=10)	5.8×10^{-5}	54	$\sim \mathbf{0.2}$	$\sim \mathbf{55}$
8	CRLQAS	1.4×10^{-4}	95	~ 3.4	~ 360
	OptCRLQAS (m=3)	1.5×10^{-4}	81	~ 1.1	~ 220
	OptCRLQAS (m=5)	$\mathbf{1.1 \times 10^{-4}}$	82	~ 0.6	~ 120
	OptCRLQAS (m=7)	1.8×10^{-4}	84	~ 0.5	~ 100
	OptCRLQAS (m=10)	1.2×10^{-4}	87	$\sim \mathbf{0.4}$	$\sim \mathbf{65}$
10	CRLQAS	$\mathbf{9.0 \times 10^{-4}}$	97	~ 3.4	~ 490
	OptCRLQAS (m=3)	1.2×10^{-3}	93	~ 1.2	~ 240
	OptCRLQAS (m=5)	1.5×10^{-3}	94	~ 0.7	~ 150
	OptCRLQAS (m=7)	9.2×10^{-4}	94	~ 0.6	~ 110
	OptCRLQAS (m=10)	1.0×10^{-3}	99	$\sim \mathbf{0.4}$	$\sim \mathbf{75}$

At 6 and 8 qubits, OptCRLQAS matches or improves minimum error at comparable or lower gate counts across all m. At 10 qubits, results are more mixed: CRLQAS achieves the lowest error of the group, with OptCRLQAS at $m = 7$ coming closest, indicating amortization remains competitive but not uniformly superior at this scale. In all cases, quantum-simulation and classical-optimization time drop substantially and monotonically as m increases, confirming that amortization does not come at a systematic quality cost even where it does not strictly improve on CRLQAS.

K Scaling to 20 qubits via tensor network warm-starts

This appendix details the experiment summarized in Section 4.2, in which we combine ReaPER+ and amortized evaluation with a matrix product state (MPS) warm-start to extend our empirical scaling study from the 12-qubit ceiling of OptCRLQAS to 15, 17, and 20 qubits.

Following the developments of TensorRL-QAS [24], we first approximate the target ground state with a fixed bond dimension MPS ($\chi = 2$), then convert this MPS into an initial quantum circuit using a sequential single- and two-qubit gate decomposition. This conversion circuit is fixed and *not* discovered by the RL agent; it provides the initial entanglement structure that the agent subsequently refines. Table 10 reports the resulting resource cost of this conversion at each system size, prior to any RL-driven modification.

Table 10: **Matrix product state → quantum circuit conversion cost.** Gate counts and depth of the fixed MPS warm-start circuit at each qubit count, using bond dimension $\chi = 2$, before any RL-agent edits.

Qubits	RX gates	RY gates	RZ gates	CNOTs	Base depth
15	14	85	156	42	27
17	16	97	178	48	27
20	19	115	211	57	27

The RL agent is then trained on top of this fixed warm-start circuit using ReaPER+ replay and m-step amortized evaluation ($m = 2$ at 15 and 17 qubits, $m = 4$ at 20 qubits), following the same amortization mechanism as OptCRLQAS (Section 3) but with a substantially smaller m, since the MPS warm-start already removes most of the burden of discovering long-range entangling structure from scratch. Table 11 reports the full empirical scaling, extending Table 2-style comparisons up to 20 qubits. The starred gate counts report only the *additional* gates placed by the RL agent; the total circuit gate count is the sum of the MPS base circuit (Table 10) and these additions – for example, the total circuit at 15 qubits contains 297 MPS-derived gates plus 20 RL-discovered gates, for 317 gates overall.

Table 11: **Empirical scaling of RL-agent performance from 6 to 20 qubits.** OptCRLQAS + ReaPER+ (random initialization, $m = 10$) is shown up to its practical 12-qubit ceiling; Amortized TensorRL-QAS + ReaPER+ (MPS warm-start) extends to 15–20 qubits with reduced m. Starred gate counts indicate only the additional structural edits made by the RL agent on top of the MPS warm-start circuit (Table 10). "*" indicated that the minimum number of gates are equal to the total number of steps in an episode.

