QRFM: Quantum Recursive Feature Machine

Hyunjin Yi[1] Phat Thanh Tran[1] Jake Scally[2] Austin Myers[3]
Ryan Carmichael[1] Xiuwen Liu[1] Weikuan Yu[1]

[1]Department of Computer Science, Florida State University
[2]Department of Physics, Florida State University
[3]Department of Mathematics, Florida State University
{hy22c,ptt17,js21h,atm22b,rcc22c,xliu,wyu3}@fsu.edu

***Abstract*—We present the Quantum Recursive Feature Machine (QRFM), a framework that combines the classical Recursive Feature Machine (RFM) algorithm with quantum kernel methods. The goal is to adaptively focus on relevant features in a hybrid quantum-classical pipeline using a metric-based update rule inspired by the Average Gradient Outer Product (AGOP). Our experiments on a down-sampled MNIST (4x4) dataset and small MNIST tasks demonstrate that QRFM can refine quantum feature maps, although classical RFM often achieves higher accuracy with substantially lower computational cost. We discuss runtime considerations, overfitting risks, and potential directions for improving quantum kernel training in resource-limited environments.**

***Index Terms*—Quantum machine learning (QML), Recursive feature machine (RFM), Quantum recursive feature machine (QRFM), Average gradient outer product (AGOP)**

Code Availability

The source code, data, and/or other artifacts have been made available at https://github.com/Kasmesq/RFMQ.

I. Introduction

Recent advances in quantum machine learning (QML) have sparked interest in using quantum kernels to tackle high-dimensional or structured data [1]. Typically, such quantum kernels rely on fixed or variationally trained quantum feature maps. However, the overhead of gradient-based quantum kernel alignment (QKA) can be significant [2], and the question of *which features* matter most often remains open.

Classical Recursive Feature Machines (RFMs) [3], [4] address feature learning via an iterative metric update: at each iteration, the model's gradient structure is used to re-weight or re-scale input dimensions, thereby significantly improving generalization and reducing dimensionality without exhaustive search. Systematic comparisons across diverse datasets, including image datasets like CIFAR-10 and ImageNet, and NLP datasets like SST-2, show that Recursive Feature Machines (RFMs) are competitive with and often outperform deep neural networks (DNNs), such as ResNet and BERT-base, achieving similar accuracy (e.g., 92% on CIFAR-10 vs. 91% for ResNet-18; 90% on SST-2 vs. 91% for BERT-base) while training up to 10x faster and using 50% less memory, making RFMs a computationally efficient alternative to deep learning models [3].

In this paper, we propose an adaptation of RFM to the quantum kernel domain, yielding a *Quantum Recursive Feature Machine* (QRFM). Our core contribution is to merge the classical AGOP-based metric update (which accumulates gradient outer products) [5] with quantum circuits that encode data as rotation gates or entangling operations, creating a new framework to develop quantum machine learning algorithms that could outperform classical algorithms by leveraging quantum computation advantages. As a first step, we empirically demonstrate the feasibility of the framework by investigating how QRFM behaves on small datasets and comparing it to classical RFM in terms of accuracy and mean-squared error (MSE). Our results on image benchmarks suggest that QRFM offers a new method to apply feature-sensitive quantum kernels to other problems such as adversarial robustness [6], tabular learning, and structured sequence data. As quantum kernel methods grow more powerful in their expressiveness, techniques such as QRFM that prioritize relevant features may prove critical for scaling toward real-world applications.

II. Related Work

A. Quantum Kernel Methods

Quantum kernel methods (QKMs) [7] represent a powerful framework for leveraging quantum computing in supervised learning. By using quantum circuits to define implicit feature embeddings, QKMs enable the computation of kernel (Gram) matrices through quantum state overlaps, potentially unlocking exponential speedups in certain tasks [1], [7]. Two primary strategies exist for constructing quantum kernels: Fidelity Quantum Kernels (FQKs) [8] and Projected Quantum Kernels (PQKs) [8].

