

Quantum feature-map learning with reduced resource overhead

Jonas Jäger^{1,2,*}, Philipp Elsässer^{3,4,*} and Elham Torabian^{4,2}

¹*Department of Computer Science and Institute of Applied Mathematics, University of British Columbia (UBC), Vancouver, British Columbia V6T 1Z4, Canada*

²*Stewart Blusson Quantum Matter Institute (QMI), Vancouver, British Columbia V6T 1Z4, Canada*

³*Institute of Physics, University of Freiburg, 79104 Freiburg (Breisgau), Germany*

⁴*Department of Chemistry, University of British Columbia (UBC), Vancouver, British Columbia V6T 1Z1, Canada*



(Received 7 October 2025; revised 2 March 2026; accepted 9 May 2026; published 5 June 2026)

Current quantum computers require algorithms that use limited resources economically. In quantum machine learning, success hinges on quantum feature maps, which embed classical data into the state space of qubits. We introduce Quantum Feature-Map Learning via Analytic Iterative Reconstructions (Q-FLAIR), an algorithm that reduces quantum resource overhead in iterative feature-map circuit construction. It shifts workloads to a classical computer via partial analytic reconstructions of the quantum model, using only a few evaluations. For each probed gate addition to the ansatz, the simultaneous selection and optimization of the data feature and weight parameter is then entirely classical. Integrated into quantum neural network and quantum kernel support vector classifiers, Q-FLAIR shows state-of-the-art benchmark performance. Since resource overhead decouples from feature dimension, we train a quantum model on a real IBM device in only 4 h, surpassing 90 % accuracy on the full-resolution MNIST dataset (784 features, digits 3 versus 5). Such results were previously unattainable, as the feature dimension prohibitively drives hardware demands for fixed and search costs for adaptive ansätze. Furthermore, Q-FLAIR demonstrates dequantization robustness against direct classical modeling, satisfying a benchmark rare in the literature and a necessary condition for potential quantum advantage. By rethinking feature-map learning beyond black-box optimization, this work takes a concrete step toward enabling quantum machine learning for real-world problems and near-term quantum computers.

DOI: [10.1103/v29j-rh32](https://doi.org/10.1103/v29j-rh32)

I. INTRODUCTION

Quantum machine learning (QML) aims to harness the computational and representational power of quantum computing and quantum information systems to enhance data-driven tasks in machine learning [1–3]. While quantum speedups over classical algorithms have traditionally been a central topic in QML [1,2], the focus of research has recently shifted toward representational advantages of quantum models [3–5]. Quantum models may provide access to richer model classes, potentially enhancing generalization and capturing complex data patterns and structure. Beyond practical motivations [6–9], this research draws on the theoretical premise that relevant datasets may stem from classically hard problems [10,11]. For recent comprehensive primers on QML, see Refs. [12,13].

However, the realization of such QML algorithms necessarily requires a mapping from the data (feature) space

$\mathcal{X} = \mathbb{R}^d$ into the exponentially large Hilbert space $\mathcal{H} = \mathbb{C}^{2^n}$ of n qubits. Such maps are known as *quantum feature maps* [4,5]. Concretely, a unitary transformation $U(\mathbf{x}, \boldsymbol{\theta})$ embeds classical data points $\mathbf{x} \in \mathcal{X}$ into $\mathcal{H}$ as quantum states of the form

$$|\psi(\mathbf{x}, \boldsymbol{\theta})\rangle = U(\mathbf{x}, \boldsymbol{\theta})|\psi_0\rangle. \quad (1)$$

They are prepared based on common parameters $\boldsymbol{\theta}$ and initial state $|\psi_0\rangle$. This work adopts the prominent gate-based quantum computing paradigm. Thus, the transformation $U(\mathbf{x}, \boldsymbol{\theta})$ is realized by a (variational or parametrized) quantum circuit, formed of multiple individual quantum gates ($U_1, U_2, \dots$) as

$$U(\mathbf{x}, \boldsymbol{\theta}) = \dots U_2(\theta_2, x_{k_2}) U_1(\theta_1, x_{k_1}). \quad (2)$$

The exact arrangement of gates is referred to as the *ansatz*.

Despite their central role in QML, there is, in general, no principled approach for designing suitable quantum feature maps, making this one of the most significant open challenges in the field [3]. The common practice of using fixed, often generic and problem-agnostic, circuit ansätze and only aligning the free (continuous) parameters $\boldsymbol{\theta}$ to the training data [14–26] often results in trainability issues and overly cumbersome models, which are inadequate for current hardware. The trainability issues arise particularly from the absence of inductive biases [27]. It is the lack of concrete prior knowledge (inductive bias) for the ansatz construction and related

*These authors contributed equally to this work.

[†]Contact author: jojaeger@cs.ubc.ca

[‡]Contact author: philipp.elsaesser@physik.uni-freiburg.de

shortcomings of fixed-ansatz approaches that motivate a fully data-driven quantum feature-map design, including the ansatz.

Recent efforts have therefore shifted toward automated and adaptive ansatz design, which iteratively modifies the ansatz to maximize relevant metrics. This approach achieves task-specific ansätze tailored for the dataset at hand, rather than optimizing the parameters alone, to balance expressivity, trainability, and hardware compatibility. Successful realizations of this *quantum architecture search* range in complexity from simple searches [28–30] over optimization guided by reinforcement learning [31], graph theory [32], or evolutionary and genetic algorithms [33–36], to graph neural network surrogate modeling [37]. However, the inherent compositional search over the (discrete) space of possible ansätze might quickly entail a combinatorial *blowup*, leading to detrimental run times. Especially considering the resource overhead scaling with the feature dimension d , their applicability to relevant high-dimensional datasets becomes impractical. Hence, learning quantum feature maps from scratch must be rethought from a resource perspective to make QML applicable in real-world scenarios in the near-term noisy intermediate-scale quantum (NISQ) or early fault-tolerant era.

We devise an algorithm, Quantum Feature-Map Learning via Analytic Iterative Reconstructions (Q-FLAIR), to learn quantum feature maps with the clear design goal of reducing quantum resource overhead. We view this reduction as twofold: First, Q-FLAIR conserves hardware resources, such as the number of gates and qubits required to implement the feature map. This controlled usage is achieved by a bottom-up design: Starting from an empty ansatz, Q-FLAIR incrementally increases complexity by allocating quantum resources only when required to improve model performance. Hence, feature-map circuits remain as shallow as possible, and qubits are entangled selectively. The latter allows to respect restrictive qubit topologies of current devices. A key characteristic of Q-FLAIR is that each growth step (iteration) involves the *simultaneous* (and greedy) optimization over three choices:

- (1) quantum gate choice (from a pool of M possibilities),
- (2) feature selection (of d features),
- (3) weight optimization (with $T_{\max}$ loss evaluation bound).

Second, Q-FLAIR reduces the number of circuit evaluations required for the actual feature-map learning process. Instead of naively querying the quantum model to determine the feature (2) and weight (3) for every gate candidate, we employ *analytic reconstructions* of the model output as a function of the gate rotation angle. While analytic reconstructions have been utilized in fixed-ansatz gradient-free optimization [38–41], and related ideas for adaptive ansätze in quantum simulation, e.g., in selection heuristics [42,43] for adaptive VQE [44], Q-FLAIR introduces this paradigm to QML to efficiently address its unique challenge of encoding classical input data. Specifically, this additional data dependence must be introduced carefully: A naive adaptation would compromise scalability with a quantum resource overhead in the data dimension d , preventing high-dimensional applications. Crucially, the fact that Q-FLAIR employs classical reconstructions of local gate effects does not render the resulting quantum circuit efficiently simulable. Necessarily, the use of universal gate sets generally rules out efficient exact classical simulation [45].

The analytic reconstructions are processed on a classical computer, efficiently off-loading choices (2) and (3). Choice (1) requires a constant overhead per gate candidate only to obtain a full set of reconstructions for a fixed-size batch of training data: one quantum circuit evaluation more than querying the candidate model output alone. This constant overhead stems from the functional class of sine curves induced by the gates studied here, although Q-FLAIR is not limited to such gates in principle. As a result, the scaling of circuit evaluations on the quantum computer reduces from $\mathcal{O}(MdT_{\max})$ to $\mathcal{O}(M)$ per iteration, shifting the $\mathcal{O}(dT_{\max})$ dependence entirely to classical processing. Figure 1 illustrates this shift: Whereas traditional architecture search repeatedly queries the quantum computer for all three choices, Q-FLAIR sweeps and reconstructs the gate pool once per iteration, thereby avoiding the aforementioned combinatorial blowup.

Q-FLAIR decouples quantum resource overhead from the feature dimension d through analytic reconstructions, a crucial advantage that avoids the detrimental twofold scaling overhead of prior quantum feature-map approaches on high-dimensional datasets. In fixed-ansatz feature maps, either the number of gates or qubits typically grows at least linearly with d [14–16], while adaptive ansatz approaches require circuit evaluation counts increasing with d [28,29,33] (cf. traditional learning in Fig. 1). *Feature selection* plays an implicit yet central role in Q-FLAIR. Feature selection techniques in classical machine learning can be applied to QML, as in prior works [30,46–49]. Some methods act as preprocessing, however, ignoring feedback from the downstream model (mutual information [50] or variance [51] filtering) or the supervision signal [dimensionality reduction such as principal component analysis (PCA) or autoencoders]. Other methods include the model in the loop (recursive feature elimination [52] or LASSO regularization [53]), reintroducing feature-dimension-dependent quantum resource scaling. In contrast, Q-FLAIR performs *quantum-aware* feature selection, meaning that it explicitly evaluates the precise (analytic) impact of candidate features on the quantum model output during training.

We utilize Q-FLAIR to learn quantum feature maps for the prominent quantum neural network (QNN) and quantum support vector machine (QSVM) classifiers (defined in Sec. II and connected to Q-FLAIR in Sec. III). In Sec. IV, we evaluate Q-FLAIR using established benchmark datasets [54], both in numerical simulations and on real NISQ hardware. Hereby, we demonstrate its practical applicability on the full-resolution MNIST dataset, achieving over 90 % accuracy in distinguishing handwritten digits 3 and 5 on NISQ devices for the first time. We also supplement experiments for extended analyses of Q-FLAIR. Beyond verifying practical hardware scalability, we conclude with empirical evidence for potential quantum advantage. Using an established benchmark [55], we rule out that, for most datasets, Q-FLAIR models could be readily dequantized. This algorithmic advantage contrasts both the data-restricted regime of the to us only known prior positive benchmark outcome [56] and the readily dequantized fixed-ansatz models benchmarked originally [55]. The broader relevance of Q-FLAIR and the results are discussed in Sec. V.

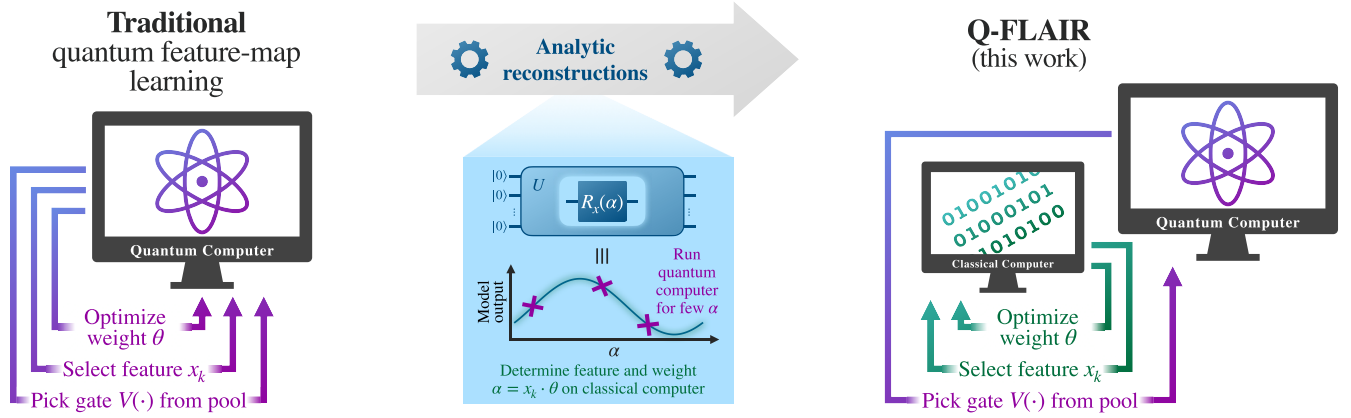


FIG. 1. Schematic comparison of traditional quantum feature-map learning and Q-FLAIR. The figure illustrates the distribution of computational workloads between quantum (violet) and classical (green) computers. Multiple subroutines that are repeatedly executed on a quantum computer in traditional approaches are shifted to classical computers in Q-FLAIR. By retrieving partial analytic reconstructions of the quantum model from only a few quantum evaluations (blue box), Q-FLAIR reduces the resource overhead of quantum feature-map learning.

II. BACKGROUND

To learn a (binary) classification model from data, a training dataset $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$ is provided, consisting of N pairs of input data points $\mathbf{x}_i$ of dimension d and true classification labels $y_i \in \{-1, +1\}$. The data dimension d is also referred to as the number of input features. The two QML methods employed in this work, which integrate quantum feature maps to construct classification models, are briefly presented here.

A. Quantum neural networks

QNN [57,58] are a widely adopted analog of classical (deep) neural networks, where trainable quantum gates replace artificial neurons. To clarify the various definitions of QNNs, variational quantum classifier [4,58] is a more specific name for the QNN paradigm used in the present work, and is defined as follows. The QNN output is obtained from an observable O as the expectation value when measured after the data point $\mathbf{x}$ has been embedded via the quantum feature map,

$$m(\mathbf{x}; \theta) = \langle \psi_0 | U^\dagger(\mathbf{x}, \theta) O U(\mathbf{x}, \theta) | \psi_0 \rangle. \quad (3)$$

The choice of the observable O should be subject to $\|O\|_{\text{op}} = 1$ to bound the model output to the interval $[-1, 1]$. Thresholding this output, defined as

$$m_{\pm}(\mathbf{x}; \theta) = \text{sign}[m(\mathbf{x}; \theta) - b], \quad (4)$$

provides the binary classification prediction for a data point $\mathbf{x}$. The threshold parameter $b \in [-1, 1]$ is also learnable.

To train the QNN, a loss function is defined to assess the fit of the model to the dataset $\mathcal{D}$. The standard choice in classification tasks is the *negative log-likelihood loss* [59], or solely *log loss*, between a true label y and model prediction $\hat{y}$:

$$l_{\log}(y, \hat{y}) = -\ln(0.5 + y\hat{y}/2). \quad (5)$$

Rescaling inside the $\ln$ interprets the model output $\hat{y} = m(\mathbf{x}; \theta)$ as the probability that the data point $\mathbf{x}$ is of class ± 1 :

$$\Pr(y = \pm 1 | \mathbf{x}) = 0.5 \pm m(\mathbf{x}; \theta)/2. \quad (6)$$

Compared to a linear loss function [57], low-probability predictions for the true class y , leading to a misclassification, dominate the log loss nonlinearly.

For generalization to the entire data distribution, the model should ideally learn to minimize the *expected loss*

$$\mathcal{L}_{\text{in}} = \mathbb{E}_{\mathbf{x}, y}[l_{\text{in}}(y, m(\mathbf{x}; \theta))]. \quad (7)$$

Given that this data distribution is presumed to be inaccessible in applications, the expected loss is estimated from finite samples (training data $\mathbf{x}_i, y_i \in \mathcal{D}$) and predictions $\hat{y}_i = m(\mathbf{x}_i; \theta)$:

$$L_{\text{in}}(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{N} \sum_{i=1}^N l_{\text{in}}(y_i, \hat{y}_i) \approx \mathcal{L}_{\text{in}}, \quad (8)$$

which is known as the *empirical loss*.

B. Quantum kernel support vector machines

Quantum kernel support vector machines (QSVMs) or, more specifically, quantum kernel support vector *classifiers*, quantum enhance the classical support vector machine (SVM) algorithm [4,5,25]. This is achieved by providing access to a quantum kernel function, which is thus evaluated on a quantum computer and constructed from quantum feature maps as

$$\kappa(\mathbf{x}, \mathbf{x}'; \theta) = |\langle \psi(\mathbf{x}', \theta) | \psi(\mathbf{x}, \theta) \rangle|^2 \quad (9)$$

$$= |\langle \psi_0 | U^\dagger(\mathbf{x}', \theta) U(\mathbf{x}, \theta) | \psi_0 \rangle|^2. \quad (10)$$

Note that other quantum kernel formulations exist besides this quantum *fidelity* kernel definition [60]. This QSVM paradigm leverages the kernel trick concept in SVMs to perform a linear classification in a potentially high-dimensional feature space, such as the quantum state space in the case of quantum kernels. The “trick” is that the feature space does not have to be accessed explicitly, but only kernel values $\kappa(\mathbf{x}, \mathbf{x}'; \theta)$ of pairs of data points $\mathbf{x}, \mathbf{x}'$ have to be known for both learning and inference. Learning such a linear SVM classification is efficient via quadratic optimization with desirable generalization guarantees, such as the maximum margin property [61]. At the same time, it is capable of modeling complex

decision boundaries in the original data space. The QSVM model provides the binary classification prediction for a data point $\mathbf{x}$ as

$$m_{\pm} = \text{sign} \left[\sum_{i=1}^N a_i y_i \kappa(\mathbf{x}_i, \mathbf{x}; \boldsymbol{\theta}) + b \right]. \quad (11)$$

This decision is based on the *support vectors* among the training data points $\mathbf{x}_i$ corresponding to a nonzero parameter a_i . QSVM training yields these parameters $\mathbf{a}$ (and b), which is classical given the fixed kernel matrix $K = [\kappa(\mathbf{x}_i, \mathbf{x}_j, \boldsymbol{\theta})]_{i,j=1}^N$, with details omitted here [62]. Importantly, the parameters $\boldsymbol{\theta}$ of the quantum kernel/feature map are unaffected.

