2024 IEEE International Conference on Quantum Computing and Engineering (QCE)

Hype or Heuristic? Quantum Reinforcement Learning for Join Order Optimisation

Maja Franz
Technical University of Applied Sciences Regensburg
Regensburg, Germany
maja.franz@othr.de

Tobias Winker
University of Lübeck
Lübeck, Germany
t.winker@uni-luebeck.de

Sven Groppe
University of Lübeck
Lübeck, Germany
sven.groppe@uni-luebeck.de

Wolfgang Mauerer
Technical University of Applied Sciences Regensburg
Siemens AG, Technology
Regensburg/Munich, Germany
wolfgang.mauerer@othr.de

***Abstract*—Identifying optimal join orders (JOs) stands out as a key challenge in database research and engineering. Owing to the large search space, established classical methods rely on approximations and heuristics. Recent efforts have successfully explored reinforcement learning (RL) for JO. Likewise, quantum versions of RL have received considerable scientific attention. Yet, it is an open question if they can achieve sustainable, practical advantages with improved quantum processors.**

In this paper, we present a novel approach that uses quantum reinforcement learning (QRL) for JO based on a hybrid variational quantum ansatz. It is able to handle general *bushy* join trees instead of resorting to simpler *left-deep* variants as compared to approaches based on quantum(-inspired) optimisation, yet requires multiple orders of magnitudes fewer qubits, which is a scarce resource even for post-NISQ systems.

Despite moderate circuit depth, the ansatz exceeds current NISQ capabilities, requiring an evaluation by numerical simulations. While QRL may not significantly outperform classical approaches in solving the JO problem with respect to result quality (albeit we see parity), we find a drastic reduction in required trainable parameters. This benefits practically relevant aspects ranging from shorter training times compared to classical RL, less involved classical optimisation passes, or better use of available training data, and fits data-stream and low-latency processing scenarios. Our comprehensive evaluation and discussion delivers a balanced perspective on possible practical quantum advantage, provides insights for future systemic approaches, and allows for quantitatively assessing trade-offs of quantum approaches for a crucial problem of database management systems.

***Index Terms*—Quantum Machine Learning, Reinforcement Learning, Query Optimisation, Database Management Systems**

I. Introduction

In database research and industrial practice, finding good orders in which joins between table columns are executed in a query—the so-called join order (JO) problem—counts amongs the most fundamental issues of database management systems (DMBS) [1]–[9]. The chosen order substantially impacts query execution time. While the problem does only need little amounts of input information (the query to be executed, and characteristics of the payload data obtained from statistical samples), the problem is know to be NP-hard in general, and also for common restricted scenarios [10]. An optimal JO cannot be efficiently found deterministically. The last few decades have seen various classical heuristics that can find suboptimal JOs in polynomial time [11]–[13]. Recent classical work [14]–[21] explores the application of reinforcement learning (RL) to tackle the JO problem. RL is considered to be beneficial in scenarios where the solution to a problem can be determined by a series of subsequent decision steps, and where finding one such good sequence for a problem generalises well to others, or when highly dynamic problems are considered. By learning from experience of past query evaluation, RL can find good decision sequences in vast search spaces, and only requires information about the current state of the system. This is particularly advantageous for the JO problem, as very typical scenarios in database systems need to process information at a high temporal frequency.

In this paper, we approach RL for JO from the perspective of quantum machine learning (QML), an emerging technique that leverages the principles of quantum mechanics for potential computational speed-ups. It has been shown that certain problems [22], [23] can be solved more efficiently using quantum algorithms over classical approaches. However, the practical utility of these algorithms is limited on the current generation of quantum computers, so-called noisy intermediate-scale quantum (NISQ) systems [24], as they only offer a limited amount of qubits and are prone to noise and imperfections [25] that strongly limit possible circuits depth and thus the length of quantum computations. To address these limitations, *hybrid* quantum-classical algorithms are proposed, where only a limited number of steps is performed on a quantum computer and the remaining steps on classical machines. As Pirnay *et al.* [26] show, fault-tolerant quantum computers can provably provide super-polynomial advantage for optimisation problems over classical algorithms. Hybrid *variational* algorithms [27]–[29] are considered key candidates for exploiting advantages of near-term quantum devices, but could also be beneficial in post-NISQ systems because of their resource efficiency.

Within the class of hybrid variational algorithms, quantum machine learning (QML) has shown promise by moving certain parts of classical machine learning to quantum computers. QCs will, despite common misperceptions, likely be inapt for handling large amounts of data [30]. This makes quantum reinforcement learning (QRL) [31]–[33], which requires little training data, a promising approach. As established JO approaches mostly rely on statistical estimates of properties of the database, JO seems a good match for QRL.

QML in general has been shown to outperform classical machine learning for certain tasks [34]–[40]. Specifically, for QRL it is hypothesised that fewer parameters are required than for classical neural networks (NNs) to address RL tasks [31], [41]. Several studies also suggest that QRL can solve tasks that are intractable to classical machine learning [42], or that it may have an advantage over classical NNs in terms of sampling complexity, that is, fewer interactions with the environment are required to achieve optimality for certain problems [32], [33]. For these reasons, the application of QML to database problems is also considered promising [43].

However, as detailed in Ref. [44], many approaches for QML that claim quantum advantage rest on artificially constructed scenarios (*e.g.*, [37], [38], [40]). Consequently, a practical definition of QML goals is required, which should not imply an exponential speed-up compared to classical approaches, but rather is a matter of details.

We have chosen to use a recent classical RL-based approach to join ordering by Marcus and Papaemmanouil [14] as baseline that is well aligned with intensively studied quantum variants of reinforcement learning [45]. It is known that a careful consideration of various factors is necessary to gauge potential improvements. This includes a sound classical baseline, data representation, quantum circuit structure, and hyperparameters. Further, we provide a high-level evaluation of hardware requirements. Our detailed contributions are:

- We systematically replicate[1] the classical baseline [14] and generalise it to the quantum case. As the baseline does not provide source code or hyperparameters, this is an important prerequisite to ascertain a fair comparison, and allows us to consider all aspects of the DBMS.
- We comprehensively simulate the performance of our approach on the join order benchmark (JOB), which is a universally accepted touchstone in the database community, and compare it against the classical baseline and a single-step QML technique [46] that was shown to outperform established classical approaches. Multi-step QRL can achieve up to 17% lower median costs than single-step QML on the selected dataset and cost model.
- We identify potentials for improvement in view of future hardware development, and carefully address the issue of judging realistic potentials for practical improvements over classical heuristics.
- We provide an open-source reproduction package [47] that makes our code transparent to the community, and can serve as basis to build further experiments upon, and benchmark alternative approaches against.

We aim to provide a comprehensive perspective on the quantum advantage landscape in RL for the JO problem. By combining optimistic hypotheses with an acknowledgement of established challenges and limitations, we strive to present a balanced view. This balance is important to guiding future research directions and manage expectations regarding the (near- and far-term) practical benefits of quantum algorithms in the field of database management systems.

[1]We follow ACM terminology on Artifact Review and Badging: A *replication* describes measurements obtained by a different team using a different experimental setup. The term *re-implementation* is also common in the literature, with identical meaning.

The paper is structured as follows: Sec. II reviews existing literature on classical approaches for the JO problem and QC for databases. Sec. III describes the theoretical background for the application of the JO problem and the method of classical and quantum RL, followed by an overview of our methodology in Sec. IV. Sec. V outlines our experiments, which are discussed in Sec. VI. We conclude in Sec. VII.

II. Related Work

The problem of query optimisation, which is formally defined in Sec. III-A, has been studied for over 40 years [13], and new results appear frequently [3], [4], [8], [9]. Since the search space for the JO problem scales factorial [10], an exhaustive search for the optimal JO is only feasible for a small number of relations, even when relying on dynamic programming (DP) approaches [13], [48]–[51], necessitating heuristic methods [1], [11], [52]–[55] for large queries.

Heuristics require to calculate costs; for instance, execution time or number of intermediate results. These, in turn, depend on estimates of the cardinalities of subqueries. Ref. [56] reviews cardinality estimation techniques and their impact on JO optimisation. Some approaches apply machine learning for cardinality or cost estimation [57]–[59], to improve the DP optimiser, or to directly determine the JO [14], [16]–[19].

Using quantum approaches to address database problems is a relatively new field of research, even with early work by Trummer and Koch on solving multi-query optimisation with quantum annealers only going back to 2016 [60]. A recent review [61] summarises existing work and classifies potential use-cases. For instance, transaction scheduling [43], [62]–[64] schema matching [65] or tuning index configurations [66] have been addressed using quantum methods.

The join order problem has been cast as an optimisation problem in quadratic unconstrained binary optimisation (QUBO) form by Schönberger *et al.* based on known transformations to mixed-integer linear programming [67], and using a direct encoding that has also been evaluated on quantum-inspired hardware [68]. These two solutions for the JO problem are restricted to left-deep join trees; alternative formulations that allow for handling general *bushy* join trees were given by Nayak *et al.* [69] and Schönberger *et al.* [70] (we discuss differences in their scalability in Sec. VI). Finally, Ref. [46] introduces an RL inspired approach for the JO problem using VQCs. It uses rewards to measure the quality of different join orders, but creates a join order in a single step and not over multiple interactions with an environment.

III. Preliminaries

This section introduces the three main concepts relevant to this work, namely the JO problem (Sec. III-A), and classical (Sec. III-B) and quantum (Sec. III-C) RL.