FQKs physically prepare two quantum states, each encoding an input datapoint, and estimate their overlap using techniques such as the SWAP test or the "compute-uncompute" strategy to compute $|\langle\phi(x)|\phi(z)\rangle|^2$ [8], [9]. Qiskit provides an implementation of this via the FidelityQuantumKernel, which, by default, uses a 2-qubit ZZFeatureMap when no custom circuit is specified. This map uses parameterized $R_Z(x_j)$ rotations followed by entangling ZZ gates between each qubit pair [10]. Through these entanglements, the ZZFeatureMap captures both individual features and their pairwise correlations, making it expressive enough to model non-linear decision

boundaries. Figure 1 shows an example circuit for two features using the ZZFeatureMap [10].

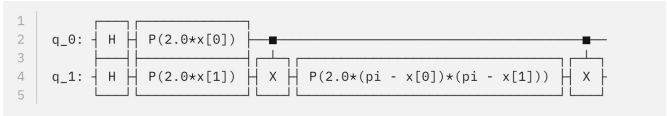

Fig. 1. Example ZZFeatureMap circuit for two qubits and one repetition. This circuit uses Hadamard gates, parameterized phase gates, and controlled entangling operations to encode classical input features into quantum states.

In contrast, PQKs simulate each quantum state vector classically (if feasible given memory limits) and compute inner products using standard linear algebra. This approach avoids quantum noise and circuit execution time, though it is limited by classical computational resources when the number of qubits exceeds 20 or so [8], [11].

QKMs have been applied in both toy and real-world datasets. Havlíček et al. showed classification on a 2D synthetic dataset using a 2-qubit quantum SVM (QSVM) [1], [7], [12], while other studies applied QKMs to reduced-dimensionality versions of MNIST—e.g., reducing the original 784 features to 4 or 8 principal components for input into 4–8 qubits [1]. The results were promising: one benchmark achieved 94% test accuracy on MNIST (digits 3 vs 5) [13] with 8 PCA features, while reducing to 4 features led to only 67% accuracy. This highlights both the potential of quantum kernels and the importance of effective dimensionality reduction.

Other benchmarks on UCI datasets [14], such as Iris and Breast Cancer, found that quantum kernel classifiers often performed similarly to classical SVMs [11]. However, these studies emphasized that performance gains from quantum kernels were inconsistent and often came at a higher computational cost, especially during kernel matrix evaluation.

B. Challenges from High-Dimensional Feature Maps

A recurring challenge in QKM is kernel concentration, where high-dimensional quantum feature maps make most input pairs nearly indistinguishable, reducing the kernel's ability to differentiate samples [8]. Additionally, the cost of computing the full $N \times N$ kernel matrix can be prohibitive. For FQKs, this requires $O(N^2)$ quantum circuit executions, each possibly needing thousands of shots to estimate overlaps accurately. As a result, QKMs are generally limited to datasets with fewer than 1,000 examples, unless approximation strategies or batching techniques are applied [8], [15].

Efforts to improve QKMs include using trainable or learned feature maps. For example, Quantum Kernel Alignment (QKA) aims to optimize feature map parameters to better align the quantum kernel with target labels [16]. Qiskit supports this via QuantumKernelTrainer, which adjusts parameters in a TrainableFidelityQuantumKernel to increase the margin in a QSVM. Though promising in theory, recent studies show that trainable quantum kernels yield only modest performance improvements in real-world tasks compared to well-chosen static feature maps [11].

To go further, some researchers propose automating quantum feature map design. Lei et al. introduced a Neural Auto-Designer that uses classical neural networks and search algorithms to learn good quantum circuit architectures based on task performance [17]. Their method includes dimensionality reduction, surrogate modeling, and architecture search, ultimately generating data-specific quantum embeddings. Similarly, evolutionary methods like genetic algorithms have been used to prune unhelpful gates or entanglement, occasionally resulting in highly efficient, almost classical circuits that perform comparably to entangled ones [18].

Finally, concepts such as quantum metric learning [19], [20] have been introduced to adapt the geometry of the quantum feature space. These methods adjust parameters of the quantum circuit to bring similar inputs closer together in the Hilbert space and push dissimilar ones apart, analogous to Automatic Relevance Determination (ARD) in classical kernels [21], [22].

C. Recursive Feature Machine (RFM)

As reviewed in Section II-A, quantum kernel methods offer a general recipe to quantum-enhance classical algorithms like SVM, ridge regression, and kernel PCA. But realizing quantum advantage in practice requires careful feature design, dimensionality reduction, and training strategies. As we explore in the next sections, combining the structure of recursive kernel updates—like those in Recursive Feature Machines (RFMs)—with quantum feature spaces could open up new frontiers for adaptive quantum learning.