For the QSVM to be successful, it is crucial to choose a suitable quantum feature map $U(\mathbf{x}, \boldsymbol{\theta})$ to construct the quantum kernel $\kappa(\mathbf{x}_i, \mathbf{x}_j, \boldsymbol{\theta})$. However, this is usually unknown *a priori*. Therefore, kernel target alignment (TA) formulates an (empirical) loss [63] that can be leveraged to learn the quantum kernel parameters $\boldsymbol{\theta}$ and/or construction U from data:

$$L_{\text{TA}}(\mathbf{y}, K) = -\frac{\mathbf{y}^T K \mathbf{y}}{N \|K\|_F}. \quad (12)$$

This formulation is motivated by the fact that nontrivial quantum feature maps should achieve a low fidelity for $|\psi(\mathbf{x}_i, \boldsymbol{\theta})\rangle$ and $|\psi(\mathbf{x}_j, \boldsymbol{\theta})\rangle$ when the data points $\mathbf{x}_i, \mathbf{x}_j$ belong to different classes, reflected in a low kernel value. Conversely, data points $\mathbf{x}_i, \mathbf{x}_j$ belonging to the same class should yield a higher kernel value by bringing their quantum feature states closer together. The TA problem formulation can also be viewed as a separate classification problem in its own right. Concretely, the input data consist of data point tuples $(\mathbf{x}_i, \mathbf{x}_j)$ along with ± 1 class labels $y_i y_j$. Ideally, the quantum kernel could be aligned with a target kernel that is 1 for data points in the same class ($y_i y_j = 1$) and zero otherwise ($y_i y_j = -1$).

III. METHOD

The proposed algorithm, Q-FLAIR, is based on the fundamental connection that the expectation value (of an observable) as a function of the angle in a rotation gate in the quantum circuit has a sinusoidal form [38–41]. Formally, consider any unitary gate being generated by a Hermitian operator A as

$$R(\alpha) = \exp\left(-\frac{i}{2}\alpha A\right), \quad \text{with } A^2 = I. \quad (13)$$

When acting with $R(\alpha)$ on some state $|\phi\rangle$, the expectation value of any observable $\hat{M}$ obeys an analytic form with respect to the (independent) rotation angle α , reading as

$$\langle \phi | R^\dagger(\alpha) \hat{M} R(\alpha) | \phi \rangle = a \sin(\alpha - b) + c. \quad (14)$$

Solely, estimations of the expectation value at three different rotation angle positions α on the quantum computer can determine the coefficients $a, b, c \in \mathbb{R}$, and, hence, be utilized to obtain a reconstruction $f(\alpha)$. Concretely, given expectation values z_0, z_+ , and z_- for the angles $\alpha_0, \alpha_0 + \pi/2$, and $\alpha_0 - \pi/2$, respectively, Ref. [38] provides direct formulas for the coefficients

$$a = \frac{1}{2} \sqrt{(2z_0 - z_+ - z_-)^2 + (z_+ - z_-)^2}, \quad (15)$$

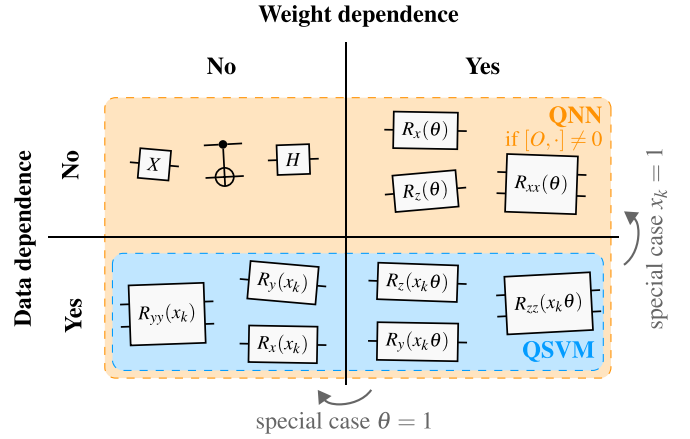


FIG. 2. Types of variational dependence in gates. Gates may depend on a data feature x_k , a weight parameter θ , both, or neither (fixed, nonvariational). Colors indicate which gate types contribute in Q-FLAIR for QNN (orange) and QSVM (blue). Arrows denote reductions to special cases by setting variational parameters to 1.

$$b = \alpha_0 - \arctan2(2z_0 - z_+ - z_-, z_+ - z_-), \quad (16)$$

$$c = \frac{1}{2}(z_+ + z_-). \quad (17)$$

Appendix A 1 details the reconstruction and implementation.

Before introducing further details of Q-FLAIR, we define the different types of gates used. Figure 2 provides a compact summary. In many variational quantum algorithms, the gates depend on a weight parameter θ or are fixed entirely. To map the data to quantum feature states in QML, however, gates may also depend on a (classical) data feature, i.e., one of the d components x_k of the data points $\mathbf{x}$. Therefore, four types of gates are possible: fixed gates, purely weight-dependent gates as defined in Eq. (13), purely data-dependent gates by setting the rotation angle α in Eq. (13) to a data feature x_k , and both weight-data-dependent gates. The latter type is of the form

$$R(\theta, x_k) = \exp\left(-\frac{i\theta x_k}{2} A_j\right), \quad \text{with } A^2 = I. \quad (18)$$

Due to the 2π gate periodicity, we assume features to be normalized to $[-\pi, \pi]$, where rescaling can be inferred from the training data $\mathcal{D}$ in advance. Weights are constrained to $[-1, 1]$.

Q-FLAIR learns the quantum feature map of a QML method with an adaptive circuit ansatz, rather than optimizing circuit parameters alone. Initialized with the empty ansatz $U^{(1)}(\mathbf{x}, \boldsymbol{\theta}^{(1)}) = I$, Q-FLAIR grows the ansatz iteratively, probing possible candidates from a pool $\mathcal{V}$ of M gates. When selecting the (1) gate that yields the largest immediate loss reduction, analytic reconstructions [Eq. (14)] yield the optimal (2) feature and (3) weight at no extra quantum cost. Concretely, in the ℓ th iteration of the algorithm, the M possible candidates $V_m \in \mathcal{V}$ are appended to the current circuit $U^{(\ell-1)}(\mathbf{x}_i; \boldsymbol{\theta})$:

$$V_m(\alpha) U^{(\ell)}(\mathbf{x}_i; \boldsymbol{\theta}) |\psi_0\rangle = V_m(\alpha) |\psi^{(\ell)}(\mathbf{x}_i; \boldsymbol{\theta})\rangle. \quad (19)$$

To evaluate these candidate quantum feature maps, a case distinction follows, based on the type of gate V_m (see Fig. 2).

If gate V_m is fixed, evaluate the loss function under this m th candidate feature map, denoted by L_m^* , which involves quantum circuit evaluations for *each* training data point $\mathbf{x}_i$. If gate V_m is instead variational in α , reconstruct ($\equiv$) the model output under this m th candidate for *each* training data point $\mathbf{x}_i$:

$$f_{m,i}(\alpha) \equiv \langle \psi^{(\ell)}(\mathbf{x}_i; \boldsymbol{\theta}) | V_m^\dagger(\alpha) \hat{M} V_m(\alpha) | \psi^{(\ell)}(\mathbf{x}_i; \boldsymbol{\theta}) \rangle. \quad (20)$$

This reconstruction requires two additional estimates (of the expectation value) of the candidate quantum circuit under shifting α , described in Appendix A 1.

The remaining evaluation of this m th gate candidate relies solely on *classical* processing of the reconstructions $f_{m,i}(\alpha)$, requiring no further quantum hardware queries. If the gate V_m is both weight- and data-dependent, we substitute $\alpha = \theta_{x_{i,k}}$ in $f_{m,i}(\alpha)$ to classically infer the model output reconstruction for each feature index k as a function of θ . We denote this as $f_{m,i,k}(\theta) = f_{m,i}(\theta_{x_{i,k}})$. Conversely, if V_m is weight- but not data-dependent, we simply set $\alpha = \theta$ to obtain $f_{m,i}(\theta)$.

Consequently, the (empirical) loss $L(\mathbf{y}, \hat{\mathbf{y}})$ can be formulated entirely in closed form. By relating the candidate reconstructions, which vary in θ (and k), to the true training labels $\mathbf{y}$, we construct the m th candidate loss function L_m as

$$L_m(\theta, k) = L(\mathbf{y}, [f_{m,i,k}(\theta)]_{i=1}^N). \quad (21)$$

For data-independent gates, the feature index k is naturally omitted, yielding $L_m(\theta)$. The optimization problem for (2) feature and (3) weight choice is *classical* as

$$\text{minimize}_{k \in \{1, \dots, d\}} (\text{minimize}_{\theta \in [-1, 1]} L_m(\theta, k)), \quad (22)$$

where θ_m^* (and k_m^*) denote optimal solutions with value L_m^* . Since the inner optimization generally has no closed-form solution, it can be solved numerically or by fine sampling with modest classical effort due to its one-dimensional bounded domain. The outer optimization, relevant for data-dependent gates only, selects the feature index k corresponding to the lowest inner-optimal value. Together with the fixed gate loss evaluations L_m^* , the optimal (1) gate choice minimizes the loss:

$$m^* \in \arg \min \{L_m^*\}_{m=1}^M. \quad (23)$$

We then append the selected gate V_{m^*} to the current feature-map ansatz $U^{(\ell)}$, record the feature index $k_{m^*}^*$, and extend the parameter vector $\boldsymbol{\theta}$ with $\theta_{m^*}^*$ if applicable. This process repeats, adding candidates from the pool until convergence, i.e., until the loss improvement falls below a threshold ΔL .

To apply Q-FLAIR to a QML method, it must be ensured that appending gate candidates (with a generator $A^2 = I$) to the feature-map circuit obeys the form of a measurement

expectation value after a state was evolved by this gate as in Eq. (14). We will provide such verification along with implementation details for the QNN and QSVM in the following two sections. Figure 2 highlights differences in terms of gate types.

A. Q-FLAIR for quantum neural networks

The immediate applicability of Q-FLAIR to the QNN method is evident by comparing the model output in Eq. (3) to the required form in Eq. (14). They coincide when appending a rotation gate $R(\alpha)$ to the end of an (arbitrary but fixed) quantum feature-map circuit $U(\mathbf{x}, \boldsymbol{\theta})$:

$$m(\mathbf{x}; \boldsymbol{\theta}, \alpha) = \langle \psi(\mathbf{x}, \boldsymbol{\theta}) | R^\dagger(\alpha) O R(\alpha) | \psi(\mathbf{x}, \boldsymbol{\theta}) \rangle, \quad (24)$$

with $|\psi(\mathbf{x}, \boldsymbol{\theta})\rangle = |\phi\rangle$ and $O = \hat{M}$ between Eqs. (3) and (14). Any loss based on the QNN output $m(\mathbf{x}; \boldsymbol{\theta})$, such as the log loss, can then be assessed classically via analytic reconstructions of the form in Eq. (14) for the QNN output.

Since Q-FLAIR adds gates exclusively to the end of the quantum feature-map circuit, any gate V_m in the pool $\mathcal{V}$ must not commute with the observable O , $[V_m, O] \neq 0$. Otherwise, such gates cannot impact the model output. Therefore, it is essential to carefully balance both the pool design and observable choice. The gate pool used in our experiments comprises

$$\mathcal{V} = \{R_x(\theta), R_y(\theta), R_x(\theta, x_k), R_y(\theta, x_k), R_{xx}(\theta), R_{yy}(\theta), R_{xx}(\theta, x_k), R_{yy}(\theta, x_k), H\}. \quad (25)$$

The two-qubit gates in this pool always act on two neighboring qubits. While all gates exhibit sine-curve reconstructions, the gate set universality [45] is crucial to preclude efficient exact classical simulation of the full learned quantum feature maps in general. The observable chosen is the all-zero-state projector $O = (|0\rangle\langle 0|)^{\otimes n}$, which realizes the probability of observing all n qubits in zero upon a computational basis measurement.

B. Q-FLAIR for quantum kernel support vector machines

To confirm the applicability of Q-FLAIR to the QSVM method, we verify that the quantum kernel function as defined in Eq. (10) obeys the required form as specified in Eq. (14). Compared to the QNN model output, we always encode pairs of data points $\mathbf{x}_i, \mathbf{x}_{i'}$ in the quantum feature map $U(\mathbf{x}, \boldsymbol{\theta})$. Therefore, we append rotation gates (with identical generator), but with rotation angles that may differ, i.e., $R(\alpha_i)$ and $R(\alpha_{i'})$, to encode x_i and $x_{i'}$ into their respective quantum feature states. The required form of Eq. (14) is then recovered as follows:

$$\kappa(\mathbf{x}_i, \mathbf{x}_{i'}; \boldsymbol{\theta}, \alpha_i, \alpha_{i'}) = |\langle \psi(\mathbf{x}_{i'}; \boldsymbol{\theta}) | R^\dagger(\alpha_{i'}) R(\alpha_i) | \psi(\mathbf{x}_i; \boldsymbol{\theta}) \rangle|^2 \quad (26)$$

$$= |\langle \psi(\mathbf{x}_{i'}; \boldsymbol{\theta}) | R(\alpha_i - \alpha_{i'}) | \psi(\mathbf{x}_i; \boldsymbol{\theta}) \rangle|^2 \quad (27)$$

$$= \underbrace{\langle \psi(\mathbf{x}_i; \boldsymbol{\theta}) |}_{=|\phi\rangle} R^\dagger(\alpha_i - \alpha_{i'}) \underbrace{|\psi(\mathbf{x}_{i'}; \boldsymbol{\theta})\rangle}_{=\hat{M}} \underbrace{\langle \psi(\mathbf{x}_i; \boldsymbol{\theta}) |}_{=|\phi\rangle} R(\alpha_i - \alpha_{i'}) \underbrace{|\psi(\mathbf{x}_{i'}; \boldsymbol{\theta})\rangle}_{=|\phi\rangle}. \quad (28)$$

Two properties of rotation gates are applied in Eq. (27). First, the adjoint flips the rotation direction $R^\dagger(\varphi) = R(-\varphi)$.

Second, two subsequent rotation gates with matching generators combine into a single rotation as $R(\alpha)R(\beta) =$

$R(\alpha + \beta)$. Equation (28) uses the property $|z|^2 = zz^*$ for any $z \in \mathbb{C}$.

Compared to the QNN derivation in Eq. (24), not only does the state depend on a data point $\mathbf{x}_i$, but the observable also depends on a second data point $\mathbf{x}_{i'}$. The application of Q-FLAIR to QSVMs becomes more transparent when the kernel target alignment setting is viewed as a classification problem, as explained in Sec. II B. Thus, per probed gate candidate, an analytic reconstruction of the quantum kernel is required for each *pair* of data points (as opposed to a single data point). For weight-data-dependent gates as in Eq. (18), the appended rotation gate then depends on the feature difference of the two data points weighted by a parameter $\theta(x_{i,k} - x_{i',k})$, which is what the rotation angle α is substituted by when processing the analytic reconstruction classically for such gates.

A well-known phenomenon, sometimes referred to as *gate erasure bug* [64,65], occurs when constructing quantum (fidelity) kernels directly from quantum feature maps $U(\mathbf{x}, \theta)$. Precisely, data-independent gates at the end of the quantum feature-map circuit cancel out, due to the adjoint construction in the quantum kernel, as it becomes evident when setting $\alpha_i = \alpha_{i'}$ in Eq. (26). Since we use Q-FLAIR to grow the quantum feature-map circuits by adding gates from the pool at the end, we therefore restrict the gate pool to gates that always exhibit data dependence. Data dependence is achieved by either weight-data-dependent gates as in Eq. (18) or by fixed gates preceding a (weight-) data-dependent gate [66]. Eventually, the gate pool used in our experiments comprises

$$\mathcal{V} = \{R_z(\theta, x_k), (CZ, H, R_z(\theta, x_k)) \\ R_{xx}(\theta, x_k), R_{yy}(\theta, x_k), R_{zz}(\theta, x_k), \}. \quad (29)$$

Like in the QNN gate pool, the two-qubit gates only act on neighboring qubits, except the CZ gate combination, which connects qubits with up to two qubits between them. Furthermore, all gates possess sine-curve reconstructions and constitute a universal gate set [45] to prevent exact classical simulability of the learned feature-map circuits.

IV. EXPERIMENTAL RESULTS

We present experimental results and analyses for Q-FLAIR, starting with demonstrating the algorithm's ability to learn high-quality QNN and QSVM models. For QNNs, we then illustrate the scalability of Q-FLAIR to high-dimensional, real-world datasets and conduct an ablation study to analyze the impact of individual components within its simultaneous optimization mechanism. We finally report benchmarking results of Q-FLAIR on a real IBM quantum computer for both QNN and QSVM. See Appendix B 1 for hyperparameter settings.

The performance and flexibility of Q-FLAIR are evaluated on a variety of datasets, summarized in Table I. Four of these are well-established classification benchmarks [54]: *Linearly separable* is easily solved with classical models, but has been reported to pose challenges for QML approaches [54]. *Bars & stripes* consists of images designed explicitly to test translational invariance in models such as convolutional architectures. *Two-curves* is based on a work in which the influence of the curvature of two curves on the accuracy of

TABLE I. Details of datasets employed in this work. Reported are the number (#) of training and test samples, the feature dimension d , and the color used in plots for each dataset. For *MNIST PCA*, the QNN is trained and tested on the full datasets, while the QSVM uses smaller subsets. The data are based on PennyLane datasets [67], except for *MNIST* 28×28 and its downsampled variants (*MNIST* 7×7 and *MNIST* 14×14), which are based on the raw MNIST data [68]. All MNIST datasets are restricted to two classes (digits 3 and 5).