A. Background on the Join Order Problem

The JO problem constitutes of three basic elements:

1) Query: A query formulated in the structured query language (SQL) (see left of Fig. 2 for an example), can be represented as an expression of relational algebra to be optimised before execution [71]. In this work, we focus on the important problem of JO optimisation with consideration of selection (*i.e.*, filter) operations while the query, in general, may also consist of other operations. Here, a query Q can be characterised by a join graph and predicates, which can be further decomposed into join predicates and selection predicates. A join graph for a query is defined by relations that represent the vertices of the graph and filter on which two relations can be joined. These are called join predicates (*e.g.*, `a1.a=D.a` in Fig. 2) and correspond to the edges of the join graph. The join graph is given by a symmetric adjacency matrix $G \in \mathbb{F}_2^{r \times r}$, where r is the number of relations. If there is a join predicate in Q connecting the relations r_i and r_j, the entry $g_{i,j}$ in G is 1. Selection predicates are additional filters that act on one relation (*e.g.*, `D.c > 5` in Fig. 2), and can be formalised as described in Par. IV-C1b or Par. IV-C2a.

2) Join Tree: In contrast to a query graph, which serves as the input for the JO problem, a join tree embodies a solution. Its leaf nodes represent the base relations to be joined, while its intermediate nodes denote join operations. Each join node, requiring two operands, has two predecessors: either a) a base relation or b) another join tree node, which itself will be further joined. The result of a join serves as an operand for another join, indicated by an outgoing edge connecting to its successor. The only exception is the final join, which does not serve as an operand for any subsequent join. In this study, we refer to intermediate join trees as "sub-trees". The top of Fig. 3 illustrates the sequence for constructing a complete join tree.

While these requirements apply universally to join trees, certain JO methods impose additional constraints on their structure to enhance efficiency by reducing the search space. Particularly, some methods exclusively consider *left-deep* join trees, necessitating at least one base relation as an operand for each join. Consequently, directly joining two pairs of relations is precluded, as it necessitates a join operation on the results of two preceding joins. Valid left-deep join orders must therefore represent a permutation of relations. This restriction to left-deep trees was employed in two quantum approaches for JO [67], [68]. Nonetheless, the detrimental impact of this constraint on solution quality can be significant, as demonstrated, for instance, by the empirical analysis conducted by Neumann and Radke [3]. Hence, our QML approaches consider general or *bushy* join trees, devoid of further structural constraints. The divergence in scalability between existing quantum-based left-deep and bushy variants is described in Sec. VI-B.

3) Cost Functions: Finally, a cost function evaluates the join tree, by assigning it a cost value. The literature proposes various definitions of cost functions [72]; some are straightforward yet less precise, while others are more intricate, taking into account multiple factors and closely reflecting real costs (*i.e.*, query execution time including I/O costs). To evaluate the selected join order, we use the established cost function C_{out} [10], which considers the cardinalities (*i.e.*, the number of tuples in a query result set) as an approximation of query complexity:

$$C_{\text{out}}(T) = |T| + C_{\text{out}}(T_1) + C_{\text{out}}(T_2), \tag{1}$$

where n is the maximum number of joins in the query, a join tree is defined as $T = T_1 \bowtie T_2$, and $|T|$ represents the true cardinality of T ($C_{\text{out}}(T) = 0$ if $T \in \{r_1, r_2, \dots\}$ is a leaf).

B. Background on Reinforcement Learning

The setup in RL is typically described by the notion of a *Markov decision process* (MDP) [73], where an *agent* interacts with an *environment* at discrete time steps t. In each time step, the current configuration of the agent in the environment is summarised by the *state* $S_t \in \mathcal{S}$, where $\mathcal{S}$ is the set of all possible states. Based on this information, the agent selects an *action* A_t from a set of possible actions $\mathcal{A}$ according to a *policy* $\pi(s,a) = \mathbb{P}[A_t = a \mid S_t = s]$, which gives the probability $\mathbb{P}$ of taking action a in state s. Executing the selected action causes the environment to transition to a next state $S_{t+1} \in \mathcal{S}$. Simultaneously, the agent receives a scalar *reward* $R_{t+1} \in \mathcal{R}$ that quantifies the contribution of the selected action towards solving the task, with $\mathcal{R} \subset \mathbb{R}$ being the set of all rewards. S_{t+1} and R_{t+1} are determined by the environment's dynamics $p : \mathcal{S}\times\mathcal{R}\times\mathcal{S}\times\mathcal{A}$, which characterises the probability distribution of a transition $(S_t, A_t, R_{t+1}, S_{t+1})$.

The agent's goal is to maximise the return [73] $G_t = \sum_{t'=t}^{T} \gamma^{t'-t} R_{t'+1}$, that is, the discounted sum of rewards, until a terminal timestep T is reached, where the discount factor $\gamma \in (0,1]$ controls how much the agent favours immediate over future rewards. The period between the initial time step and T is often referred to as an *episode*.

To find a good policy that maximises the return, various RL methods exist [73]. As our baseline [14], in this work we focus on *Proximal Policy Optimization* (PPO) from the class of policy gradient methods [74]. The goal of policy gradient methods is to directly learn the parameterised policy $\pi_{\boldsymbol{\theta}} : \mathcal{S} \times \mathcal{A} \rightarrow [0,1]$, where $\boldsymbol{\theta}$ denote trainable parameters of a function approximator, such as a neural network (NN), or a variational quantum circuit (VQC). In PPO, the parameters $\boldsymbol{\theta}$ can be optimised using a gradient ascent method, maximising the following objective, consisting of three parts:

$$L_t^{\text{clip+VF+S}}(\boldsymbol{\theta}) = \mathbb{E}_t\left[L_t^{\text{clip}}(\boldsymbol{\theta}) - c_1 L_t^{\text{VF}}(\boldsymbol{\theta}) + c_2 S(\pi_{\boldsymbol{\theta}})\right]. \tag{2}$$

The PPO algorithm alternates between sampling and optimisation stages. Therefore, $\mathbb{E}_t$ indicates the average over a finite batch of samples, which is gathered prior to each optimisation stage. c_1 and $c_2 \in \mathbb{R}^+$ are hyperparameters. The clip-objective $L^{\text{clip}}(\boldsymbol{\theta})$, is defined as $r_t(\boldsymbol{\theta})\mathbb{A}_t$, where the ratio $r_t(\boldsymbol{\theta}) = \frac{\pi_{\boldsymbol{\theta}}(a_t,s_t)}{\pi_{\boldsymbol{\theta}_{\text{old}}}(a_t,s_t)}$ is clipped in $1 \pm \epsilon$ with $\boldsymbol{\theta}_{\text{old}}$ being the parameters before the update, $\epsilon \in \mathbb{R}$ a hyperparameter and $\mathbb{A}_t$ an advantage estimation of the current policy. The advantage estimation $\mathbb{A}_t$ itself can be learned by a function approximator based on the value function in an MDP $V(s) = \mathbb{E}_t[G_t \mid S_t = s]$, that is an estimation of the return, and optimised through the objective L_t^{VF}, which is a squared-error loss function of estimated values from the function approximator and target values, collected in the sampling stage. The third part in Eq. 2 $S[\pi_{\boldsymbol{\theta}}]$ denotes the entropy of $\pi_{\boldsymbol{\theta}}$, which is added to ensure sufficient exploration. For a detailed discussion on PPO, we point readers to Ref. [74].

We refer to the policy function approximator that mainly contributes to $L^{\text{clip}}(\boldsymbol{\theta})$ as the *actor*, as it represents the policy that "acts" in the environment and to the advantage estimator, which is optimised through $L^{\text{VF}}(\boldsymbol{\theta})$ as the *critic*, which evaluates a current policy. We investigates classical and quantum versions of actor/critic in Sec. V-B.

C. Background on Quantum Machine Learning

As a variational quantum circuit (VQC) is proven to be a universal function approximator [75], similar to a classical NN [76], it can be employed as a set-in for NNs in a variety of settings (*e.g.*, [27], [77]), including PPO. A VQC's structure often follows the data processing flow of a classical NN and comprises three fundamental components: In the first part, a quantum state is prepared to represent the classical input data $\boldsymbol{x}$ through applying a unitary gate $\hat{U}_{\text{enc}}(\boldsymbol{x})$ to the initial quantum state, which by convention is $\otimes_n |0\rangle$ for a configuration with n qubits [78]. In the second so-called *variational* part, the quantum state is then transformed by applying a parameterised unitary $\hat{U}_{\text{var}}(\boldsymbol{\theta})$. An exemplary gate sequence for the encoding and the variational part is depicted in Fig. 4. Finally, classical information $\langle\hat{O}\rangle$ is obtained from the quantum circuit by measuring the state. The notation $\langle\hat{O}\rangle$ refers to the *expectation value* of an observable $\hat{O}$.

The parameters of the VQC are optimised using classical approaches such as gradient ascent to maximise an objective function, where the gradient of a parameter with respect to the measurement can be calculated using the *parameter-shift rule* [77], [79]. As algorithms involving VQCs perform calculations on both, the quantum processing unit (QPU) and CPU, they are called *hybrid* approaches.

1) Data Encoding: The encoding unitary $\hat{U}_{\text{enc}}(\boldsymbol{x})$ depends on the encoding strategy; Weigold *et al.* [80], [81] survey common strategies. Among these, we focus on *angle encoding*, which uses a Pauli-rotation gate to encode one real value into one qubit. The corresponding unitary can comprise one (*e.g.*, [32]) or multiple (*e.g.*, [31]) parameterised rotation gates per qubit. Given that the gates are periodic, each input element must be scaled to an interval smaller than 2π.

Even if payload data are not required to encode JO problems, a simple angle encoding scheme for JO exceeds the capability of NISQ devices for even small instances. We therefore employ *incremental data uploading* [82] to spread the encoding gates for the input elements throughout the quantum circuit with parameterised unitaries in between them, which increases *circuit depth* (*i.e.*, the longest gate sequence), but decreases qubit count. As there is no limit on the maximum number of repetitions of input elements, encoding unitaries can be re-introduced multiple times into the VQC. This approach, known as *data re-uploading* (DRU) [75], is suggested to increase the expressivity of a VQC [83], which in turn determines the class of functions a VQC can approximate. In Sec. V-B we empirically evaluate and compare the combination of incremental data uploading and DRU.

