Data-Centric Quantum Machine Learning: data-to-models continuum

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

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

[2] Postgraduate Program on Robotics and Artificial Intelligence, UTEC – Technological University of Uruguay, Rivera, Uruguay

Abstract

Quantum machine learning (QML) is a machine learning approach that uses quantum computation to improve learning performance. Research in this area has primarily followed a model-centric approach. However, near-term quantum hardware shows that data preparation also plays an important role, and can influence the quality and scalability of the learning process. Existing QML research focuses on circuits, kernels, and variational models; however, the data requirements that shape these methods remain less explored. This topical review addresses two questions. First, what forms of data processing are required across representative QML methods? Second, how can a data-centric approach improve the accuracy, scalability and resource use in QML models? A taxonomy was introduced to classify QML architectures according to their requirements for dimensionality reduction, encoding, dataset balance and compression. Based on this taxonomy, a data-centric QML (*dc-qml*) framework is proposed. It integrates various subprocesses into a coherent data flow, such as data ingestion, abstraction, tokenization, quantum encoding, and synthetic data generation. Results from recent benchmarks and case studies show that improvements in near-term QML optimizations often stem from data-focused practices rather than model refinement alone.

Key-words: Quantum machine learning, data-centric QML, quantum data encoding, quantum data tokenization, model-centric QML

1 Introduction

Quantum machine learning (QML) brings quantum computation into the machine learning workflow to increase problem-solving capacity and expand the range of problems that can be addressed [1–3]. Research in this area has focused mainly on algorithm design and the associated refinement of quantum circuit architectures. In contrast, the roles of data preparation and representation have received less attention [3, 4].

QML may offer advantages over classical machine learning (ML) methods. However, current quantum processors can only process limited-size datasets [5–7]. Quantum computer systems commonly represent information as qubits by using amplitude vectors. This representation poses a challenge when classical[1] ML datasets are mapped to quantum form [8]. This mapping becomes a bottleneck when the dataset is large or when the data structure does not align with the constraints of the near-term quantum hardware.

A data-centric approach provides a method for examining these issues from a broader perspective. For instance, dimensionality reduction methods, such as singular value decomposition (SVD), can

*This work has been partially supported by the CAPES graduate school program of the Ministry of Education, Brazil.

[1]Classical refers to the datasets, algorithms, and methods commonly used in classical machine learning. Quantum refers to data representations and methods, including architectures and circuits, designed for a target quantum computing platform.

reduce the complexity of the data input before quantum encoding. This can help to prepare large datasets for current quantum technologies. In this way, it makes sense to put effort into understanding how data should be represented before constructing quantum circuits and associated gates. Therefore, data representation is an important consideration in quantum machine learning, because the structure of the quantum state directly influences the performance and feasibility of the learning procedures [9].

If a classical dataset can be reliably mapped to a quantum information format, quantum machine learning is a natural extension of the classical ML approaches [10]. However, this mapping process is challenging. The transfer of data between classical and quantum systems requires appropriate data representation, scaling and precision. Such design choices influence the feasibility of the downstream workflow computations.

Results from classical machine learning demonstrate that many improvements arise from a data-centric perspective rather than from model refinements alone [11, 12]. This perspective favors the quality, structure and consistency of the data. Similar considerations apply to the QML. The performance of a quantum model, measured in terms of accuracy and scalability, depends on how the data are presented in a quantum circuit [10,11]. Well-designed data representations reduce the burden on quantum algorithms and align the dataset with the geometry of the quantum state space [13,14]. These data representations play a role in determining whether a QML method can exploit the available underlying quantum resources.

The resource limitations of the current quantum processor hardware, together with restricted access to larger quantum systems for simulation, necessitate a data-centric approach. A typical QML processing workflow begins by handling a dataset, followed by quantum state preparation (data encoding), parameterized quantum circuits, measurement and optimization. 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 entire algorithm [2]. For example, the encoding strategy affects both the number of qubits and the required circuit depth, which in turn determines the accuracy of the system outputs and extent of resource usage. A recent benchmarking work on angle versus amplitude encoding found that the choice of data embedding can lead to accuracy drops or gains of up to 30% for a fixed circuit architecture [12]. Optimizing data handling provides measurable improvements to address these constraints [15–17].

This topical review addresses two questions:

[Q1] What data requirements arise in representative QML methods?

[Q2] How can a data-centric approach improve the quality and scalability of QML algorithms?

Data-centric quantum machine learning (dc-qml) framework offers a way to answer these questions. It introduces a unified view in which datasets are partitioned into smaller data units called **tokens**, whichcan be encoded and manipulated within quantum workflows [15]. This approach allows for system modularity and reduces the complexity of the increased data dimensionality. It also aligns the workflow with the parallel processing used in quantum computing environments [18,19]. The framework can be deployed on local systems or cloud-based platforms, depending on computing resource availability and operational constraints [20,21].

The contributions of this topical review are as follows. This study examines the data-centric view of quantum machine learning and introduces the following elements:

- *dc-qml*: A framework that provides a data–to–model design workflow, that integrates data ingestion, tokenization, encoding and generation.
- *Tokenization and compression*: Discusses token-based representations and QRAC-inspired encoding schemes that reduce qubit usage and circuit depth.
- *Synthetic data*: reviews recent generative models and feedback cycles in a workflow that co-optimizes the data and models.

Table 1 lists the acronyms and terms used throughout this topical review, which are structured as follows. Section 2 presents the proposed taxonomy for data processing in QML. Section 3 discusses the tokenization and compression. Section 4 describes the proposed *dc-qml* framework. Section 5 provides a more detailed discussion along with the research challenges presented in Section 6. Finally, Section 7 concludes the paper.

Table 1: List of acronyms and terms used in the paper

Acronym Term	Description
Datasets	Classical ML datasets: MNIST, Fashion-MNIST, Iris and Wine are used for hybrid and baseline comparisons. Quantum datasets: QSFA and GASP-generated datasets simulate encoded qubit states or quantum feature distributions.
Embedding	Representation of data within a high-dimensional latent space, either quantum or classical, to preserve data relational properties.
Encoding	The process of mapping classical or tokenized data into quantum states.
Model-centric	A conventional paradigm in classical ML that prioritizes model architecture and hyperparameter updates over data refinements.
GASP	Quantum Generative Adversarial Sampling Pipeline.
NISQ	Noisy Intermediate-Scale Quantum. Describes current generations of quantum hardware with limited qubit counts and non-negligible noise, constraining circuit depth and training stability.
Performance	Combination of model quality (measured by *accuracy*) and scalability (assessed on how well the systems handle growth).
Performance-Accuracy	Measure of correct predictions to total predictions. Used to assess the quality of models such as QSVM, QNN, and QGAN.
Performance-Scalability	It refers to the ability of a QML model to maintain accuracy as the dataset size or circuit depth increases.
QASA	Quantum Adaptive Self-Attention. Transformer-like model using parameterized quantum circuits to learn adaptive attention weights through quantum superposition.
QGAN	Quantum Generative Adversarial Network.
QKSAN	Quantum Key–Sequence Attention Network. Quantum analogue of self-attention, representing key–query–value relations as quantum states for interference-based attention computation.
QNN	Quantum Neural Network.
QRAC	Quantum Random Access Code.
QSVM	Quantum Support Vector Machine.
QRWKV	Quantum Recurrent Weighted Key–Value. Quantum version of RWKV transformers integrates recurrent dynamics and key–value memory using quantum gates to capture temporal dependencies.
QSAM	Quantum Self-Attention Mechanism.
QSFA	Quantum Synthetic Feature Augmentation.
Token	The smallest unit of structured data used in the proposed *dc-qml* framework. Tokens may represent attributes or compressed data units.
Tokenization	The transformation of raw data into discrete tokens, preparing the data for downstream QML workflow processing.
Transformers	Classical ML architectures based on attention mechanisms. Serve as the baseline for comparing quantum counterparts such as QKSAN, QRWKV, and QASA.

2 Data Processing Taxonomy in QML

A data-processing taxonomy was developed specifically for this study to classify QML methods according to the data type (classical, quantum) and algorithm type (classical, quantum).

The survey presented in this paper reveals that most contributions in the literature fall under the classical data-quantum algorithm paradigm, with relatively few approaches addressing quantum data–quantum algorithms or quantum data–classical algorithms. Classical data-quantum algorithms dominate across topics such as quantum convolutional neural networks [22–24], quantum support vector machines (QSVM) [25, 26], generative models [27, 28] and reinforcement learning [29, 30]. Quantum data and quantum methods remain limited in the literature but include quantum native datasets [31], QSVMs trained on circuit data [32] and TensorFlow Quantum prototypes [33]. Quantum data and classical approaches were mainly introduced in early research, as proposed in [34].

In contrast, data handling practices varied across the board. High-dimensional classical data are typically reduced to a lower-dimensional space using techniques such as Principal Component Analysis (PCA) or autoencoders [35]. QML methods using specific quantum datasets rely on quantum PCA and tensor networks [18, 36, 37]. Hybrid approaches integrate these groups of methods with research that proposes token-based representations and QRAC-inspired embeddings for mapping to quantum circuits [38]. The following common challenges were observed: encoding efficiency, data dimensionality reduction and the associated circuit design issues.

The application context guided the selection of a paradigm for this review. In chemistry and materials science, quantum states provide relevant inputs for both quantum data-quantum algorithm and quantum data-classical methods. Finance and healthcare rely almost exclusively on classical data-quantum algorithm processing workflows [39, 40]. Natural language processing, an entirely classical

approach, has motivated classical data-quantum algorithms studies onto encoding data tokens into quantum states for downstream processing [13, 41]

Overall, the literature surveyed in Tables 2 and 3 demonstrates that despite methodological variety, data handling is a common aspect that improves QML model performance in terms of accuracy and scalability.

Another identified issue is the methodological redundancy. Several pieces of work replicate ideas under slightly different names—QCNN variants [22, 23, 42], or QSVM applications [25, 26, 43] without providing a theoretical analysis or even comparative experimental evaluation. Moreover, encoding methods are usually introduced without rigorous analysis of the system resource requirements or the need for system scalability. This raises questions regarding their applicability beyond simplified or small-scale problems.