To bridge this gap between fixed kernels and learned embeddings, the Recursive Feature Machine (RFM) [3], [4] was introduced for classical ML. It learns a task-adaptive kernel by updating a feature weighting matrix based on the Average Gradient Outer Product (AGOP)—effectively emphasizing directions in the input space most important for the prediction [3].

Algorithm 1 Recursive Feature Machine (RFM)

Require: Training data: (X, y), kernel K_M, number of iterations T

Ensure: Learned coefficients: α, and updated feature metric: M

Initialize $M \leftarrow I_{d\times d}$ {Identity metric}
for $t = 1$ to T **do**
  $K_{\text{train}} \leftarrow K_M(X, X)$
  $\alpha \leftarrow$ SolveKernelProblem(K_{train}, y) {e.g., regression or SVM dual}
  for each x in X **do**
    Compute $f(x) = \alpha^T K_M(X, x)$
    Compute gradient $\nabla f(x)$ w.r.t. x
  end for
  $G \leftarrow \frac{1}{|X|} \sum_{x\in X} \nabla f(x), \nabla f(x)^T$
  $M \leftarrow$ UpdateMetric(M, G)
end for
return α, M

This procedure, shown in Algorithm 1, iteratively refines the kernel by computing input-output sensitivities (gradients) and adjusting the feature metric M accordingly. The result is a kernel that becomes more aligned with the task over time.

III. Quantum Recursive Feature Machine (QRFM)

To address the gap between fixed-feature quantum kernels and data-adaptive methods, we propose the Quantum Recursive Feature Machine (QRFM)—the first framework to extend the classical Recursive Feature Machine (RFM) to the quantum setting. As noted in Section II-B and II-C, quantum kernels suffer from kernel concentration and lack effective ways to adapt feature maps to a task. Meanwhile, classical RFM has shown success in emphasizing important features through AGOP-based updates. QRFM creates a new framework by merging RFM's gradient-based feature selection with quantum kernel methods, aiming to alleviate kernel concentration and improve learning in limited-data quantum environments.

A. Overview

As Algorithm 2 illustrates, the proposed Quantum Recursive Feature Machine aims to integrate the AGOP-based recursive feature learning mechanism (from classical RFM) into a quantum kernel context. At a high level, QRFM loops through the following steps:

1) Choose an initial quantum feature map (e.g., a parameterized circuit with default settings).
2) Train a quantum kernel model (e.g., QSVM or kernel ridge regressor) using the current feature map to obtain a predictor $f(x)$.
3) Evaluate some measure of feature relevance (for instance, the gradient of $f(x)$ with respect to input features).
4) Update the feature map or an associated metric to emphasize the high-impact features.

This process repeats until convergence or for a fixed number of iterations. Ideally, the quantum feature map becomes tailored to the dataset, potentially reducing qubit requirements or boosting accuracy beyond a static embedding.

B. Defining the Feature Map and Metric

Quantum Recursive Feature Machines (QRFM) combine the key idea of a learned metric from classical RFM with a quantum kernel approach. One straightforward way is to maintain a matrix M that is applied to the classical inputs before they are encoded into the quantum state. For instance, if M is a positive-definite matrix $d \times d$ (with d being the number of original features), then the transformed feature vector is

$$x' = M^{\frac{1}{2}} x.$$

A diagonal M simply re-scales each input dimension, while off-diagonal entries correspond to mixing features before encoding — similar to a principal component (PCA) rotation.

An alternative is to let the circuit parameters themselves serve as the metric. For example, a variational feature map

$$U(\theta, x) = U_{\text{enc}}\big((x \odot \theta),\, V(\theta)\big),$$

Algorithm 2 Quantum Recursive Feature Machine (QRFM)

Require: Training data (X, y), parameterized feature map $U(\theta, x)$, number of iterations T, learning rate η
Ensure: Updated parameters θ, final decision function f_θ

Initialize feature-map parameters $\theta \leftarrow \theta^{(0)}$
for $t = 1$ to T **do**
  (1) Train QSVM:
    Compute quantum kernel:

$$K_\theta(x_i, x_j) = \left|\langle 0 \mid U(\theta, x_i)^\dagger U(\theta, x_j) \mid 0\rangle\right|^2$$