Method	m-step	Qubits	Time/episode (s)	Min. error (ΔE)	Min. gates
OptCRLQAS + ReaPER+	10	6	55.0	5.8×10^{-5}	54
	10	8	65.0	1.2×10^{-4}	87
	10	10	75.0	1.0×10^{-3}	99
	10	12	120.7	2.2×10^{-2}	135
m-step TensorRL-QAS + ReaPER+	2	15	11.6	1.5×10^{-2}	20*
	2	17	63.0	1.8×10^{-2}	30*
	4	20	100.4	2.1×10^{-2}	40*

Interpretation. Two trends are notable. First, wall-clock time per episode at 15 qubits (11.6s) is over $10\times$ lower than at 12 qubits under OptCRLQAS with random initialization (120.7s), despite operating on a larger system. This reduction comes from both the smaller m enabled by the warm-start and the fact that the agent needs far fewer corrective edits per episode. Second, the RL-discovered gate additions grow only mildly with system size ($20 \rightarrow 30 \rightarrow 40$ across 15, 17,

and 20 qubits), indicating that the combination of MPS warm-start and reliability-aware replay continues to suppress the unconstrained gate-count growth we caution against elsewhere in this paper (Section 4.2).

Caveats. We report this experiment as a proof-of-concept demonstration of compatibility between our contributions and physics-inspired initialization, not as a chemically accurate result: minimum energy errors at this scale (1.5-2.1 $\times 10^{-2}$ Ha) remain roughly an order of magnitude above the chemical-accuracy threshold used in our 5–12 qubit experiments ($\varepsilon_{\text{chem}} = 1.6 \times 10^{-3}$ Ha). We also note that, unlike the multi-seed reporting used for our 5–12 qubit QAS results (Table 2, Figure 4), the 15-20 qubit results reported here reflect single-seed runs; extending this study to multiple seeds, and to convergence at full chemical accuracy, is left for future work (Appendix A).

L Transfer score weight sensitivity

Recalling the transfer score reported in Section 4.3 through Equation 5:

$$S = w_1\Delta_{\text{steps}} + w_2\Delta_{\text{ROT}} + w_3\Delta_{\text{CNOT}} + w_4\Delta_{\text{err}}, \tag{54}$$

aggregates four relative-improvement terms into a single scalar using fixed weights $(w_1, w_2, w_3, w_4) = (0.40, 0.10, 0.20, 0.30)$. Since this weighting is a design choice rather than a property of the transfer mechanism itself, a natural concern is whether the reported transfer benefit is an artifact of this particular weighting rather than a robust effect. We address this by recomputing S under six alternative weightings, chosen to emphasize each individual term in turn (steps, error, CNOT, ROT), a fully balanced weighting, and a joint gate-count-heavy weighting that combines CNOT and ROT emphasis. Table 12 reports the resulting scores at 6- and 12-qubit scale; we restrict this comparison to two system sizes for space, as the 8-qubit case follows the same qualitative pattern.

Table 12: **Transfer score sensitivity to weight choice.** Composite score S under alternative (w_1, w_2, w_3, w_4) weightings at 6- and 12-qubit scale. The original weighting used in Section 4.3 is shown first.