Data requirements by algorithm type. QML algorithms impose specific data requirements and constraints, which are summarized as follows:

- Quantum support vector machines (QSVMs) [59]: Input data should be scaled numerically. Dimensionality reduction is typically used to minimize downstream encoding overhead in the workflow.
- Quantum neural networks (QNNs) [1, 10, 60]: Input data should be vectorized and numerically bounded (embedding). Encoding should be handled appropriately given the noise and circuit depth constraints.
- Hybrid classical-quantum neural networks [60, 61]: Data conversion should preserve numerical range compatibility between classical outputs and quantum data encoding.
- Quantum generative adversarial networks (QGANs) [62]: Requires large datasets. Appropriate encoding and measurement strategies are required to integrate the components of the classical and quantum systems.

In summary, QML algorithms require well-structured, numerically bounded and correctly scaled data to ensure compatibility with the quantum encoding, circuit depth and noise constraints of available computing platforms.

3 Data Compression and Tokenization

A token represents a minimal structured data unit that encapsulates small abstractions or attributes extracted from the raw data. Tokenization, as defined in this topical review, is a core scheme that converts continuous or unstructured data into discrete representations. This process allows compression, dimensionality reduction and data preparation for quantum encoding.

3.1 Prior Approaches to Tokenization

Existing QML and classical ML mechanisms demonstrate how tokenization has been explored for data handling:

- *Cognitive and semantic aspects:* Beyond engineering implementations, tokenization can be observed through semantic compression. Shani et al. [63] introduced a language model that maps data inputs into compact tokens, such as representations in classical ML, which balances compression with semantic fidelity. This view addresses tokenization as a technical mechanism as well as a conceptual bridge between the efficiency and data meaning. In a quantum computing setting, the same principle holds for hardware constraints: the number of qubits and feasible circuit depth remain restricted by current quantum processor technologies [64]. Tokenization can be used as a practical mechanism to reduce the dimensionality of data inputs, in which complex data are transformed into smaller token vectors.
- *Modular quantum operations (OTP) [65]*: Probabilistic one-time programs encode logical functions (e.g., G1 gates) onto single-qubit quantum states that operate as modular tokens. This design separates quantum resource exchange from classical execution, resulting in an increase in the gate rate by four orders of magnitude compared to earlier schemes.

Table 2: QML paradigms are categorized by data types and algorithmic approaches - Part I. To provide historical context, the studies are summarized in publication date order.

	Algorithm Type	*Key Results*
Bosco et al., 2024 [44]	Integrated quantum encoding strategy	Achieved acceptable accuracy with integrated encoding, outperforming classical CNNs in some configurations.
Dorsey et al., 2024 [45]	Quantum Fourier Transform Classifier	A quantum algorithm for antimalarial drug discovery.
Jeng et al., 2024 [46]	Multidimensional Quantum Convolutional Classifier (MQCC)	An algorithm for multidimensional quantum convolution.
Lee et al., 2024 [22]	Quantum CNN (QCNN)	An architecture to handle a large number of data dimensions.
Monnet et al., 2024 [47]	Hybrid quantum-classical QCNN	Studied effects of data encoding on QCCNN performance.
Andrés et al., 2023 [30]	Quantum neural network (QNN)	Proposed dimensionality reduction strategies.
Herman et al., 2023 [48]	Variational Linear Quantum Model	Introduced quantum models on a Boolean cube.
Mashtura et al., 2023 [49]	Quantum CNN (QCNN)	Achieved 90.9% accuracy on Fashion MNIST and 93.3% on MNIST using parallel quantum feature encoding.
Ovalle Magallanes et al., 2023 [42]	Quantum CNN (QCNN) using angle encoding	Achieved 90% accuracy on MNIST and 78.50% on Fashion MNIST using angle encoding.
Placidi et al., 2023 [32]	Quantum support vector machine (QSVM)	MNISQ, a quantum circuit dataset, achieving 97.91% accuracy.
Saxena and Saxena, 2023 [26]	Quantum support vector machine (QSVM)	Achieved 48.3% to 50% accuracy on pancreatic cancer classification.
West et al., 2023 [17]	Genetic Algorithm for State Preparation (GASP) and Variational Circuit	Achieved up to 96% accuracy with reduction in circuit depth.
Hur et al., 2022 [23]	Quantum Convolutional Neural Network (QCNN)	Achieved 99% accuracy on MNIST and 94% on Fashion MNIST with fewer parameters than classical CNNs.
Melvin, 2022 [28]	Quantum Generative Adversarial Network (QGAN)	A framework for high-dimensional signal processing.
Otgonbaatar et al., 2022a [50] Otgonbaatar et al., 2022c [51]	Amplitude and entangling N-layer QML. Quantum Transfer Learning	Quantum transfer learning for small and large-scale datasets.
Peddireddy et al., 2022 [52]	Tensor Ring Variational Quantum Circuit (QC)	Tensor ring parameterized VQC, achieving 83.33% accuracy on the Iris dataset.
Zeguendry et al., 2022 [53]	Quanvolutional Neural Networks (QNNs), Variational Quantum Classifier (VQC)	High accuracy and low loss results compared to classical ML.
Alam et al., 2021 [54]	Quantum convolutional neural network (QCNN)	A hybrid quantum-classical model for image classification.
Batra et al., 2021 [39]	Quantum support vector machine (QSVM)	Comparable accuracy to classical ML methods for drug discovery.
Kariya and Behera, 2021 [25]	Quantum support vector machine (QSVM)	Studied QSVM performance in the NISQ era.
Schatzki et al., 2021 [31]	Quantum Neural Network (QNN)	Entangled datasets are inherently quantum and can be used directly without encoding.
Sierra-Sosa et al., 2021 [40]	Variational Quantum Circuit (VQC)	Achieved 70% (2-qubit) and 72% (3-qubit) accuracy on diabetes type 2 classification.
Skolik et al., 2021 [29]	Quantum Deep Q-Learning	A variational quantum algorithm for deep reinforcement learning.
Stein, 2021 [55, 56]	Quantum Neural Network (QNN)	Up to 12.51% improvement in classification accuracy and 98.5% reduction in parameters compared to classical methods.
Thumwanit et al., 2021 [38]	Quantum Random Access Coding (QRAC)	Trainable discrete feature embeddings for QML.
Zoufal, 2021 [27]	Quantum generative adversarial network	Generative QML algorithms and their applications.

Table 3: QML paradigms are categorized by data types and algorithmic approaches - Part II. To provide historical context, the studies are summarized in publication date order.

	Algorithm Type	*Key Results*
Zoufal, 2021 [27]	Quantum generative adversarial network	Generative QML algorithms and their applications.
Chen et al., 2020 [35]	Hybrid model: tensor network and variational quantum circuit	A hybrid model that outperforms principal component analysis (PCA) as a feature extractor.
Kerenidis and Luongo, 2020 [16]	Quantum slow feature analysis (QSFA) and Quantum Frobenius Distance (QFD)	Achieved 98.5% accuracy on MNIST dataset.
Kumar et al., 2020 [57]	Variational QML algorithm using a *dressed* quantum network	An encoding scheme for quantum circuit implementation.
Sierra-Sosa et al., 2020 [33]	TensorFlow QCNN	Discussed 8.9% improvement in classification accuracy using amplitude encoding
Kerenidis et al., 2019 [58]	Quantum CNN (QCNN)	A shallow quantum circuit based on a classical CNN.
Yu et al., 2019 [36]	Quantum PCA-based data compression	A quantum algorithm for data compression exponentially faster than classical PCA.
Rupp, 2015 [34]	Classical ML algorithms (e.g., SVM)	Surveyed classical ML applied to quantum mechanics problems.

- *Representation prediction architectures (V-JEPA) [66]:* Raw video data are decomposed into spatio-temporal patches that serve as tokens, creating a 1-D data sequence. Operating in this vector space allows the model to ignore irrelevant details while retaining its predictive capability.
- *QML equivalents:* Quantum random access coding (QRAC) encodes multiple classical bits into a single qubit, which compresses information into quantum tokens [38]. Similarly, quantum operations can be treated as tokens in Quantum Assembly Language (QASM) within the context of the MNISQ dataset, as discussed in [32].

Hardware constraints reinforce the same previously mentioned principle: the number of qubits and maximum feasible circuit depth remain limited by currently available technologies [64]. Thus, tokenization is a practical mechanism for creating small abstractions from data inputs.

3.2 Quantitative Preliminaries

Quantitative studies have suggested that handling data more systematically may lead to performance improvements in QML that are likely to exceed those obtained from purely algorithmic refinements. The extraction of small abstractions from data and the use of quantum data-encoding strategies have been shown to directly affect the model accuracy. For instance, Stein et al. [15] reported an accuracy improvement of up to 12.51% with a 98.5% reduction in the number of system parameters compared to classical approaches. Kerenidis et al. [16] achieved 98.5% accuracy on MNIST using QSFA with polylogarithmic runtimes. West-Maxwell et al. [17] demonstrated that approximate encoding via GASP achieved 96% accuracy with a reduced circuit depth. Other results cover a wide spectrum, from 48–50% in pancreatic cancer classification using QSVMs [26], to 97.9% with QSVMs on the MNISQ dataset [32], and up to 99% with QCNNs [23]. Parallel feature-encoding data abstractions achieved 90.9% accuracy on the data and 90.9% on the Fashion-MNIST dataset [49].

Fig. 1 shows the performance results obtained from the surveyed literature. It shows how data handling, such as dimensionality reduction, approximate encoding and hybrid preprocessing affects performance in terms of accuracy and scalability. Such quantitative analysis substantiates the core argument of a data-centric approach: systematic data preparation, further compression and optimized encodings can play an important role in quantum machine learning methodologies [67].

