dc-qml: data-centric quantum machine learning

Everson da Silva Flores[1], Kauã Dalla Riva Cucco Barbosa[1], Bruna de Vargas Guterres[2], Silvia Silva da Costa Botelho[1] and Marcelo Rita Pias[1]

[1]Computer Science Center (C3) – Federal University of Rio Grande (FURG)

[2]Technological University of Uruguay (UTEC)

{eversonflores,kaua,silviacb,mpias}@furg.br, bruna.guterres@utec.edu.uy

Abstract. *Quantum machine learning research (QML) has largely followed a model-centric perspective that focuses on circuit design and algorithm refinement. However, near-term quantum hardware constraints place equal importance on how data are prepared and encoded before quantum processing. This short paper presents preliminary work on a data-centric quantum machine learning methodology termed* ***dc-qml****. The approach structures data ingestion, tokenization, encoding and synthetic data generation within a unified data-to-model workflow. The goal is to explore how systematic data preparation can improve the performance of QML systems. This work in progress is intended to generate discussion at the first SBCCQ workshop.*

1. Introduction

Quantum machine learning (QML) integrates quantum computation into the machine learning workflow to improve problem-solving capacity and expand the range of problems that can be addressed [Hong and Josh Domingo Lopez 2025]. Research in this area has primarily focused on algorithm design and the refinement of quantum circuit architectures. Current QML approaches have been validated on noisy intermediate-scale quantum (NISQ) hardware, where quantum gates, qubit connectivity and coherence times create constraints on problem sizes. Such constraints cause a bottleneck in data encoding and loading high-dimensional data into a small number of qubits often dominates the computational cost of the algorithm. In fact, the encoding strategy affects both the number of qubits and the required circuit depth, which in turn determine the accuracy of the system outputs and the extent of resource usage [Raisa et al. 2023, Yang et al. 2023]. Well-designed data representations reduce the burden on quantum algorithms and align the dataset with the geometry of the quantum state space. Optimizing data handling provides measurable improvements to address these constraints [Hong and Josh Domingo Lopez 2025].

This short paper presents work in progress that examines how a data-centric approach can improve the quality and scalability of QML algorithms. Two contributions are presented in this work. First, the *dc-qml* methodology is introduced as a data-to-model design workflow that integrates data ingestion, tokenization, encoding and synthetic data generation. Second, the paper briefly discusses data-oriented mechanisms that include token representations, QRAC based compression and generative models for dataset expansion.

2. QML data requirements

Quantum machine learning algorithms impose constraints on how classical data are represented before quantum processing. Methods such as quantum support

vector machines [Havlíček et al. 2019], quantum neural networks [Yang et al. 2023], hybrid classical quantum models and quantum generative adversarial networks [Hong and Josh Domingo Lopez 2025] require input data to be numerically scaled, vectorized and compatible with the limits associated with circuit depth, qubit count and noise. To meet these constraints, dimensionality reduction and bounded numerical embeddings are commonly applied before quantum encoding. These transformations reduce encoding overhead and maintain compatibility between classical preprocessing stages and quantum circuit execution.

3. Data-centric QML design: models and data

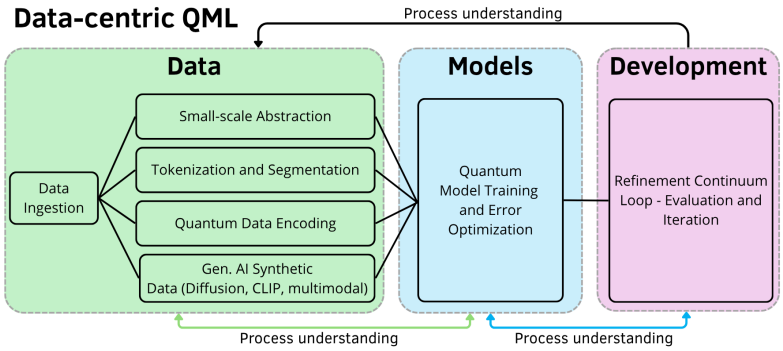

Figura 1. Evaluation and iteration in data-centric QML (*dc-qml*). Unlike model-centric workflows, *dc-qml* links model evaluation back to data preparation.

QML adopts a model-centric design approach by default that focuses on architecture adjustments, hyperparameter tuning and the definition of alternative loss functions. By contrast, a data-centric design approach treats data preparation, encoding and broader data handling as equally important. It focuses on the properties of the data used for model training and inference, and recognizes that iterative refinement is required to obtain more accurate models. This approach is aligned with data-centric principles in classical machine learning, which prioritize data quality and representation over architectural adjustments [Jarrahi et al. 2023]. The design space of data-centric QML (*dc-qml*) is therefore conceived as an integrated ”continuum”of data and model workflows along with deployment stages, as shown in Fig. 1.

Data Ingestion. It collects experimental data, checks data provenance, standardizes measurement protocol data and produces dataset annotations. This stage provides reliable data inputs to the downstream workflow and prepares the data for quantum processing. Dimensionality reduction and compression strategies are commonly applied to align high-dimensional datasets with hardware constraints and encoding limits. Approaches include quantum-based dimensionality reduction methods, tensor-based representations for high dimensional inputs [Di Meglio et al. 2024] and compression-oriented encoding strategies that reduce circuit depth and qubit requirements [Raisa et al. 2023].

