Technical Explanation of QCardEst and QCardCorr: Hybrid Quantum–Classical Cardinality Estimation

Based on *QCardEst/QCardCorr: Quantum Cardinality Estimation and Correction*[3]

January 11, 2026

Abstract

This document explains the technical mechanism of the two models proposed in the paper *QCardEst/QCardCorr: Quantum Cardinality Estimation and Correction* [3]. Both models are query-driven hybrid quantum–classical regressors built from three components: (i) a compact query encoding that maps an SQL join query to an n-qubit quantum state using one qubit per table, (ii) a variational quantum circuit (VQC) that transforms the encoded state using trainable parameters, and (iii) a classical post-processing layer that converts measurement probabilities into a single real-valued output. QCardEst predicts a query cardinality directly; QCardCorr predicts a multiplicative correction factor for an existing classical estimator.

Contents

1 Problem Setting

Cardinality estimation predicts the number of rows produced by a query plan operator (especially joins). These estimates steer join order selection and physical operator choices. Classical estimators often rely on simplifying assumptions (for example, independence assumptions) and can exhibit large errors on correlated data or complex predicates. The paper proposes two quantum machine learning models to learn a mapping from query features to cardinality values in a query-driven manner.

2 Related Work (Positioning)

The authors of the paper [3] contrast their approach with two prior quantum cardinality estimators:

- **SQL2Circuits** [2] encodes query syntax via grammar-based compilation into parameterized circuits and performs classification into cardinality bins rather than regression.
- **QardEst** [1] replaces parts of MSCN with many small quantum circuits and pools their outputs; it inherits a high-dimensional encoding and requires multiple circuits per query.

A key design goal of QCardEst/QCardCorr is feasibility on near-term devices: process a complete join query with a single VQC, which motivates a compact encoding.

3 Quantum Background and Hybrid Learning

3.1 From Classical Queries to Quantum States

Quantum computers operate on qubits rather than classical numbers. A single qubit has two computational basis states, $|0\rangle$ and $|1\rangle$, but it can also exist in a superposition of both:

$$|\psi\rangle = a_0|0\rangle + a_1|1\rangle, \quad \text{with } |a_0|^2 + |a_1|^2 = 1. \tag{1}$$

The coefficients a_0 and a_1 are called *amplitudes.* Measuring the qubit collapses the state to $|0\rangle$ with probability $|a_0|^2$ or to $|1\rangle$ with probability $|a_1|^2$.

When we move from a single qubit to multiple qubits, an important property emerges. A system of n qubits is described by a joint quantum state with 2^n amplitudes:

$$|\psi\rangle = \sum_{i=0}^{2^n-1} a_i|i\rangle, \quad \sum_{i=0}^{2^n-1} |a_i|^2 = 1. \tag{2}$$

This exponential state space allows a small number of qubits to represent a large number of joint configurations at the same time. Our goal is to define such a quantum state so that its amplitudes encode the structure of an SQL query.[3]

4 Compact Query Encoding

4.1 Design Principle

Not every part of an SQL query affects the result cardinality. Encoding unnecessary information would waste qubits and circuit depth, which is infeasible on current quantum hardware.

The compact encoding therefore follows a strict rule: *only encode query components that influence cardinality.*

Table 1 summarizes which query parts are relevant.

Query Part	Affects Cardinality	Encoding Decision
Selected columns	No	Ignored
Tables	Yes	Each table gets an ID t_i
Join conditions	Yes	Ignored under PK–FK assumption
Filter predicates	Yes	Aggregated into selectivity s_i

Table 1: Compact encoding only retains query parts relevant for cardinality.

4.2 Reducing a Query to $2n$ Classical Values

Consider a join query over n tables. After removing irrelevant components, the query is reduced to exactly two values per table:

- a table identifier t_i
- a selectivity value $s_i \in [0, 1]$ representing the fraction of rows that remain after filtering

All filters on the same table are combined into a single selectivity. As a result, a query over n tables is reduced to $2n$ classical values:

$$(t_0, s_0), (t_1, s_1), \ldots, (t_{n-1}, s_{n-1}).$$

This reduction step is essential because it ensures that the size of the quantum input grows linearly with the number of tables.