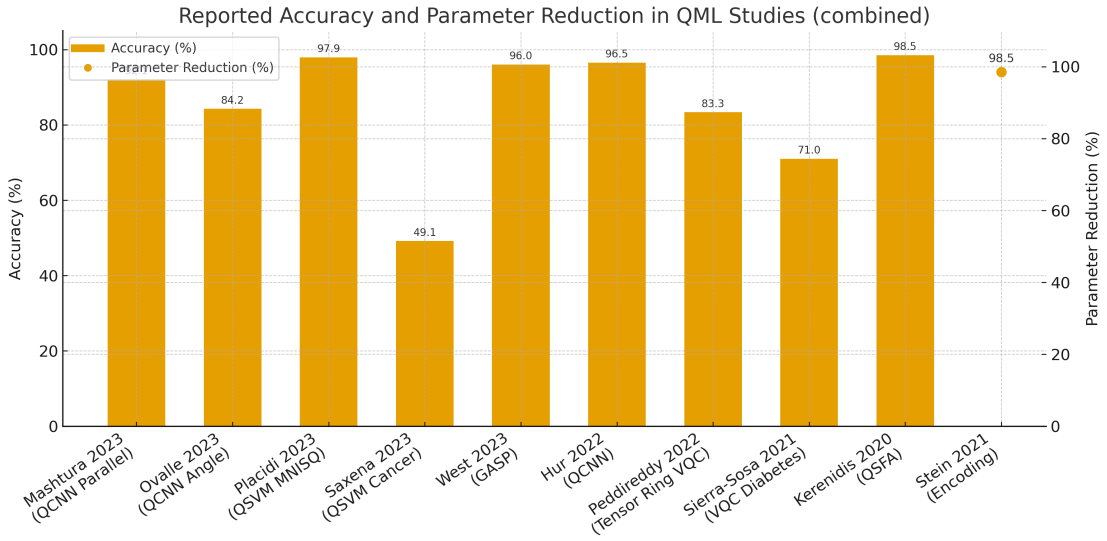

Figure 1: Reported accuracy and system parameter reduction in QML studies. Results range from 48–50% in medical datasets [26] to nearly 99% on MNIST with QCNNs [23], with intermediate values such as 97.9% using QSVMs [32], 96% with GASP [17], and 93.3% with parallel QCNNs [49]. Stein [56] reported a 12.5% accuracy gain with a 98.5% reduction in system parameters. This result suggests that data preparation and quantum data encoding may have stronger effects on accuracy and scalability than algorithmic refinement alone.

4 Data-Centric QML design: models and data

As summarized in Tables 2 and 3, QML adopts a model-centric design approach by default that focuses on architecture adjustments, hyperparameter tuning and the definition of alternative loss functions. A recent study provided a detailed review of model-centric QML methods and system architectures for engineering applications [3].

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 training and inference, and recognizes that iterative refinement is required to obtain more accurate scalable models. This approach is aligned with data-centric principles in classical machine learning, which prioritize data quality and representation over architectural adjustments [4]. In QML, these ideas are constrained by the limits of current quantum systems, where the efficiency of data encoding and compression determines whether an algorithm remains feasible in near-term quantum processor hardware.

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. 2. The following discussion focuses on the proposed *dc-qml* framework and its rationale as a guideline for future QML work development.

In this framework, *process understanding* defines each workflow stage. In traditional model-centric QML, process understanding occurs mainly between model development and model evaluation, and rarely extends to the data phase. In *dc-qml*, process understanding spans the entire workflow. It iterates through data, models, and development phases and cycles back until the accuracy and scalability requirements for a target application are met.

4.1 Data Ingestion

High-quality datasets are required for reliable learning. Typically, the data ingestion process consists of a subsystem capable of collecting experimental data, checking data provenance, standardizing measurement protocols and creating data annotation. Data ingestion ensures fidelity of the physical simulations downstream. In line with data-centric AI principles, priority is given to improving the data annotation (e.g., labels in classification tasks) and reducing noisy samples before designing and evaluating complex quantum circuits. Classical data preprocessing steps such as normalization, outlier

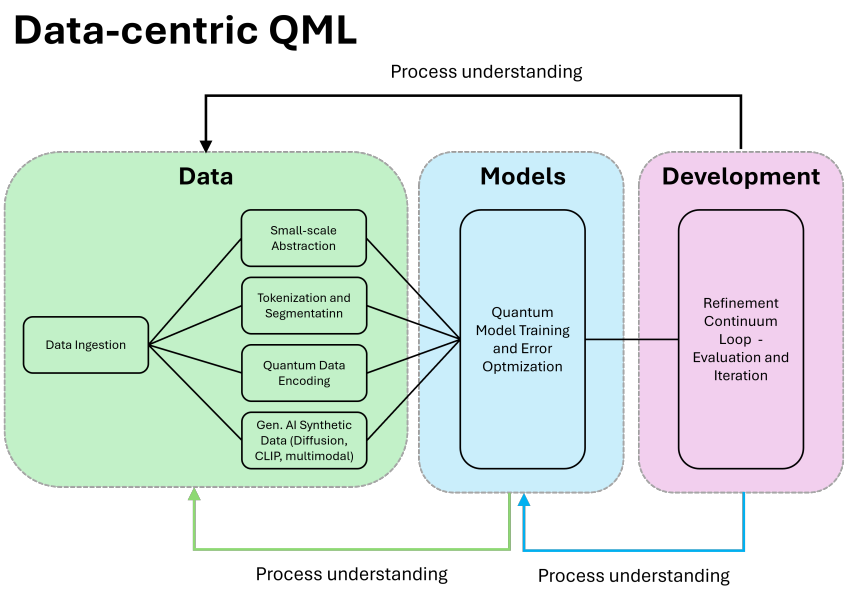

Figure 2: Evaluation and iteration in data-centric QML (*dc-qml*). Unlike model-centric workflows, *dc-qml* links model evaluation back to data preparation. The loop adjusts parameters and also revisits data design through augmentation, tokenization and synthetic data generation [62, 68, 69]. Tasks run across QPUs within a unified classical-quantum workflow. This structure reflects the idea of a universal quantum computer introduced by Deutsch, in which classical control coordinates quantum operations [70].

removal and data imputation can be used prior to quantum data encoding. Recent work has also shown that quantum processing can directly improve data quality [71]. In this approach, data ingestion serves as both an entry point in the design space and as a possible stage for quantum-assisted quality improvement. The partitioning of data into training and testing sets should be statistically justified to reduce overfitting and any domain shifts should be recorded. This is standard practice in classical machine learning.

4.2 Small-scale Abstraction

Small-scale abstraction, a superset of feature mapping, reorganizes raw data into structured, high-level inputs. This workflow step extracts important characteristics from the original data. Such features can be obtained using classical algorithms running on classical hardware prior to quantum data encoding. PCA or autoencoders can be used to remove redundancy and noise, resulting in more compact downstream quantum inputs.

Quantum Principal Component Analysis (QPCA) can achieve dimensionality reduction with favorable scaling compared with classical PCA methods [36]. Tensor ring representations also provide better handling of high-dimensional data inputs [52]. Approximate encoding can produce low-depth circuits that are robust to noise, with balanced accuracy and scalability, as discussed in [17]. In contrast, other compression encoding schemes reduce circuit resource demand as described previously [57]. Hybrid methods integrate classical dimensionality reduction, such as PCA and autoencoders with quantum encoding [39, 49, 54].

Limited quantum processing resources can be addressed by using compression schemes that mitigate errors and noise in the datasets. Current issues include minimizing information loss to address algorithm-specific compression requirements and integrating more quantum-driven methods, such as tensor networks [18, 37]. Material discovery is an application area in which small-scale abstraction has been evaluated with reasonable results [72].

In the *dc-qml* design approach, small-scale abstraction is the stage in which raw or preprocessed data signals are compressed to prepare them for tokenization and encoding.

4.3 Tokenization and Segmentation

Following small-scale abstraction, tokenization splits a high-dimensional data sample into smaller data pieces (*tokens*), which can be processed independently or in parallel. Token-based representations, commonly used in classical ML Transformer-based language models [73], have also been considered for QML. In this case, tokens are specified as fixed-length data segments of a classical vector mapped to an individual or small group of qubits.

This modular approach allows parallel quantum state preparation; hence, it reduces the quantum circuit depth [15, 19, 59, 74]. For example, the data input of V tokens can be represented in $\lceil \log_2(V) \rceil$ qubits. Pre-trained classical embeddings can also be translated into quantum embeddings in a manner that preserves the structural relationships between these modalities. Some of these challenges include encoding extended token sequences and managing associated quantum circuit complexity. Techniques such as quantum recurrent processing and tensor networks have been proposed to mitigate these issues [18, 37]. Tokenization also aids with compression encoding through quantum random access codes (QRAC) [38], in which n classical bits are encoded into a single qubit with success probability $p > 1/2$ [75]. For instance, by grouping features into tokens and applying QRAC, qubit resources can be traded against success probability.

Discrete feature embeddings produce embeddings similar to those of classical tokens in natural language processing (NLP). Placidi et al. [32] introduced the MNISQ dataset, which represents quantum circuits as tokenized sequences in quantum assembly language (QASM). In this case, the operations are treated as tokens. Data re-uploading strategies, as discussed in [76], resemble sequential tokenization, in which data are repeatedly encoded into a sequence of quantum circuits.