Setting	(w_1, w_2, w_3, w_4)	6-qubit score	12-qubit score
Original	0.40, 0.10, 0.20, 0.30	19.2%	12.3%
Steps-heavy	0.50, 0.10, 0.20, 0.20	23.8%	13.0%
Err-heavy	0.30, 0.10, 0.20, 0.40	14.5%	11.6%
CNOT-heavy	0.40, 0.10, 0.30, 0.20	20.2%	16.5%
ROT-heavy	0.40, 0.20, 0.20, 0.20	17.2%	14.1%
Balanced	0.25, 0.25, 0.25, 0.25	9.7%	16.1%
Gate-heavy	0.30, 0.20, 0.30, 0.20	13.6%	17.7%

At 6 qubits, the score ranges from 9.7% (balanced weighting) to 23.8% (steps-heavy weighting), a spread of 14.1 percentage points, indicating that the composite score is moderately sensitive to weight choice at this scale. At 12 qubits, the same seven weightings produce scores ranging from 11.6% (error-heavy) to 17.7% (gate-heavy), a substantially narrower spread of 6.1 percentage points. This tightening indicates that the transfer advantage becomes more uniformly distributed across the individual objectives (steps, ROT, CNOT, error) as system size increases, rather than being concentrated in whichever objective happens to receive the largest weight.

This pattern is consistent with, and helps explain, an important qualification of the 12-qubit result: unlike at 6- and 8-qubit scale, where transfer improves both convergence speed and final energy accuracy (main text Section 4.3), at 12 qubits transfer consistently improves steps-to-convergence and CNOT count but does *not* improve final energy accuracy relative to the from-scratch baseline. Because the error-heavy weighting ($w_4 = 0.4$) yields one of the *lowest* scores at 12 qubits (11.6%) while the gate- and CNOT-heavy weightings yield the *highest* scores (17.7% and 16.5%, respectively), the weight-sensitivity analysis directly corroborates this scale-dependent pattern: the 12-qubit transfer benefit is real but concentrated in efficiency (steps, gate count) rather than solution quality, and any single fixed weighting that under-emphasizes error would otherwise risk overstating the accuracy benefit at this scale. We therefore report this qualification explicitly in the main text rather than relying on the composite score alone to characterize the 12-qubit result.

M Buffer transfer × exploration ablation

To isolate the effect of buffer transfer from the choice of initial exploration rate ϵ, we run a 2×2 ablation on the 6- and 8-qubit tasks, varying buffer initialization (empty vs. transferred under several noise-strength conditions) and ϵ. Table 13 and Table 14 report the results.

Table 13: **6-qubit buffer transfer × exploration ablation.** Steps, gate counts, and final error under empty vs. transferred buffer initialization, across noise strengths and two exploration rates.

Buffer / transfer condition	ϵ	Steps	ROT	CNOT	Error
Empty buffer (noiseless)	0.55	52401	27	14	5.77×10^{-5}
Empty buffer (noiseless)	1.0	52401	21	17	5.27×10^{-5}
Transfer (p_2=0.001)	0.55	47161	29	29	5.67×10^{-5}
Transfer (p_2=0.001)	1.0	47161	11	32	5.77×10^{-5}
Transfer (p_2=0.005)	0.55	47758	24	13	5.42×10^{-5}
Transfer (p_2=0.005)	1.0	47758	16	24	$\mathbf{7.72 \times 10^{-6}}$
Transfer (p_1=0.001, p_2=0.005)	0.55	58952	25	17	5.78×10^{-5}
Transfer (p_1=0.001, p_2=0.005)	1.0	58952	23	27	5.86×10^{-5}

Table 14: **8-qubit buffer transfer × exploration ablation.**

Buffer / transfer condition	ϵ	Steps	ROT	CNOT	Error
Empty buffer (noiseless)	0.5	5223	45	54	0.00105
Empty buffer (noiseless)	1.0	4223	55	45	0.00125
Transfer (p_2=0.001)	0.5	31259	68	98	0.00144
Transfer (p_2=0.001)	1.0	29457	86	99	0.00134
Transfer (p_1=0.001, p_2=0.005)	0.5	12503	76	65	0.00121
Transfer (p_1=0.001, p_2=0.005)	1.0	13986	65	71	**0.00101**