4.3 Encoding Two Values into One Qubit

Each table is assigned exactly one qubit. Two values are encoded on the same qubit using rotation gates around different axes of the Bloch sphere.

- the table identifier t_i is encoded using a rotation around the x-axis
- the selectivity s_i is encoded using a rotation around the z-axis

Both values are first scaled so that the resulting rotation angles remain strictly below 2π.

Formally, the encoding applied to qubit i is

$$U_e^{(i)}(t_i, s_i) = R_x(\alpha_i)\, R_z(\beta_i), \tag{3}$$

where α_i encodes the table identifier and β_i encodes the selectivity.

The encoding of the full query is the joint n-qubit operation

$$U_e(x) = \prod_{i=1}^{n} R_x(\alpha_i) R_z(\beta_i). \tag{4}$$

The product denotes that each pair of rotations is applied independently to its corresponding qubit.

This means:

- one qubit per table
- one rotation gate per encoded value
- linear scaling in the number of tables

4.4 Why the Encoding Is Compact

The proposed encoding satisfies all required properties for near-term feasibility [3]:

- **Uniqueness:** Different (t_i, s_i) pairs map to different rotation angles, which makes the encoding injective.
- **Qubit complexity:** Exactly one qubit per table is used.
- **Circuit depth:** Each value is encoded with a single rotation gate.

As a result, the ratio between tables and qubits is exactly 1:1, allowing a complete SQL query to be processed by a single variational quantum circuit.[3]

5 Variational Quantum Circuit Architecture

5.1 Encoding Layer

The encoding layer $U_e(x)$ initializes the quantum system. It performs the compact query encoding and creates an initial superposition whose amplitudes reflect the structure of the query.

5.2 Computation Layer

The computation layer $U(\theta)$ contains the trainable parameters. It combines:

- parameterized single-qubit rotation gates
- controlled Pauli gates that entangle qubits

The purpose of this layer is to learn correlations between tables. Entanglement is crucial here because join cardinalities depend on interactions between tables rather than independent behavior.

5.3 Measurement Layer

Measurement collapses the quantum superposition into classical bit strings. For n qubits, each measurement yields an n-bit outcome, and repeating the circuit produces a probability vector

$$\boldsymbol{x} \in [0,1]^{2^n}, \quad \sum_i x_i = 1.$$

This probability vector captures the information learned by the quantum model and is passed to the classical post-processing layer.[3]

6 Classical Post-Processing Layer

The variational quantum circuit outputs a probability vector $\boldsymbol{x}$ whose entries lie in $[0,1]$ and sum to 1. Cardinality estimation, however, requires a single real-valued scalar. The classical post-processing layer therefore acts as a decoder that maps the quantum measurement output to a numeric estimate.

Different post-processing functions induce different output ranges and numeric distributions, which significantly influences estimation accuracy. For this reason, the paper evaluates multiple post-processing strategies rather than committing to a single fixed mapping. We briefly summarize the intuition behind each family below and refer to the original paper for the exact mathematical definitions.[3]

6.1 Linear Mapping

The linear layer produces a cardinality estimate by scaling a single probability component. It is simple and stable, but its output range is bounded, which can limit expressiveness for large cardinalities or strong corrections.

6.2 Rational and RationalLog Mappings

Rational mappings compute a ratio between probability components. This allows the output to grow or shrink dynamically and supports unbounded ranges. The logarithmic variant maps ratios to a symmetric real scale, which is particularly useful when modeling multiplicative correction factors that may be greater or smaller than one.