Algorithm 1 Tokenization and segmentation

Require: Classical feature vector $x \in \mathbb{R}^d$, segment size m, encoding method $\mathcal{E}$
Ensure: Quantum state prepared on $\lceil d/m \rceil$ qubits qu_n
1: Partition x into $K = \lceil d/m \rceil$ segments $x^{(1)}, \ldots, x^{(K)}$, each of length m (padding with zeros if necessary).
2: **for** $k = 1$ to K **do**
3: $\tilde{x}^{(k)} \leftarrow$ Preprocess$(x^{(k)})$ {e.g., scaling or PCA}
4: Prepare qubit q_k in state $|0\rangle$
5: Apply encoder $\mathcal{E}(\tilde{x}^{(k)})$ to q_k {e.g., amplitude, angle or QRAC}
6: **end for**
7: **return** Quantum register $(q_1, \ldots, q_K)$ representing the encoded data

The pseudo-code in Listing 1 illustrates a simplified tokenization and segmentation algorithm. A feature vector is partitioned into fixed-size segments, each of which can be optionally preprocessed (e.g., scaling or PCA) before being encoded by a quantum encoder. This modular scheme supports heterogeneous encoders and achieves parallelism across the quantum registers. Parallelism exploits quantum superposition to simultaneously process multiple tokens [18]. In summary, the proposed tokenization stage provides building blocks for the next step, termed quantum data encoding, in which tokens are transformed into quantum states that are suitable for computation.

4.4 Quantum Data Encoding

After tokenization, each token is mapped to a quantum state. This mapping, referred to as encoding, influences the model performance. Comparative studies have reported accuracy differences of 10–30% between amplitude and angle encoding when the circuit architecture is fixed [12]. Therefore, the choice of encoding requires a balance between the number of qubits and the complexity of the state preparation. High-dimensional data do not always require a direct correspondence between dimensionality and qubit count. Many tasks can be performed using compact quantum representations.

QML inherently relies on vector space embeddings, in which quantum states are represented in high-dimensional Hilbert spaces. Multiple encoding families of methods have been explored in the literature. Amplitude encoding allows a classical vector of length N to be compressed into $\log_2 N$ qubits. However, this may require complex state preparation circuits [12, 17, 33, 36, 49, 53, 77]. By contrast, angle encoding maps each feature component to a rotation angle on an individual qubit [42, 47]. Angle encoding increases the number of qubits required by using simpler gates. Discrete Boolean data embedding

was introduced by Herman et al. [48]. Structured embeddings, such as tensor networks, can capture correlations in a compact manner [35, 52]. Quantum feature maps, motivated by kernel methods, embed classical data into larger Hilbert spaces [26].

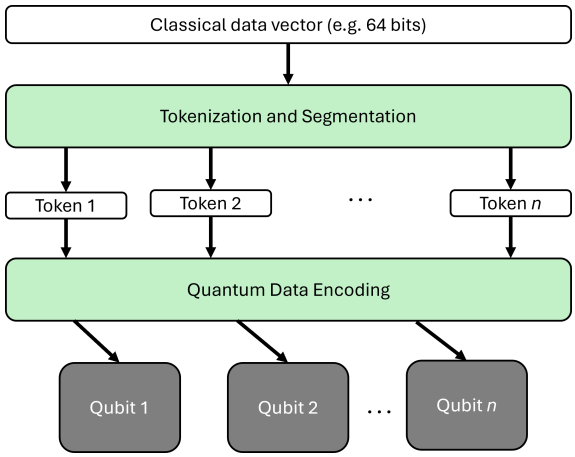

Figure 3: Illustrative scheme of quantum data encoding. A classical vector data (e.g., 64 bits) is partitioned into fixed-length tokens, which are mapped to qubits using methods such as amplitude encoding or QRAC. The diagram illustrates a specific configuration and is not intended to represent a general case. In the context proposed in this paper, tokenization combined with quantum data encoding allows for (i) modular circuit design, (ii) explicit control of trade-offs between qubit count and decoding accuracy, and (iii) parallel execution of tasks across quantum registers in system-level implementations.

Encoding is the bridge between tokenized data and quantum computation, where accuracy and scalability should be balanced. It is connected to the next workflow stage (Fig. 2), where the states are processed and optimized within the quantum circuits.

4.5 Data preparation for quantum states

Universal quantum computer model

A universal quantum computer (UQC) is a conceptual computing machine that manipulates data using quantum circuits that operate on qubits [70]. This framework serves as a pertinent abstraction for quantum computation [10]. In this model, qubits allow for superposition and equip the system with capabilities that lack direct counterparts in classical computing. Multi-qubit systems expand the QML computational space via tensor products, and quantum circuits implement algorithms by arranging quantum gates into ordered sequences.

A universal gate set can approximate any unitary transformation to arbitrary accuracy. This property provides the UQC model with additional capacity that allows different physical system implementations to capture correlations within a compact framework. Representative platforms include photonic, superconducting, atomic and spin-based system architectures, all of which are governed by the same underlying formalism [10]. The universal UQC model provides a theoretical basis for data encoding and representation in *dc-qml*.

Quantum states and quantum gates

Quantum states follow a compact vector notation. A single qubit is represented as

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle,$$

with amplitudes that satisfy

$$|\alpha|^2 + |\beta|^2 = 1.$$

Multi-qubit states extend this structure through tensor products. A two-qubit system follows the form:

$$|\psi\rangle = \alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle,$$

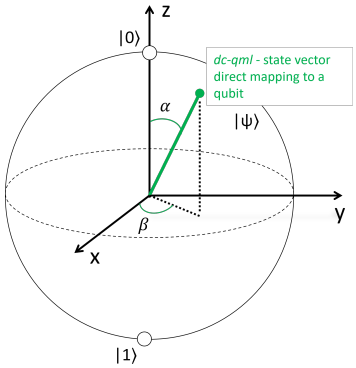

Figure 4: One qubit state represented on the Bloch sphere. The amplitudes α and β determine the position of the state vector. This geometric view provides an intuitive way to interpret classical data in terms of a quantum-computing counterpart.

as presented in [10]. Such states represent correlations and entanglement when they cannot be separated into independent components within a particular vector space.

The quantum gates change these states. For instance, single-qubit gates rotate amplitudes or introduce phases, whereas multi-qubit gates implement controlled operations or tensor-product construction [3].

Data preparation for state vector simulation

Data preparation in *dc-qml* builds on the ideas introduced in the previous subsections. A classical data value is mapped into amplitudes that follow the expression:

$$\alpha|0\rangle + \beta|1\rangle.$$

This establishes a direct link between real-valued features and their quantum representations. The mapping also provides a geometric interpretation, because each qubit corresponds to a point on the Bloch sphere. This relationship is shown in Fig. 4, where each qubit corresponds to a binary index in the global state vector. The entire quantum state is stored as a one-dimensional complex vector of size 2^N [18]. This data representation matches the Hilbert space dimension and preserves the full amplitude without reduction. The vector should be normalized before use and ordered on a computational basis.

Large quantum states are likely to introduce memory constraints into the systems. In this case, the state vector can be distributed across multiple devices when it no longer fits into single-device memory. The gate operation was described in [18] as the major computational cost in the state-vector simulation. This is because each transformation affects an entire vector. Performance improves when gate operations targeting multiple qubits are executed in parallel.

Gate fusion further improved the performance. Several small gates are combined into a single larger operation. This fused operator completes execution in a single step, reducing memory data transfer and shortening the execution path. These optimizations are particularly useful for *dc-qml* workloads in quantum circuits with many small gates.

The state-vector representation describes the Hilbert space. In contrast, the following section presents tensor network methods that provide structural abstraction and compression to address complexity.

4.6 Data preparation for tensor network states

Tensor networks provide a more compact representation of quantum states, distinct from the full-state-vector approach discussed earlier. Instead of keeping track of all amplitudes, the global state is represented as a network of smaller connected tensors. Such connections are represented by indices that capture correlations across the system to reduce memory usage when entanglement remains limited [18, 37].

In this context, quantum simulations proceed by contracting pairs of connected tensors. In a network comprising N tensors, a total of $N-1$ contractions are required. Contraction is associative, meaning that there are multiple possible sequences of contractions. The choice affects the system performance. Therefore, data preparation should consider the network structure and the planned contraction strategy. A matrix product state (MPS) is a particular case of a tensor network arranged as a linear chain. Each qubit is represented by a local tensor with two bond indices and a physical index [37]. To create an MPS from a general state, a dimensionality-reduction technique such as singular value decomposition (SVD) is applied recursively. This method produces a suitable representation of the state, provided that all the resulting SVD singular values are retained. Other dimensionality reduction techniques may also be used, including the Gram-Schmidt QR method. Memory usage can be reduced by removing small singular values during the decomposition. The number of retained values defines bond dimensions (χ). A smaller (χ) reduces the computational cost and introduces an approximation that resembles lossy compression. This approach remains useful for quantum machine learning tasks when data correlations are not uniformly strong [20].

These properties also support qubit compression. A four-qubit system requires typically a vector with a size of 2^4. MPS decomposition reduces the rank of the quantum state and produces a more compact form when correlations remain limited. A compressed representation needs only two effective qubits.

Tensor networks also allow simulations that are beyond the limits of state-vector methods. The memory required for a state vector grows to 2^N, which restricts simulations on a single device once the number of qubits becomes large. Simulations above 33 qubits exceed the capacity of a device, and a 50-qubit state vector already requires several petabytes of memory [78]. In contrast, MPS simulations can be run on a single device when the bond dimensions are controlled. Schieffer et al. [37] reported simulations with 60 qubits for several quantum circuits and up to 90 qubits for a GHZ circuit. Runtime scaling differs from the exponential behavior observed in state-vector simulations and follows power-law or linear patterns that depend on the quantum circuit structure.

4.7 AI Generator for Synthetic Data

As shown in the left block of Fig. 2, the data phase can be expanded using generative AI models, which produce large-scale synthetic data samples to augment, re-balance and evaluate quantum workflows. Beyond conventional preprocessing, synthetic data generation step extends the token space and allows for controlled perturbations of the data inputs. Gaussian-process generators and diffusion models, including CLIP-guided stable diffusion [79], can inject structured noise into the seed data and regenerate new data variants (e.g., diverse tensor network states) [68, 69]. The resulting samples can be modularized into tokens and encoded into quantum states in the *dc-qml* downstream processing workflow (on the right-hand side of Fig. 2).