The preferred ϵ flips depending on the transfer source rather than following a single consistent rule. In the 6-qubit empty-buffer baseline, ϵ=1.0 slightly improves error over ϵ=0.55. Under transfer with p_2=0.001, the reverse holds: ϵ=0.55 is preferable. Under the stronger noise setting p_2=0.005, however, ϵ=1.0 produces an order-of-magnitude error improvement (7.72×10^{-6} vs. 5.42×10^{-5} at ϵ=0.55). A similarly metric-dependent pattern appears at 8 qubits, where combined-noise transfer under ϵ=1.0 achieves the best overall error (1.01×10^{-3}) despite ϵ=1.0 increasing error in the empty-buffer baseline at the same scale. We conclude that while buffer transfer can yield state-of-the-art results, its benefit is conditional on its interaction with the exploration rate rather than a fixed, transferable hyperparameter choice – practitioners applying this transfer scheme should treat ϵ as jointly tunable with the transfer source rather than fixing it independently.

N ReaPER ω sensitivity and variant selection

In this appendix, we first provide a thorough sensitivity analysis of ReaPER+ with respect to the hyperparameter ω, and then discuss how to select the PER-to-ReaPER annealing schedule for ReaPER+ through a thorough hyperparameter sweep.

N.1 Fixed-ω ReaPER variant selection

In the main text, Figure 3 reports gate counts for a single fixed-ω ReaPER variant per molecular system: $\omega = 0.4$ for 6-qubit BEH_2 and $\omega = 0.6$ for 8-qubit H_2O. This appendix documents the full sweep over ω that motivates these choices. For each system we train agents with $\omega \in \{0.2, 0.4, 0.6\}$ (6-qubit) and $\omega \in \{0.2, 0.4, 0.6, 0.8\}$ (8-qubit), using three independent random seeds $\{1, 42, 786\}$ per setting. All other hyperparameters are identical to the main-text QAS experiments (Appendix P). We report mean and standard deviation over seeds for CNOT count and single-qubit rotation (ROT) count. Results are given in Table 15.

Since two-qubit gates dominate both circuit noise and compilation cost on near-term hardware, we select the ω value that minimizes mean CNOT count subject to the agent achieving chemical accuracy ($\Delta E \leq \varepsilon_{\text{chem}}$) across all seeds, using ROT count as a tiebreaker.

Table 15: **CNOT and ROT gate counts (mean ± 1-sigma over 3 seeds) for ReaPER ω variants**. All entries achieve chemical accuracy ($\Delta E \leq \varepsilon_{\text{chem}} = 1.6 \times 10^{-3}$ Ha). Bold rows indicate the variant selected for the main text. PER, ReaPER+, and Vanilla are included for reference.

	6-qubit BEH_2		**8-qubit H_2O**	
ReaPER	CNOT	ROT	CNOT	ROT
$\omega = 0.2$	30.7 ± 18.8	20.7 ± 9.9	84.3 ± 75.4	39.3 ± 32.0
$\omega = \mathbf{0.4}$	$\mathbf{23.7 \pm 9.0}$	$\mathbf{29.3 \pm 21.5}$	71.7 ± 46.6	38.3 ± 14.6
$\omega = \mathbf{0.6}$	30.7 ± 7.4	30.3 ± 2.1	$\mathbf{51.0 \pm 25.1}$	$\mathbf{24.3 \pm 20.6}$
$\omega = 0.8$	–	–	78.7 ± 46.2	35.0 ± 5.0

For 6-qubit BEH_2, $\omega = 0.4$ achieves the lowest mean CNOT count of 23.7 ± 9.0 gates among all ReaPER variants, compared with 30.7 ± 18.8 for $\omega = 0.2$ and 30.7 ± 7.4 for $\omega = 0.6$. For 8-qubit H_2O, $\omega = 0.6$ achieves the lowest mean CNOT count of 51.0 ± 25.1 gates–a 41% reduction relative to PER (86.3 ± 37.5) and 29% relative to $\omega = 0.4$ (71.7 ± 46.6).