6.3 Threshold-Based Mappings

Threshold-based layers introduce conditional behavior. One probability component acts as a gate that determines whether another component contributes to the output. This results in piecewise behavior, enabling the model to represent abrupt changes or regime shifts in cardinality, such as systematic underestimation or overestimation patterns.

6.4 Place-Value Mappings

Place-value mappings interpret multiple probability components as digits in a positional number system. By combining several outputs with increasing weights, these layers can represent a wide numeric range. Extended variants allow negative contributions, which is especially important for correction tasks where the output may need to scale estimates downward.

7 Model 1: QCardEst (Direct Cardinality Estimation)

QCardEst is a query-driven hybrid quantum–classical regression model that predicts the cardinality of an SQL join query directly. It builds on the components introduced in the previous sections: compact query encoding, a variational quantum circuit, and a classical post-processing layer.

Given a query q, the encoding and VQC produce a probability vector $\boldsymbol{x}(q;\theta)$. The classical post-processing layer then maps this vector to a scalar cardinality estimate $\hat{t}(q)$:

$$\hat{t}(q) = h(\boldsymbol{x}(q;\theta);\phi), \tag{5}$$

where θ denotes the trainable parameters of the quantum circuit and ϕ denotes the parameters of the classical post-processing function (for example scaling constants or thresholds).

Training objective. QCardEst is trained on a workload Q of queries with known true cardinalities $t(q)$. The parameters of both the quantum circuit and the classical layer are optimized jointly by minimizing a regression loss:

$$\min_{\theta,\phi} \sum_{q\in Q} e\big(\hat{t}(q),\, t(q)\big). \tag{6}$$

Feasibility. A central benefit of QCardEst is its feasibility on near-term quantum hardware. As shown in section 2 and the compact encoding section, a query joining n tables is processed using exactly n qubits and a single variational quantum circuit. This distinguishes QCardEst from prior quantum approaches that require multiple circuits or encodings whose size grows with query complexity.[3]

8 Model 2: QCardCorr (Cardinality Correction Factor)

QCardCorr addresses the same task as QCardEst but follows a different modeling strategy. Instead of predicting cardinalities from scratch, it learns to correct the output of an existing classical estimator.

Let $f(q)$ denote the cardinality estimate produced by a classical query optimizer (for example PostgreSQL), and let $t(q)$ be the true result. QCardCorr predicts a multiplicative correction factor $g_x(q)$ such that

$$\hat{t}(q) = f(q) \cdot g_x(q). \tag{7}$$

Training objective. During training, the classical estimator f is kept fixed. Only the parameters x of the quantum correction model are optimized:

$$\min_x \sum_{q \in Q} e\big(f(q) \cdot g_x(q),\, t(q)\big). \tag{8}$$

The optimal correction factor for a query q is

$$g^*(q) = \frac{t(q)}{f(q)}, \tag{9}$$

which means that training can be interpreted as learning to approximate the ratio between the true and the estimated cardinality.

Why correction is easier than direct estimation. Correction factors live in $\mathbb{Q}^+$ and are centered around 1 for a reasonably accurate baseline estimator. This substantially simplifies the learning task compared to predicting absolute cardinalities. To further stabilize training, the authors of the paper [3] apply a logarithmic transformation, mapping correction factors to $\mathbb{R}$ and assigning the neutral value 0 to perfectly corrected predictions.

9 End-to-End Example (Running Query)

To illustrate how all components interact, we consider a single running example throughout the pipeline:

```sql
SELECT *
FROM Movies M, Ratings R, Users U
WHERE M.id = R.movie_id
  AND R.user_id = U.id
  AND M.year > 2000
  AND U.country = 'Germany';
```

Encoding. The query joins three tables, so the compact encoding uses three qubits. Tables (M, R, U) are mapped to identifiers (t_0, t_1, t_2), and all filters on each table are aggregated into selectivities (s_0, s_1, s_2). Each qubit i encodes its corresponding (t_i, s_i) pair using rotations $R_x(\alpha(t_i))$ and $R_z(\beta(s_i))$.