Recent work has suggested that synthetic data can be co-designed with quantum workflows. Diffusion methods in classical ML have been adapted to generate parameterized quantum circuits (PQCs) by proposing circuit structures and gate parameters. These elements can then be evaluated downstream [80]. Quantum Generative Adversarial Networks (QGANs) have been investigated as quantum-native generators for data augmentation. A recent study surveyed QGAN architectures and training protocols, from theory to early demonstrations, as approaches capable of producing representative data distributions with hardware-amenable training loops [81]. Empirically, QGAN-based augmentation has been reported to outperform common classical heuristics (e.g., random geometric transforms) on computer vision benchmarks [82]. Hybrid quantum–classical QGAN variants have also been explored for tabular time-series data with quantitative evaluations of the distance between the real and generated data distributions [83].

In this context, Uotila et al. [71] discussed how quantum processor resources can be explored for data preprocessing and transformation. This particular work shows that extracting values from noisy or incomplete data is consistent with data-centric QML, in which data preparation is regarded as equally important as algorithm design. These results suggest that synthetic data generation, whether classical or quantum, can be used not only to address unbalanced data but also to produce controllable, token-level variations tailored to specific quantum data encoders.

AI generators for synthetic data are configurable steps that can precede or run in parallel with tokenization and encoding in the proposed *dc-qml* framework. Two complementary paths are considered:

- (i) *Classical generators for quantum encoding:* diffusion and CLIP-guided pipelines create token candidates that are later mapped to quantum states [68, 69].
- (ii) *Quantum-native generators:* QGANs produce distributions directly on quantum processing units (or simulators), generating samples or circuit components for subsequent quantum modules [81, 82].

Ablation studies are encouraged on both paths. The ablation methodology isolates the role of each component by removing or varying it and observing the resulting changes in the performance. Such studies can compare configurations with and without generated synthetic data, classical versus quantum generators and how different encoders (such as amplitude or angle) respond to the synthetic data variability. Although this type of data generation has not yet been systematically applied to QML, it illustrates how the *dc-qml* framework can address the shortage of high-quality datasets. Generative AI workflows that co-design data and models under quantum resource constraints can be further developed within a continuous cycle of QML refinement [84].

4.8 Quantum Model Training and Error Optimization

The QML model parameters can be adjusted manually at the beginning of the design process. Small circuits can be probed, gates can be tuned manually and the computed loss values are inspected to obtain a coarse parameter setting before the start of automated model training. After this initialization, parameterized quantum circuits are trained using gradients computed using the parameter-shift rule or finite differences. When classical training is used, the optimization can proceed end-to-end using algorithms such as backpropagation. The weights are updated in the backward pass, and standard loss functions such as cross-entropy or mean-squared error are used in the optimization procedure. Following this, optimizers are selected from a pool that includes stochastic gradient descent, Adam or gradient-free methods with pre-configured learning rates, early optimization stopping applied and gradient clipping used to stabilize optimization updates. This represents a model training procedure similar to a standard machine learning workflow.

However, Martin et al. [85] elaborated on the issue of model training, which has evolved beyond conventional routines. This new methodology is termed *evolving machine learning* and it proposes more adaptive and automated pipelines in which optimization, model architecture selection and data handling are integrated within a continuous processing loop. From this perspective, QML training can benefit from approaches that co-adapt encodings, tokenization schemes and data augmentation as part of the optimization cycle. Such approaches extend the role of training within a broader model–data co-design process. Process understanding begins in the data handling phase, proceeds through model development and then returns to the data phase, as represented by the top arrow in Fig. 2.

Hyperparameters are still tuned by adjusting "higher level variables" such as circuit depth, entangling patterns, number of layers in chosen model architectures, batch size and learning rate - to cite a few. Optimization search strategies may be manually configured, grid-based, Bayesian or evolutionary, but they all share the goal of minimizing a specific loss function within the existing hardware constraints [64]. In this case, the search also aims to strike a balance between classical and quantum processing splits [86]. Model training in quantum machine learning should also address *barren plateaus*, where gradients vanish across regions of the cost/loss function exploration space and therefore hinder optimization of quantum circuits [87]. Mitigation strategies include the use of local cost/loss functions, data re-uploading when the number of qubits is limited and frugal measurement schemes. Parameter initialization and "handcrafted" circuit design have also been discussed as additional approaches to the barren plateaus problem [88]. Thanasilp et al. [89] reported encouraging results showing that data-centric designs for encodings and kernel families can be used in this context.

A workflow for QML model training based on these methodologies is presented in the box below, and is directly linked to the surveyed models in Tables 2 and 3.

QCNN models trained with angle or amplitude encoding have achieved near state-of-the-art accuracy on the MNIST and Fashion-MNIST datasets under compact parameterizations [22, 23, 47, 49]. Variational classifiers and QNNs follow similar model regimes where compression and data re-uploading ensure feasibility on current hardware [30, 54, 57]. In contrast, QSVMs rely on well-conditioned small-scale abstractions (feature maps). Therefore, model training and error optimization focus on kernel selection and regularization rather than circuit depth [26, 39, 59]. QGAN models are trained adversarially and the alternating generator and discriminator are updated under measurement constraints [27, 62].

Quantum Model Training

Initialize model/circuit → Choose loss function →
Select optimizer → Tune hyperparameters →
Minimize errors → Monitor barren plateaus →
Validate → Early stopping

The results reported in [90] demonstrate that hybrid QNN models can perform entity matching in classification tasks with compact data representations. Across such families of QML algorithms, success measured as high accuracy and scalability of trained models still depends on disciplined encoding, controlled circuit depth and improved data handling. Data-centric practices reinforce such efforts: data balance is maintained, batches are stratified and outliers are reviewed. Data representations are refined to reduce variance, which typically aids in model convergence.

4.9 Refinement Continuum: Evaluation and Iteration

Evaluation and refinement are not isolated end stages but rather integral parts of the entire workflow (Fig. 2). Traditional model-centric workflows tend to cycle from model to model, adjusting the model architecture constructs, weight matrices and hyperparameters. In contrast, a data-centric workflow extends this by introducing a development→data loop, where the evaluation outcomes inform new data ingestion, augmentation and tokenization strategies.

Rigorous evaluation of the QML "end solution" should consider multiple dimensions, including not only accuracy (how good the outputs are), but also scalability aspects (how well the system handles growth). Such aspects include the circuit depth, qubit count and robustness to noise under existing hardware constraints [64]. Ablation experiments using factorial design methodologies should be conducted to determine the effects of each system design choice, both individually and in combination. Whenever accuracy and scalability bottlenecks are detected, the process understanding iteration does not simply require adjusting learning rates or the number of layers. This implies revisiting the data quality, expanding the data sample size with real and synthetic data, or even re-evaluating tokenization granularity using compression techniques. In a data-centric workflow, hyperparameter optimization is meaningful only when paired with renewed data design.

AI data generators help create new tokenized datasets that enrich the training data and probe the limits of quantum encodings. Such tools can evaluate feedback to trigger data renewal, rather than being confined to circuit adjustments. Recent research on classical ML confirms this argument by showing that improvements in dataset design and augmentation often outweigh model-side refinements in terms of the overall performance (both accuracy and scalability) [85].

Table 4: Analysis of alignment of selected recent studies with the *dc-qml* framework. A checkmark (✓) is alignment, a tilde (∼) represents partial alignment and a dash (–) is a lack of alignment. Synthetic data workflows remain unexplored.

Related work	Tokenization	Quantum Data Encoding	Dataset Balancing	Synthetic Data Generation	Development → Data Loop	Hybrid Workflow
Routh & Singh (2025) [91]	✓	∼	–	–	–	✓
Uddin et al. (2025) [92]	✓	∼	✓	–	–	✓
Sharma et al. (2025) [93]	✓	✓	–	–	∼	✓

5 Discussion

The following subsections discuss how the *dc-qml* rationale appears in recent work and broader data-centric practices in QML. Table 4 shows that the latest work has already adopted tokenization and hybrid workflow. However, data-centric elements, such as feedback loops, dataset balancing and synthetic data, have only been partially addressed.

5.1 Linkage of recent studies to the *dc-qml* framework

5.1.1 Hybrid quantum–classical data generation

Routh and Singh (2025) [91] integrated a classical ML transformer architecture with a parameterized quantum circuit for text generation. Tokenization and encoding were identified as key aspects of the performance in this study; however, the evaluation was not sufficiently elaborated and the token-to-qubit mapping was unclear. From a *dc-qml* standpoint, systematic encoding strategies and generative tokens (e.g., diffusion-based) would enrich representations and better capture circuit depth.

5.1.2 Hybrid transfer learning with SMOTE balancing

Uddin et al. (2025) [92] proposed a hybrid model using GPT-Neo embeddings with LoRA and quantum circuits. This is complemented by Synthetic Minority Over-sampling Technique (SMOTE), which generates synthetic data for minority classes to improve the class balance in classification applications. This work demonstrates data-side processing across GPU hardware and simulators. However, its dependence on SMOTE and its limited encoding details constrain its scalability. The *dc-qml* can address this by exploring synthetic data pipelines to generate token-level diversity rather than relying on heuristic oversampling.

5.1.3 Quantum-enhanced natural language generation

Sharma et al. (2025) [93] developed benchmarks for QKSAN, QRWKV and QASA against classical transformer architectures [73]. This study demonstrates that dataset curation and multi-metric evaluation can improve data-related diversity. The *dc-qml* extends this work by linking evaluation outcomes back to the data design, thus producing dataset variation and synthetic augmentation. These recent studies confirm the argument of this topical review that QML approaches can benefit from data-side practices, such as tokenization, balancing, and curation, rather than relying solely on algorithmic adjustments.

5.2 Data-Centric Practices in QML

Small-scale abstraction, encoding optimization and data resampling strategies show that careful data preparation can lead to measurable improvements even when model configuration parameters are fixed during the loop of process understanding. For instance, tailored abstractions in the form of feature maps [44] and domain-specific, small-scale abstractions [42] may reduce the circuit overhead by embedding relevant patterns into quantum states.