Across both systems, increasing ω shifts the replay distribution toward reliability-aware sampling more aggressively, which tends to reduce CNOT count at the cost of slightly higher energy variance. The optimal ω is therefore task-dependent: shorter-horizon problems with lower qubit counts benefit from moderate reliability weighting ($\omega = 0.4$), while longer-horizon, larger systems favor stronger weighting ($\omega = 0.6$) to suppress noisy TD targets in a deeper search space. This task-dependence is precisely what motivates the annealed ReaPER+ construction, which avoids committing to a fixed ω by scheduling the transition adaptively during training (Section 3).

N.2 Joint sweep of ω_{min}, ω_{max}, and T_{ann} (1-qubit)

We conduct a joint sensitivity sweep over the three ReaPER+ annealing hyperparameters (ω_{min}, ω_{max}, T_{ann}) on the 1-qubit compiling task; due to GPU compute constraints, this full joint sweep was not extended to the 12-qubit setting (a targeted T_{ann}-only ablation at 12-qubit scale is reported instead in Section N.3). The results in Table 16 show that T_{ann} must be scaled with the training

Table 16: **Joint sensitivity sweep of the ReaPER+ annealing schedule on 1-qubit compiling.** Success rate, mean fidelity, and mean circuit length for alternative (ω_{min}, ω_{max}, T_{ann}) configurations. The chosen configuration (ω_{min}=0.2, ω_{max}=0.7, T_{ann}=20000) achieves the best success-fidelity-length tradeoff among the sweep.

ω_{min}	ω_{max}	T_{ann}	Success	Mean fidelity	Mean length
0.2	**0.7**	**20000**	**0.89**	**0.93**	**85.8**
0.1	0.7	10000	0.84	0.89	96.0
0.1	0.5	5000	0.84	0.87	95.1
0.3	0.9	40000	0.83	0.87	103.1

horizon: transitioning too early or too late actively harms performance. They further show that pushing $\omega_{max} \geq 0.9$ forces the agent to overcommit to reliability scores before the value function has fully converged, degrading both success rate and fidelity relative to the chosen configuration ($\omega_{min} = 0.1, \omega_{max} = 0.7$).

N.3 12-qubit annealing-schedule validation

The 12-qubit advantage of ReaPER+ is best framed around circuit compactness rather than raw speed, as reflected in the main text in Section 4.2. To validate that the $T_{ann} = T_{total}/2$ scaling rule remains sound at scale, we run the proposed ablation on the 12-qubit task, varying T_{ann} while holding ω_{min}, ω_{max} fixed. Table 17 shows that all schedules reach a comparable energy error, but the base schedule yields the fewest total gates, while the overcommitting schedule ($T_{ann} = 40000$) yields the fewest two-qubit (CNOT) gates. Uncalibrated schedules (aggressive or narrow) instead

Table 17: T_{ann} **scaling ablation on 12-qubit H_2O.** All profiles reach a comparable energy error; the base schedule ($T_{ann} = T_{total}/2$) minimizes total gate count, while the overcommitting profile minimizes CNOT count.

T_{ann} (Profile)	Error (Ha)	Gates	CNOT
10000 (Aggressive)	2.2×10^{-2}	121	64
5000 (Narrow)	2.2×10^{-2}	241	138
40000 (Overcommitting)	2.2×10^{-2}	76	**32**
$T_{total}/2$ (Base)	2.2×10^{-2}	**61**	34

produce deeper circuits overall, confirming that linking T_{ann} directly to the training horizon provides a reliable, interpretable timescale that maintains structural efficiency at scale.

O Hamiltonians

In the following we first provide the geometry of the chemical Hamiltonian and then a description og the Heisenberg model and its corresponding Hamiltonian.

O.1 Chemical Hamiltonian

In Table 18 we provide the detailed configuration of molecules utilized throughout the quantum architecture search. All molecular Hamiltonians are obtained via PySCF [71] with the STO-3G (6-qubit, 8-qubit) and 6-31G (10-, 12-qubit) basis sets, using the Jordan-Wigner fermion-to-qubit mapping [72]. No active space freezing is applied; all electrons and orbitals are included within the chosen basis.