2) Data Decoding: Several techniques are known to map "outputs" of a VQC (*i.e.*, the expectation value of multiple measurements) to a set of output values that is smaller than or equal to the number of qubits [42], [84]. Few existing approaches [85] decode quantum states to larger output spaces. As described in Sec. IV-C1, action and output space are typically larger than the number of qubits for JO. We therefore determine the expectation value for each qubit individually using $\hat{Z}$ observables and feed the outcomes into one classical NN layer with the correct output size for the actor. For the critic model, which only requires one output component to estimate the advantage, circuit outcome is determined by observable $\otimes_n \hat{Z}$. Since the expectation value of $\hat{Z}$ lies in $[-1, 1]$ the critic model outcome is scaled using an additional trainable classical parameter and bias.

IV. Methodology

To understand how RL can be utilised for the JO problem on QCs, we commence with discussing the differences between building the join order step-wise or returning the full join order within one single step using a machine learning (ML) model. We also introduce a single-step approach based on QML. Subsequently, we outline our classical baseline *ReJoin* and the adjustments required for quantum RL.

A. Single-Step versus Multi-Step Join Ordering

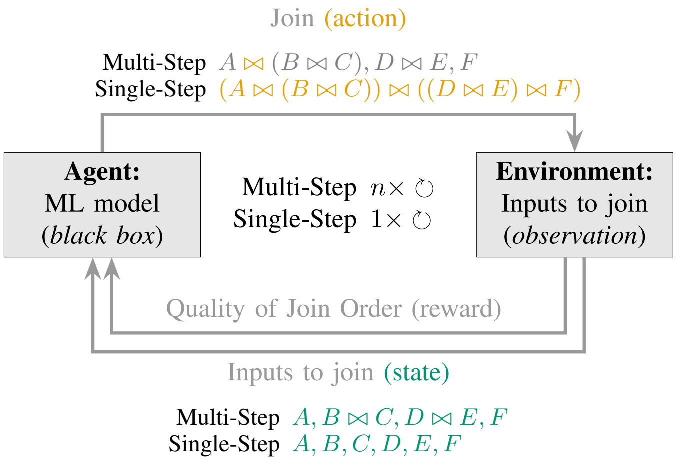

Join (action)
Multi-Step $A \bowtie (B \bowtie C), D \bowtie E, F$
Single-Step $(A \bowtie (B \bowtie C)) \bowtie ((D \bowtie E) \bowtie F)$
Agent:
ML model
(black box)
Multi-Step $n\times$ ↻
Single-Step $1\times$ ↻
Environment:
Inputs to join
(observation)
Quality of Join Order (reward)
Inputs to join (state)
Multi-Step $A, B \bowtie C, D \bowtie E, F$
Single-Step A, B, C, D, E, F

Fig. 1. Single-step versus multi-step approach presented in an RL fashion. Here, A to F are the relations to join. We neglect selection predicates.

A join tree can be created by an ML model in multiple steps or in a single step. Fig. 1 summarises the differences: The state of the environment contains (1) already determined subjoins, (2) relations to be joined, and (3) selection predicates. An ML model acts as an agent in the RL context. It predicts and emits the next best subjoin (*i.e.*, the action) connecting two of the subjoins and relations of the previous state. Thereby, the environment of already determined subjoins and to be joined relations is updated. By determining the quality (*i.e.*, the reward) of the intermediate join order, the model can be trained to better predict the next best subjoin. In the *multi-step approach*, a single join is added to the join tree in each step until all input relations are joined (*i.e.*, a terminal state is reached). In the *single-step approach* [46], the model directly generates a complete join order without intermediate steps (*i.e.*, a terminal state is reached after one step).

B. Single-Step QML

In the single-step QML approach [46], all join orders are enumerated, and each join tree is associated with a quantum state. The join order associated with the most commonly measured quantum state is taken as join tree. The quality of join orders can vary greatly and multiple join orders can have equal or nearly equal good quality. For instance, the second-best join order might only be slightly worse, while another might have a large difference in quality. Thus, it is not a good approach to make a binary choice between right or wrong for a join order. Instead, each join order is assigned a reward depending on its quality. We use the VQC to predict these rewards and choose the join order with the highest reward.

C. Multi-Step QRL

Fig. 2 visualises our QRL multi-step approach. By using a state representation based on Ref. [14], a VQC can choose the next join in an iterative process until a complete join order is built. The classical baseline as well as the modifications required for the application of QRL are described below.

1) Classical Baseline—ReJoin*:* For our multi-step approach, we utilised the method described in Ref. [14]. Although the literature proposes various RL methods for the JO problem (cf. Sec. II), we opted for *ReJoin* as a foundation because of its compact input feature space. Other approaches, such as *RTOS* [19] or *JOGGER* [18], utilise sophisticated classical machine learning techniques to represent states of queries and databases, which lack a direct equivalent in the domain of quantum computing. Investigating novel methods that apply these advanced classical machine learning techniques to a quantum domain is beyond the scope of this study. Instead, our QRL approach should evaluate the capabilities of existing QML methods on small input spaces of the JO problem to establish a lower bound for the potential of using QRL, or QC in general. Additionally, due to the limited number of qubits on current NISQ devices and each quantum circuit gate being a potential source for noise and imperfections, it is beneficial to reduce the classical data encoded into the quantum gates to a minimum. As outlined below, *ReJoin* employs a total of $a + 2r^2$ input features, where r denotes the number of tables and a represents the total number of attributes in the database with $a > r$. As we show in Par. IV-C2a we are able to reduce the input space even further. In contrast, for example *DQ* [16] necessitates roughly $r \times (a+1)$ features, resulting in a larger input space considering that the number of attributes typically outweighs the number of relations in the database.

a) MDP: The MDP's state for the JO problem is represented by a query Q and a set of relations or (sub-)join-trees $\mathcal{F}$. The PPO agent sequentially combines two sub-trees $T_k, T_l \in \mathcal{F}$, which corresponds to an action, until a complete join order is build. Building the join order for one query, represents an episode. The agent aims for a join order that achieves minimum costs respectively a maximum reward.

b) State Representation: Formally, one part of the state representation is the join graph G, defined in Sec. III-A. Additionally, selection predicates in the query Q are represented by a vector of length a, which is the number of attributes in the database. Selection predicates are one-hot encoded: If a predicate is present in Q, the corresponding value in the predicate vector P is one; otherwise zero. Furthermore, each intermediate sub-tree $T_k \in \mathcal{F}$, that is the tree structure, is encoded as a row vector τ_k of size r. If a relation r_i is equal to T_k (r_i is a leaf) or is present in T_k, then the corresponding value in the row vector $\tau_{k,i}$ is $\frac{1}{h(i,k)}$, where $h(i,k)$ is the height of r_i in T_k. To ensure an evenly sized input space throughout the training process, for each subtree T_k that is successfully joined to another subtree T_l, τ_k is set to $\vec{0}$. There exist r sub-tree row vectors $\mathring{T}$ in total, since at the beginning of each join-process each relation correspond to one sub-tree. An exemplary sequence of row vectors that is encountered until a full join order is built is depicted in Fig. 3, which uses the reduced encoding introduced in Par. IV-C2a. The complete state for the baseline can be expressed through concatenation, $S_t = G^f \oplus P \oplus (\oplus_{\tau_k \in \mathring{T}} \tau_k)$, where G^f denotes the flattened join graph as a vector and $\oplus$ concatenation with $|S_t| = a + 2r^2$.

c) Action Representation: The PPO actor returns a probability distribution over all actions $A_t \in \mathcal{A}$. The set of actions $\mathcal{A}$ comprises all combinations of two sub-trees $(T_k, T_l) \forall T_k, T_l \in \mathcal{F}, k \neq l$, resulting in an action space of size $r \times (r-1)$. It encompasses actions with relations that are not present in the query, or lead to a cross join (*i.e.* a join between relations that are not connected by a join predicate). As these typically involve high costs, we apply a mask to the policy by multiplying each value that represents an invalid action with zero to prevent them from being sampled.