Data resampling and augmentation have contributed to these improvements. Borrowing from classical ML, these methods re-balance datasets or augment data samples to help mitigate bias, thus improving the generalization properties of the model [48]. Transfer learning further illustrates a data-centric approach: hybrid models trained on one task can be fine-tuned on another, reusing the data-induced knowledge rather than starting from scratch [39]. Hybrid preprocessing pipelines demonstrate the synergy between classical and quantum transformations [94]. Using dimensionality reduction or filtering steps before encoding, the data are simplified and better aligned with the existing resource limitations of quantum processing devices. Compression methods have the similar purpose of ensuring that high-dimensional data can be expressed within the limited availability of qubits [17].

5.3 Towards Standardization and Automation

A key implication of this new perspective is the need for standardized data-oriented development practices in QML. Standardized benchmarks, encoding protocols, and more importantly, adequate evaluation metrics are necessary for reproducibility across platforms. Beyond standardization, automation is a promising avenue for future research. Similar to AutoML [95, 96], which automates hyperparameter exploration in classical machine learning, preliminary research results suggest that next-generation AutoQML systems will automate the search for encoding strategies, small abstractions and data augmentation workflows in quantum machine learning [97]. Native quantum augmentation is an emerging research field. QGANs and diffusion-based data generators provide the basis for producing representative token-level distributions tailored to quantum encoders [81].

6 Research Challenges

The issues below provide a snapshot of the areas in which current practices are still underdeveloped.

6.1 Scaling and system co-design

End-to-end performance is limited by state preparation and data loading. Amplitude encodings offer efficient data compression; however, their gate complexity increases with the input size. Angle encoding trades qubits for simpler preparation. Choosing among them is a system decision rather than a purely algorithmic decision. Workflows should schedule tasks across GPUs for data ingestion, small-scale abstraction and tokenization. Progress depends on coordinated efforts in hardware and software design and on the integration of QPUs at both the node and system levels [98–100]. Large-scale simulators are important tools for development and ablation studies [18]. Recent studies have examined the use of customized FPGA accelerators to scale quantum circuit simulations, although the cost of processing high-dimensional data remains a challenge [101].

Open issues. Computing cost-aware encoding under high dimensionality, placement and scheduling strategies for QPU–CPU/GPU pipelines, and appropriate resource system monitoring including depth, qubit count and wall-clock.

6.2 Training stability, noise and mitigation

Variational training in QML faces the barren plateaus problem, where vanishing gradients stall the optimization procedure [87]. Local cost/loss, layer-wise growth, careful initialization and circuit design have been demonstrated to address this issue [88].

On hardware, error-mitigation methods such as zero-noise extrapolation, probabilistic error cancellation, randomized compiling and readout calibration should be integrated into the training loop rather than being applied only after the computation is complete [60].

Open issues. Tuning of encoding to control expressivity; measurement-frugal training protocols that preserve sample efficiency; and standardized monitoring of mitigation actions and their effects on cost/loss metrics.

6.3 Quantum data constraints

The limitations of measurement and the no-cloning theorem complicate data annotation, as the latter states that it is not possible to create an exact copy of an arbitrary unknown quantum state [102]. This constraint affects the diagnostics and reproducibility of quantum data. In practice, repeated sampling, careful shot allocation and token-aware batching are necessary to derive statistics in the presence of noise.

Open issues. Protocols for data access that minimize state disturbance, labeling annotation workflows compatible with non-duplicable data and evaluation methodologies that reflect realistic sampling limits.

6.4 Synthetic data: validation, safety and quantum-native generation

Generative AI extends the data processing beyond cleaning, resampling and augmentation. This introduces the possibility of generating artificial data seeded from real datasets across many iterations and experiments. These generators align well with the *dc-qml* design philosophy because they introduce important data variability that can improve the performance of the QML model. However, AI-based data generation should be exploited with caution to avoid error amplification and downstream bias. Diffusion and multimodal methods, such as CLIP-guided generation, can broaden token coverage in classical to quantum workflows. QGANs provide quantum native generators with hardware-compatible training [68, 69, 81]. Early results showed that QGAN-based augmentation outperformed classical heuristics across computer vision and tabular time-series tasks. In parallel, domain-specific research has begun to quantify the statistical distance between real and synthetic data distributions [83, 103].

Open issues. Small-sample fidelity metrics suitable for QML, diagnostics of mode collapse and instability in QGANs, audits for bias and safety, and downstream monitoring that link synthetic generation to accuracy, robustness, depth and qubit usage.

6.5 Standardization, benchmarks and automation

Comparable claims require shared datasets, encoding protocols and evaluation metrics. Recent work has shown how encoding choices shift the accuracy using double-digit percentages in fixed circuits. This suggests that encoding-aware benchmarks are required [47]. MNISQ offers a large-scale circuit dataset for method comparison [32], and studies on data locality and QCNN design further motivate controlled data ablation [22, 46].

Open issues. Task-specific, encoding-aware benchmarks, *dc-qml* checklists in experimental work and AutoQML software tools that co-optimize data, encoding, and circuits.

6.6 Alignment with the *dc-qml* framework

To make *dc-qml* studies interpretable and reproducible, the following should be documented in the form of metadata [104]: (i) data lineage, data annotation protocols and changes made to dataset; (ii) tokenization granularity, quantum data encoding method family and data loading cost; (iii) QML model training "pathologies" and mitigations; (iv) synthetic-data generators, validation metrics, and safety audits for real-world deployments and pilots; (v) resource profiles that include qubit counts, circuit depth and transpilation results, together with system level placement of workflow tasks on CPU, GPU and QPU resources in-house or in cloud infrastructures [20, 21]; and (vi) ablations based on experimental factorial design that isolate data-side gains from model circuit-side gains.

7 Conclusions

This topical review addresses two questions: (1) What data requirements exist for representative QML methods? (2) How can a data-centric approach improve the quality and scalability of QML algorithms?

Regarding the first question, this review aims to describe the data preparation requirements of methods such as QSVMs, QNNs, and QGANs. Such methods require dimensionality reduction, efficient encoding schemes, balanced datasets and compression techniques to operate within the limits of NISQ hardware. The analysis showed that poorly conditioned data inflated circuit depth and qubit demands. In contrast, structured data quantum encodings and careful preprocessing improve accuracy.

To answer the second question, the proposed *dc-qml* framework was introduced as a "continuum" that connects data and models. This framework suggests that tokenization, modular encoding and synthetic data generation are equally important for quantum circuit refinement and evaluation.

This study recognized the importance of model-driven design for QML model training and online retraining. These practices remain core to current workflows. However, the results presented here suggest that QML performance and scalability also depend on a continuous feedback loop between data and model refinement.

In this sense, *dc-qml* does not replace the model-driven design. Instead, it provides a complementary approach in which data and models are co-designed within an iterative quantum–classical workflow.

Acknowledgment

The authors acknowledge the partial financial support from the Brazilian government through the Brazilian Federal Agency for Support and Evaluation of Graduate Education (CAPES). In addition, we thank Pedro Botelho for providing feedback on the earlier versions of this manuscript.

References

[1] M. Schuld and F. Petruccione, *Machine learning with quantum computers*. Springer, 2021, vol. 676.

[2] D. Peral-García, J. Cruz-Benito, and F. J. García-Peñalvo, "Systematic literature review: Quantum machine learning and its applications," *Computer Science Review*, vol. 51, p. 100619, 2024.

[3] Y.-Y. Hong and D. Josh Domingo Lopez, "A review on quantum machine learning in applied systems and engineering," *IEEE Access*, vol. 13, pp. 144 607–144 631, 2025.

[4] M. H. Jarrahi, A. Memariani, and S. Guha, "The principles of data-centric ai," *Commun. ACM*, vol. 66, no. 8, p. 84–92, Jul. 2023.

[5] Z. Abohashima, M. Elhosen, E. H. Houssein, and W. M. Mohamed, "Classification with quantum machine learning: A survey," 2020.

[6] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum machine learning," *Nature*, vol. 549, no. 7671, pp. 195–202, Sep 2017.

[7] J. Amin, M. Sharif, N. Gul, S. Kadry, and C. Chakraborty, "Quantum machine learning architecture for covid-19 classification based on synthetic data generation using conditional adversarial neural network," *Cognitive Computation*, vol. 14, no. 5, pp. 1677–1688, Sep 2022.

[8] S. Aaronson, "Read the fine print," *Nature Physics*, vol. 11, no. 4, pp. 291–293, Apr 2015.

[9] M. Schuld, I. Sinayskiy, and F. Petruccione, "An introduction to quantum machine learning," *Contemporary Physics*, vol. 56, no. 2, pp. 172–185, 2015.

[10] Z. Yang, M. Zolanvari, and R. Jain, "A survey of important issues in quantum computing and communications," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 2, pp. 1059–1094, 2023.

[11] H.-Y. Huang, M. Broughton, M. Mohseni, R. Babbush, S. Boixo, H. Neven, and J. R. McClean, "Power of data in quantum machine learning," *Nature communications*, vol. 12, no. 1, p. 2631, 2021.

[12] A. Tudisco, A. Marchesin, M. Zamboni, M. Graziano, and G. Turvani, "Evaluating angle and amplitude encoding strategies for variational quantum machine learning: their impact on model's accuracy," 2025.

[13] M. Benedetti, E. Lloyd, S. Sack, and M. Fiorentini, "Parameterized quantum circuits as machine learning models," *Quantum Science and Technology*, vol. 4, no. 4, p. 043001, nov 2019.

[14] M. Schuld, I. Sinayskiy, and F. P. and, "An introduction to quantum machine learning," *Contemporary Physics*, vol. 56, no. 2, pp. 172–185, 2015.

[15] S. A. Stein, R. L'Abbate, W.-Q. Mu, Y. Liu, B. Baheri, Y. Mao, Q. Guan, A. Li, and B. Fang, "A hybrid system for learning classical data in quantum states," *2021 IEEE International Performance, Computing, and Communications Conference (IPCCC)*, pp. 1–7, 2020.

[16] I. Kerenidis and A. Luongo, "Classification of the MNIST data set with quantum slow feature analysis," *Physical Review A*, vol. 101, no. 6, jun 22 2020.