Table 18: **The geometry and basis of molecules used in this research**. The coordinates are in Angstrom units.

Molecule	Geometry	Basis
6-BEH_2	H (0,0,-1.33); Be (0,0,0); H (0,0,1.33)	STO-3G
8-H_2O	H (-0.02,-0,0); O (0.84,0.45,0); H (1.48,-0.27,0)	STO-3G
10-H_2O	H (-0.02,-0,0); O (0.84,0.45,0); H (1.48,-0.27,0)	6-31G
12-H_2O	H (-0.02,-0,0); O (0.84,0.45,0); H (1.48,-0.27,0)	6-31G

O.2 Heisenberg model Hamiltonian formulation

In addition to molecular chemistry benchmarks, we evaluate OptCRLQAS and replay buffer design on the n=5 qubit one-dimensional isotropic Heisenberg model with a uniform longitudinal field. The Hamiltonian is defined on a chain of n spin-$\frac{1}{2}$ particles with open boundary conditions as

$$H_{\text{Heis}} = \sum_{i=1}^{n} \Big(X_i X_{i+1} + Y_i Y_{i+1} + Z_i Z_{i+1} \Big) + \sum_{i=1}^{n} Z_i, \tag{55}$$

P Hyperparameters for quantum architecture search

Tables 19 summarize the environment and agent configurations used across the three QAS benchmarks. All experiments share the same agent architecture (multi-step DQN with n-step returns), optimizer, and curriculum type, differing only where the problem scale demands it.

Environment. The maximum circuit depth (`num_layers`) grows with qubit count, from 70 (6-qubit) to 250 (8-qubit) to 300 (12-qubit), to accommodate the deeper circuits required

Table 19: **Environment, agent, and optimizer configuration** for quantum architecture search benchmarks.

Parameter	6-qubit	8-qubit	12-qubit
Environment			
Episodes	5000	5000	1000
Qubits	6	8	12
Max layers	70	250	300
Accept error (Ha)	5.5	5.0	5.0
Shift threshold ball		0.001	
Shift threshold time		2000	
Success switch (Ha)	5.5	5.0	5.0
Success threshold		50	
`opt step interval` (m)	10	10	15
`energy interval` (m)	10	10	15
Agent (DQN with n-step returns [73])			
Network layers	$[1000]^3$	$[1000]^4$	$[1000]^4$
Batch size		1000	
Replay buffer size		20 000	
Learning rate		3×10^{-4}	
n-step return	5	6	6
Target net update		Every 500 steps	
γ (discount)		0.005	
Dropout		0.0	
Exploration (ϵ-greedy)			
ϵ_{start}		1.0	
$\epsilon_{\min}$		0.05	
ϵ_{decay}		0.99995	
Classical optimizer (COBYLA [52])			
Iterations		1000	

for larger Hamiltonians. The curriculum uses a `MovingThreshold` introduced in ref. [19] schedule in all cases, with the acceptance error initialized at 5.0-5.5 Ha and tightened by 0.001 Ha every 2000 episodes. OptCRLQAS evaluates the circuit every m architectural edits (`opt_step_interval` = `energy_interval`): m=10 for the 6- and 8-qubit problems and m=15 for the 12-qubit problem, reflecting the longer episodes at larger scale.

Agent. The DQN agent uses a 3-layer MLP ($[1000, 1000, 1000]$) for the 6-qubit task and a 4-layer MLP ($[1000, 1000, 1000, 1000]$) for the 8- and 12-qubit tasks. The n-step return horizon is n=5 for 6-qubit and n=6 for 8- and 12-qubit. Exploration follows an ϵ-greedy schedule decaying from ϵ=1.0 at rate 0.99995 per step to a floor of $\epsilon_{\min}$=0.05. The replay buffer holds 20 000 transitions in all cases, with a batch size of 1000.