Dataset	# Training	# Test	Dimension d	Color
<i>Linearly separable</i>	1000	1000	10	orange
<i>Two-curves</i>	1000	1000	10	blue
<i>Bars & stripes</i>	1000	200	16	red
<i>MNIST PCA</i>	11 552/1000	1902/1000	10	purple
<i>MNIST</i> 7×7	1000	1000	49	light green
<i>MNIST</i> 14×14	1000	1000	196	green
<i>MNIST</i> 28×28	1000	1000	784	dark green

a neural network has been investigated [69]. *MNIST PCA* is a dimensionality-reduced version, using principal component analysis, of the MNIST handwritten digit data [68] restricted to the digits 3 and 5. This classification task is considered one of the most challenging digit pairings [54]. We further extend the analysis of Q-FLAIR to the MNIST in full 28×28 -pixel resolution [68]. To specifically examine scaling behaviors with the feature dimension, we study downsampled resolutions labeled as *MNIST* 7×7 and *MNIST* 14×14 . This dataset not only extends our study to real-world data, as opposed to the synthetic or preprocessed datasets, but data of significantly higher dimensionality (up to $d = 784$) than the other benchmarking datasets $d \leq 16$.

We report accuracies using *average (balanced)* accuracy,

$$\bar{A} = \frac{1}{2}(t_p + t_n), \quad (30)$$

averaging over the ratio of correct predictions t_p and t_n for the positive and negative classes, respectively. Since the datasets we use deviate up to 6 % from perfect balance, this measure avoids the bias that standard accuracy would introduce.

A. Q-FLAIR performance benchmark for QNN and QSVM

For the four benchmarking datasets, Figs. 3 and 4 show the accuracies achieved with Q-FLAIR for QNNs and QSVMs, together with the gate-by-gate construction of the learned quantum feature maps. With QNNs, test accuracies above 92 % are reached in all tasks except for *two-curves*, which saturates at 85 %. The best result is obtained on *linearly separable*, with a test accuracy of 96 %. Overfitting is negligible across datasets as the gap between final training and test accuracies never exceeds 1.5 %. With QSVMs, test accuracies above 89 % are achieved on all datasets. The best accuracy on *two-curves*, with 99 %, marks a sharp improvement over the QNN setting, where this dataset had the lowest accuracy, and even surpasses the QNNs on any dataset. The weakest QSVM result is on *bars & stripes*. Unlike QNNs, QSVMs show slight overfitting (train-test gap of up to 5 % on *linearly separable*).

Both methods achieve similar accuracies, with QSVM performing better on *two-curves* and QNN more reliably capturing *bars & stripes*. The strong QSVM performance on

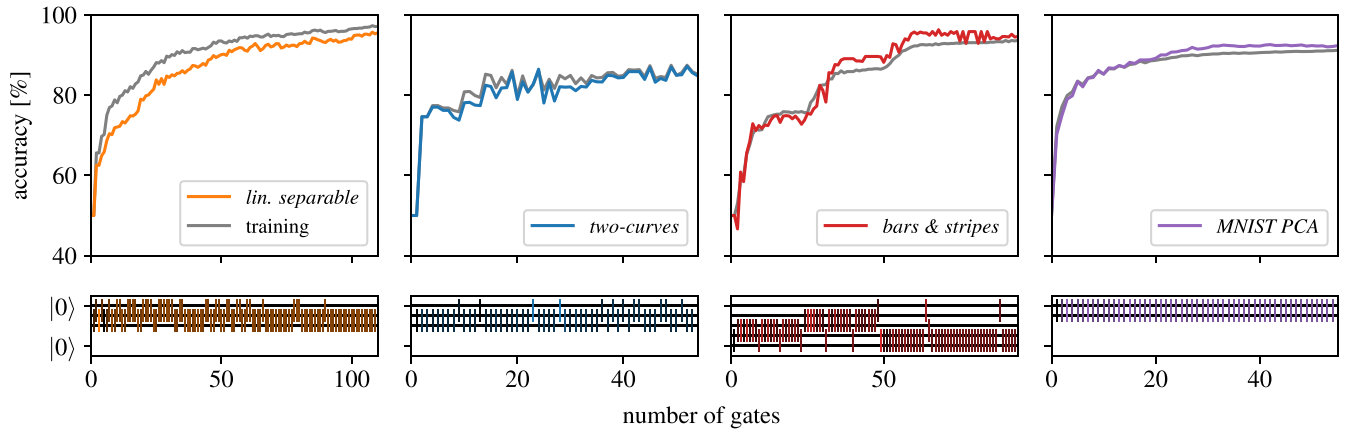


FIG. 3. QNN performance benchmark on four different datasets. Top panels: Accuracy over the number of gates appended to the feature-map circuit by Q-FLAIR, where each point corresponds to the QNN with the intermediate circuit obtained after each gate addition. Gray curves show training accuracy, and colored curves show test accuracy. Bottom panels: The corresponding feature-map circuits built gate by gate with Q-FLAIR. Each rectangle denotes a gate (single or two qubit), with coloring indicating its parameter value between -1 (black) and 1 (color).

two-curves is notable given its reported difficulty, though this complexity measure [70] should be interpreted with caution [54]. For instance, despite the good performance of Q-FLAIR for both models, *linearly separable* poses difficulties for some QML methods [54], but can be readily learned by most classical models. Appendix C 2 shows that reoptimizing weights offers negligible gains, indicating near-optimal weights.

Comparing the quantum circuits learned for the QNN and QSVM models (cf. bottom panels in Figs. 3 and 4) reveals clear differences. On the one hand, the circuit *width* (i.e., number of qubits) is typically higher for QSVM. Two QSVM circuits (Fig. 4) use TEN qubits, reaching the maximum available in the Q-FLAIR simulations, whereas the QNN circuits (Fig. 3) never exceed five qubits, as observed for *bars & stripes*. In this dataset, the QNN makes particular use of entangling qubits, with each qubit addition corresponding to

a noticeable increase in accuracy. On the other hand, the QSVM circuit *depth* (i.e., number of gates) is yet lower across all datasets. None of the QSVM circuits use more than 25 gates, while the QNN circuit for *linearly separable* exceeds 100 gates before the loss function change falls below the convergence threshold. Two subtleties must be considered, however, when comparing circuit depths in this context. First, the convergence threshold itself drives the circuit length, and because it is applied to different loss functions for QNN and QSVM, the depth is not directly comparable. To address this, Appendix C 3 analyzes the minimum circuit depth needed to reach certain accuracy levels, confirming that QSVM circuits are shallower in the higher accuracy regime (above 75 %). Second, the effective depth of QSVM circuits is doubled because each kernel entry corresponds to preparing two distinct states, whereas QNNs use only one state per data point.

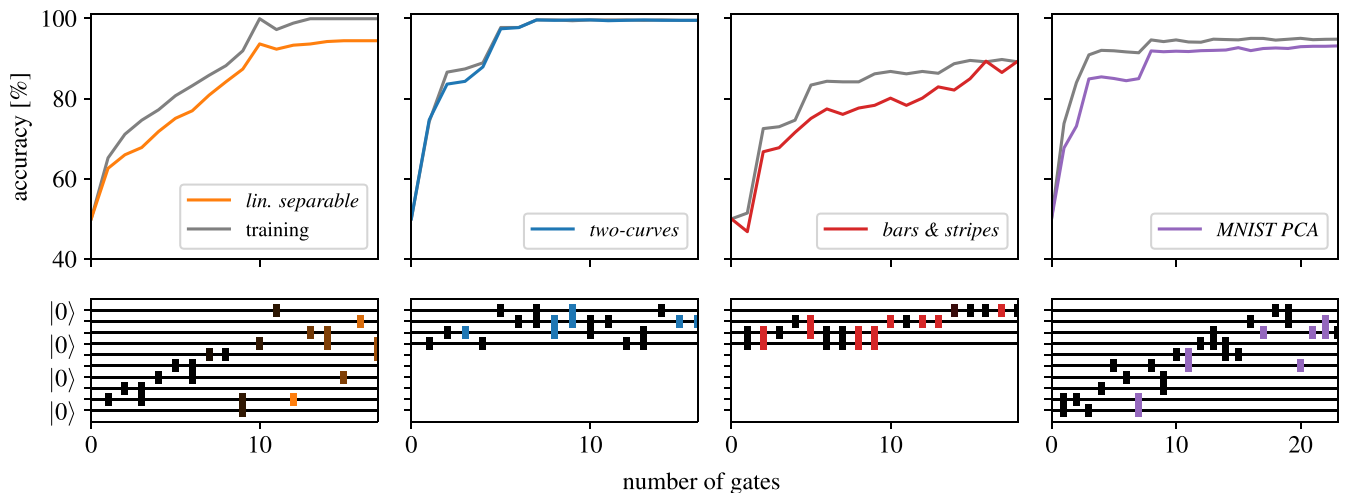


FIG. 4. QSVM performance benchmark on four different datasets. Top panels: Accuracy over the number of gates appended to the feature-map circuit by Q-FLAIR, where each point corresponds to the QSVM with the intermediate circuit obtained after each gate addition. Gray curves show training accuracy, and colored curves show test accuracy. Bottom panels: The corresponding feature-map circuits built gate by gate with Q-FLAIR. Each rectangle denotes a gate (single or two qubit), with coloring indicating its parameter value between -1 (black) and 1 (color).

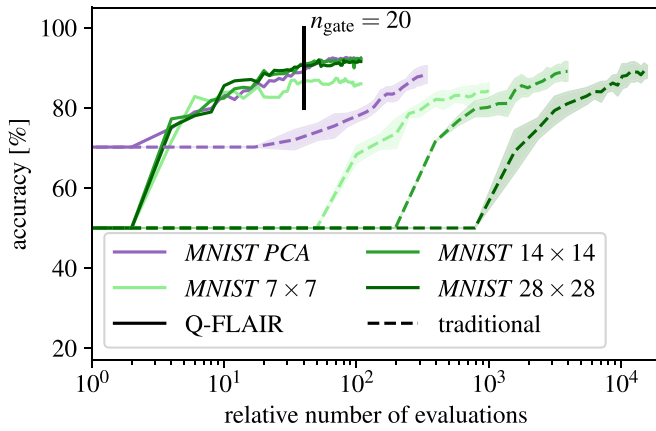


FIG. 5. Comparison of Q-FLAIR with traditional feature-map learning on MNIST (digits 3 vs 5). Accuracies are shown for *MNIST PCA* (purple) and pixel (light to dark green) variants. Solid lines show Q-FLAIR results, while dashed lines correspond to the traditional feature-map learning without analytic reconstruction. Here, training is terminated after 20 gates (Q-FLAIR curves are marked at 20 gates for comparison). A random parameter is drawn for each gate-feature combination. Each curve represents the mean over ten independent runs, with shaded bands indicating standard deviations. Traditional learning scales quantum evaluation overhead with feature dimension, exceeding Q-FLAIR by more than 2 orders of magnitude on *MNIST 28 × 28*.

Another difference is their scaling with training data, since evaluating the full $N \times N$ kernel matrix requires many more circuit evaluations for QSVMs throughout training. Overall, the combination of lower measurement and qubit requirements makes QNNs more suitable for near-term quantum hardware, particularly when qubits are limited.

B. Scaling with feature dimension

We investigate the effect of scaling the feature dimension d by incorporating, in addition to the PCA-reduced version ($d = 10$), raw pixel variants of MNIST (7×7 , 14×14 , and 28×28) for digits 3 versus 5. Varying the pixel resolution allows us to study scaling on nonsynthetic data across nearly 2 orders of magnitude in feature dimension (see Table I). In these experiments, only QNNs were trained with Q-FLAIR.

Figure 5 shows that Q-FLAIR maintains stable convergence and accuracy across all pixel resolutions compared to the PCA variant, with a slight drop for *MNIST 7 × 7*. We hypothesize that this arises from limited visual information in these most downsampled images rather than from the learning algorithm itself. Matching the *MNIST PCA* accuracies demonstrates that advanced preprocessing, such as PCA, is not necessary: Q-FLAIR integrates feature selection into the feature-map circuit composition and can directly process raw, high-dimensional data. In fact, accuracy initially increases faster for the pixel variants at higher resolutions (*MNIST 14 × 14* and *MNIST 28 × 28*) than for PCA-reduced data. While accuracies above 90 % are reached for the 3–5 digit classification task on the higher-resolution datasets, Q-FLAIR achieves 99 % and 97 % accuracy on the simpler 0–1 and 0–2 tasks, respectively, with a single data-dependent gate targeting a cen-

tral pixel. This underscores the effectiveness of Q-FLAIR’s inherent feature selection in high-dimensional spaces while also highlighting the relative simplicity of the 0–1 task [54], motivating our focus on the more challenging 3–5 scenario. At the same time, the result for 0–2 demonstrates that our algorithm is competitive with other established methods [71]. Besides the high accuracy of higher-resolution datasets, it is essential to note that the quantum resources overhead is unchanged for all MNIST variants. Although *MNIST PCA* is 16-dimensional, while *MNIST 28 × 28* has 784 dimensions, the number of circuit evaluations is independent of the feature dimension, and the number of qubits is unchanged.

In contrast, traditional feature-map learning, where the model is queried for probing individual features, scales unfavorably, because circuit evaluations are directly coupled to the number of features. To demonstrate this, we trained QNNs with up to 20 gates without reconstructions, repeating each run ten times with randomly drawn weight parameters. Because random parameters are drawn for each gate-feature combination, redundant features and similar gates effectively provide finer sampling of the parameter space, so that only a negligible decrease in accuracy is expected in comparison to the full optimization. This ensures that the number of evaluations is determined by the features rather than by weight optimization, resulting in a fair comparison with Q-FLAIR. As presented in Fig. 5, quantum circuit evaluation overhead grows rapidly with feature dimension, exceeding Q-FLAIR by more than 2 orders of magnitude for *MNIST 28 × 28* to construct quantum feature maps for similar accuracy and depth. In turn, the constant reconstruction overhead in Q-FLAIR is negligible relative to the high feature dimensions present.

C. Ablation study of simultaneous optimization in Q-FLAIR

We analyze the role of the three simultaneous optimizations in each Q-FLAIR iteration over gates, features, and weight parameters. Hence, we conduct ablation experiments in which each optimization is separately replaced with random choices. This design isolates the relative contribution of each optimization. We focus on QNNs trained on the *bars & stripes* dataset, which has the highest feature count, and on *MNIST PCA*, the only nonsynthetic dataset in our benchmarks. Training is terminated once circuits reach the same number of gates as in the unmodified Q-FLAIR runs. To account for stochasticity introduced by randomization, each ablation is repeated ten times, and Fig. 6 reports the resulting mean accuracies with standard deviations.

In the first ablation (Fig. 6, dotted), gate selection is no longer optimized but randomized. The best feature and parameter are then optimized as usual for this random but fixed gate. To ensure that the gate has an effect, the choice is restricted to those that modify the loss function and thus the model output. Learning behavior remains visible by improving beyond random guessing ($\sim 50\%$), but accuracy quickly saturates at about 70 % for *bars & stripes* and 80 % for *MNIST PCA*, well below the full Q-FLAIR performance.

In the second ablation (Fig. 6, dashed), the weight parameter is no longer optimized but randomly drawn (uniformly from $[-1, 1]$). The best gate and feature are then optimized as usual for this random but fixed weight. Note that this ablation

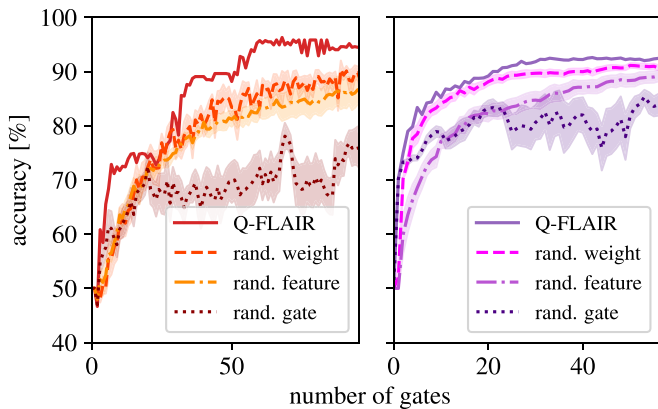


FIG. 6. Ablation study of Q-FLAIR optimizations for QNNs on (left) *bars & stripes* and (right) *MNIST PCA*. Solid curves (labeled “Q-FLAIR”) correspond to the full algorithm comparison (same as in Fig. 3). Dotted, dash-dotted, and dashed curves show the effect of replacing gate, feature, or parameter optimization, respectively, by random choices instead of including them in the simultaneous optimization. Each curve represents the mean over ten independent runs, with shaded bands indicating standard deviations.

differs from the traditional learning scheme in the previous study (Sec. IV B), where repeated resampling of weights across gate-feature combinations could implicitly optimize parameters through, e.g., redundant features. For *MNIST PCA*, this leads to a consistent but modest decrease in accuracy across all (intermediate) circuit depths, whereas for *bars & stripes* the drop is more pronounced. The latter is evident in both slower convergence (more gates for the same accuracy) and substantially lower final accuracy.

In the third ablation (Fig. 6, dash-dotted), the feature is no longer optimized but randomly selected. The best gate and weight are then determined for this random but fixed feature. The impact of random feature selection is for *bars & stripes* similar to the random-parameter ablation, which means that the final accuracy is nearly $\Delta\bar{A} = 10\%$ lower compared to full Q-FLAIR. For *MNIST PCA*, the effect is initially stronger, with poor performance for shallow circuits, but the gap narrows at larger depths, eventually approaching the random-parameter ablation.