[17] T. W. Maxwell, C. N. Azar, H. J., M. C. Floyd, H. L., S. M., and U. Muhammad, "Drastic Circuit Depth Reductions with Preserved Adversarial Robustness by Approximate Encoding for Quantum Machine Learning," *Intelligent Computing*, 2023.

[18] H. Bayraktar, A. Charara, D. Clark, S. Cohen, T. Costa, Y.-L. L. Fang, Y. Gao, J. Guan, J. Gunnels, A. Haidar, A. Hehn, M. Hohnerbach, M. Jones, T. Lubowe, D. Lyakh, S. Morino, P. Springer, S. Stanwyck, I. Terentyev, S. Varadhan, J. Wong, and T. Yamaguchi, "cuQuantum SDK: A High-Performance Library for Accelerating Quantum Science ," in *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*. Los Alamitos, CA, USA: IEEE Computer Society, Sep. 2023, pp. 1050–1061.

[19] K. Meichanetzidis, S. Gogioso, G. de Felice, N. Chiappori, A. Toumi, and B. Coecke, "Quantum natural language processing on near-term quantum computers," *Electronic Proceedings in Theoretical Computer Science*, vol. 340, p. 213–229, Sep. 2021.

[20] A. Liaqat, A. Darwish, A. Roman, and S. DiAdamo, "Qaoa in quantum datacenters: Parallelization, simulation, and orchestration," in *2025 IEEE International Conference on Quantum Software (QSW)*, 2025, pp. 195–205.

[21] N. Kakuko, S. Gokita, N. Masumoto, K. Matsumoto, K. Miyaji, T. Miyanaga, T. Mori, H. Nakayama, K. Sasada, Y. Takamiya, S. Tsukano, R. Uchida, and M. Yamaguchi, "A practical open-source software stack for a cloud-based quantum computing system," in *2025 IEEE International Conference on Quantum Computing and Engineering (QCE)*, vol. 01, 2025, pp. 797–803.

[22] C. Lee, I. F. Araujo, D. Kim, J. Lee, S. Park, J.-Y. Ryu, and D. K. Park, "Optimizing Quantum Convolutional Neural Network Architectures for Arbitrary Data Dimension," *arXiv.org*, 2024.

[23] T. Hur, L. Kim, and D. K. Park, "Quantum convolutional neural network for classical data classification," *Quantum Machine Intelligence*, vol. 4, no. 1, feb 10 2022.

[24] I. Kerenidis, A. Prakash, and D. Szilágyi, "Quantum algorithms for portfolio optimization," in *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, 2019, pp. 147–155.

[25] A. Kariya and B. K. Behera, "Investigation of quantum support vector machine for classification in nisq era," *arXiv preprint arXiv:2112.06912*, 2021.

[26] A. Saxena and S. Saxena, "Pancreatic Cancer Data Classification with Quantum Machine Learning," *Journal of Quantum Computing*, vol. 5, no. 1, pp. 1–13, 2023.

[27] Zoufal, Christa, "Generative Quantum Machine Learning," 2021.

[28] M. Theresa, "High-Dimensional Signal Processing using Classical-Quantum Machine Learning Pipelines with the TensorFlow Stack, Cirq-NISQ, and Vertica," *International Conference on Quantum Computing and Engineering*, 2022.

[29] S. Andrea, J. S., and D. V., "Quantum agents in the Gym: a variational quantum algorithm for deep Q-learning," *Quantum*, 2021.

[30] E. Andrés, M. P. Cuéllar, and G. Navarro, "Efficient Dimensionality Reduction Strategies for Quantum Reinforcement Learning," *IEEE Access*, vol. 11, pp. 104 534–104 553, 2023.

[31] S. Louis, A. A., J. C. Patrick, and C. M., "Entangled Datasets for Quantum Machine Learning," *arXiv.org*, 2021.

[32] L. Placidi, R. Hataya, T. Mori, K. Aoyama, H. Morisaki, K. Mitarai, and K. Fujii, "Mnisq: A Large-Scale Quantum Circuit Dataset for Machine Learning on/for Quantum Computers in the NISQ era," *arXiv.org*, 2023.

[33] D. Sierra-Sosa, M. Telahun, and A. Elmaghraby, "Tensorflow Quantum: Impacts of Quantum State Preparation on Quantum Machine Learning Performance," *IEEE Access*, vol. 8, pp. 215 246–215 255, 2020.

[34] M. Rupp, "Special issue on machine learning and quantum mechanics," *International Journal of Quantum Chemistry*, vol. 115, no. 16, pp. 1003–1004, 2015.

[35] Y.-C. C. Samuel, H. Chih-Min, H. Chia-Wei, and K. Y., "Hybrid quantum-classical classifier based on tensor network and variational quantum circuit," *arXiv.org*, 2020.

[36] C.-H. Yu, F. Gao, S. Lin, and J. Wang, "Quantum data compression by principal component analysis," *Quantum Information Processing*, vol. 18, no. 8, jul 1 2019.

[37] G. Schieffer, S. Markidis, and I. Peng, "Harnessing cuda-q's mps for tensor network simulations of large-scale quantum circuits," 2025.

[38] N. Thumwanit, C. Lortaraprasert, H. Yano, and R. Raymond, "Trainable Discrete Feature Embeddings for Quantum Machine Learning," in *2021 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, 10 2021, pp. 479–480.

[39] B. Kushal, M. Z. Kimberley, H. F. Daniel, M. Eni, G. V., L. T., and E. S., "Quantum Machine Learning Algorithms for Drug Discovery Applications," *Journal of Chemical Information and Modeling*, 2021.

[40] D. Sierra-Sosa, J. D. Arcila-Moreno, B. Garcia-Zapirain, and A. Elmaghraby, "Diabetes Type 2: Poincardata Preprocessing for Quantum Machine Learning," *Computers, Materials & Continua*, vol. 67, no. 2, pp. 1849–1861, 2021.

[41] S. Corli, L. Moro, D. Dragoni, M. Dispenza, and E. Prati, "Quantum machine learning algorithms for anomaly detection: A review," *Elsevier Future Generation Computer Systems*, vol. 166, p. 107632, 2025.

[42] E. Ovalle-Magallanes, D. E. Alvarado-Carrillo, J. G. Avina-Cervantes, I. Cruz-Aceves, and J. Ruiz-Pinales, "Quantum angle encoding with learnable rotation applied to quantum–classical convolutional neural networks," *Applied Soft Computing*, vol. 141, p. 110307, 7 2023.

[43] K. Batra, K. M. Zorn, D. H. Foil, E. Minerali, V. O. Gawriljuk, T. R. Lane, and S. Ekins, "Quantum machine learning algorithms for drug discovery applications," *Journal of chemical information and modeling*, vol. 61, no. 6, pp. 2641–2647, 2021.

[44] D. L. Bosco, B. Portelli, and G. Serra, "Integrated Encoding and Quantization to Enhance Quanvolutional Neural Networks," *arXiv.org*, 2024.

[45] A. D. Matthew, D. Kelvin, R. Dhruv, S. H. Joshua, R. L. Thomas, U. Fabio, and E. Sean, "Near-Term Quantum Classification Algorithms Applied to Antimalarial Drug Discovery," *Journal of Chemical Information and Modeling*, 2024.

[46] J. M., N. Alvir, J. Vinayak, L. David, K. Dylan, C. Manu, I. Ishraq, F. Audrey, S. Manish, B. Evan, V. Eade, A. Abina, and E.-A. E., "Leveraging Data Locality in Quantum Convolutional Classifiers," *Entropy*, 2024.

[47] M. Monnet, N. Chaabani, T.-A. Drăgan, B. Schachtner, and J. M. Lorenz, "Understanding the Effects of Data Encoding on Quantum-Classical Convolutional Neural Networks," in *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, sep 15 2024, pp. 1436–1446.

[48] D. Herman, R. Raymond, M. Li, N. Robles, A. Mezzacapo, and M. Pistoia, "Expressivity of Variational Quantum Machine Learning on the Boolean Cube," *IEEE Transactions on Quantum Engineering*, vol. 4, pp. 1–18, 2023.

[49] M. Raisa, M. Jishnu, F. S., and S. Mohammad, "A Parallel Quantum Feature Encoding Scheme for Effective Classical Data Classification in Quantum Convolutional Neural Networks," *IEEE Region 10 Conference*, 2023.

[50] S. Otgonbaatar, G. Schwarz, M. Datcu, and D. Kranzlmüller, "Quantum Transfer Learning for Real-World, Small, and High-Dimensional Datasets," *arXiv*, 2022.

[51] S. Otgonbaatar, G. Schwarz, M. Datcu, and D. Kranzlmüller, "Quantum transfer learning for real-world, small, and high-dimensional datasets," 09 2022.

[52] P. Dheeraj, B. V., J. Z., and A. V., "Tensor Ring Parametrized Variational Quantum Circuits for Large Scale Quantum Machine Learning," *arXiv.org*, 2022.

[53] Z. Amine, J. Zahi, and Q. M., "Quantum Machine Learning: Practical Cases," *International Symposium on INnovations in Intelligent SysTems and Applications*, 2022.

[54] M. Alam, S. Kundu, R. O. Topaloglu, and S. Ghosh, "Quantum-classical hybrid machine learning for image classification (iccad special session paper)," in *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. IEEE, 2021, pp. 1–7.

[55] S. A. Stein, R. L'Abbate, W. Mu, Y. Liu, B. Baheri, Y. Mao, Q. Guan, A. Li, and B. Fang, "A hybrid system for learning classical data in quantum states," 2021.

[56] S. A. Stein, "Quantum computing aided machine learning through quantum state fidelity," *Preprints*, March 2021.

[57] K. Saurabh, D. Siddharth, and B. D., "Supervised Learning Using a Dressed Quantum Network with "Super Compressed Encoding": Algorithm and Quantum-Hardware-Based Implementation," *arXiv.org*, 2020.