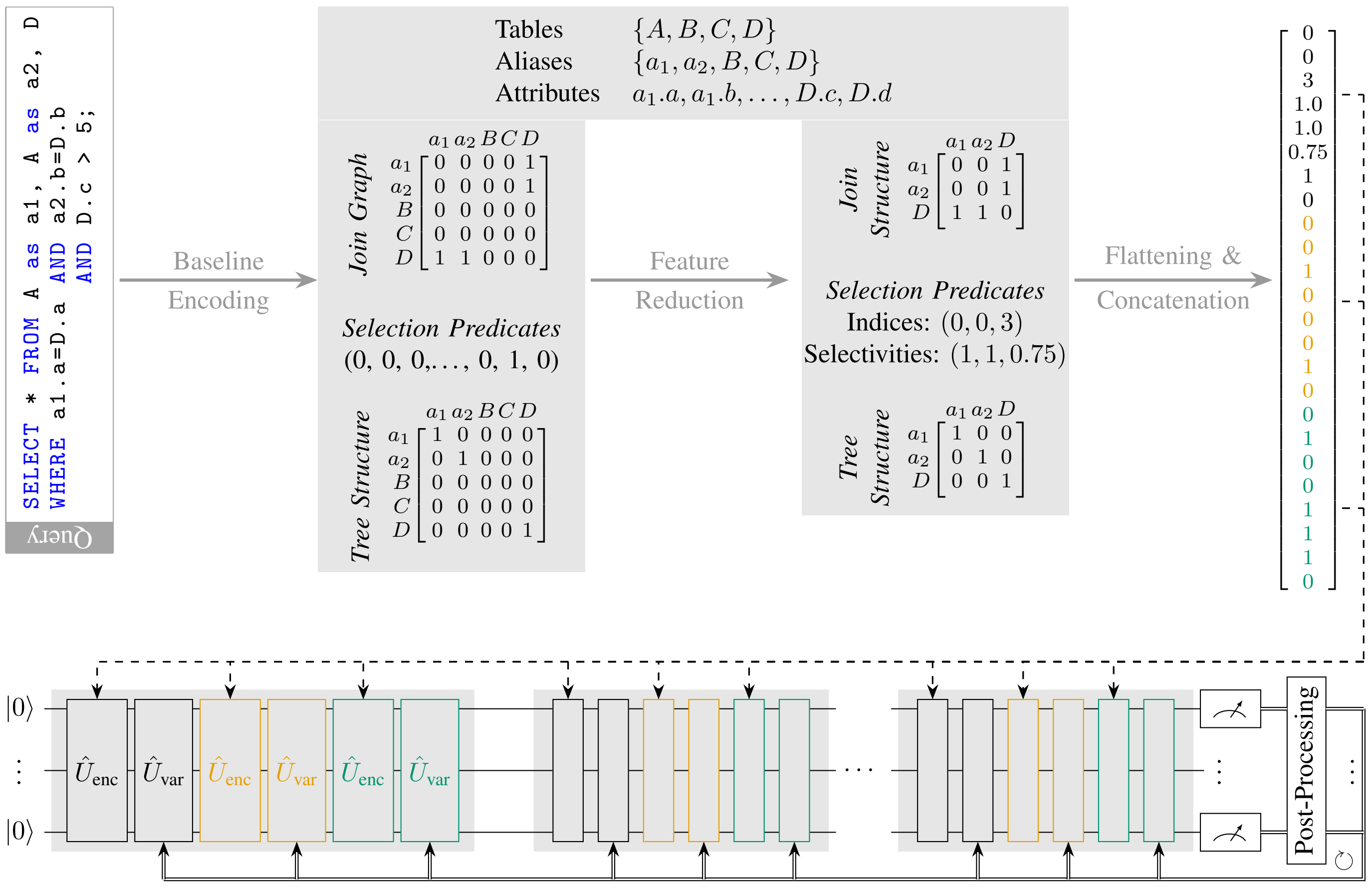
d) Reward Signal: In previous studies on RL for JO (*e.g.*, Refs. [14], [16], [19]) the reward, as function of cost, is only assigned at the end of each episode when the full join order is built by the RL policy. Intermediate steps receive a zero reward. This seems counter-productive, given that one property of RL is to determine an action based on a current state and reward signal[2]. Therefore, we propose a multi-step reward signal: Assuming the cost difference C_t between timesteps t and $t-1$ with costs c_k for subtrees $T_k \in \mathcal{F}_t$ in a state S_t is

$$C_t = \begin{cases} \sum_{T_k \in \mathcal{F}_t} c_k - \sum_{T_l \in \mathcal{F}_{t-1}} c_l & \text{if } t > 0 \\ 0 & \text{if } t = 0 \end{cases}, \tag{3}$$

and the cost assigned to the best join order of the full query determined by a DP exhaustive search is C_{DP}, we propose the clipped reward at t as

$$R_t = \frac{1}{n-1}\left[-\min\left(\frac{C_t}{C_{\text{DP}}}, n-1\right) + 2\right]. \tag{4}$$

[2]We provide a comparison to a method, which awards zero to intermediate steps in the supplementary material in our reproduction package



SELECT * FROM A as a1, A as a2, D
WHERE a1.a=D.a AND a2.b=D.b
AND D.c > 5;
Query
Baseline Encoding
Tables $\{A, B, C, D\}$
Aliases $\{a_1, a_2, B, C, D\}$
Attributes $a_1.a, a_1.b, \ldots, D.c, D.d$
Join Graph
Selection Predicates (0, 0, 0,…, 0, 1, 0)
Tree Structure
Feature Reduction
Join Structure
Selection Predicates Indices: (0, 0, 3) Selectivities: (1, 1, 0.75)
Tree Structure
Flattening & Concatenation
$|0\rangle$
$\hat{U}_{\text{enc}}$
$\hat{U}_{\text{var}}$
Post-Processing

Fig. 2. Interplay between data encoding (top) and variational quantum circuit (bottom) processing in our approach. Starting from the query and the baseline encoding of Ref. [14], we prune unnecessary features and flatten the core input data into a vector that is statically fed into the encoding quantum gates $\hat{U}_{\text{enc}}$. The variational quantum circuit (using a configurable number of qubits) is initialised with qubits in state $|0\rangle$, and iteratively executes block of intermingled encoding and variational ($\hat{U}_{\text{var}}$) gates; following a measurement, a classical optimisation procedure delivers new parameter estimates for the variational gates, and the updated circuit is iteratively re-executed. Following established conventions, solid lines indicate quantum information, double lines concern classical information (measurement results that may change in each run of the quantum circuit), and dashed lines represent parameters that are statically fed into the quantum circuit (remaining constant across circuit runs). Grey, thick lines symbolise logical flow.

This requires $n-1$ joins (and actions) to build the join tree for a query with n relations. Clipping, shifting and normalising the ratio reduces the chances of steeper gradients during training, which is a known cause of suboptimal training [86].

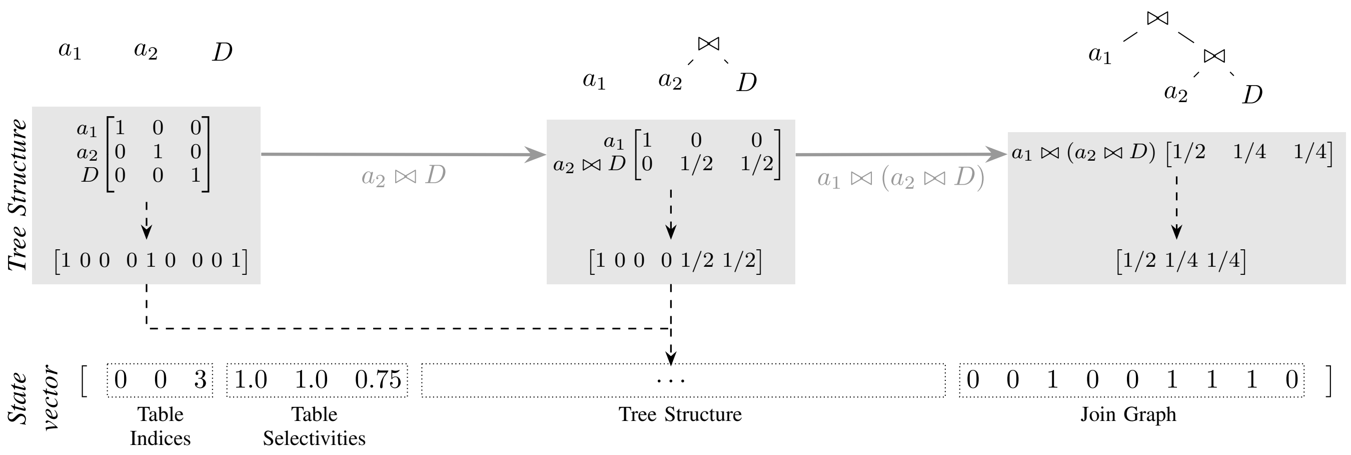
2) *Quantum* ReJoin*:* For *ReJoin*, a VQC can be employed as the actor-, as well as critic-part of PPO, or both. In both cases, the VQC encodes the state S_t. Policy or advantage estimations are obtained using the approach of Sec. III-C2.

As the number of inputs that a QPU can process is restricted by the hardware capabilities of QPUs, it is advantageous to minimise this number. As described in Sec. IV-C1, the state representation of the classical baseline suggests a state space with $a + 2r^2$ features for a database with r relations[3] and a attributes. For the JOB, which encompasses 208 attributes across 39 different aliases throughout the JOB query set, there are 3 250 input elements for one state.

[3]We assume r is the number of different aliases occurring in the dataset, and a is the number of attributes corresponding to these aliases. One author of Ref. [14] confirmed that multi-aliases were handled as an additional tables.

a) Reducing the Input Size: To reduce the observation space, we specify a maximum number of relations n that can be joined. As for the baseline, we employ a join graph and a tree structure representation, which are defined analogous to the baseline over the n relations present in a given query. This leads to n^2 elements in both, the join graph and the sub-tree structure representation. To specify, which tables are referenced in a query, the tables in the database are enumerated and assigned with an index $I : \mathcal{T} \rightarrow [0, r-1]$, where $\mathcal{T}$ is the set of all tables and r is the number of tables in the database. The indices $i \in \bigcup_{\mathcal{T}_q \in Q} I(\mathcal{T}_q)$ for a query Q are added to the input components. To represent the information, which is given through the selection predicates, we obtain the selectivity (*i.e.*, the fraction of tuples present in a result when filtering for the corresponding selection predicates of a specific table) for every table in a query and add these to the input components.