Overall, these experiments demonstrate that all three optimizations make meaningful contributions to Q-FLAIR. Optimal gate selection has the most significant impact, but feature and parameter optimization also yield clear and non-negligible improvements. Importantly, Q-FLAIR consistently outperforms all ablation variants, establishing an effective upper bound on expected accuracies. It should also be emphasized that none of the ablation schemes could be implemented to reduce the quantum resource overhead in a substantial way, highlighting the information efficiency of Q-FLAIR through analytic reconstructions.

D. Benchmarking on IBM quantum computers

To conclude the experimental study, we transition from numerical simulations to benchmarks on real NISQ hardware, running Q-FLAIR for both QNN and QSVM. Experiments were conducted on IBM Quantum System One devices

(ibm_brussels, ibm_sherbrooke and ibm_brisbane, ibm_strasbourg), featuring the Eagle architecture with up to 127 qubits. Q-FLAIR was executed with only minor modifications from the numerically simulated setup, and all quantum calculations, including training and testing, were performed on the IBM devices, without any numerical simulations. We switch from full training sets in simulation to *minibatching* [72] on hardware to process a random subset at each iteration. Implementation details are provided in Appendix B 2. To increase practical relevance, we replace the simple synthetic *linearly separable* with the full-resolution 28×28 MNIST dataset (digits 3 versus 5).

As Fig. 7 shows, the QNN achieves test accuracies above 90 % on the IBM device across all datasets, most notably surpassing 92 % on full-resolution *MNIST* 28×28 with only eight gates. This performance is consistent with, and in the case of *MNIST* 28×28 even exceeds, the numerical simulations, where similar accuracies required deeper circuits. This seemingly counterintuitive result is likely due to stochasticity from minibatching and noise in loss function estimates, as discussed in Sec. V. Compared to prior full-resolution MNIST benchmarks on real quantum hardware, to our knowledge, this is the first demonstration of MNIST classification with feature maps learned from scratch on real hardware without heavy preprocessing. Importantly, no substantial overfitting occurs, and ideal simulations with the learned feature maps reproduce the IBM results.

Figure 8 presents the QSVM results. However, the performance on the IBM hardware is consistently lower than in numerical simulation, yet it achieves a test accuracy of at least 80 % across all tested datasets. This discrepancy can be attributed primarily to the minibatching approach, as reported in Ref. [73], which indicates that the resulting 32×32 training minikernels lead to overfitted SVM models. This phenomenon is also evident in the test-training accuracy gaps observed (Fig. 8). Larger kernels or approximations (e.g., Nyström method [73]) could mitigate the overfitting. Moreover, the target alignment loss minibatch estimation itself appears insufficient for learning effective quantum feature maps, as opposed to observations in Ref. [73]. This is evident from the poor SVM performance using the quantum feature maps learned on the IBM device, even for kernels incorporating the full training dataset evaluated in ideal numerical simulation (Fig. 8, dashed curves). Otherwise, these models should approach the high test accuracies of the simulated experiments in Sec. IV A, which is only the case for, e.g., *two-curves*.

Overall, the IBM benchmarks demonstrate Q-FLAIR’s applicability to real NISQ hardware. This success stems from two main factors: the minimal impact of hardware noise, largely mitigated by the quantum resource efficiency of Q-FLAIR in learning lightweight circuits with few gates and qubits that map naturally to the hardware’s native gate set. The other factor is the low run-time, achieved by off-loading subroutines to classical computation, a mechanism crucial for avoiding severe resource overhead with high-dimensional data. Notably, Q-FLAIR learns the QNN feature map from scratch and achieves over 90 % test accuracy entirely on a real IBM quantum computer within roughly 4 h of total quantum computation, including test evaluations at each iteration, all in a single IBM run-time session. By circumventing the high

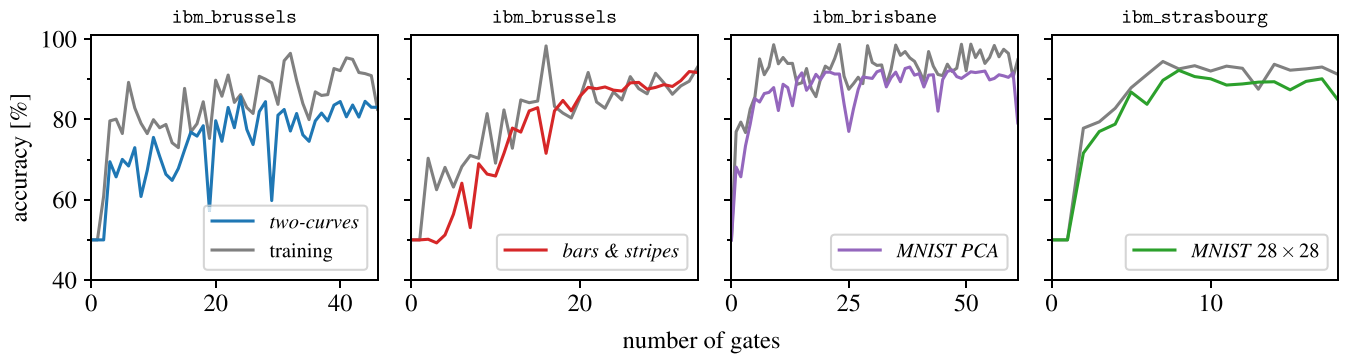


FIG. 7. NISQ benchmark of QNN performance on four datasets, including full-resolution $MNIST\ 28 \times 28$. All results stem entirely from IBM quantum hardware (device names above panels), without numerical simulation. Minibatch training is used. Curves are interpreted as in Fig. 3.

resource overhead of traditional methods, which scale directly with the data dimension d (cf. Sec. IV B), Q-FLAIR bypasses what would otherwise need an estimated four months to run.

E. Benchmarking quantum advantage via classical surrogates

To conclude the empirical analysis of Q-FLAIR, we shift our focus from demonstrating practical (hardware) scalability to establishing concrete evidence of potential quantum advantage. While rigorous proofs of quantum advantage rely on theoretical frameworks that often lack direct applicability to relevant real-world data [10,11], an established empirical benchmarking technique to bridge this gap was proposed by Schreiber *et al.* [55]. It relies on multivariate truncated Fourier series as *classical surrogate models* to capture the input-output relation of quantum circuits, such as those learned by Q-FLAIR. The data-dependent quantum gates induce a frequency spectrum that the classical surrogate uses to match the accessible frequencies of the quantum model, allowing for Fourier coefficients that match the classical surrogate with the *learned* quantum model.

While the classical surrogate with such matched Fourier coefficients could replace the learned quantum model at inference time to make predictions for new data points $\mathbf{x}$, it is most intriguing to ask whether there exists a choice of Fourier coefficients such that the classical surrogate outperforms the quantum model. Given a training dataset, the globally optimal

Fourier coefficients can be computed classically, entirely *independent* of the learned quantum model. Hence, such training of the classical surrogate models “provides a natural benchmark [...] for any claim of ‘quantum advantage’” [55] of a quantum model. We conduct classical surrogate benchmarking on Q-FLAIR QNNs. Details on the benchmark and its adaptation from Ref. [55] for the Q-FLAIR framework are provided in Appendix B 3.

Figure 9 presents the benchmark for *bars & stripes* with a classical surrogate model trained for the QNN ansatz at each Q-FLAIR iteration. The test accuracy is initially similar but separates when Q-FLAIR reaches its first performance plateau. This performance gap (about 90 % versus 75 %) marks a clear classical regime because the classical surrogate achieves better performance from the same frequencies that the Q-FLAIR model has access to, and highlights the limitation of the greediness of Q-FLAIR where choices made in previous iterations cannot be reverted. In contrast, coefficient learning in the classical surrogates corresponds to the freedom of adjusting gate types (e.g., rotation axes) and placements in the quantum circuit. The frequency spectrum, which is dictated by the selected features and corresponding weights via Q-FLAIR, is fixed for both. Overall, the benchmark implies that the shallow quantum circuits (≥ 17 gates) at the beginning of the Q-FLAIR algorithm cannot exhibit a quantum advantage and “enter a ‘classical regime’ where one could just equivalently train the classical surrogate” [55].

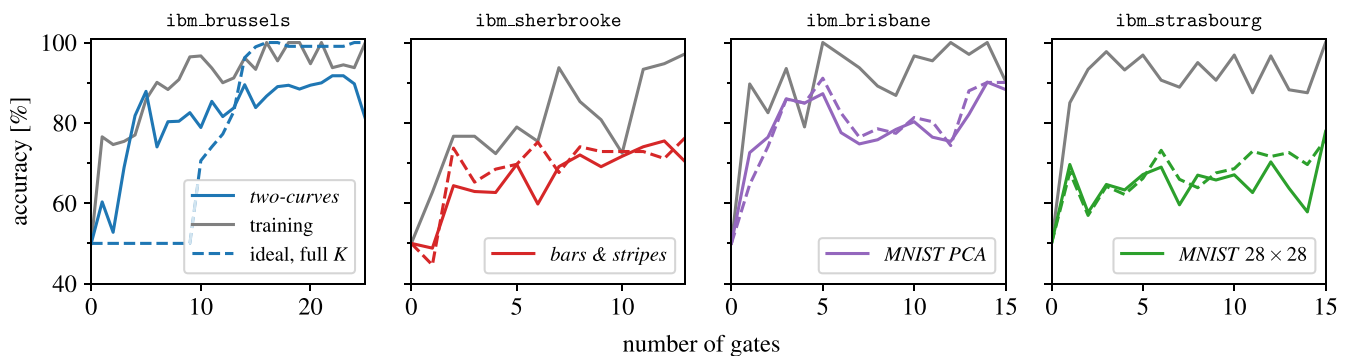


FIG. 8. NISQ benchmark of QSVM performance on four datasets, including full-resolution $MNIST\ 28 \times 28$. Solid curves show results obtained entirely on IBM quantum hardware (device names above panels), without numerical simulation, using minibatch kernels for training. Dashed curves correspond to the SVM trained on the full kernel matrix in ideal numerical simulations. Curves are interpreted as in Fig. 3.

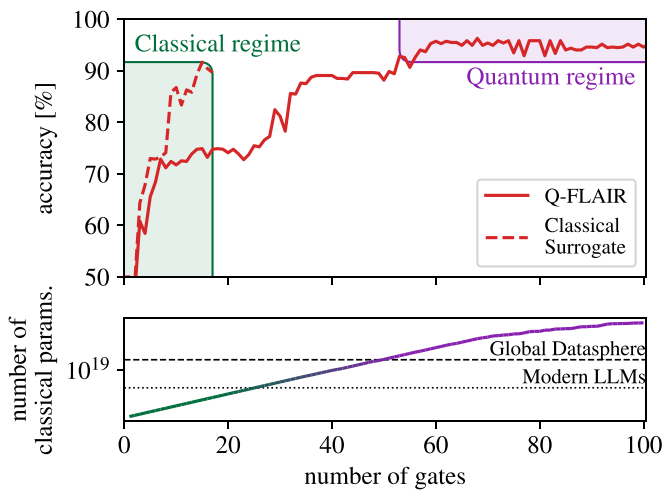


FIG. 9. Classical surrogate benchmark with *bars & stripes* dataset. Top: Comparison of the test accuracies of Q-FLAIR QNNs (solid) and the trained classical surrogate models (dashed). For each Q-FLAIR iteration (added gate), the reported surrogate model test accuracy corresponds to the highest-performing coefficients from 1000-epoch training, evaluated on the training dataset every 100 epochs. The classical regime (green) indicates where surrogates are tractable and can outperform intermediate Q-FLAIR models. The quantum regime (purple) begins at 53 gates, where Q-FLAIR surpasses the highest accuracy achieved by any classical surrogate. Bottom: Parameter count required for the classical surrogate model to match the quantum circuit at each Q-FLAIR iteration. For reference, we indicate the scale of trillion-parameter state-of-the-art large language models [74] and the projected 2025 global datasphere of 175 zettabytes [75] (i.e., a model with $\approx 10^{23}$ parameters would match the total data stored globally, assuming single-byte precision).

However, classical surrogates exhibit a scaling limitation stemming from the number of feature encodings involved. Since the frequency spectrum is obtained from Minkowski sums of the spectra per encoding gate, its size generally grows exponentially in this gate count [76]. Crucially, because Q-FLAIR learns continuous weights, it inherently induces incommensurable frequencies (i.e., frequencies with irrational ratios). Unlike standard (unweighted) gates with integer frequencies [55], these incommensurate spectra prevent the frequency collisions that would otherwise collapse the spectrum to a polynomial size in the repeated encodings per feature. Because the number of Fourier coefficients scales with the frequency spectrum, classical surrogate models become intractably large as Q-FLAIR produces deeper circuits. While Fig. 9 indicates a classical regime only up to the point where we conducted surrogate training (for models with at most 10^8 frequencies, tractable on a high-performance computing), it shows context for how this exponential trend rapidly exceeds any realistic computational requirements.

For deeper feature-map circuits later in Q-FLAIR training, Q-FLAIR passes the benchmark. In this “quantum regime,” test accuracies surpass that of classical surrogates at earlier stages, when the model was classically tractable. Due to the prohibitive parameter counts of the classical surrogate of deeper Q-FLAIR circuits, this “impracticability of the classical surrogate can indicate a possible regime of advantage” [55].

TABLE II. Comparison of classical surrogate and Q-FLAIR QNN accuracies. Maximum training and test accuracies achieved for each dataset. Bold values highlight the highest test accuracy.

Dataset	Classical surrogate		Q-FLAIR QNN	
	Training	Test	Training	Test
<i>Linearly separable</i>	90.3%	84.4%	98.1%	97.3%
<i>Bars & stripes</i>	95.0%	92.1%	94.0%	96.3%
<i>Two-curves</i>	98.3%	98.4%	87.6%	87.6%
<i>MNIST PCA</i>	94.0%	94.5%	92.5%	93.6%
<i>MNIST 28 × 28</i>	99.6%	91.6%	98.4%	92.9%

To rule out that inferior performance was merely due to incomplete features at early Q-FLAIR stages, a surrogate that uses all 16 features simultaneously (one frequency/rotation gate per feature), despite slight but overfit improvements, still falls short of deeper Q-FLAIR QNNs. Crucially, the benchmark does not overall imply that no classical model exists that could match Q-FLAIR’s performance on these datasets. Rather, it highlights intractability within the specific function class native to the quantum models, i.e., incommensurable Fourier models for the frequencies determined by Q-FLAIR. Nevertheless, from a practical learning perspective, it remains an open question to what extent this specific incommensurable complexity is an essential resource for learning, since classical approximations with commensurable frequencies may still yield sufficient learning outcomes.

For the remaining datasets, benchmarking results in Table II show that Q-FLAIR repeatedly produces QNNs that cannot be dequantized via classical surrogates. However, as evidenced by slightly higher surrogate test accuracies on some datasets, this restriction to classically tractable surrogate models does not universally imply a deficiency in learning capability. According to Ref. [55], claiming a quantum advantage over its surrogate requires the quantum model to exhibit superiority in trainability, generalization, or expressivity. The Q-FLAIR benchmarking results align with these criteria: First, Q-FLAIR exhibits a clear *trainability* advantage, as surrogate models become intractable for deeper QNNs. Second, a slight *generalization* advantage is consistently reflected in a smaller overfitting gap. Finally, Q-FLAIR provides a distinct structural advantage by building an ansatz that captures the dataset’s inductive bias, effectively structuring the inherently vast spectrum of noncanceling incommensurable frequencies. Since classical surrogates fail to replicate this mechanism by unconstrained coefficient training, they cannot access this rich function class, granting Q-FLAIR a distinct *expressivity* advantage.

The benchmarks provide a concrete demonstration of a quantum regime driven by incommensurate frequencies. To successfully exploit this spectral richness, Q-FLAIR’s adaptive learning is crucial compared to fixed quantum models. Traditionally, immense expressivity poses a severe trainability hazard [77,78] for standard fixed-ansatz feature maps that generate a generic, complex spectrum. Q-FLAIR overcomes this by dynamically tailoring the inductive bias of the ansatz to the specific dataset, selectively utilizing expressivity only where necessary. As verified in Appendix C 1, this adap-

tive approach maintains a tractable, low effective dimension [measured via the quantum Fisher information matrix (QFIM) rank] and remains trainable as the data dimension d is increased. In contrast, the fixed feature maps saturate at full QFIM rank and become untrainable at larger dimension d .

While most benchmarks in the literature report that quantum models fail to outperform their classical surrogates [55,79,80] or similar classical baselines [54], we are aware of only one prior instance where QNNs successfully surpass classical surrogates, specifically in scenarios with scarce and noisy data [56]. Because their scenario of restricted data access is strictly complementary to our demonstration of a rich, quantum-trainable but classically intractable model class, the positive benchmark outcome provides empirical evidence for an alternative route to quantum advantage.

V. DISCUSSION

We introduce an algorithm, Q-FLAIR, to learn quantum feature maps. These maps are utilized in the widely adopted QML models QNN and QSVM. The contributions and central elements of this algorithm are summarized briefly. Q-FLAIR iteratively probes different gates to grow the ansatz. It relies on partial analytic reconstructions of the model outputs for each gate. Because the loss function becomes efficiently evaluable on classical computers through these reconstructions, feature selection and weight parameter optimization do not require further access to the quantum computer. In addition to this reduced quantum resource overhead in terms of circuit executions and shot counts, Q-FLAIR learns resource-frugal quantum models by adapting to qubit topology constraints and maintaining shallow circuits due to sparse feature selection. We demonstrate strong performance in experiments across typical QML benchmarking datasets [54], including both numerical simulations and NISQ experiments, supported by further analyses for a comprehensive interpretation.

Building on these results, we emphasize the importance of simultaneously optimizing both the ansatz (gate and feature selection) and parameters at each iteration. Ablation studies show that removing either component degrades performance, highlighting their complementary roles. We further extend experiments to datasets with much higher feature dimensions (up to $d = 784$) than typically benchmarked [54] ($d \lesssim 20$), demonstrating that Q-FLAIR scales without compromising performance by decoupling quantum resource overhead from the feature dimension. This contrasts with fixed-ansatz feature maps, where one qubit per feature renders high-dimensional data virtually inaccessible [81]. Finally, the reproducibility of simulated results on NISQ hardware further supports that Q-FLAIR is both resource efficient and viable in the near term.