[58] K. Iordanis, L. Jonas, and P. A., "Quantum Algorithms for Deep Convolutional Neural Networks," *International Conference on Learning Representations*, 2019.

[59] V. Havlíček, A. Córcoles, K. Temme, A. Harrow, A. Kandala, J. Chow, and J. Gambetta, "Supervised learning with quantum-enhanced feature spaces," *Nature*, vol. 567, pp. 209–212, 03 2019.

[60] S. Endo, Z. Cai, S. C. Benjamin, and X. Yuan, "Hybrid quantum-classical algorithms and quantum error mitigation," *Journal of the Physical Society of Japan*, vol. 90, no. 3, p. 032001, 2021.

[61] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, "Quantum circuit learning," *Phys. Rev. A*, vol. 98, p. 032309, Sep 2018.

[62] V. Dunjko and H. J. Briegel, "Machine learning and artificial intelligence in the quantum domain: a review of recent progress," *Reports on Progress in Physics*, vol. 81, no. 7, p. 074001, jun 2018.

[63] C. Shani, D. Jurafsky, Y. LeCun, and R. Shwartz-Ziv, "From tokens to thoughts: How llms and humans trade compression for meaning," *arXiv preprint arXiv:2505.17117*, 2025.

[64] A. Carrera Vazquez, C. Tornow, D. Ristè, S. Woerner, M. Takita, and D. J. Egger, "Combining quantum processors with real-time classical communication," *Nature*, vol. 636, no. 8041, pp. 75–79, Dec 2024.

[65] M.-C. Roehsner, J. A. Kettlewell, J. Fitzsimons, and P. Walther, "Probabilistic one-time programs using quantum entanglement," *npj Quantum Information*, vol. 7, no. 1, p. 98, Jun 2021.

[66] A. Bardes, Q. Garrido, J. Ponce, X. Chen, M. Rabbat, Y. LeCun, M. Assran, and N. Ballas, "Revisiting feature prediction for learning visual representations from video," 2024.

[67] S. Raubitzek, S. Schrittwieser, A. Schatten, and K. Mallinger, "Quantum inspired kernel matrices: Exploring symmetry in machine learning," *Elsevier Physics Letters A*, vol. 525, p. 129895, 2024.

[68] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 684–10 695.

[69] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PmLR, 2021, pp. 8748–8763.

[70] D. Deutsch, "Quantum theory, the church–turing principle and the universal quantum computer," *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, vol. 400, no. 1818, pp. 97–117, 07 1985.

[71] V. Uotila, S. Sahri, and S. Groppe, "Utilizing quantum computing to improve the quality of data," in *Advances in Databases and Information Systems (ADBIS 2025)*, ser. Lecture Notes in Computer Science, P. K. Chrysanthis, K. Nørvåg, K. Stefanidis, and Z. Zhang, Eds. Springer, Cham, 2025, vol. 16043, pp. 280–294.

[72] M. Akrom, S. Rustad, H. K. Dipojono, R. Maezono, and H. Kasai, "Quantum machine learning for abo3 perovskite structure prediction," *Elsevier Computational Materials Science*, vol. 250, p. 113694, 2025.

[73] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023.

[74] Z. Zhang, Y. Wu, and X. Ma, "Quantum machine learning based wind turbine condition monitoring: State of the art and future prospects," *Elsevier Energy Conversion and Management*, vol. 332, p. 119694, 2025.

[75] A. Ambainis, D. Leung, L. Mancinska, and M. Ozols, "Quantum random access codes with shared randomness," *arXiv preprint arXiv:0810.2937*, 2008.

[76] B. Kushal, M. Z. Kimberley, H. F. Daniel, M. Eni, G. V., R. L. Thomas, and E. S., "Quantum Machine Learning for Drug Discovery," 2020.

[77] A. M., K. Satwik, T. R., and G. Swaroop, "Quantum-Classical Hybrid Machine Learning for Image Classification (ICCAD Special Session Paper)," *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, 2021.

[78] A. Berezutskii, M. Liu, A. Acharya, R. Ellerbrock, J. Gray, R. Haghshenas, Z. He, A. Khan, V. Kuzmin, D. Lyakh, D. Lykov, S. Mandrà, C. Mansell, A. Melnikov, A. Melnikov, V. Mironov, D. Morozov, F. Neukart, A. Nocera, M. A. Perlin, M. Perelshtein, M. Steinberg, R. Shaydulin, B. Villalonga, M. Pflitsch, M. Pistoia, V. Vinokur, and Y. Alexeev, "Tensor networks for quantum computing," *Nature Reviews Physics*, vol. 7, no. 10, pp. 581–593, Oct 2025.

[79] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021, pp. 8748–8763.

[80] D. Barta, D. Martyniuk, J. Jung, and A. Paschke, "Leveraging diffusion models for parameterized quantum circuit generation," 2025.

[81] M. Islam, S. Turkeli, and F. Ozaydin, "A survey of quantum generative adversarial networks: Architectures, use cases, and real-world implementations," 2025.

[82] R.-Z. He, J.-J. Su, S.-J. Qin, Z.-P. Jin, and F. Gao, "Qgan-based data augmentation for hybrid quantum–classical neural networks," *Elsevier Chinese Journal of Physics*, vol. 97, pp. 1453–1463, 2025.

[83] O. Pires, M. Nooblath, Y. Silva, M. Silva, L. Galvão, and A. Simen, "Synthetic data generation with hybrid quantum-classical models for the financial sector," *The European Physical Journal B*, vol. 97, 11 2024.

[84] I. Tyagin, M. H. Farag, K. Sherbert, K. Shirali, Y. Alexeev, and I. Safro, "Qaoa-gpt: Efficient generation of adaptive and regular quantum approximate optimization algorithm circuits," in *2025 IEEE International Conference on Quantum Computing and Engineering (QCE)*, vol. 01, 2025, pp. 1505–1515.

[85] I. C. Martin, S. Mukherjee, A. Baimagambetov, J. Vanschoren, and N. Polatidis, "Evolving machine learning: A survey," 2025.

[86] V. G. Pineda, A. Valencia-Arias, F. E. L. Giraldo, and E. A. Zapata-Ochoa, "Integrating artificial intelligence and quantum computing: A systematic literature review of features and applications," *Elsevier International Journal of Cognitive Computing in Engineering*, vol. 7, pp. 26–39, 2026.

[87] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, "Barren plateaus in quantum neural network training landscapes," *Nature Communications*, vol. 9, no. 1, p. 4812, Nov 2018.

[88] J. Cunningham and J. Zhuang, "Investigating and mitigating barren plateaus in variational quantum circuits: a survey," *Quantum Information Processing*, vol. 24, no. 2, p. 48, Jan 2025.

[89] S. Thanasilp, S. Wang, M. Cerezo, and Z. Holmes, "Exponential concentration in quantum kernel methods," *Nature Communications*, vol. 15, no. 1, p. 5200, Jun 2024.

[90] L. Bischof, S. Teodoropol, R. M. Füchslin, and K. Stockinger, "Hybrid quantum neural networks show strongly reduced need for free parameters in entity matching," *Scientific Reports*, vol. 15, no. 1, p. 4318, Feb 2025.

[91] A. Routh and J. Singh, "A hybrid quantum-classical language model for text generation: A case study on the gift of the magi," in *2025 IEEE MRIE Conference*, 2025, pp. 52–57.

[92] M. Uddin *et al.*, "Robust hybrid classical-quantum transfer learning model for text classification using gpt-neo 125m with lora & smote enhancement," *arXiv preprint arXiv:2501.10435*, 2025.

[93] S. Sharma *et al.*, "Quantum-enhanced natural language generation: A multi-model framework with hybrid quantum-classical architectures," *arXiv preprint arXiv:2508.21332*, 2025.

[94] S. Otgonbaatar, G. Schwarz, M. Datcu, and D. Kranzlmüller, "Quantum Transfer Learning for Real-World, Small, and High-Dimensional Remotely Sensed Datasets," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 16, pp. 9223–9230, 2023.

[95] X. He, K. Zhao, and X. Chu, "Automl: A survey of the state-of-the-art," *Knowledge-Based Systems*, vol. 212, p. 106622, 2021.

[96] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter, "Efficient and robust automated machine learning," in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28. Curran Associates, Inc., 2015.

[97] M. Roth, D. A. Kreplin, D. Basilewitsch, J. F. Bravo, D. Klau, M. Marinov, D. Pranjic, H. Stuehler, M. Willmann, and M.-A. Zöller, "Autoqml: A framework for automated quantum machine learning," 2025.

[98] K. A. Britt and T. S. Humble, "High-performance computing with quantum processing units," *J. Emerg. Technol. Comput. Syst.*, vol. 13, no. 3, Mar. 2017.

[99] R. Ramsauer and W. Mauerer, "Towards system-level quantum-accelerator integration," 2025.

[100] H. Safi, K. Wintersperger, and W. Mauerer, " Influence of HW-SW-Co-Design on Quantum Computing Scalability ," in *2023 IEEE International Conference on Quantum Software (QSW)*. Los Alamitos, CA, USA: IEEE Computer Society, Jul. 2023, pp. 104–115.

[101] K. Wei, H. Amano, R. Niwase, Y. Yamaguchi, and T. Miyoshi, "Qu-trefoil: Large-scale quantum circuit simulator working on fpga with sata storages," *IEEE Transactions on Computers*, vol. 74, no. 4, pp. 1306–1321, 2025.

[102] W. K. Wootters and W. H. Zurek, "A single quantum cannot be cloned," *Nature*, vol. 299, no. 5886, pp. 802–803, 1982.

[103] V. Hernandes and E. Greplova, "Exploring biological neuronal correlations with quantum generative models," *Cell Reports Physical Science*, vol. 6, no. 8, Aug 2025.

[104] A. Di Meglio, K. Jansen, I. Tavernelli, C. Alexandrou, E. Fernández-Combarro *et al.*, "Quantum computing for high-energy physics: State of the art and challenges," *PRX Quantum*, vol. 5, p. 037001, Aug 2024.