The reduced state representation leads to $2(n^2 + n)$ input elements. For $n = 17$ as maximum size in the JOB, this results in 612 elements, over 80% less than in the baseline.



a_1 a_2 D
Tree Structure
$\begin{array}{c|ccc} a_1 & 1 & 0 & 0 \\ a_2 & 0 & 1 & 0 \\ D & 0 & 0 & 1 \end{array}$
[1 0 0 0 1 0 0 0 1]
$a_2 \bowtie D$
$\begin{array}{c|ccc} a_1 & 1 & 0 & 0 \\ a_2 \bowtie D & 0 & 1/2 & 1/2 \end{array}$
[1 0 0 0 1/2 1/2]
$a_1 \bowtie (a_2 \bowtie D)$
$a_1 \bowtie (a_2 \bowtie D)$ [1/2 1/4 1/4]
[1/2 1/4 1/4]
State vector
0 0 3
1.0 1.0 0.75
0 0 1 0 0 1 1 1 0
Table Indices
Table Selectivities
Tree Structure
Join Graph

Fig. 3. Processing sequence to iteratively determine join orders. Once the query has been parsed and encoded, subsequent invocations of the variational quantum circuit as illustrated in Fig. 2, determine more and more joins, until a complete order has been found.

b) Circuit dimensions: One advantage of quantum algorithms involving VQCs is that they allow for a certain degree of controllability of the circuit depth and number of qubits, which is especially desirable for NISQ devices [24]. Utilising the incremental data-uploading [82] and the DRU [75] approaches, we can choose the structure of the quantum circuit. We opted to divide the $2(n^2+n)$ input features in n equally sized parts p_l. Each feature $f_i \in p_l$ is then scaled to a range of $[0, \pi]$ and used as s rotation angle for a $\hat{R}_x$ gate acting on qubit i in the *layer* l. The input parts are interleaved with parameterised gates $\hat{R}_y$ and $\hat{R}_z$ that act on each qubit and introduce trainable parameters, and a circular sequence of C–$\hat{Z}$ gates between two adjacent qubits, which create entanglement. Fig. 4 visualises this gate sequence for one encoding and one variational layer. This layer structure is chosen as it is seen as highly expressive throughout the literature [32], [87]. We considered two types of circuits: In the first, we apply DRU and repeat the encoding pattern several times, which can increase quantum expressivity [75]. In the second, we omit the input encoding part after each input feature is present in the circuit once, that is, we do not apply DRU, resulting in a flatter circuit. Both variants are evaluated empirically in Sec. V-B.

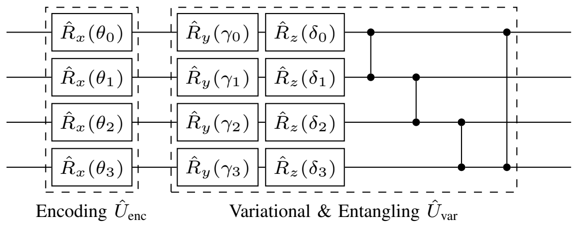


Fig. 4. Details of quantum state manipulation: Parametrised rotations around the x axis ($\hat{R}_x(\theta)$) encode information. The variational part comprises parametrised rotations around the y and z axes, implemented by $\hat{R}_y(\gamma)$ and $\hat{R}_z(\delta)$, followed by a cyclic sequence of C–$\hat{Z}$ gates that create entanglement.

V. Experiments

We commence with the experimental setup (fully reproducible with our reproduction package), followed by the training results for quantum-based versions of *ReJoin* in Sec. V-B.

A. Experimental Setup

1) Training and Test Data: Following the approaches presented in Sec. II that evaluate their methods using various industrial benchmark datasets [88]–[90], for classical *ReJoin*, we used the 113 queries from the join order benchmark (JOB) [88]. As we lack access to a sufficiently large quantum machine to process data for all queries in the JOB, we concentrate on training with four relations per query. Since the JOB only provides three queries with four relations, we generate new queries based on subplans to enlarge the dataset, following Krishnan *et al.* [16]. However, instead of obtaining subplans from the traditional optimiser, we rely on a single *ReJoin* training run, generating over 12 000 subqueries, from which we randomly select 497 that join four relations, and combine them with three JOB queries. To the resulting dataset of size 500, we apply a ten-fold cross-validation scheme [91], whereby the dataset is split into ten distinct parts. Each part is excluded from the training set once to be utilised for testing, leading to ten different train-test-splits.

2) Python Libraries: Since the original source code for *ReJoin* is not available, and other implementations for solving the JO problem by the means of RL [92]–[94] utilise different RL methods [95], and a different encoding for states and actions [93], [94] we modified and fine-tuned a third-party replication [96] based on the descriptions in Ref. [14] in collaboration with one of the original authors using the Python machine learning library *Tensorflow* [97] for the machine learning specific parts. For the quantum specific parts of our experiments, we additionally utilised the quantum frameworks *Tensorflow Quantum* [98] to simulate ideal quantum systems and *Qiskit* [99] to simulate noisy systems. Given the lack of capable quantum machines, we rely on simulations. The implementation can be found in our reproduction package.

3) Classical Baseline Replication: We were able to successfully replicate *ReJoin*, despite some minor deviations from the findings in Ref. [14], which could possibly attained to differing hyperparameters or settings that were not specified in the original study. To further enhance the outcomes of our replication, and to allow for the reduced encoding described in Par. IV-C2a, we combined methods from other RL approaches for JO [16], [19] to improve the learning convergence in cost training. For more information on the classical replication and the baseline modification, the reader is referred to the supplementary material in the reproduction package.

4) *PPO Models:* We consider the following configurations:

a) *Classical Model:* As baseline, we use a classical NN with two hidden layers (128 units each) for actor and critic.

b) *Quantum Model—Single-Step [46]:* This model uses one qubit per relation in the query, resulting in 4 qubits for our dataset. For each relation in the query, the ID of the relation is encoded with an $\hat{R}_x$ gate and the combined selectivity of all filters on the relation is encoded with a $\hat{R}_y$ gate. As there are at most 15 possible join orders for 4 relations, 2^4 quantum states are enough to have a state for each join order.

c) *Quantum Models—Multi-Step:* We consider three configurations: (a) *Q-Critic*, where a VQC is employed as the critic part of PPO, and a classical NN with the same dimensions as for the classical model serves as the actor; (b) *Q-Actor* with a VQC as actor in PPO, and classical critic; (c) *Fully Quantum* with VQCs for actor and critic. All quantum models use classical post-processing layer (see Sec. III-C2).

5) *Data Re-Uploading (DRU) Setup:* For each quantum model, we evaluate setups with and without DRU. The version utilising DRU employs 2–5 repetitions of the gates necessary to encode all input features once. With four relations this results in 8, 12, 16 and 20 variational layers for the multi-step QRL approach. To ensure a fair comparison with the single-step QML approach, we repeat the input features, consisting of indices and selectivities, every four variational layers for the configurations with single-step QML and DRU. This results in the same number of input repetitions and variational layers as for the multi-step QRL approach. For the second configurations without DRU, we use the same number of variational layers and introduce an additional experiment with four layers to encode every input feature once without extra variational layers for multi-step QRL. Analogously, the configurations for single-step QML and without DRU encode the input features once followed by the respective number of variational layers.

6) *Training and Evaluation:* While incorporating noise during training, whether through direct execution on real QPUs or via noisy simulations utilising snapshots of actual devices, provides the most accurate assessment of our approach's performance on present or near-term quantum hardware, the computational demands of noisy simulation, particularly for large input sizes during optimisation, are substantial. Given these constraints, a complete training iteration exceeds the scope of this study. Nonetheless, to quantify the adverse effects of noise, we assess models trained in an ideal simulation in a noisy environment using the same test sets from the tenfold cross-validation. Specifically, we introduce depolarising errors [78], a prevalent error type in noisy simulations, with a predetermined probability applied to each gate within the models utilising a quantum actor (*i.e.*, *Q-Actor* and *Fully Quantum*). For this probability, we select values ranging from 1% to 5%, representing upper bounds of gate errors, to which current QPUs are prone [25]. The findings from our noisy evaluation are detailed in V-B2.

B. Experimental Results

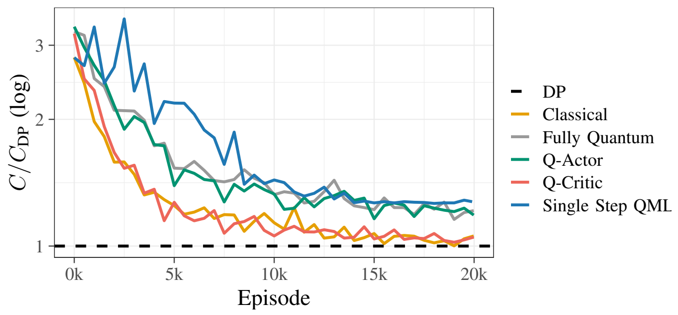


Fig. 5. Relative cost median during training. For the methods involving a quantum part the models with DRU and 20 variational layers are depicted.

1) *Training Results from Ideal Simulations:* As shown in Fig. 5, the classical model can achieve the median for optimal join orders after sampling roughly 8 000 queries (episodes), while Q-Critic delivers comparable results. Since the single-step approach surpasses conventional JO heuristics [43], [46] when trained on query execution times (*i.e.*, true cost), it can be regarded as quantum baseline. We either outperform or match it in all three QRL variants. Specifically, the Q-Critic configuration can achieve up to 17% lower median costs than single-step QML. This implies that although the configurations that employ a VQC as an actor, as well as the single-step QML method, do not achieve an optimal cost median during training, the QRL approaches are competitive with established classical heuristics, assuming that careful hyperparameter tuning and incorporating true costs leads to better join orders. Since our focus is on the specific implications for quantum computing, and as training on a cost model may not necessarily translate to actual query execution times, we consider costs as performance indicator, following Refs [14], [16], [19].

Our results suggest that as the classical component of computation increases, the quality of results improves. This finding appears to contradict claims for quantum advantage in QML literature [37], [38], [40]. However, it aligns with a recent observation by Bowles *et al.* [100] who conducted benchmarks across various QML configurations and noted that models with a substantial portion of classical parameters often outperform those with a higher quantum component. Understanding the dynamic between classical and quantum methods remains an important future challenge.