We further place the Q-FLAIR results in the broader context of prior full-resolution MNIST benchmarks on real quantum hardware. Earlier works typically relied on classical preprocessing: downscaling or dimensionality reduction, such as PCA or neural (autoencoder) networks [82–87], or exponentially compressed, such as amplitude-type, encodings [88–92]. Most demonstrations remain limited to simulations, and real-device studies are rare. Using downscaled 4×4

digits, Ref. [83] reached only about 80 % accuracy for the 0 versus 1 task on IBM (Eagle) hardware, while otherwise exponentially deep amplitude-encoding circuits had to be approximated to shallower depth first to load a MNIST fashion item variant on earlier IBM devices [91]. Notably, Ref. [93] reported 96 % for full-resolution 0 versus 1 classification on the more recent IBM Heron hardware, but with a fixed ansatz derived from prior simulation-based architecture search rather than feature maps learned directly on hardware—An *in situ* search would likely take months using traditional methods. In contrast, Q-FLAIR achieves over 92 % test accuracy for the more challenging 3 versus 5 task [54], fully on IBM hardware within 4 h of training. Only hybrid models that employ rich classical processing report marginally higher accuracies on this task [94–96]. Moreover, for the simpler 0 versus 1 task, Q-FLAIR achieves 99 % accuracy in simulation with only a single data-dependent rotation gate, which could likely be reproduced on hardware within minutes with minimal loss. To our knowledge, our IBM device experiments constitute the first demonstration of full-resolution MNIST classification on NISQ hardware with feature maps learned entirely from scratch.

Efficiently loading classical data into quantum states remains a central challenge in QML [3], with quantum feature maps at the core of this process. Their design is often difficult due to limited prior knowledge of effective encodings. We demonstrate the viability of Q-FLAIR not only on synthetic toy datasets but also on modest yet real-world datasets. This proof of principle is a concrete step toward overcoming the challenge of embedding classical data for QML. Q-FLAIR is particularly well suited for near-term quantum devices, as it is adaptable to specific hardware architectures, including NISQ systems and early fault-tolerant quantum computers, and it scales even to high-dimensional datasets encountered in real-world scenarios. However, learning quantum feature maps faces not only practical resource constraints but also a key theoretical challenge in QML: the *barren plateau* phenomenon [27,77]. This arises primarily in fixed ansätze, where model outputs concentrate exponentially with the number of qubits, rendering learning at scale intractable. By growing the ansatz, Q-FLAIR may delay the onset of barren plateaus and identify effective quantum feature maps before the entire gate pool suffers from concentration [97]. Indeed, incrementally learning QNN weight layers has proven effective for mitigating barren plateaus [98]. However, such methods still rely on generic, fixed-ansatz feature maps that inherently lack a dataset-specific inductive bias, requiring deep and parameter-heavy QNNs to compensate. More broadly, by leveraging partial analytic reconstructions, Q-FLAIR can also be seen as a contribution toward developing an interface between classical and quantum models, opening efficient pathways for their integration and helping to bridge the gap between traditional machine learning and quantum approaches. For instance, enabling seamless backpropagation through reconstructions in hybrid models without additional quantum overhead.

While Q-FLAIR relies on efficient local reconstruction of individual quantum gates, this does not imply efficient exact classical simulability via global reconstructions, provided a universal gate set [45]. Simultaneous global reconstruction, yielding classical surrogates of the full circuits [55], scales

exponentially with gate count, a barrier enforced by Q-FLAIR's continuous weights that induce a rich spectrum of noncanceling incommensurable frequencies. Although such immense expressivity typically renders fixed ansätze untrainable, Q-FLAIR's adaptive learning successfully avoids trainability hazards. Consequently, surrogate training becomes intractable before matching the quantum model, marking an empirical demonstration of a natural advantage over classical surrogates. Such positive outcomes are rare in the literature [56]. Although we carefully adapted the benchmark to this adaptive setting, classical strategies should be considered in the future to challenge this regime. Specifically, an independent incremental search for frequencies could decouple the surrogate from the quantum learning entirely. Furthermore, applying regularization could improve generalization, while approximations via pruning similar frequencies or utilizing alternative surrogate architectures [80,99] could delay—though are unlikely to eliminate—exponential scaling.

Despite its effectiveness on QNN and QSVM, Q-FLAIR is limited to a subset of learnable quantum feature maps, and its inherently greedy nature may restrict broader exploration. The first limitation stems from restricting gates to those whose generators satisfy $A^2 = I$. While this condition is necessary for a simple sine-curve reconstruction, it constrains the design of the gate pool, ultimately limiting the expressivity of the quantum models and potentially hindering task-specific designs. It appears that a consequence of these limitations is that Q-FLAIR learns quantum feature maps that utilize only a few qubits, in most experiments, even though it is not explicitly constrained. Whether this is “a bug or a feature” is a matter of perspective: Low-qubit scenarios limit the quantum advantage potential, as such models can often be efficiently simulated classically, yet a single qubit is theoretically sufficient for a universal quantum classifier via data reuploading [100], a capability that Q-FLAIR can exploit.

The second limitation is the greediness of Q-FLAIR: It biases the search toward exploitation over exploration. From a computational perspective, this favors efficiency at the potential cost of task performance. This greediness is largely due to the following algorithm design choices: deterministically selecting the gate, feature, and parameter that immediately minimizes the loss, only exploring a single circuit at a time, and growing the ansatz exclusively by appending gates. The latter also requires caution in the gate pool design as certain gates may not affect the model output and loss, e.g., gates commuting with the QNN observable or data-independent gates causing the QSVM gate erasure bug [64,65].

Finally, we propose potential research directions to overcome the two main limitations identified in Q-FLAIR. Extending Q-FLAIR likely improves classification accuracy, e.g., on the benchmark datasets, but at the expense of extra quantum resources. First, to increase the expressivity of quantum feature maps, the algorithm could be extended from simple sine-curve reconstructions to higher-order truncated Fourier series. While K th order reconstructions require K times more circuit evaluations per gate candidate, they allow more general gate generators with $A^2 \neq I$ [14,43,101,102] and can capture parameter repetitions, correlations, or sharing across the circuit [39]. Starting from a more expressive initial state or ansatz, e.g., applying Hadamard layers or

data-dependent qubit rotations [29], may guide the algorithm to richer solutions faster. Complementary approaches include classical nonlinear feature maps $[x \mapsto \phi(x)]$ as input, such as polynomial features for ZZ feature maps [4] in the quadratic case [103], which increase input dimension without adding quantum overhead because feature selection remains classical.

Second, to mitigate the greediness of selecting a single ansatz expansion (for the largest immediate loss reduction), each iteration of the algorithm could maintain a set of top-performing circuits, pruning the low-performing ones after a few steps. Alternatively, stochasticity could enhance the otherwise deterministic selection process by randomly sampling the next gate candidate, weighted by the determined loss improvements. So far, stochasticity has appeared solely in the NISQ experiments, originating from minibatch sampling, finite measurement shots, and hardware noise. For example, improved 28×28 MNIST performance over exact simulation provides concrete initial evidence of the benefits of stochasticity. Furthermore, the ansatz construction strategy could be switched from appending gates to prepending or extended to simultaneous bidirectional growth. In the QNN case, this corresponds to transitioning from data-dependent evolution of the initial state (Schrödinger picture) to reverse evolution of the observable (Heisenberg picture). Alternative strategies involve inserting gates at arbitrary positions in the circuit or adding new qubits through richer entanglement structures beyond linear configurations. Even though these strategies pose a risk of combinatorial blowup, they can also facilitate gate pool and model design. For instance, enabling data-independent gates in a QSVM pool without directly triggering the *gate erasure bug*, and further broadening the range of QNN observable choices, since commuting gates can have an effect at intermediate positions within the ansatz. Overall, combining analytic reconstructions with such compositional search strategies, including evolutionary or reinforcement learning approaches, could balance exploitation with broader exploration.

ACKNOWLEDGMENTS

We thank R. V. Krems for his advice and support on this project. Furthermore, we thank both him and T. Schilling for making this project possible. We would like to thank the Plateforme d'Innovation Numérique et Quantique (PINQ²), a nonprofit organization based in Québec, Canada, for the access to the IBM QS1 machines and the computation time needed for this study. We are particularly thankful to M.-È. Boulanger (PINQ²) and P. Baca (QMI, UBC) for their dedicated efforts in facilitating this partnership. The authors acknowledge support by the state of Baden-Württemberg through bwHPC and the German Research Foundation (DFG) through Grant No. “INST 39/963-1 FUGG” (bwForCluster NEMO), as well as “INST 39/1232-1 FUGG” (bwForCluster NEMO 2). This work was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada. J.J. and E.T. further acknowledge the NSERC CREATE in Quantum Computing Program, Grant No. 543245. P.E. has received support by the DFG funded Research Training Group “Dynamics of Controlled Atomic and Molecular Systems” (RTG 2717).

J.J. conceived the project and developed the algorithm. P.E. and E.T. contributed variants and adjustments to the algorithmic framework, some of which are outlined as future work. P.E. implemented the algorithm, as well as the QML models, training, and evaluation code, with J.J. and E.T. assisting the process. E.T. contributed a literature review. P.E. conducted the numerical simulation experiments, with J.J. performing the classical surrogate benchmarking. J.J. also carried out the quantum hardware experiments and implemented the interface specifics for the IBM platform. All authors analyzed the data and contributed to the interpretation of the results, with P.E. producing the plots. All authors contributed to writing the manuscript, with J.J. and P.E. leading the effort.

The views expressed are those of the authors and do not reflect the official policy or position of IBM or the IBM Quantum team.

DATA AVAILABILITY

The data that support the findings of this article are openly available [104].

APPENDIX A: ALGORITHMIC DETAILS

Algorithm 1 provides the pseudocode for the Q-FLAIR algorithm introduced in Sec. III. To clarify which operations are executed on quantum versus classical hardware, we utilize the color coding established in the schematic overview of Fig. 1. As is evident from the nested loop structure, the optimizations over features and weights rely solely on classical resources, while only the gate selection loop queries the quantum computer. Consequently, the run-time complexity with respect to the three optimization problem sizes per Q-FLAIR iteration decouples into a quantum complexity of $\mathcal{O}(M)$ and a classical complexity of $\mathcal{O}(dT_{\max})$. In contrast, traditional feature-map search methods lacking these analytical reconstructions must execute all three loops on the quantum computer, resulting in a significantly steeper quantum scaling of $\mathcal{O}(MdT_{\max})$. Note that in either approach, the quantum run-time inevitably scales with the training dataset size N , making minibatching an important strategy for practical implementation.

Reconstruction implementation

As detailed in Sec. III, the reconstruction of Eq. (14) is determined from expectation values at three different angles, e.g., via Eqs. (15)–(17). This technique applies independent of whether the gate depends on data or not, because the weight-data-dependent gate in Eq. (18) does not differ from the standard rotation gate definition of Eq. (13) when substituting the rotation angle by the parameter times the data feature $\alpha = \theta x_k$. Note that if the parameter shifts are centered at $\alpha_0 = 0$, the value $z_0 = f(\alpha_0)$ at the nonshifted position α_0 only needs to be evaluated once per data point x_i . It is independent of the probed gate candidate because the gate action becomes identity when the parameter vanishes, and it suffices to evaluate only the two shifted positions $\alpha_0 \pm \pi/2$ for each gate candidate. Furthermore, when the reconstructions are computed in an iterative manner as in Q-FLAIR, the loss/expectation value for the unshifted parameter position $\alpha_0 = 0$ is known from the previous iteration (as the minimal

Algorithm 1. Q-FLAIR: Labels highlight steps executed on quantum (purple Q) vs classical (green L) resources.

Require: Training data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, gate pool $\mathcal{V} = \{V_m\}_{m=1}^M$, loss function $L(\mathbf{y}, \hat{\mathbf{y}})$, loss convergence threshold ΔL

- 1: **C** Initialize: $U \leftarrow I, \theta \leftarrow \emptyset, L \leftarrow \infty$, init. const. model output z_0
- 2: **repeat** ▷ Incremental ansatz growth
- 3: **C** $L_{\text{prev}} \leftarrow L$
- 4: **for each** candidate gate $V_m \in \mathcal{V}$ **do**
- 5: **C** Append V_m gate to current circuit $U(x_i; \theta)$
- 6: **if** V_m is a fixed gate **then**
- 7: **Q** Evaluate loss L_m^* over all train. data points $x_i \in \mathcal{D}$
- 8: **else** ▷ $V_m(\alpha)$ is a variational gate
- 9: **for each** training data point $x_i \in \mathcal{D}$ **do**
- 10: **C** For $\alpha_0 = 0$, reuse z_0 from prev. ansatz iter.
- 11: **Q** Estimate model output $z_{\pm}$ at two shifts $\alpha_0 \pm \frac{\pi}{2}$
- 12: **C** Reconstruct $f_{m,i}(\alpha) = a \sin(\alpha - b) + c$ with
- 13: $a = \frac{1}{2} \sqrt{(2z_0 - z_+ - z_-)^2 + (z_+ - z_-)^2}$
- 14: $b = \alpha_0 - \arctan 2(2z_0 - z_+ - z_-, z_+ - z_-)$
- 15: $c = \frac{1}{2}(z_+ + z_-)$
- 16: **end for**
- 17: **if** V_m is weight- and data-dependent **then**
- 18: **for** feature index $k = 1, \dots, d$ **do**
- 19: **C** $\alpha \leftarrow \theta x_{i,k}$
- 20: **C** $f_{m,i,k}(\theta) \leftarrow f_{m,i}(\alpha)$
- 21: **end for**
- 22: **C** Candidate loss $L_m(\theta, k) \leftarrow L(\mathbf{y}, [f_{m,i,k}(\theta)]_{i=1}^N)$
- 23: $L_m^*, \theta_m^*, k_m^* \leftarrow \text{minimize}_{\theta, k} L_m(\theta, k)$ ▷ $T_{\max}$
- 24: **else if** V_m is only weight-dependent **then**
- 25: **C** $\alpha \leftarrow \theta$
- 26: $f_{m,i}(\theta) \leftarrow f_{m,i}(\alpha)$
- 27: Candidate loss $L_m(\theta) \leftarrow L(\mathbf{y}, [f_{m,i}(\theta)]_{i=1}^N)$
- 28: $L_m^*, \theta_m^* \leftarrow \text{minimize}_{\theta} L_m(\theta)$ ▷ $T_{\max}$
- 29: **end if**
- 30: **end if**
- 31: **end for**
- 32: **C** Select optimal candidate $m^* \leftarrow \arg \min_m \{L_m^*\}$
- 33: Append to ansatz $U \leftarrow V_{m^*} U$
- 34: Extend $\theta \leftarrow \theta \cup \{\theta_{m^*}^*\}$ and record $k_{m^*}^*$ (if applicable)
- 35: $L \leftarrow L_{m^*}^*$
- 36: **until** $L_{\text{prev}} - L < \Delta L$
- 37: **return** Learned quantum feature-map $U(x; \theta)$

loss when appending the last gate) and can be reused. At the initialization time of the algorithm, the loss for the unshifted parameter position, corresponding to the empty ansatz, is also usually known or efficiently computable classically. Hence, the quantum resource cost per reconstruction is effectively reduced from 3 to 2 per data point. Note that two quantum circuit evaluations exactly match the number required for the parameter-shift rule [15, 105, 106] to determine the partial derivative of the parameter in a rotation gate at a certain parameter position.

A further reduction of evaluations can be achieved if the same gates, i.e., identical generators A , appear in the gate pool $\mathcal{V}$ both as weight-dependent gates and as weight-data-dependent gates. Thus, the quantum resource demand can be reduced by reconstructing with these gates only once per data point and iteration. Similarly, fixed gates that correspond to a variational gate in the pool, such as $X = R_X(\pi)$, do not need to be evaluated on the quantum computer; instead, they can be

inferred from the respective reconstruction. The same holds for purely data-dependent gates, which are a special case of the weight-data-dependent gates in which the parameter in Eq. (18) is set to 1 [107].

APPENDIX B: EXPERIMENTAL SETUP AND IMPLEMENTATION DETAILS

1. Hyperparameter settings

For the training of the QNN, the logarithmic loss function defined in Eq. (5) is used for the empirical loss in Eq. (8). The stopping criterion is set to $\Delta L_{\log} < 10^{-3}$. In order to determine what threshold parameter b on the QNN output is required for the decision function as defined in Eq. (4), we select the best value from the receiver operating characteristic curve analysis. Thus, the threshold parameter b provides the optimal training set separation, considering correct classifications equally important for both classes, which matches the choice of the balanced accuracy metric as in Eq. (30).

The quantum feature maps for the QSVM kernel are selected according to the target alignment of Eq. (12). We use as well a stopping criterion of $\Delta L_{TA} < 10^{-3}$ between the attachment of two gates. For the QSVMs, the hyperparameter of the regularization strength C is determined through a grid search (equidistant in log space) to improve predictive performance.

In either parameter search (for b or C in QNNs and QSVMs, respectively), only minor changes were observed. Importantly, these searches do not involve any model evaluations on the quantum computer, but are conducted entirely classically.