VQC and measurement. The encoded quantum state is processed by the variational quantum circuit, which applies parameterized rotations and entangling gates to learn correlations between tables. After repeated circuit executions, measurement yields a probability distribution over all computational basis states.

For a query encoded using $n = 3$ qubits, the measurement output is a probability vector of dimension $2^3 = 8$. For illustration, consider the following example output:

$$\boldsymbol{x} = [0.18,\ 0.07,\ 0.12,\ 0.21,\ 0.05,\ 0.14,\ 0.09,\ 0.14], \tag{10}$$

where each entry corresponds to one of the eight possible basis states and the entries sum to one.

Classical decoding. The probability vector is converted into a scalar using the selected post-processing layer. A linear layer might scale a single component, a rational layer might form a ratio of components, and a place-value layer would combine multiple components as weighted digits.

Correction view. Assume a classical optimizer predicts $f(q) = 3$ rows, while the true cardinality is $t(q) = 4$. The optimal correction factor for this query is therefore $g^*(q) = t(q)/f(q) = 4/3 \approx 1.33$.

In QCardCorr, the query is encoded and processed by the same VQC architecture described above. After measurement, the circuit produces a probability vector $\boldsymbol{x}(q)$, which is mapped by the classical post-processing layer to a predicted correction factor $g(q)$. If the model outputs $g(q) \approx 1.30$, the final estimate becomes $f(q) \cdot g(q) \approx 3.9$, which is much closer to the true cardinality than the original estimate.

This illustrates how QCardCorr uses the quantum model to learn and apply systematic corrections to a fixed classical estimator.

10 Implementation and Training

Both QCardEst and QCardCorr are trained using the same hybrid quantum–classical workflow. They share the same model components and differ only in what the model is trained to predict: an absolute cardinality value in QCardEst, and a multiplicative correction factor in QCardCorr.

Training loop. Each training iteration follows the same sequence of steps:

1. extract the query features (t_i, s_i) and encode them into qubits using the compact encoding $U_e(x)$,
2. apply the variational computation layer $U(\theta)$ with parameterized rotations and entangling gates,
3. measure the circuit repeatedly to obtain a probability vector $\boldsymbol{x}$,
4. convert $\boldsymbol{x}$ into a scalar using the selected classical post-processing function,
5. compute the loss and update parameters with a classical optimizer.

This loop is executed on classical hardware using a quantum circuit simulator. A detailed empirical evaluation, including benchmarks and quantitative results, is reported separately in the document *Evaluation of QCardEst/QCardCorr on JOB-light and STATS Benchmarks.*

Summary. QCardEst and QCardCorr rely on the same hybrid architecture consisting of compact query encoding, a variational quantum circuit, and a classical post-processing layer. QCardEst learns to predict cardinalities directly, while QCardCorr learns how to adjust an existing estimator. The compact encoding is the main factor that makes both approaches practical, as it allows complete SQL join queries to be processed using a single quantum circuit on near-term hardware.[3]

References

[1] Florian Kittelmann, Pavel Sulimov, and Kurt Stockinger. “QardEst: Using Quantum Machine Learning for Cardinality Estimation of Join Queries”. In: *Proceedings of the Q-Data Workshop (SIGMOD/PODS)*. ACM, 2024, pp. 2–13. DOI: 10.1145/3665225.3665444.

[2] Valter Uotila. “SQL2Circuits: Estimating Metrics for SQL Queries with a Quantum Natural Language Processing Method”. In: *arXiv preprint arXiv:2306.08529* (2023). arXiv: 2306.08529 [cs.DB]. URL: https://arxiv.org/abs/2306.08529.

[3] Tobias Winker, Jinghua Groppe, and Sven Groppe. “QCardEst/QCardCorr: Quantum Cardinality Estimation and Correction”. In: *Proceedings of the ACM SIGMOD Workshops*. New York, NY, USA: ACM, 2025.