As illustrated in Fig. 6, the quantity of variational layers impacts configurations with a higher proportion of VQC parameters (Q-Actor, Single-Step QML and Fully Quantum QRL), especially with DRU. We observe, consistent with findings in the literature [32], [34], [42], [46] that more layers lead to lower costs. In all other instances, optimal training

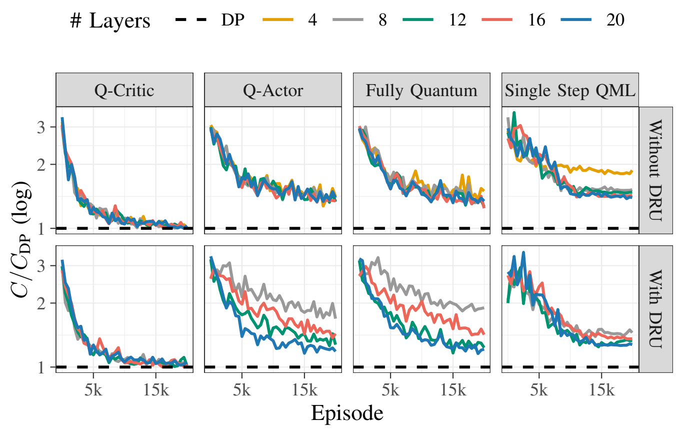


Fig. 6. Relative cost median during training.

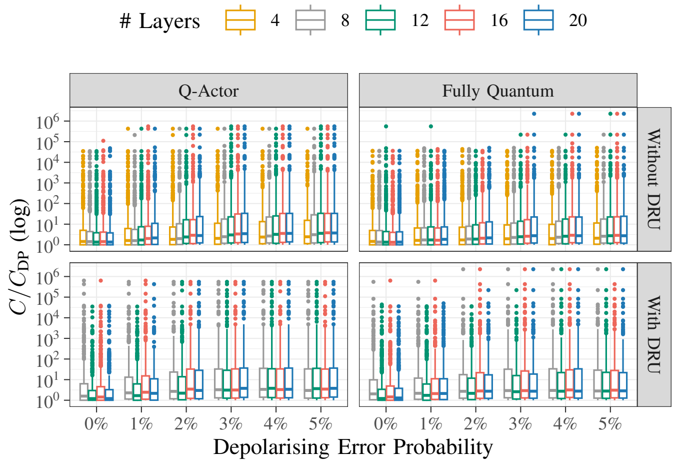


Fig. 8. Relative costs after training with different noise probabilities.

convergence is attainable with fewer variational layers, which translates to fewer parameters and shallower circuits.

2) Evaluation Results from Noisy Simulations: The configurations utilising a quantum actor (*i.e.*, *Q-Actor* and *Fully Quantum*) demonstrate the capability to achieve nearly optimal results conducted on ideally simulated QPUs. However, when incorporating gate errors, the performance of quantum models tends to deteriorate. Fig. 7 shows that the median relative cost increases almost linearly across all configurations, with steeper increases observed for deeper circuits—those with more layers and DRU–, which inherently present more opportunities for errors. While this observation is sobering, it aligns with the expectation that models trained in a ideal environment may struggle when confronted with noise. Other studies [32], [101] suggest that incorporating noise during training, coupled with hyperparameter tuning tailored to such noise models, can yield successful outcomes even in the presence of noise. The exploration of noise's impact during training on the JO problem could be deferred to future investigations. However, as shown in Fig. 8, a comprehensive examination of results reveals that significant outliers persist, even in ideal and classical scenarios, indicating that while median performance appears reasonable, pronounced instabilities persist within the (Q)RL approach to the JO problem, necessitating further theoretical and empirical investigation of the methods itself.

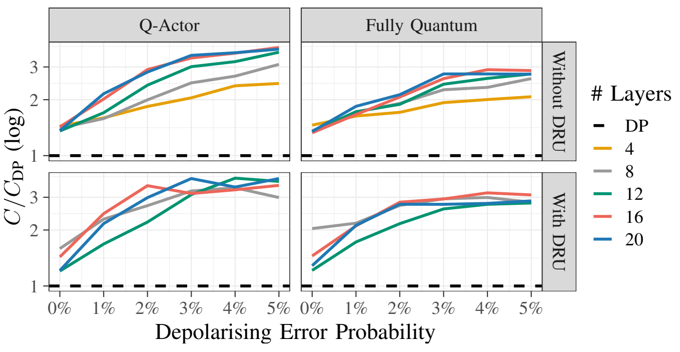


Fig. 7. Relative cost median after training with different noise probabilities.

VI. Evaluation

While the quantum models may not outperform classical models in terms of cost efficiency post-training, other factors are pertinent to assess the effectiveness of QML methods for JO. This section discusses the influence of the circuit dimensionality on overall trainability, examining the number of parameters and scalability of our approaches compared to alternative quantum approaches for the JO problem.

A. Parameter Efficiency

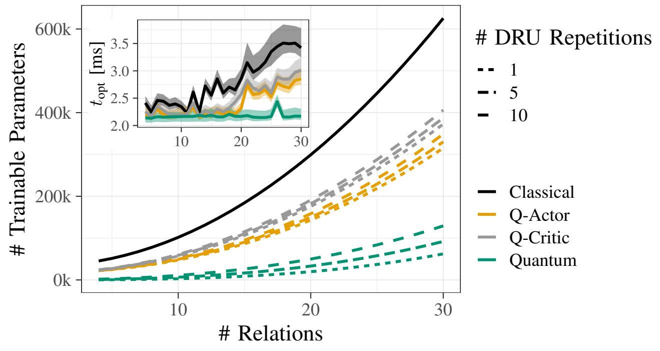


Fig. 9. Number of parameters for classical and quantum methods. The pure quantum case requires substantially less parameters than the classical baseline, which reduces optimisation complexity. Partial quantum variants (Q-actor, Q-critic) are less parsimonious, yet retain advantages against the baseline. The inset shows the optimisation time of the Adam [102] optimiser, to apply gradients to the parameters. Lines show the median of 1 000 measurements for each configuration; shaded areas depict first and third quartiles. Note that gradient calculation is excluded from our measurements.

Fig. 9 shows the total number of parameters (variational and classical) dependent on the number of relations in a query for the different methods. We considered the VQC structure with the dimensions described in Par. IV-C2b, including the classical post-processing layer and a fully-connected NN with two hidden layers and a constant hidden dimension of 128[4].

[4]Typically, the number of hidden units grows with input space [103], so the number of parameters for the classical model gives a lower bound.

While the classical baseline achieves lower and more stable costs, the QRL variants require fewer parameters. Considering the Q-Critic configuration, which achieves costs comparable to the baseline, we found that 47% less parameters suffice for four relations using 20 variational layers, that is, five DRU repetitions, and about 38% less parameters for 30 relations.

The corresponding run-times for the *Adam optimiser* [102] are shown in the inset of Fig. 9. We did not consider the time taken to calculate the gradients in our optimisation time measurements, as (1) gradient calculation or estimation methods for VQCs are still an ongoing area of research [79], [104], [105] and (2) our experiments are conducted on simulators instead of real QPUs, so the execution times may differ significantly. The parameter-shift rule [79], commonly used with VQCs, is computationally and necessitates two circuit executions per shot and parameter. Optimised techniques for gradient calculations have appeared [106]–[109], similar to classical ML over the past decades [110]. Yet, a comprehensive evaluation is beyond the scope of this paper. Based on our measurements, it is possible to achieve up to 12% improvement in median optimisation time for the Q-Critic configuration with one DRU repetition, in comparison to the classical model per optimisation step for 30 relations. As ML methods update parameters over multiple thousand iterations, this could significantly impact overall training time.

B. *Scalability of Quantum Approaches for Join Ordering*

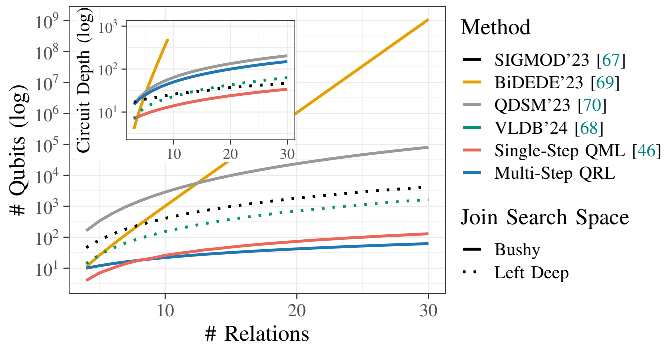


Fig. 10. Number of qubits and circuit depth required to encode the JO problem for different quantum optimisation strategies.

As outlined in Sec. II, other quantum-based techniques address the JO problem. The number of qubits necessary to encode up to 30 relations for each of these strategies is depicted in Fig. 10. Refs. [67]–[70] aim to solve a specific class of problems, namely quadratic unconstrained binary optimisation (QUBO) problems, where the number of qubits required depends on the QUBO formulation. In contrast, QML approaches provide greater flexibility in the utilisation of qubits and circuit depth. As shown in the figure, both proposed QML approaches, single-step QML and multi-step QRL, are more efficiently in terms of qubit numbers, compared to the QUBO approaches. Furthermore, circuit depth is a widely accepted quantum runtime proxy, for which we provide bounds[5] in Fig. 10. QRL generally requires only low circuit depth, comparable to the QUBO approach for bushy joins presented in Ref. [70]. However, not unlike with classical machine learning [111], while substantial progress with understanding capabilities of VQCs has been made [112], the learning dynamics based on the circuit dimension and theoretical underpinnings are not yet fully understood [44] and require further empirical and theoretical evaluation.