Classical optimizer. All three tasks use COBYLA [52] with a local search size of 8. The 6- and 8-qubit tasks run 1000 global iterations per evaluation, as does the 12-qubit task; the reduced number of evaluations per episode at 12-qubit due to the larger amortization interval m=15 keeps the total classical optimization cost manageable.

Q State encoding for quantum architecture search

In both gateset configurations, the partially constructed ansatz at each step is represented as a three-dimensional state tensor $\mathbf{S} \in \mathbb{R}^{L \times R \times n}$, where L is the maximum number of circuit layers (moments), n is the number of qubits, and R is the number of row channels encoding gate placement and, optionally, variational angle parameters. A moment-tracking vector $\boldsymbol{\mu} \in \mathbb{Z}^n$ records the next available layer index per qubit, ensuring causal gate ordering. At each time step the agent places at most one 2-qubit gate and one 1-qubit rotation, updates $\boldsymbol{\mu}$ accordingly, and returns a flattened view of $\mathbf{S}$ as the observation.

Encoding I (for gateset $\{\texttt{RX}, \texttt{RY}, \texttt{RZ}, \texttt{CX}\}$). This a a complete binary encoding. The action is a four-tuple $a = [a_0, a_1, a_2, a_3]$, where a_0 is the CNOT control qubit ($a_0 = n$ signals no CNOT), the target is $t = (a_0 + a_1) \bmod n$, a_2 is the rotation qubit ($a_2 = n$ signals no rotation), and $a_3 \in \{1, 2, 3\}$ encodes the axis (X/Y/Z). Gate placement writes are strictly binary:

$$\mathbf{S}[\ell][t][a_0] \leftarrow 1 \quad \text{(CNOT)}, \tag{56}$$

$$\mathbf{S}[\ell][n + a_3 - 1][a_2] \leftarrow 1 \quad \text{(rotation)}, \tag{57}$$

where ℓ is the resolved moment from $\boldsymbol{\mu}$. Because the gate set contains only one two-qubit gate type (CX), all non-zero connectivity entries are uniformly 1 and gate identity is unambiguous.

Encoding II (for gateset $\{\texttt{RXX}, \texttt{RYY}, \texttt{RZZ}, \texttt{RX}, \texttt{RY}, \texttt{RZ}\}$). This encoding is integer-valued type. The action is an eight-tuple $a = [a_0^{xx}, a_1^{xx}, a_0^{yy}, a_1^{yy}, a_0^{zz}, a_1^{zz}, a_{\text{rot}}, a_{\text{axis}}]$, providing a control-qubit and an offset for each of the three two-qubit gate types plus a single-qubit rotation. Targets are $t_g = (a_0^g + a_1^g) \bmod n$ for $g \in \{xx, yy, zz\}$. Rather than introducing separate binary planes per gate type, distinct integer labels are assigned within a single shared connectivity plane:

$$\mathbf{S}[\ell][t_{xx}][a_0^{xx}] \leftarrow 1, \tag{58}$$

$$\mathbf{S}[\ell][t_{yy}][a_0^{yy}] \leftarrow 2, \tag{59}$$

$$\mathbf{S}[\ell][t_{zz}][a_0^{zz}] \leftarrow 3, \tag{60}$$

$$\mathbf{S}[\ell][n + a_{\text{axis}} - 1][a_{\text{rot}}] \leftarrow 1 \quad \text{(rotation)}. \tag{61}$$

A fully binary encoding for K distinct two-qubit gate types requires K separate $n \times n$ connectivity planes, giving a connectivity footprint of $\mathcal{O}(Kn^2)$. Encoding II consolidates all K types into a single plane via integer labels, reducing this to $\mathcal{O}(n^2)$ regardless of K, which is particularly advantageous as both the gate vocabulary and qubit count grow. The tradeoff is that integer labels introduce an implicit ordinal relationship among gate types that carries no physical meaning; a fully binary scheme provides a strictly one-hot signal per gate type that may be easier for the network to disentangle. A systematic comparison of the two encoding strategies across a controlled set of benchmarks is beyond the scope of the present work and is left as a direction for future investigation.