2. Specifics for IBM benchmarks

For both models, QNN and QSVM, for each quantum circuit evaluated for a fixed parameter binding, the all-zero state probability was estimated through computational basis measurements using 1024 shots. These measurements are implemented in the IBM hardware without basis transformations. This probability directly corresponds to the all-zero projector observable $O = |0\rangle^{\otimes n} \langle 0|^{\otimes n}$ in the QNN model and, hence, a direct realization of our numerical simulations. For the QSVM, the all-zero state probability emerges from implementing the fidelity kernel $\kappa(\mathbf{x}_i, \mathbf{x}_j)$ as the transition probability when applying the quantum feature-map circuit to the initial state $|\psi_0\rangle = |0\rangle^{\otimes n}$ once for data point $\mathbf{x}_i$ and once inverted for data point $\mathbf{x}_j$. This results in the quantum circuit

$$U^\dagger(\mathbf{x}_j)U(\mathbf{x}_i) \quad (\text{B1})$$

to be executed starting from the all-zero computational basis state and collecting the (relative) frequencies of measuring all qubits in zero to estimate $\kappa(\mathbf{x}_i, \mathbf{x}_j)$ [4]. Note that other realizations of the quantum state fidelity exist [4], which are theoretically equivalent in the shot limit, but nevertheless can behave differently on real hardware or show different estimation guarantees under finite shots. For each pair of training data points $\mathbf{x}_i, \mathbf{x}_j$, the kernel value is only estimated in one order $\kappa(\mathbf{x}_i, \mathbf{x}_j)$ and used to populate the counterpart $\kappa(\mathbf{x}_j, \mathbf{x}_i)$ in the symmetric training kernel matrix. Furthermore, the diagonal of the training kernel matrix is never estimated using

the quantum computer but set to the theoretically expected value $\kappa(\mathbf{x}_i, \mathbf{x}_i) = 1$.

Due to the (additive) shot noise present in the kernel entries, the training kernel matrix is no longer guaranteed to be positive semidefinite. The underlying optimization problem when training the SVM loses its convexity/concavity property, which is usually used by the underlying optimization routine to find the global optimum. Possible corrections include shifting the kernel diagonal by the largest magnitude of the negative eigenvalues or clipping negative eigenvalues to zero in an eigendecomposition. However, these corrections did not yield any impact in terms of the model test accuracy. This finding aligns with other studies that estimate quantum fidelity kernels on real hardware [4]. In fact, we could observe increased overfitting in improved accuracies on training data only. Therefore, all reported results are without any kernel correction.

A key algorithmic difference between the experiments conducted in numerical simulation and those run on the real quantum computer is the *minibatching* technique, a standard in machine learning [72] to evaluate the (empirical) loss function on. Here, the training dataset is randomly split into smaller subsets, i.e., minibatches, which are used individually per Q-FLAIR iteration. This approach reduces computational effort when evaluating the (empirical) loss function, albeit at the cost of a less accurate (higher variance) estimate of the true loss function. For the QNN experiments, the minibatch size is 200 for *MNIST* 28×28 and 64 for all other datasets. The test dataset is not minibatched but limited to the first 200 samples. For the QSVM experiments, the minibatch size is generally 32 (minikernel of size 32×32) and 100 test data points (mini-test-kernel of size 32×100). Complementary to reducing quantum circuit executions by limiting the number of reconstructions to the minibatch size, quantum resources are further saved by limiting how gate candidates are probed on new qubits. Specifically, for single-qubit gate candidates, only a single new qubit (added at the bottom of the quantum circuit) is assessed. For two-qubit gates, only two qubits (one at the top and one at the bottom of the circuit) are considered to form linear entanglement. This selective probing strategy significantly reduces the combinatorial overhead of gate placement. Because the gate pool for the QNN in Eq. (25) only contains basic operations, it is not changed for the experiments. The gate pool for the QSVM is slightly modified to further improve the run-time on the quantum hardware. It consists of

$$\mathcal{V} = \{R_x(\theta, x_k), R_y(\theta, x_k), R_z(\theta, x_k), \\ R_{xx}(\theta, x_k), R_{yy}(\theta, x_k), R_{zz}(\theta, x_k)\}. \quad (\text{B2})$$

Recall that only a single reconstruction must be retrieved to probe the weight-data-dependent and purely weight-dependent variant of the same rotation gate.

Each model in the IBM benchmarks was trained on a single hardware instance per dataset to guarantee the same device-specific noise characteristics and allow the optimization to tailor the quantum feature maps to this noise specifically. While as many iterations of the algorithm as possible were performed within a single IBM run-time session, multiple sessions were needed for the number of iterations reported here. An IBM run-time session guarantees exclusive and

uninterrupted access to a single hardware instance to iteratively execute quantum circuits for up to 8 h. Typically, around 10–15 Q-FLAIR iterations could be executed in a single session for the QNN model, which was only about five iterations due to the more computationally demanding kernel matrix evaluations in the QSVM case. Between sessions, it is not guaranteed that model performance will be maintained, as other computations or calibrations may run on the hardware instance, altering the noise characteristics.

In order to mitigate hardware errors, matrix-free measurement mitigation (M3) is employed, a prominent error mitigation technique compatible with the computational basis measurement sampler in the IBM run-time [108]. The quasiprobability distribution as a result of the mitigation is mapped to the closest probability distribution under the $L2$ norm [108]. However, the effects of M3 were found to be negligible as they did not improve over the raw measurements to any extent that would have a noticeable impact on the model output and performance. On the other hand, the calibration (performed at the beginning of each Q-FLAIR iteration) only took a few seconds. The M3 calibration was performed using the efficient balanced method restricted to the physical qubits that were incorporated by the quantum feature-map candidate circuits over the gate pool in the current iteration.

For the hyperparameter tuning in the QSVM models (regularization strength C), ten repetitions of the fivefold cross-validation are now performed on the minibatches, to improve the stability of this tuning as the (balanced) accuracy estimates on folds of the minibatches exhibit a higher variance than on the complete training dataset.

3. Classical surrogate benchmark

The classical surrogate benchmark, conducted in Sec. IV E, was established in Ref. [55]. After a recap of the classical modeling and training procedures, we describe the key modifications necessary to accommodate the Q-FLAIR framework.

As covered and leveraged in this work, the outputs of a quantum model (as the expectation values of an observable after executing a quantum circuit) as a function of a single rotation gate angle α are simple sine curves [Eq. (14)]. Hence, such functions have a single (unit) frequency. As per the gate definition in Eq. (13), this functional form stems from the gate generators A being involutory ($A^2 = I$). A more generalized view is obtained through the eigenvalues $\sigma(A)$ of the generator A . The involution property then follows as a special case of eigenvalues being exclusively ± 1 . In this generalization, the functional form extends to trigonometric functions in α with frequencies defined by the generator eigenvalue spectra $\sigma(A)$.

When providing data $\mathbf{x}$ through quantum gates encoding individual features x_k , the quantum model can therefore be described as a multivariate trigonometric function of $\mathbf{x}$, also known as a truncated Fourier series

$$s(\mathbf{x}) = a_0 + \sum_{\omega \in \Omega_+} a_\omega \sin(\omega^\top \mathbf{x}) + b_\omega \cos(\omega^\top \mathbf{x}), \quad (\text{B3})$$

where Ω_+ contains the frequency vectors induced by the gate generators [55]. This full multivariate frequency spectrum is precisely composed from gate frequency spectra of both repeated encodings of the same feature x_k (reuploading) and

encodings of different features via Minkowski sums

$$\Omega = \sum_{k=1}^d \sum_{\ell=1}^{L_k} \{(\lambda_i - \lambda_j) \mathbf{e}_k \mid \lambda_i, \lambda_j \in \sigma(A_\ell^{(k)})\}, \quad (\text{B4})$$

where the k th feature x_k is encoded via the L_k gates with generators $A_\ell^{(k)}$ [76]. Ω_+ in Eq. (B3) constitutes a positive half-set of the full frequency spectrum Ω , such that $\Omega = \Omega_+ \cup (-\Omega_+) \cup \{0\}$ and $\Omega_+ \cap (-\Omega_+) = \emptyset$ [55]. It is important to emphasize that these classical surrogate models $s(\mathbf{x})$ are functions of all data features x_k simultaneously and thus characterize the functional form of the full quantum model, not only the local effect of a single gate parameter as exploited in Q-FLAIR. Furthermore, the coefficients $\{a_0\} \cup \{a_\omega, b_\omega \mid \omega \in \Omega_+\}$ are referred to as the Fourier coefficients and parameters of the classical surrogate model interchangeably with shorthand notation $\mathbf{a}, \mathbf{b}$. Clearly, the number of Fourier coefficients relates to the number of frequencies as $|\Omega| = 2|\Omega_+| + 1$. As discussed in the main text, the parameter count of the surrogate models scales exponentially with the number of features as well as the number of encodings, which becomes clear from the two Minkowski sums in Eq. (B4), constructing the full multivariate frequency spectrum.

Reference [55] proposes the classical surrogate benchmark to assess the quality of the learned quantum model by comparing it to the classical model with its Fourier coefficients being directly fit to the training dataset instead. Conveniently, the classical surrogate model is linear in its Fourier coefficients, such that the optimal coefficients are given by the least-squares solution, which guarantees global optimality due to convexity. Hence, the classical surrogate model can equivalently be treated as a classical neural network with a single (fully connected) layer (without activation function) on Fourier features $\{\sin(a_\omega^\top x_k), \cos(b_\omega^\top x_k) \mid \omega \in \Omega_+, k \in \{1, \dots, d\}\}$. For large datasets and numerical stability, a stochastic gradient descent approach that samples random training batches is recommended [55].

To adapt the classical surrogate benchmark proposed by Schreiber *et al.* [55] to the Q-FLAIR framework, we introduce three careful but necessary modifications. First, while the original benchmark was designed for fixed-ansatz models, Q-FLAIR constructs the ansatz adaptively. Because the frequency spectrum is intrinsic to the specific gate selection (including feature and weight choices), we extend the benchmark procedure to construct a new classical surrogate model that matches the current ansatz at each iteration of the Q-FLAIR algorithm and is fully trained on the training data. This approach not only offers granular assessment and insight into the progression of the algorithm's classicality, but it is also essential for computational tractability, as the spectral complexity of the final converged Q-FLAIR model typically exceeds the capacity of a tractable classical surrogate.

Second, Q-FLAIR utilizes gates with an additional weight parameter $\theta \in [-1, 1]$ [see Eq. (18)]. These weights directly scale the generator eigenvalues, resulting in noninteger and generally incommensurate frequencies (e.g., eigenvalues become $\pm\theta$ rather than ± 1). While the original benchmark [55] focused on integer spectra, the framework remains valid for these generalized Fourier series [76] and was already formulated as such in Eqs. (B3) and (B4). However, a critical

consequence of incommensurability is the absence of frequency cancellations. In the fixed integer frequency case [55], repeated encodings of the same feature x_k result in a basis size that scales linearly with repetitions L_k due to cancellations in the inner Minkowski sum in Eq. (B4). In contrast, the distinct, real-valued weights in Q-FLAIR generally cause the number of unique frequencies per feature x_k to grow exponentially with the number of encoding gates L_k .

Finally, we adapt the loss function for binary classification tasks where $y_i \in \{-1, +1\}$. In contrast to Ref. [55], which employs a squared loss for regression on continuous labels, we utilize the negative log-likelihood loss to train the classical surrogate for binary classification labels $y_i \in \{-1, +1\}$:

$$\text{minimize}_{a,b} \sum_{j=1}^M \ln(1 + \exp(-y_i s_{a,b}(\mathbf{x}_i))). \quad (\text{B5})$$

Beyond being the preferred loss for discrete classification tasks [109], this loss aligns with the Q-FLAIR log loss [Eq. (5)] and recasts surrogate training as a logistic regression problem. Crucially, because the classical surrogate remains a linear model in the Fourier feature space defined by the frequency spectrum, this linear logistic regression is convex with respect to the trainable Fourier coefficients [109]. Therefore, surrogate training identifies the globally optimal trigonometric function that matches the Q-FLAIR gate choice for the given iteration.

Alternative approaches for adapting the classical surrogate benchmark of Q-FLAIR are discussed in Sec. V.

APPENDIX C: ADDITIONAL EXPERIMENTAL STUDIES

1. Expressivity-trainability benchmark with fixed ansätze

The fixed-ansatz second-order Pauli-Z (or ZZ) feature map [4] is an established choice due to its difficulty to classically simulate [4]. However, as most fixed-ansatz feature maps, the ZZ feature map frequently suffers from trainability issues [77,78,110]. As the data dimensionality d increases, the expressivity of these fixed ansätze grows rapidly, leading directly to these expressivity-trainability issues. The quantum Fisher information matrix serves as a critical diagnostic tool in this context [111,112]: A saturated, full-rank QFIM indicates an overexpressive model whose outputs become indistinguishable from random noise, rendering the circuit untrainable. We empirically verify that Q-FLAIR avoids these expressivity-induced trainability issues. As an adaptive approach, Q-FLAIR expands circuit expressivity only as mandated by the dataset. This controlled growth is reflected in the moderate scaling of the QFIM rank, aligning with the lower *effective* dimension (excluding redundant and correlated features) rather than blindly scaling with the data dimension d .

The downscaled MNIST datasets, evaluated in Sec. IV B to study Q-FLAIR performance and resource dependence on the data dimension d , provide an ideal test bed to isolate this effect. Varying the image resolution scales the raw data dimension d while maintaining the underlying image semantics (and thus, the effective dimension). Here, we compare Q-FLAIR to the ZZ feature map for short, implemented via the ZZFeatureMap class provided by Qiskit [113,114]. While the accuracy of models may be obtained again from Eq. (30),

quantifying differences in the expressivity of the models is more challenging. Recently, new methods have been proposed that use measures to determine the difference between distributions to quantify the expressivity [115].

We utilize the data QFIM and take its rank as a measure of the feature-map expressivity [111,112,116]. This rank corresponds to the number of independent data-embedding directions in the quantum state, thereby revealing which input features are strictly necessary and which transport redundant or irrelevant information. For a quantum feature map preparing a pure state $|\psi(\mathbf{x})\rangle$ parametrized by a d -dimensional data point $\mathbf{x}$, the QFIM is a $d \times d$ matrix $F(\mathbf{x})$, defined as

$$F_{kl}(\mathbf{x}) = 4\Re[\langle \partial_k \psi(\mathbf{x}) | \partial_l \psi(\mathbf{x}) \rangle - \langle \partial_k \psi(\mathbf{x}) | \psi(\mathbf{x}) \rangle \langle \psi(\mathbf{x}) | \partial_l \psi(\mathbf{x}) \rangle], \quad (\text{C1})$$

with the k th partial derivative with respect to the feature component x_k denoted as $|\partial_k \psi(\mathbf{x})\rangle = \partial |\psi(\mathbf{x})\rangle / \partial x_k$ [117,118]. Rather than relying on the local geometry at a single data point, we capture the global geometry over the full data distribution by evaluating the expected data QFIM $\mathcal{F} = \mathbb{E}_{\mathbf{x}}[F(\mathbf{x})]$. In practice, this is estimated via the empirical data QFIM averaged over a finite sample set $\{\mathbf{x}_i\}_{i=1}^N$ [116], given by

$$F = \frac{1}{N} \sum_{i=1}^N F(\mathbf{x}_i) \approx \mathcal{F}. \quad (\text{C2})$$

For the sake of brevity, we refer to this empirical expected data QFIM simply as the QFIM, denoted as F , henceforth.

We have trained QSVMs using both, Q-FLAIR and ZZ feature map, on the MNIST datasets with pixel resolutions of 7×7 , 14×14 , and 28×28 . For better data compatibility, the ZZ feature map is folded onto seven qubits, thus containing some structure of the data by mapping pixels of one image row on the same qubit, since the data dimensions are multiples of 7. The number of qubits is for Q-FLAIR restricted to five qubits. QSVMs are preferred over QNNs to strictly assess the intrinsic quality of fixed nonparametric feature maps, such as the ZZ feature map. QSVMs are guaranteed to find the optimal separating hyperplane, whereas training an additional circuit or observable in QNNs risks suboptimal separation that could obscure the true effectiveness of the feature map [119].

While the number of gates in the quantum circuit of the ZZ feature map scales with the number of features, the circuits obtained by Q-FLAIR consist of 15 gates, of which 14 are data-dependent. Nevertheless, test datasets are classified by Q-FLAIR with an accuracy of close to 90 %, independent of the dimension [see Fig. 10]. The predictions with the ZZ feature map reach only an accuracy of up to 60 % first and decrease for higher dimensions. For the full 28×28 case, there is no significant learning effect left. This difference in prediction accuracy indicates that by adapting its ansatz to each specific dataset, Q-FLAIR successfully encodes more task-relevant properties than the fixed ZZ feature map.

That additionally the expressivity is larger can be seen from the rank of the QFIM: The maximum achievable rank of the QFIM is strictly bounded by the data dimension d [112]. For the MNIST datasets evaluated here, this corresponds to theoretical upper limits of $d = 49$, 196, and 784 for the 7×7 , 14×14 , and full 28×28 pixel resolutions, respectively.

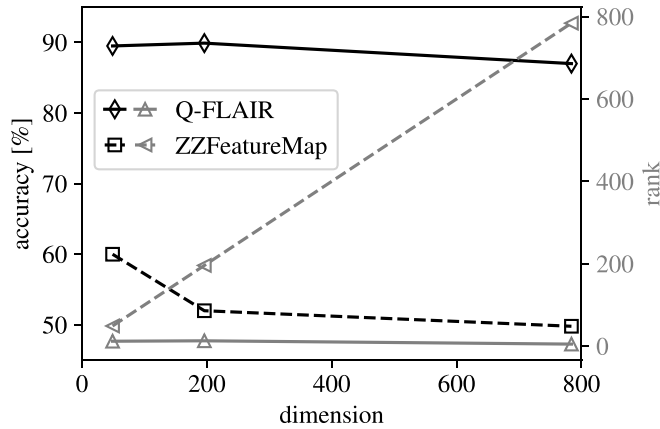


FIG. 10. Classification performance and expressivity scaling with data dimension. Comparison of QSVM accuracies (black, left axis) and data QFIM ranks (gray, right axis) using the adaptive Q-FLAIR feature map (solid lines) vs the fixed ZZ feature map (dashed lines). Models are evaluated on *MNIST* dataset variants at 7×7 , 14×14 , and 28×28 pixel resolutions, corresponding to increasing data dimension (horizontal axis). Connecting lines are added to guide the eye.