VII. Discussion and Outlook

We introduced a quantum reinforcement learning based approach to solve the long-standing, seminal join order problem, and replicated a classical reinforcement learning approach as suitable baseline to compare against the state of the art. In a systematic and comprehensive evaluation based on numerical simulation of quantum systems, we found that our approach at least matches classical performance in terms of result quality, which is not universally observed throughout the literature [45] for quantum algorithms. Apart from significantly reducing the input feature space of the classical baseline, we could show that substantially fewer trainable parameters are required, which is likely rooted in enhanced quantum expressivity. We believe the resulting reduction in classical optimisation efforts particularly benefits two scenarios: (a) Frequently changing data characteristics that necessitate continuous re-computation of join orders, and (b) low response latency requirements. Both appear in important commercial settings like stream data processing and high-frequency operation [113].

We also showed that our approach improves upon the scalability of existing quantum-RL solutions by nearly ten orders of magnitude in terms of qubit count. Given that this is the most scarce resource in current and future QPUs, we believe this is an important step towards practical utility.

Current NISQ capabilities prevents us from enjoying practical advantages right now. The limitations might, however, be circumvented even prior to the arrival of fully error-corrected hardware that is capable of delivering the behaviour predicted in our simulations by using custom-designed hardware. Additionally, it has recently been observed that the JO problem on quantum-inspired hardware can outperform established approaches [68]. Similar observations could generalise to other types of hardware, potentially applicable to the domain of variational algorithms or machine learning that our approach is based on. Finally, progress in the foundational understanding of QML could improve performance using more sophisticated quantum baseline methods, or data encoding strategies.

Acknowledgements MF, TW, SG and WM were supported by the German Federal Ministry of Education and Research (BMBF), funding program "Quantum Technologies—from Basic Research to Market", grants #13N15647 and #13NI6092 (MF and WM), and #13N16090 (TW and SG). WM acknowledges support by the High-Tech Agenda Bavaria.

[5]Bounds are based on circuit depth for one data uploading block (QML approaches) and lower bounds on the circuit depth for the respective QAOA [27] circuit with $p = 1$ (QUBO-based approaches). We consider the maximum number of entangling gates/quadratic terms that act on two qubits, and gates required for initialisation and mixer Hamiltonian.

References

[1] M. Steinbrunn, G. Moerkotte, and A. Kemper, "Heuristic and randomized optimization for the join ordering problem," *The VLDB Journal The Int. Journal on Very Large Data Bases*, vol. 6, no. 3, 1997.

[2] T. Neumann, "Query simplification: Graceful degradation for join-order optimization," in *Proc. of the 2009 ACM SIGMOD Int. Conf. on Management of data*, 2009.

[3] T. Neumann and B. Radke, "Adaptive optimization of very large join queries," in *Proc. of the 2018 Int. Conf. on Management of Data*, 2018.

[4] I. Trummer and C. Koch, "Solving the join ordering problem via mixed integer linear programming," in *Proc. of the 2017 ACM Int. Conf. on Management of Data*, 2017.

[5] W.-S. Han and J. Lee, "Dependency-aware reordering for parallelizing query optimization in multi-core cpus," in *Proc. of the 2009 ACM SIGMOD Int. Conf. on Management of data*, 2009.

[6] I. Kolchinsky and A. Schuster, *Join query optimization techniques for complex event processing applications*, 2018.

[7] F. A. Gonçalves, F. G. Guimarães, and M. J. Souza, "Query join ordering optimization with evolutionary multi-agent systems," *Expert Systems with Applications*, vol. 41, no. 15, 2014.

[8] V. Leis *et al.*, "Query optimization through the looking glass, and what we found running the join order benchmark," *The VLDB Journal*, vol. 27, 2018.

[9] G. Moerkotte. "Building query compilers." (2023), [Online]. Available: pi3.informatik.uni-mannheim.de/~moer/querycompiler.pdf.

[10] S. Cluet and G. Moerkotte, "On the complexity of generating optimal left-deep processing trees with cross products," in *Proc. of the 5th Int. Conf. on Database Theory*, 1995.

[11] A. Swami, "Optimization of large join queries: Combining heuristics and combinatorial techniques," in *Proc. ACM SIGMOD*, 1989.

[12] R. Krishnamurthy, H. Boral, and C. Zaniolo, "Optimization of nonrecursive queries," in *Proc. of the 12th Int. Conf. on Very Large Data Bases*, 1986.

[13] P. G. Selinger *et al.*, "Access path selection in a relational database management system," in *Proc. of the 1979 ACM SIGMOD Int. Conf. on Management of Data*, 1979.

[14] R. Marcus and O. Papaemmanouil, "Deep reinforcement learning for join order enumeration," in *Proc. of the 1st Int. Workshop on Exploiting Artificial Intelligence Techniques for Data Management*, 2018.

[15] R. Marcus *et al.*, "Neo: A learned query optimizer," *Proc. VLDB Endow.*, vol. 12, no. 11, 2019.

[16] S. Krishnan *et al.*, "Learning to optimize join queries with deep reinforcement learning," 2018.

[17] I. Trummer *et al.*, "Skinnerdb: Regret-bounded query evaluation via reinforcement learning," *ACM Trans. Database Syst.*, vol. 46, no. 3, 2021.

[18] J. Chen *et al.*, "Efficient join order selection learning with graph-based representation," in *Proc. of the 28th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining*, 2022.

[19] X. Yu *et al.*, "Reinforcement learning with tree-lstm for join order selection," in *2020 IEEE 36th Int. Conf. on Data Engineering (ICDE)*, 2020.

[20] J. Wang *et al.*, *Adopt: Adaptively optimizing attribute orders for worst-case optimal join algorithms via reinforcement learning*, 2023.

[21] L. Ji *et al.*, "Query join order optimization method based on dynamic double deep q-network," *Electronics*, vol. 12, no. 6, 2023.

[22] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Review*, vol. 41, no. 2, 1999.

[23] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proc. of the twenty-eighth annual ACM symposium on Theory of computing*, 1996.

[24] J. Preskill, "Quantum Computing in the NISQ era and beyond," *Quantum*, vol. 2, 2018.

[25] F. Greiwe, T. Krüger, and W. Mauerer, "Effects of imperfections on quantum algorithms: A software engineering perspective," en, in *2023 IEEE Int. Conf. on Quantum Software (QSW)*, 2023.

[26] N. Pirnay *et al.*, *An in-principle super-polynomial quantum advantage for approximating combinatorial optimization problems*, 2023.

[27] E. Farhi, J. Goldstone, and S. Gutmann, *A quantum approximate optimization algorithm*, 2014.

[28] J. R. McClean *et al.*, "The theory of variational hybrid quantum-classical algorithms," *New Journal of Physics*, vol. 18, no. 2, 2016.

[29] M. Cerezo *et al.*, "Variational quantum algorithms," *Nature Reviews Physics*, vol. 3, no. 9, 2021.

[30] T. Hoefler, T. Häner, and M. Troyer, "Disentangling hype from practicality: On realistically achieving quantum advantage," *Commun. ACM*, vol. 66, no. 5, 2023.

[31] S. Y.-C. Chen *et al.*, "Variational quantum circuits for deep reinforcement learning," *IEEE Access*, vol. 8, 2020.

[32] A. Skolik, S. Jerbi, and V. Dunjko, "Quantum agents in the Gym: A variational quantum algorithm for deep Q-learning," *Quantum*, vol. 6, 2022.

[33] M. Franz *et al.*, "Uncovering instabilities in variational-quantum deep q-networks," *Journal of the Franklin Institute*, 2022.

[34] R. Dilip *et al.*, "Data compression for quantum machine learning," *Phys. Rev. Res.*, vol. 4, 4 2022.

[35] H.-Y. Huang *et al.*, "Quantum advantage in learning from experiments," *Science*, vol. 376, no. 6598, 2022.

[36] Y. Du *et al.*, "Expressive power of parametrized quantum circuits," *Phys. Rev. Res.*, vol. 2, 3 2020.

[37] H.-Y. Huang *et al.*, "Power of data in quantum machine learning," *Nature Communications*, vol. 12, no. 1, 2021.

[38] Y. Liu, S. Arunachalam, and K. Temme, "A rigorous and robust quantum speed-up in supervised machine learning," *Nature Physics*, vol. 17, no. 9, 2021.

[39] R. Sweke *et al.*, "On the Quantum versus Classical Learnability of Discrete Distributions," *Quantum*, vol. 5, 2021.

[40] V. Havlíek *et al.*, "Supervised learning with quantum-enhanced feature spaces," *Nature*, vol. 567, no. 7747, 2019.

[41] O. Lockwood and M. Si, "Reinforcement learning with quantum variational circuit," *Proc. of the AAAI Conf. on Artificial Intelligence and Interactive Digital Entertainment*, vol. 16, no. 1, 2020.

[42] S. Jerbi *et al.*, *Parametrized quantum policies for reinforcement learning*, 2021.

[43] U. Çalikyilmaz *et al.*, "Opportunities for quantum acceleration of databases: Optimization of queries and transaction schedules," *Proc. VLDB Endow.*, vol. 16, no. 9, 2023.

[44] M. Schuld and N. Killoran, "Is quantum advantage the right goal for quantum machine learning?" *PRX Quantum*, 2022.

[45] N. Meyer *et al.*, *A survey on quantum reinforcement learning*, 2022.

[46] T. Winker *et al.*, "Quantum machine learning for join order optimization using variational quantum circuits," in *Proc. of the Int. Workshop on Big Data in Emergent Distributed Environments*, 2023.

[47] W. Mauerer and S. Scherzinger, "1-2-3 reproducibility for quantum software experiments," in *IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, 2022.

[48] A. Meister and G. Saake, "GPU-accelerated dynamic programming for join-order optimization," 2020.

[49] G. Moerkotte and T. Neumann, "Analysis of two existing and one new dynamic programming algorithm for the generation of optimal bushy join trees without cross products," in *Proc. VLDB Endow.*, ser. VLDB '06, 2006.

[50] B. Vance and D. Maier, "Rapid bushy join-order optimization with cartesian products," in *Proc. of the 1996 ACM SIGMOD Int. Conf. on Management of Data*, ser. SIGMOD '96, 1996.