R Hyperparameters for quantum compilation

To isolate the effect of replay design, all quantum compilation experiments use a common off-policy deep Q-learning setup. The network architecture, optimization procedure, exploration schedule, and replay capacity are held fixed across agents, so that performance differences can be attributed to the replay mechanism rather than to changes in model capacity or training protocol.

Table 20 lists the hyperparameters used in the 1- and 2-qubit compiling benchmarks. The upper block gives the shared network and training configuration, while the lower blocks specify the additional method-dependent parameters for HER, PER, ReaPER, and ReaPER+. In this way, the comparison remains controlled, and the gains of ReaPER and ReaPER+ can be interpreted as arising from replay design itself.

S Compute details

All experiments were conducted on local and HPC machines, with runtimes not exceeding 48 hours. Table 21 summarizes the hardware configurations and runtimes for each experiment.

Table 20: **Hyperparameters used for 1 and 2-qubit quantum compilation experiments**. All agents share a common environment and training configuration; buffer-specific parameters are listed separately in the lower block.

Parameter	Value
Network (all agents)	
Architecture	
Hidden layer	2 (each 128 units)
Activation	ReLU
Input size	8
Output size	6 / 3 (small rotations / HRC)
Discount factor γ	0.99
Learning rate	3×10^{-4}
Batch size	200
Replay capacity	5×10^5
Target update freq.	100 episodes
Gradient clipping	1.0
ε-start	1.0
ε-min	0.01
ε-decay	0.99931
HER	
k-future relabeling	5
Strategy	`future`
PER	
α	0.6
β_0	0.4
β anneal frames	10^5
ReaPER	
α	0.4
ω	0.2
β_0	0.4
β anneal frames	10^5
ReaPER+	
α	0.4
$\omega_{\min}$	0.1
$\omega_{\max}$	0.7
T_{ann}	5×10^5
β_0	0.4
β anneal frames	10^5

Table 21: Computational resources and runtimes for all experiments.

Experiment	Qubits	GPU	CPU	Runtime (h)
Quantum compiling	1-2	RTX 4070	i7-13620H	$\sim$1.5
Quantum architecture search	5-12	Tesla V100S	EPYC 7402	$\sim$48
Lightweight buffer transfer	6-10	RTX 4070	i7-13620H	$\sim$2
	12	Tesla V100S	EPYC 7402	$\sim$5

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes],

   Justification: The main claim made in the abstract and the introduction are clearly reflected in the Section 4.

2. **Limitations**

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes],

Justification: The limitations and future work is discussed in the Section A.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes].

   Justification: The theoretical justification and the set of assumptions are elaborated in the Section D and Section E.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes].

   Justification: Yes it does as we provide the elaborated hyperparameter settings in the Section R and Section P. Also the code is available at GitHub.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes].

   Justification: Yes the code and accompanied data to fully reproduce the results is available at GitHub.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes].

   Justification: In the appendix in Section R, Section P and Section N we provide a the complete details of the hyperparameter and how they were chosen.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes].

   Justification: In the Figure 10, Figure 9, Figure 3 and Figure 2 we utilize error bars and we define in the captions clearly that they are 1-sigma error bars.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes].

   Justification: In the Table S we define the CPU and GPU utilized in the paper and the total computation time.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes].

   Justification: Yes the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes].

    Justification: The broader impact is described in the Section B.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A].

    Justification: The paper poses no such risks.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes].

    Justification: The paper does use LunarLander-v3 environment from Gymnasium which is distributed under MIT license.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes].

    Justification: Yes the new assets are well documented in the open repository GitHub.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A].

    Justification: the paper does not involve crowdsourcing nor research with human subjects.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A].

    Justification: the paper does not involve crowdsourcing nor research with human subjects.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A].

    Justification: The authors utilize LLMs in the text and paragraph level not in the methodological level.