Tokenization and segmentation. Tokenization decomposes a high-dimensional data sample into smaller data elements termed tokens, which can be processed independently or in parallel. Each token corresponds to a fixed length segment of a classical feature vector that is mapped to one or a small group of qubits. Token-based representations are used in machine learning systems and have also been explored for QML. Compression strategies

such as quantum random access codes (QRAC) encode n classical bits into a single qubit with success probability $p > 1/2$ [Thumwanit et al. 2021].

Quantum data encoding. After tokenization, each token is mapped to a quantum state through an encoding procedure. The choice of encoding can impact on model accuracy and the required circuit depth. Previous work reports performance variations between encoding strategies when the circuit architecture remains fixed [Hong and Josh Domingo Lopez 2025, Yang et al. 2023].

AI generator for synthetic data. Generative models can expand the data phase by producing synthetic samples that augment and re-balance training datasets. These models introduce controlled variations in the input data and increase the diversity of tokens used in the workflow. Approaches include Gaussian process generators and diffusion models such as CLIP guided stable diffusion [Rombach et al. 2022]. The generated samples are subsequently tokenized and encoded into quantum states, and may operate before or alongside tokenization and encoding within the proposed *dc-qml* workflow.

Quantum model training and error optimization. Quantum model training comprises tuning hyperparameters such as circuit depth, entangling patterns, number of layers, batch size and learning rate under hardware constraints [Carrera Vazquez et al. 2024]. Optimization strategies may use manual configuration, grid search, Bayesian methods or evolutionary approaches, all aiming to minimize a loss function. Training should also address *barren plateaus*, where gradients vanish and hinder quantum circuit optimization [McClean et al. 2018]. Consequently, effective training depends on coordinated adjustments of circuit parameters and data representation, including balanced datasets and refined encodings that reduce variance during optimization [Thanasilp et al. 2024].

Refinement continuum: evaluation and iteration. This process is used throughout the workflow rather than only at the final stage (Fig. 1). In a data-centric workflow, evaluation outcomes guide revisions of data ingestion, augmentation, tokenization and encoding. Model assessment therefore considers both accuracy and scalability metrics, including circuit depth, qubit count, and noise sensitivity under hardware constraints [Carrera Vazquez et al. 2024]. Whenever performance limits arise, adjustments extend beyond hyperparameter tuning and require redesign of data representation.

4. Discussion and research challenges

The proposed *dc-qml* methodology reframes the development of quantum machine learning systems by placing data preparation and representation alongside model design. In this perspective, improvements in tokenization, encoding and dataset composition can change model behavior even when quantum circuit architectures remain unchanged. Data resampling and controlled augmentation contribute to improved training behavior and scalability of QML workflows. This view aligns with broader results from classical machine learning research, where improvements in dataset augmentation often produce measurable improvements without architectural modifications [Jarrahi et al. 2023]. The *dc-qml* methodology therefore shifts the development process from a model adjustment loop to a ”data-to-model continuum”in which evaluation outcomes guide iterative refinement of data and model components.

Several open challenges remain for this perspective. QML workflows require improved coordination between encoding strategies, circuit depth and system resources in

hybrid QPU computing environments (CPU-GPU-Native processor). QML model training stability remains affected by noise and optimization difficulties, including *barren plateau* phenomena [McClean et al. 2018]. Synthetic data generation introduces additional questions related to validation, sampling reliability and bias control. At a broader level, progress also depends on standardized evaluation procedures. Automation techniques for *dc-qml* may assist this process through systems in a direction similar to automated machine learning approaches. Documentation of dataset "lineage", annotation procedures and ablation studies will also be required to ensure reproducibility and interpretability in future *dc-qml* studies [Di Meglio et al. 2024].

5. Conclusions

This short paper presented the *dc-qml* methodology, which frames QML development as a data-to-model workflow. The approach focuses on the importance of data preparation, including dimensionality reduction, quantum encoding design and dataset conditioning to operate within the constraints of NISQ hardware. The preliminary discussion suggests that performance and scalability in QML depend on the interaction between data design and model optimization. The proposed *dc-qml* approach complements model-centric development by introducing iterative refinement between data preparation and quantum circuit training.

Referências

Carrera Vazquez, A. et al. (2024). Combining quantum processors with real-time classical communication. *Nature*, 636(8041):75–79.

Di Meglio, A. et al. (2024). Quantum computing for high-energy physics: State of the art and challenges. *PRX Quantum*, 5:037001.

Havlíček, V. et al. (2019). Supervised learning with quantum-enhanced feature spaces. *Nature*, 567:209–212.

Hong, Y.-Y. and Josh Domingo Lopez, D. (2025). A review on quantum machine learning in applied systems and engineering. *IEEE Access*, 13:144607–144631.

Jarrahi, M. H. et al. (2023). The principles of data-centric AI. *Commun. ACM*, 66(8):84–92.

McClean, J. R. et al. (2018). Barren plateaus in quantum neural network training landscapes. *Nature Communications*, 9(1):4812.

Raisa, M. et al. (2023). A Parallel Quantum Feature Encoding Scheme for Effective Classical Data Classification in QCNN. In *Proc. IEEE TENCON*, pages 1–5.

Rombach, R. et al. (2022). High-resolution image synthesis with latent diffusion models. In *Proc. IEEE/CVF CVPR*, pages 10684–10695.

Thanasilp, S. et al. (2024). Exponential concentration in quantum kernel methods. *Nature Communications*, 15(1):5200.

Thumwanit, N. et al. (2021). Trainable Discrete Feature Embeddings for Quantum Machine Learning. In *Proc. IEEE QCE*, pages 479–480.

Yang, Z. et al. (2023). A survey of important issues in quantum computing and communications. *IEEE Communications Surveys & Tutorials*, 25(2):1059–1094.