[51] G. Moerkotte and T. Neumann, "Dynamic programming strikes back," in *Proc. of the 2008 ACM SIGMOD Int. Conf. on Management of Data*, ser. SIGMOD '08, Association for Computing Machinery, 2008.

[52] J.-T. Horng, C.-Y. Kao, and B.-J. Liu, "A genetic algorithm for database query optimization," in *Proceedings of the 1st IEEE Conf. on Evolutionary Computation. IEEE World Congress on Computational Intelligence*, 1994.

[53] N. Bruno, C. Galindo-Legaria, and M. Joshi, "Polynomial heuristics for query optimization," in *2010 IEEE 26th Int. Conf. on Data Engineering (ICDE 2010)*, 2010.

[54] Y. E. Ioannidis and Y. Kang, "Randomized algorithms for optimizing large join queries," *SIGMOD Rec.*, vol. 19, no. 2, 1990.

[55] I. Trummer and C. Koch, "Parallelizing query optimization on shared-nothing architectures," *Proc. VLDB Endow.*, vol. 9, no. 9, 2016.

[56] Y. Han *et al.*, *Cardinality estimation in DBMS: A comprehensive benchmark evaluation*, 2021.
[57] K. Kim *et al.*, "Learned cardinality estimation: An in-depth study," in *Proc. of the 2022 Int. Conf. on Management of Data*, 2022.
[58] R. Hasan and F. Gandon, "A machine learning approach to sparql query performance prediction," in *IEEE/WIC/ACM Int. Joint Conf.s on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, IEEE, vol. 1, 2014.
[59] M. Akdere *et al.*, "Learning-based query performance modeling and prediction," in *2012 IEEE 28th Int. Conf. on Data Engineering*, IEEE, 2012.
[60] I. Trummer and C. Koch, "Multiple query optimization on the d-wave 2x adiabatic quantum computer," *Proc. VLDB Endow.*, vol. 9, no. 9, 2016.
[61] T. Winker *et al.*, "Quantum machine learning: Foundation, new techniques, and opportunities for database research," in *Companion of the 2023 Int. Conf. on Management of Data*, 2023.
[62] S. Groppe and J. Groppe, "Optimizing transaction schedules on universal quantum computers via code generation for grovers search algorithm," in *25th Int. Database Engineering & Applications Symposium*, 2021.
[63] T. Bittner and S. Groppe, "Hardware accelerating the optimization of transaction schedules via quantum annealing by avoiding blocking," *Open Journal of Cloud Computing (OJCC)*, vol. 7, no. 1, 2020.
[64] T. Bittner and S. Groppe, "Avoiding blocking by scheduling transactions using quantum annealing," in *Proc. of the 24th Symposium on Int. Database Engineering & Applications*, 2020.
[65] K. Fritsch and S. Scherzinger, "Solving hard variants of database schema matching on quantum computers," *Proc. VLDB Endow.*, vol. 16, no. 12, 2023.
[66] L. Gruenwald *et al.*, "Index tuning with machine learning on quantum computers for large-scale database applications," in *Proc. of QDSM@VLDB23*, 2023.
[67] M. Schönberger, S. Scherzinger, and W. Mauerer, "Ready to leap (by co-design)? Join order optimisation on quantum hardware," 1, vol. 1, 2023.
[68] M. Schönberger, I. Trummer, and W. Mauerer, "Quantum-inspired digital annealing for join ordering," 3, vol. 17, 2023.
[69] N. Nayak *et al.*, "Constructing optimal bushy join trees by solving qubo problems on quantum hardware and simulators," in *Proc. of BiDEDE@SIGMOD23*, 2023.
[70] M. Schönberger, I. Trummer, and W. Mauerer, "Quantum optimisation of general join trees," in *Proc. of QDSM@VLDB23*, 2023.
[71] J. M. Smith and P. Y.-T. Chang, "Optimizing the performance of a relational algebra database interface," *Communications of the ACM*, vol. 18, no. 10, 1975.
[72] R. K. Kurella, "Systematic literature review: Cost estimation in relational databases," M.S. thesis, University of Magdeburg, 2018.
[73] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. 2018.
[74] J. Schulman *et al.*, *Proximal policy optimization algorithms*, 2017.
[75] A. Pérez-Salinas *et al.*, "Data re-uploading for a universal quantum classifier," *Quantum*, vol. 4, 2020.
[76] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, 1989.
[77] K. Mitarai *et al.*, "Quantum circuit learning," *Phys. Rev. A*, vol. 98, 3 2018.
[78] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information - 10th Anniversary Edition*. 2010.
[79] M. Schuld *et al.*, "Evaluating analytic gradients on quantum hardware," *Phys. Rev. A*, vol. 99, 3 2019.
[80] M. Weigold *et al.*, "Data encoding patterns for quantum computing," in *Proc. of the 27th Conf. on Pattern Languages of Programs*, 2022.
[81] M. Weigold *et al.*, "Encoding patterns for quantum algorithms," *IET Quantum Communication*, vol. 2, no. 4, 2021.
[82] M. Periyasamy *et al.*, "Incremental data-uploading for full-quantum classification," in *IEEE Int. Conf. on Quantum Computing and Engineering (QCE)*, 2022.
[83] M. Schuld, R. Sweke, and J. J. Meyer, "Effect of data encoding on the expressive power of variational quantum-machine-learning models," *Phys. Rev. A*, vol. 103, 3 2021.
[84] N. Meyer *et al.*, "Quantum policy gradient algorithm with optimized action decoding," in *Proc. of the 40th Int. Conf. on Machine Learning*, A. Krause *et al.*, Eds., vol. 202, 2023.
[85] O. Lockwood and M. Si, "Playing atari with hybrid quantum-classical reinforcement learning," in *NeurIPS 2020 Workshop on Pre-registration in Machine Learning*, L. Bertinetto *et al.*, Eds., vol. 148, 2021.
[86] A. Laud and G. DeJong, "The influence of reward on the speed of reinforcement learning: An analysis of shaping," in *Proc. of the 20th Int. Conf. on Machine Learning (ICML-03)*, 2003.
[87] A. Kandala *et al.*, "Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets," *Nature*, vol. 549, no. 7671, 2017.
[88] V. Leis *et al.*, "How good are query optimizers, really?" *Proc. VLDB Endow.*, vol. 9, no. 3, 2015.
[89] M. Poess *et al.*, "TPC-DS, taking decision support benchmarking to the next level," in *Proc. of the 2002 ACM SIGMOD Int. Conf. on Management of Data*, 2002.
[90] M. Poess and R. Nambiar, "Tpc benchmark h standard specification," 2010.
[91] T. Fushiki, "Estimation of prediction error by using k-fold cross-validation," *Statistics and Computing*, vol. 21, 2011.
[92] X. Yu *et al.* "Github repository: AI4DBCode." Commit: a8989bfa. (2022), [Online]. Available: https : / / github . com / TsinghuaDatabaseGroup/AI4DBCode.
[93] Z. Yang *et al.*, "Balsa: Learning a query optimizer without expert demonstrations," in *Proc. of the 2022 Int. Conf. on Management of Data*, 2022.
[94] R. Marcus *et al.*, "Bao: Making learned query optimization practical," in *Proc. of the 2021 Int. Conf. on Management of Data*, 2021.
[95] V. Mnih *et al.*, *Playing atari with deep reinforcement learning*, 2013.
[96] G. Xintong and A. Mandamadiotis. "Github repository: Rejoin." Commit: 02365ab0. (2021), [Online]. Available: https://github.com/GUOXINTONG/rejoin.
[97] Martín Abadi, Ashish Agarwal, Paul Barham, *et al.*, *TensorFlow: Large-scale machine learning on heterogeneous systems*.
[98] M. Broughton *et al.*, *Tensorflow quantum: A software framework for quantum machine learning*, 2021.
[99] Qiskit contributors, *Qiskit: An open-source framework for quantum computing*, 2023.
[100] J. Bowles, S. Ahmed, and M. Schuld, *Better than classical? the subtle art of benchmarking quantum machine learning models*, 2024.
[101] K. Borras *et al.*, "Impact of quantum noise on the training of quantum generative adversarial networks," *Journal of Physics: Conf. Series*, vol. 2438, no. 1, 2023.
[102] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2017.
[103] K. G. Sheela and S. N. Deepa, "Review on methods to fix number of hidden neurons in neural networks," *Mathematical Problems in Engineering*, vol. 2013, 2013.
[104] D. Wierichs *et al.*, "General parameter-shift rules for quantum gradients," *Quantum*, vol. 6, 2022.
[105] A. Gilyén, S. Arunachalam, and N. Wiebe, "Optimizing quantum optimization algorithms via faster quantum gradient computation," in *Proc. of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms*, SIAM, 2019.
[106] M. Periyasamy *et al.*, "Guided-spsa: Simultaneous perturbation stochastic apprimation assisted by the parameter shift rule," in *IEEE Int. Conf. on Quantum Computing and Engineering (QCE)*, 2024.
[107] J. Stokes *et al.*, "Quantum natural gradient," *Quantum*, vol. 4, 2020.
[108] L. Bittel, J. Watty, and M. Kliesch, *Fast gradient estimation for variational quantum algorithms*, 2022.
[109] J. Spall, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," *IEEE Transactions on Automatic Control*, vol. 37, no. 3, 1992.
[110] I. H. Sarker, "Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions," *SN Computer Science*, vol. 2, no. 6, 2021.
[111] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (Adaptive computation and machine learning). 2016.
[112] J. Landman *et al.*, *Classically approximating variational quantum machine learning with random fourier features*, 2022.
[113] J. Dean and L. A. Barroso, "The tail at scale," *Commun. ACM*, vol. 56, no. 2, 2013.