Because the fixed ZZ feature map indiscriminately embeds every input feature, its QFIM rank can theoretically reach this absolute limit. In contrast, Q-FLAIR selectively incorporates only task-relevant features; therefore, its maximum attainable rank is naturally constrained by the size of this chosen feature subset rather than the full data dimension. The 15 data-dependent gates that form the Q-FLAIR circuit for the 7×7 depend on 12 unique features. The circuit for 14×14 contains 13 distinct features, and the full-resolution *MNIST* is described by a circuit consisting of five gates, each of which depends on a unique feature. In all three cases, the number of unique features is equivalent to the QFIM rank. This counter-intuitive decrease in both the gate count and the QFIM rank at higher resolutions occurs because a sparser subset of pixels is sufficient for digit discrimination when individual pixels resolve finer structural details. In contrast, the QFIM rank of the fixed ZZ feature map increases drastically from the 14×14 to the 28×28 variant. This stark divergence explicitly validates our central claim: Simply increasing the effective dimension of the feature map does not drive the classification performance. Instead, by adaptively tailoring the ansatz to the dataset, Q-FLAIR increases its expressivity only as required, thereby promoting generalization and averting the pitfalls of overexpressiveness and the accompanying trainability issues.

Beyond the mere QFIM rank, a relaxation to large versus small over nonzero versus zero eigenvalues may be more suitable as the large eigenvalues determine how many data dimensions are *effectively* present in the feature map. Furthermore, due to the limited precision inherent in evaluating the QFIM from numerical derivatives (finite differences) averaged over the training dataset [120], we obtain, for all investigated cases, ranks that equal the upper limit imposed by the number of unique features encoded in the circuit. If we assume a (rather conservative) numerical uncertainty threshold of 1×10^{-3} on the eigenvalues, we obtain a reduction to an effective rank of 674 for the 28×28 ZZ feature map, while

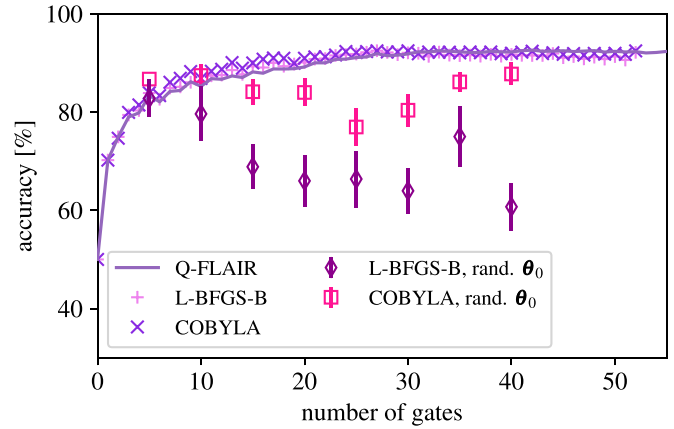


FIG. 11. Comparison between different postoptimization methods. The purple curve is the native result for the *MNIST PCA* testing dataset, directly obtained from the trained circuit. The “+” and “x” are the results if the parameters in the circuit are optimized after the training with two different optimization methods (L-BFGS-B and COBYLA). In these optimizations, the optimal parameters are taken as starting points. The brown squares are the results, again for the COBYLA optimization method, but this time with random angles θ_0 as initial configuration. The coral data points are the same for L-BFGS-B. The uncertainties are determined from three independent optimizations with up to 1000 function evaluations.

the number of nonzero eigenvalues of the two lower resolution variants does not change. For Q-FLAIR, the ranks are slightly reduced in the two higher-resolution cases: 10 for 14×14 and 3 for 28×28 . Nevertheless, the overall picture of the lower expressivity of ZZ feature map relative to Q-FLAIR is preserved. Moving beyond simple rank analysis and eigenvalue thresholding, more sophisticated metrics can be derived from QFIMs to quantify the expressivity and effective dimension of quantum feature maps more robustly [121,122].

2. Effect of parameter reoptimization

The number of evaluations in Q-FLAIR is kept low, because every parameter gets optimized only once. This leads to local optima. By reoptimizing the parameters, in theory better results should be achievable. The reoptimization of parameters is done for the QNN of the *MNIST PCA* dataset. With the limited-memory Broyden-Fletcher-Goldfarb-Shanno bounded (L-BFGS-B) algorithm [123] and Constrained Optimization by Linear Approximation (COBYLA) [124], two well-established optimization methods are used in the way in which they are implemented in SciPy [125]. These two methods were chosen, because they are able to respect finite definition ranges of the variables and can thus handle the restriction on the parameter space to $\theta \in (0, \pi)$.

In an initial attempt to increase the accuracy, the fully trained circuit is optimized with the iteratively determined parameters as starting values. The resulting mean accuracies are depicted in Fig. 11 as + and x. While initially a small enhancement of the accuracy can be observed, this vanishes for larger circuits, resulting in the same performance as the circuit with no reoptimization. The behavior does not change

between the two methods, which indicates that the local minima are rather stable.

In a second optimization scheme, the starting parameters are chosen randomly, leading to initial configurations far from a minimum. This is repeated ten times with different initial parameters and the resulting accuracies are averaged. From the distribution of the results, the standard deviations are calculated. The results of these optimizations can as well be found in Fig. 11. In shallow circuits, with only up to ten gates, it is possible to obtain slightly higher accuracies with these reoptimizations than without the additional optimization step. In none of the deeper circuits do we reach the same average accuracy through reoptimization from random starting parameters as with the iterative optimization. Throughout the optimizations, the maximally allowed number of steps toward the minimum is restricted to 1000. This constraint was set in order to limit the number of explicit evaluations of the quantum circuit. COBYLA did not succeed in fully converging for circuits with more than 15 gates, while the optimization with L-BFGS-B was reported as converged for all investigated circuit sizes. Interestingly, the optimizations with COBYLA nevertheless outperform the L-BFGS-B results in every instance. The lowest accuracy with COBYLA is obtained for circuits with 25 gates. With the increasing number of parameters, it gets more difficult to find a minimum of the loss function, which leads to similar accuracy as the iterative method. By further increasing the number of gates, the accuracy increases again and approaches our previous results. This indicates that the circuit with approximately 40 gates has enough parameters, such that the nonuniqueness of the parameter combination allows for a high enough probability to end up in a favorable minimum. For L-BFGS-B, we do not see again such a systematic increase of the accuracy. Instead, the accuracy decreases for larger gates, with the circuit consisting of 35 being an outlier due to statistic fluctuations. L-BFGS-B is thus more efficient in finding a minimum of the objective function, but, due to its gradient-based local search, struggles more than COBYLA in reaching minima with low values since it is incapable of reaching any minimum but one that is close to the initialization.

The four optimizations demonstrate overall that the iterative local optimization due to the adaptive ansatz is very efficient. It leads maybe not to the global optimum, but achieves better accuracies than optimizations from random starting parameters, which fail to reach parameter configurations that lead to similar accuracies. Also, local optimizations, starting from the parameters determined by Q-FLAIR, do not yield significantly better results. Overall, Q-FLAIR facilitates the navigation of these quantum-model-loss landscapes by optimizing parameters within an adaptive ansatz. Established optimizers, when randomly initialized in a fixed ansatz, settle in worse local optima.

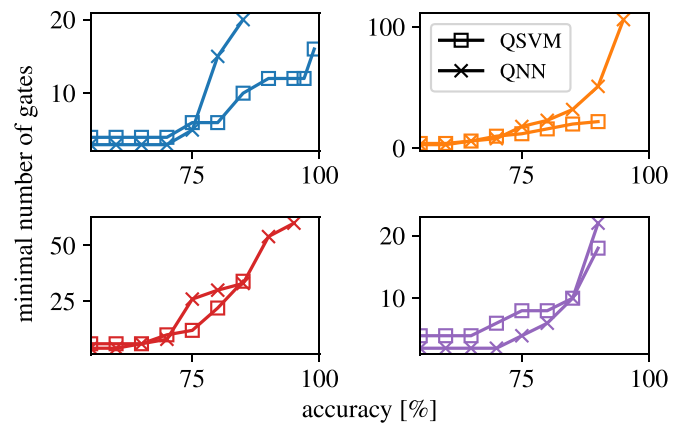


FIG. 12. Minimal number of gates needed to reach an average accuracy. The number of gates for the QSVN are, in comparison with Fig. 4, multiplied by 2. The colors indicate the datasets and correspond with those in Fig. 3.

3. Accuracy comparison of QSVN and QNN

In Sec. IV A, we concentrated mostly on the maximally achievable accuracy prior to the saturation of the loss. While this is an important quantity, it is easier to compare the performance of different models if the minimal circuit size for a given accuracy is considered. Such a comparison for the QSVNs and QNNs of *two-curves*, *linearly separable*, *bars & stripes*, and *MNIST PCA* is depicted in Fig. 12. The number of gates in the circuit for the QNNs are counted in the same way as, e.g., in Fig. 3, because this correlates directly with the number of operations that have to be calculated in Eq. (3). For the QSVNs, we double the number of gates from Fig. 4. In this way, the two states that have to be measured for the kernel entry in Eq. (10) are considered.

In the resulting data, we see that both models require in most cases a similar number of gates for the same accuracy. This is especially true for accuracies of up to 75 %. The QNNs have a slight tendency to larger numbers of gates. While this may be a disadvantage for the usage on NISQ devices, it allows in most cases for higher accuracies, because neither the loss, nor the accuracy saturate fast. It seems thus a matter of the application, as well as the available quantum computer, which of the two models is preferable: With the exception of datasets with a similar structure to *two-curves*, QNNs lead to a higher accuracy. At the same time, rather high accuracies are reached by the QSVNs with fewer gates. Overall, the two models are very similar and mediocre results can already be obtained with very few gates. The overall shallowness of the circuits is possible due to the efficiency in selecting features: For the QSVN of *bars & stripes*, this results for example in the neglect of half of the features, indicating that all the others contain redundant information.

- [1] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, *Nature (London)* **549**, 195 (2017).
- [2] V. Dunjko and H. J. Briegel, Machine learning & artificial intelligence in the quantum domain: A review

of recent progress, *Rep. Prog. Phys.* **81**, 074001 (2018).

- [3] M. Cerezo, G. Verdon, H.-Y. Huang, L. Cincio, and P. J. Coles, Challenges and opportunities in quantum machine learning, *Nat. Comput. Sci.* **2**, 567 (2022).

- [4] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, Supervised learning with quantum-enhanced feature spaces, *Nature (London)* **567**, 209 (2019).
- [5] M. Schuld and N. Killoran, Quantum machine learning in feature Hilbert spaces, *Phys. Rev. Lett.* **122**, 040504 (2019).
- [6] K. Huang, Z.-A. Wang, C. Song, K. Xu, H. Li, Z. Wang, Q. Guo, Z. Song, Z.-B. Liu, D. Zheng, *et al.*, Quantum generative adversarial networks with multiple superconducting qubits, *npj Quantum Inf.* **7**, 165 (2021).
- [7] K. Sakka, K. Mitarai, and K. Fujii, Automating quantum feature map design via large language models, [arXiv:2504.07396](https://arxiv.org/abs/2504.07396).
- [8] D. Ristè, M. P. Da Silva, C. A. Ryan, A. W. Cross, A. D. Córcoles, J. A. Smolin, J. M. Gambetta, J. M. Chow, and B. R. Johnson, Demonstration of quantum advantage in machine learning, *npj Quantum Inf.* **3**, 16 (2017).
- [9] G. Cho and D. Kim, Machine learning on quantum experimental data toward solving quantum many-body problems, *Nat. Commun.* **15**, 7552 (2024).
- [10] Y. Liu, S. Arunachalam, and K. Temme, A rigorous and robust quantum speed-up in supervised machine learning, *Nat. Phys.* **17**, 1013 (2021).
- [11] J. Jäger and R. V. Krems, Universal expressiveness of variational quantum classifiers and quantum kernels for support vector machines, *Nat. Commun.* **14**, 576 (2023).
- [12] S. Y. Chang and M. Cerezo, A primer on quantum machine learning, [arXiv:2511.15969](https://arxiv.org/abs/2511.15969).
- [13] L. Pira and P. Rebentrost, Fundamentals of quantum machine learning and robustness, in *Quantum Robustness in Artificial Intelligence Principles and Applications* (Springer, 2026), pp. 1–19.
- [14] M. Schuld, R. Sweke, and J. J. Meyer, Effect of data encoding on the expressive power of variational quantum-machine-learning models, *Phys. Rev. A* **103**, 032430 (2021).
- [15] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, Quantum circuit learning, *Phys. Rev. A* **98**, 032309 (2018).
- [16] M. Watabe, K. Shiba, M. Sogabe, K. Sakamoto, and T. Sogabe, Quantum circuit parameters learning with gradient descent using backpropagation, [arXiv:1910.14266](https://arxiv.org/abs/1910.14266).
- [17] J. Qi, C.-H. Yang, and P.-Y. Chen, QTN-VQC: An end-to-end learning framework for quantum neural networks, *Phys. Scr.* **99**, 015111 (2024).
- [18] K. Terashi, M. Kaneda, T. Kishimoto, M. Saito, R. Sawada, and J. Tanaka, Event classification with quantum machine learning in high-energy physics, *Comput. Software Big Sci.* **5**, 2 (2021).
- [19] S. Y. C. Chen, C. M. Huang, C. W. Hsing, and Y. J. Kao, Hybrid quantum-classical classifier based on tensor network and variational quantum circuit, [arXiv:2011.14651](https://arxiv.org/abs/2011.14651).
- [20] A. Blance and M. Spannowsky, Quantum machine learning for particle physics using a variational quantum classifier, *J. High Energy Phys.* **02** (2021) 212.
- [21] Y. Kwak, W. J. Yun, S. Jung, J. K. Kim, and J. Kim, Introduction to quantum reinforcement learning: Theory and PennyLane-based implementation, in *International Conference on ICT Convergence* (IEEE, Jeju Island, 2021), pp. 416–420.
- [22] D. Sierra-Sosa, J. Arcila-Moreno, B. Garcia-Zapirain, C. Castillo-Olea, and A. Elmaghraby, Dementia prediction applying variational quantum classifier, [arXiv:2007.08653](https://arxiv.org/abs/2007.08653).
- [23] S. Y. C. Chen, C. H. H. Yang, J. Qi, P. Y. Chen, X. Ma, and H. S. Goan, Variational quantum circuits for deep reinforcement learning, *IEEE Access* **8**, 141007 (2020).
- [24] S. Lloyd, M. Schuld, A. Ijaz, J. Izaac, and N. Killoran, Quantum embeddings for machine learning, [arXiv:2001.03622](https://arxiv.org/abs/2001.03622).
- [25] T. Hubregtsen, D. Wierichs, E. Gil-Fuster, P.-J. H. S. Derks, P. K. Faehrmann, and J. J. Meyer, Training quantum embedding kernels on near-term quantum computers, *Phys. Rev. A* **106**, 042431 (2022).
- [26] L. P. Henry, S. Thabet, C. Dalyac, and L. Henriët, Quantum evolution kernel: Machine learning on graphs with programmable arrays of qubits, *Phys. Rev. A* **104**, 032416 (2021).
- [27] M. Larocca, S. Thanasilp, S. Wang, K. Sharma, J. Biamonte, P. J. Coles, L. Cincio, J. R. McClean, Z. Holmes, and M. Cerezo, Barren plateaus in variational quantum computing, *Nat. Rev. Phys.* **7**, 174 (2025).
- [28] Y. Du, T. Huang, S. You, M. Hsieh, and D. Tao, Quantum circuit architecture search for variational quantum algorithms, *npj Quantum Inf.* **8**, 62 (2022).
- [29] E. Torabian and R. V. Krems, Compositional optimization of quantum circuits for quantum kernels of support vector machines, *Phys. Rev. Res.* **5**, 013211 (2023).
- [30] S. Zahid and M. A. Tahir, Unlocking quantum SVM potential: Optimal feature map generation and feature selection, *Phys. Scr.* **100**, 015120 (2025).
- [31] T. Fösel, M. Y. Niu, F. Marquardt, and L. Li, Quantum circuit optimization with deep reinforcement learning, [arXiv:2103.07585](https://arxiv.org/abs/2103.07585).
- [32] S. Chakraborty, S. H. Shaikh, A. Chakrabarti, and R. Ghosh, A hybrid quantum feature selection algorithm using a quantum inspired graph theoretic approach, *Appl. Intell.* **50**, 1775 (2020).
- [33] Z. Lu, P. X. Shen, and D. L. Deng, Markovian quantum neuroevolution for machine learning, *Phys. Rev. Appl.* **16**, 044039 (2021).
- [34] H. Wang, Y. Ding, J. Gu, Y. Lin, D. Z. Pan, F. T. Chong, and S. Han, QuantumNAS: Noise-adaptive search for robust quantum circuits, in *Proceedings of the IEEE International Symposium on High-Performance Computer Architecture* (IEEE, Seoul, 2022), pp. 692–708.
- [35] S. Altares-López, A. Ribeiro, and J. J. García-Ripoll, Automatic design of quantum feature maps, *Quantum Sci. Technol.* **6**, 045015 (2021).
- [36] R. Pellow-Jarman, A. Pillay, I. Sinayskiy, and F. Petruccione, Hybrid genetic optimization for quantum feature map design, *Quantum Mach. Intell.* **6**, 45 (2024).
- [37] Y. Liu, F. Meng, L. Wang, Y. Hu, S. Li, Z. Zhang, and X. Yu, Hardware-aware quantum kernel design based on graph neural networks, [arXiv:2506.21161](https://arxiv.org/abs/2506.21161).
- [38] M. Ostaszewski, E. Grant, and M. Benedetti, Structure optimization for parameterized quantum circuits, *Quantum* **5**, 391 (2021).
- [39] K. M. Nakanishi, K. Fujii, and S. Todo, Sequential minimal optimization for quantum-classical hybrid algorithms, *Phys. Rev. Res.* **2**, 043158 (2020).
- [40] R. M. Parrish, J. T. Iosue, A. Ozaeta, and P. L. McMahon, A Jacobi diagonalization and Anderson acceleration algorithm for variational quantum algorithm parameter optimization, [arXiv:1904.03206](https://arxiv.org/abs/1904.03206).

- [41] J. G. Vidal and D. O. Theis, Calculus on parameterized quantum circuits, [arXiv:1812.06323](#).
- [42] C. Feniou, M. Hassan, B. Claudon, A. Courtat, O. Adjoua, Y. Maday, and J.-P. Piquemal, Greedy gradient-free adaptive variational quantum algorithms on a noisy intermediate scale quantum computer, *Sci. Rep.* **15**, 18689 (2025).
- [43] J. Jäger, T. N. Kaldenbach, M. Haas, and E. Schultheis, Fast gradient-free optimization of excitations in variational quantum eigensolvers, *Commun. Phys.* **8**, 418 (2025).
- [44] H. R. Grimsley, S. E. Economou, E. Barnes, and N. J. Mayhall, An adaptive variational algorithm for exact molecular simulations on a quantum computer, *Nat. Commun.* **10**, 3007 (2019).
- [45] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, 10th ed. (Cambridge University Press, Cambridge, NY, 2010).
- [46] M. Grossi, N. Ibrahim, V. Radescu, R. Lored, K. Voigt, C. Von Altröck, and A. Rudnik, Mixed quantum—classical method for fraud detection with quantum feature selection, *IEEE Trans. Quantum Eng.* **3**, 1 (2022).
- [47] A. S. Albino, O. M. Pires, M. Q. Nooblaht, and E. G. S. Nascimento, Evolutionary quantum feature selection, [arXiv:2303.07131](#).
- [48] L. Wang, Z. Y. Chen, F. Y. Le, Z. Q. Yu, C. Xue, X. N. Zhuang, Q. Yan, Y. Yang, Y. C. Wu, and G. P. Guo, A quantum feature selection framework via ground state preparation, *Phys. Scr.* **98**, 115121 (2023).
- [49] S. Mücke, R. Heese, S. Müller, M. Wolter, and N. Piotrowski, Feature selection on quantum computers, *Quantum Mach. Intell.* **5**, 11 (2023).
- [50] R. Battiti, Using mutual information for selecting features in supervised neural net learning, *IEEE Trans. Neural Networks* **5**, 537 (1994).
- [51] M. A. F. A. Fida, T. Ahmad, and M. Ntahobari, Variance threshold as early screening to Boruta feature selection for intrusion detection system, in *Proceedings of the 13th International Conference on Information and Communication Technology and Systems* (IEEE, Surabaya, 2021), pp. 46–50.
- [52] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, Gene selection for cancer classification using support vector machines, *Mach. Learn.* **46**, 389 (2002).
- [53] R. Tibshirani, Regression shrinkage and selection via the lasso, *J. R. Stat. Soc. B* **58**, 267 (1996).
- [54] J. Bowles, S. Ahmed, and M. Schuld, Better than classical? The subtle art of benchmarking quantum machine learning models, [arXiv:2403.07059](#).
- [55] F. J. Schreiber, J. Eisert, and J. J. Meyer, Classical surrogates for quantum learning models, *Phys. Rev. Lett.* **131**, 100803 (2023).
- [56] J. Stein, M. Poppel, P. Adamczyk, R. Fabry, Z. Wu, M. Kölle, J. Nüßlein, D. Schuman, P. Altmann, T. Ehmer, V. Narasimhan, and C. Linnhoff-Popien, Benchmarking quantum surrogate models on scarce and noisy data, in *Proceedings of the 16th International Conference on Agents and Artificial Intelligence: ICAART, INSTICC* (SciTePress, Rome, Italy, 2024), Vol. 3, pp. 352–359.
- [57] E. Farhi and H. Neven, Classification with quantum neural networks on near term processors, [arXiv:1802.06002](#).
- [58] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe, Circuit-centric quantum classifiers, *Phys. Rev. A* **101**, 032308 (2020).
- [59] This loss function draws motivation from statistics and information theory, also known as log loss, logistic loss, or cross-entropy loss, and has been established as the standard choice in the classical machine learning literature—including extensions to multiclass—classification [72,109,126].
- [60] H.-Y. Huang, M. Broughton, M. Mohseni, R. Babbush, S. Boixo, H. Neven, and J. R. McClean, Power of data in quantum machine learning, *Nat. Commun.* **12**, 2631 (2021).
- [61] B. E. Boser, I. M. Guyon, and V. N. Vapnik, A training algorithm for optimal margin classifiers, in *Proceedings of the 5th Annual Workshop on Computational Learning Theory, COLT '92* (Association for Computing Machinery, New York, NY, 1992), pp. 144–152.
- [62] Note that given the evaluated quantum kernel values, commonly aggregated in the kernel matrix (also referred to as Gram matrix), the training of the QSVM does not differ from that of classical SVMs [72,109,126].
- [63] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. Kandola, On kernel-target alignment, *Adv. Neural Inf. Proc. Syst.* **14** (2001).
- [64] A. E. Paine, V. E. Elfving, and O. Kyriienko, Quantum kernel methods for solving regression problems and differential equations, *Phys. Rev. A* **107**, 032428 (2023).
- [65] I. Salmenperä, I. Kuhtarskis, A. M. van de Griend, and J. K. Nurminen, The impact of feature embedding placement in the ansatz of a quantum kernel in QSVMs, in *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE, Montreal, 2024), Vol. 1, pp. 1663–1671.
- [66] The required form to apply Q-FLAIR is still maintained when appending a fixed gate along with a (weight-) data-dependent gate because the fixed gate is absorbed by the state and observable in Eq. (28).
- [67] J. Bowles, S. Ahmed, and M. Schuld, PennyLane datasets for better than classical? The subtle art of benchmarking quantum machine learning models, PennyLane (2024), <https://pennylane.ai/datasets/linearly-separable>, <https://pennylane.ai/datasets/two-curves>, <https://pennylane.ai/datasets/bars-and-stripes>, <https://pennylane.ai/datasets/downscaled-mnist>.
- [68] Y. LeCun, The MNIST database of handwritten digits (1998), <http://yann.lecun.com/exdb/mnist/>.
- [69] S. Buchanan, D. Gilboa, and J. Wright, Deep networks and the multiple manifold problem, [arXiv:2008.11245](#).
- [70] A. C. Lorena, A. I. Maciel, P. B. de Miranda, I. G. Costa, and R. B. Prudêncio, Data complexity meta-features for regression problems, *Mach. Learn.* **107**, 209 (2018).
- [71] X. Pan, X. Cao, W. Wang, Z. Hua, W. Cai, X. Li, H. Wang, J. Hu, Y. Song, D.-L. Deng, *et al.*, Experimental quantum end-to-end learning on a superconducting processor, *npj Quantum Inf.* **9**, 18 (2023).
- [72] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*, 2nd ed. (MIT Press, Cambridge, MA, 2012).
- [73] R. Coelho, G. Kruse, and A. Rosskopf, Quantum-efficient kernel target alignment, [arXiv:2502.08225](#).
- [74] W. Fedus, B. Zoph, and N. Shazeer, Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity, *J. Mach. Learn. Res.* **23**, 1 (2022).

- [75] D. Reinsel, J. Gantz, and J. Rydning, *The Digitization of the World from Edge to Core* (International Data Corporation, Framingham, 2018), Vol. 16, p. 1.
- [76] M. C. Caro, E. Gil-Fuster, J. J. Meyer, J. Eisert, and R. Sweke, Encoding-dependent generalization bounds for parametrized quantum circuits, *Quantum* **5**, 582 (2021).
- [77] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, Barren plateaus in quantum neural network training landscapes, *Nat. Commun.* **9**, 4812 (2018).
- [78] Z. Holmes, K. Sharma, M. Cerezo, and P. J. Coles, Connecting ansatz expressibility to gradient magnitudes and barren plateaus, *PRX Quantum* **3**, 010313 (2022).
- [79] S. Mittal, M. Kumar, Y. Chand, R. Krishnakumar, and N. K. Kundu, Experimental validation of dequantization of hybrid quantum machine learning models using classical surrogates, in *2025 5th International Conference on AI-ML-Systems (AIMLSys)* (IEEE, Bangalore, 2025), pp. 412–417.
- [80] P. A. Hernicht, A. Sakhnenko, C. O’Meara, G. Cortiana, and J. M. Lorenz, Enhancing the scalability of classical surrogates for real-world quantum machine learning applications, [arXiv:2508.06131](https://arxiv.org/abs/2508.06131).
- [81] Due to qubit demands, we do not directly compare Q-FLAIR feature maps with fixed-ansatz maps matching the qubit count to the feature count, e.g., angle embedding, Z feature map, and ZZ feature map [4].
- [82] I. Kerenidis and A. Luongo, Classification of the MNIST data set with quantum slow feature analysis, *Phys. Rev. A* **101**, 062327 (2020).
- [83] M. Słysz, K. Kurowski, G. Waligóra, and J. Węglarz, Exploring the capabilities of quantum support vector machines for image classification on the MNIST benchmark, in *International Conference on Computational Science*, edited by J. Mikyka, C. de Mulatier, M. Paszynski, V. V. Krzhizhanovskaya, J. J. Dongarra, and P. M. Sloot (Springer Nature Switzerland, Cham, 2023), pp. 193–200.
- [84] A. Senokosov, A. Sedych, A. Sagingalieva, B. Kyriacou, and A. Melnikov, Quantum machine learning for image classification, *Mach. Learn.: Sci. Technol.* **5**, 015040 (2024).
- [85] R. Zhou, S. Sarkar, S. Bose, and A. Bayat, Enhanced image classification via hybridizing quantum dynamics with classical neural networks, [arXiv:2507.13587](https://arxiv.org/abs/2507.13587).
- [86] Z. Erkan, J. Rahebi, and A. Yelghi, Quantum image dataset transform (QIDT) for image processing, *Quantum Inf. Proc.* **24**, 156 (2025).
- [87] K. Chen, J. Liu, and F. Yan, Exploring quantum neural networks for binary classification on MNIST dataset: A swap test approach, *Neural Networks* **188**, 107442 (2025).
- [88] P. A. X. Tognini, L. Banchi, and G. De Palma, Solving MNIST with a globally trained mixture of quantum experts, [arXiv:2505.14789](https://arxiv.org/abs/2505.14789).
- [89] S. Manko and D. Frolovtssev, Classification and reconstruction for single-pixel imaging with classical and quantum neural networks, *Signal Image Video Process.* **19**, 277 (2025).
- [90] P. Wang, C. R. Myers, L. C. Hollenberg, and U. Parampalli, Quantum Hamiltonian embedding of images for data reuploading classifiers, *Quantum Mach. Intell.* **7**, 35 (2025).
- [91] K. Shen, B. Jobst, E. Shishenina, and F. Pollmann, Classification of the Fashion-MNIST dataset on a quantum computer, [arXiv:2403.02405](https://arxiv.org/abs/2403.02405).
- [92] F. J. Kiwit, B. Jobst, A. Luckow, F. Pollmann, and C. A. Riofrío, Typical machine learning datasets as low-depth quantum circuits, *Quantum Sci. Technol.* **10**, 045035 (2025).
- [93] P. Röseler, O. Schaudt, H. Berg, C. Bauckhage, and M. Koch, Efficient quantum convolutional neural networks for image classification: Overcoming hardware constraints, [arXiv:2505.05957](https://arxiv.org/abs/2505.05957).
- [94] Y. Zeng, H. Wang, J. He, Q. Huang, and S. Chang, A multi-classification hybrid quantum neural network using an all-qubit multi-observable measurement strategy, *Entropy* **24**, 394 (2022).
- [95] B. Dhara, M. Agrawal, and S. D. Roy, Multi-class classification using quantum transfer learning, *Quantum Inf. Proc.* **23**, 34 (2024).
- [96] F. Riaz, S. Abdulla, H. Suzuki, S. Ganguly, R. C. Deo, and S. Hopkins, Application of quantum pre-processing filter for binary image classification with small samples, *J. Data Sci. Intell. Syst.* **3**, 109 (2023).
- [97] H. R. Grimsley, G. S. Barron, E. Barnes, S. E. Economou, and N. J. Mayhall, Adaptive, problem-tailored variational quantum eigensolver mitigates rough parameter landscapes and barren plateaus, *npj Quantum Inf.* **9**, 19 (2023).
- [98] A. Skolik, J. R. McClean, M. Mohseni, P. van der Smagt, and M. Leib, Layerwise learning for quantum neural networks, *Quantum Mach. Intell.* **3**, 5 (2021).
- [99] S. R. Nair and C. Ferrie, Local surrogates for quantum machine learning, [arXiv:2506.09425](https://arxiv.org/abs/2506.09425).
- [100] A. Pérez-Salinas, A. Cervera-Lierta, E. Gil-Fuster, and J. I. Latorre, Data re-uploading for a universal quantum classifier, *Quantum* **4**, 226 (2020).
- [101] D. Wierichs, J. Izaac, C. Wang, and C. Y.-Y. Lin, General parameter-shift rules for quantum gradients, *Quantum* **6**, 677 (2022).
- [102] M. Schuld and F. Petruccione, *Machine Learning with Quantum Computers, Quantum Science and Technology* (Springer International Publishing, Cham, 2021).
- [103] The discrepancy introduced by classical nonlinear preprocessing is another reason why we refrained from directly comparing the feature maps composed by Q-FLAIR with such fixed-ansatz quantum feature maps as the ZZ feature map.
- [104] J. Jäger, P. Elsässer, and E. Torabian, Q-FLAIR: Quantum feature-map learning with reduced resource overhead, Zenodo (2026), <https://zenodo.org/records/20101655>.
- [105] J. Li, X. Yang, X. Peng, and C.-P. Sun, Hybrid quantum-classical approach to quantum optimal control, *Phys. Rev. Lett.* **118**, 150503 (2017).
- [106] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, Evaluating analytic gradients on quantum hardware, *Phys. Rev. A* **99**, 032331 (2019).
- [107] One could further consider purely weight-dependent gates as a special case of the weight-data-dependent gates if a constant feature π is included (considering a parameter range of $[-1, 1]$) in the input data points, which is a common approach for convenience in machine learning practice.
- [108] P. D. Nation, H. Kang, N. Sundaresan, and J. M. Gambetta, Scalable mitigation of measurement errors on quantum computers, *PRX Quantum* **2**, 040326 (2021).

- [109] C. M. Bishop, *Pattern Recognition and Machine Learning, Information Science and Statistics* (Springer, New York, 2006).
- [110] S. Thanasilp, S. Wang, M. Cerezo, and Z. Holmes, Exponential concentration in quantum kernel methods, *Nat. Commun.* **15**, 5200 (2024).
- [111] J. Barzen and F. Leymann, On the differential topology of expressivity of parameterized quantum circuits, *AppliedMath* **5**, 121 (2025).
- [112] J. Yao, Learning to maximize quantum neural network expressivity via effective rank, [arXiv:2506.15375](https://arxiv.org/abs/2506.15375).
- [113] A. Javadi-Abhari, M. Treinish, K. Krsulich, C. J. Wood, J. Lishman, J. Gacon, S. Martiel, P. D. Nation, L. S. Bishop, A. W. Cross, B. R. Johnson, and J. M. Gambetta, Quantum computing with Qiskit, [arXiv:2405.08810](https://arxiv.org/abs/2405.08810).
- [114] Qiskit contributors, Qiskit 0.44.0, Zenodo (2023), <https://zenodo.org/records/8190968>.
- [115] L. Monbroussou, E. Z. Mamon, J. Landman, A. B. Grilo, R. Kukla, and E. Kashefi, Trainability and expressivity of Hamming-weight preserving quantum circuits for machine learning, *Quantum* **9**, 1745 (2025).
- [116] T. Haug and M. S. Kim, Generalization of quantum machine learning models using quantum Fisher information metric, *Phys. Rev. Lett.* **133**, 050603 (2024).
- [117] J. J. Meyer, Fisher information in noisy intermediate-scale quantum applications, *Quantum* **5**, 539 (2021).
- [118] S. L. Braunstein and C. M. Caves, Statistical distance and the geometry of quantum states, *Phys. Rev. Lett.* **72**, 3439 (1994).
- [119] M. Schuld, Supervised quantum machine learning models are kernel methods, [arXiv:2101.11020](https://arxiv.org/abs/2101.11020).
- [120] A. Abbas, D. Sutter, A. Figalli, and S. Woerner, Effective dimension of machine learning models, [arXiv:2112.04807](https://arxiv.org/abs/2112.04807).
- [121] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, and S. Woerner, The power of quantum neural networks, *Nat. Comput. Sci.* **1**, 403 (2021).
- [122] O. Berezniuk, A. Figalli, R. Ghigliazza, and K. Musaelian, A scale-dependent notion of effective dimension, [arXiv:2001.10872](https://arxiv.org/abs/2001.10872).
- [123] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, A limited memory algorithm for bound constrained optimization, *SIAM J. Sci. Comput.* **16**, 1190 (1995).
- [124] M. J. Powell, Direct search algorithms for optimization calculations, *Acta Numer.* **7**, 287 (1998).
- [125] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, *et al.*, SciPy 1.0: Fundamental algorithms for scientific computing in Python, *Nat. Methods* **17**, 261 (2020).
- [126] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Vol. 2 (Springer, New York, 2009).