    Solve SVM dual to obtain $\alpha = \{\alpha_i\}$
    Define

$$f_\theta(x) = \sum_i \alpha_i \, K_\theta(x_i, x)$$

  (2) Compute gradients w.r.t. input features:
  for each training sample $x_i \in X$ **do**
    Compute $\nabla_x f_\theta(x_i)$ {e.g., via circuit differentiation or finite difference}
  end for
  (3) Form AGOP matrix:

$$G \leftarrow \frac{1}{|X|} \sum_{i=1}^{|X|} \big[\nabla_x f_\theta(x_i)\big] \big[\nabla_x f_\theta(x_i)\big]^T$$

  (4) Update feature-map parameters:
    for each parameter θ_j related to feature j **do**
      $\theta_j^{\text{new}} \leftarrow \text{UpdateRule}\big(\theta_j, G, \eta\big)$
    end for
    {Handle off-diagonal G_{jk} for entangling gates (if applicable)}
    for each pair (j, k) **do**
      $\phi_{jk}^{\text{new}} \leftarrow \phi_{jk} + \eta \, G_{jk}$
    end for
    $\theta \leftarrow \theta^{\text{new}}$
end for
return θ, f_θ

where $\odot$ denotes the element-wise (Hadamard) product between the input vector x and parameter vector θ, can incorporate separate weights θ_j that multiply each input x_j, or it can include entangling parameters ϕ_{jk} that couple features x_j and x_k. In this approach, the classical Average Gradient Outer Product (AGOP) update then determines which parameters to amplify or reduce—without resorting to black-box gradient descent on the circuit.

C. QRFM Algorithm Design

Algorithm 2 outlines the essential steps in QRFM. We assume an initial set of parameters θ, e.g., $\theta_j = 1$ for feature scales or $\phi_{ij} = 1$ for entangler phases.

a) Train QSVM.: Compute the quantum kernel matrix [1], [3], [12]

$$K_\theta(x_i, x_j) = \Big|\langle 0|\, U(\theta, x_i)^\dagger \, U(\theta, x_j)\, |0\rangle\Big|^2,$$

solve the SVM dual to obtain coefficients $\{\alpha_i\}$, and define the decision function

$$f_\theta(x) = \sum_i \alpha_i \, K_\theta(x_i, x).$$

b) Compute Gradients.: For each sample x_i, approximate

$$\nabla_x f_\theta(x_i)$$

with respect to the *input features*. This can be done via finite differences if the circuit is differentiable in x. In many quantum embeddings (e.g., angle encoding with $R_Z(x_j)$), known parameter-shift rules can be exploited.

c) AGOP Update.: Collect these gradients into an AGOP matrix:

$$G = \frac{1}{|X|} \sum_{i=1}^{|X|} \nabla_x f_\theta(x_i) \, \nabla_x f_\theta(x_i)^T,$$

which mimics the classical RFM update and serves as a new "importance" measure for each dimension (or pair of dimensions) [3], [4].

d) Update Parameters.: Map G onto the circuit parameters θ. For example, if θ_j corresponds to the weight on feature j, one may update it as

$$\theta_j^{\text{new}} = \theta_j \cdot \sqrt{G_{jj}},$$

or use an additive rule such as

$$\theta_j^{\text{new}} = \theta_j + \eta \cdot \sum_k G_{jk}.$$

[4] If off-diagonal G_{jk} is large, it suggests that the circuit should entangle features j and k more strongly; thus, one can update the corresponding entangler angles (where η is learning rate):

$$\phi_{jk}^{\text{new}} = \phi_{jk} + \eta \, G_{jk}.$$

[4]

e) Repeat.: Retrain the QSVM with the updated parameters θ^{new} and repeat the process until convergence or for a fixed number of iterations.

IV. Experimental Evaluation

A. *Experimental Setup*

We evaluate the Quantum Recursive Feature Machine (QRFM) against the Classical Recursive Feature Machine (RFM) on the coarse-grained MNIST dataset, downsampled to 4×4 pixels (16 features) [23], with labels from the standard MNIST dataset [24]. Experiments include both binary classification (digits 0 vs. 1) and one multi-class task (10 classes), using training set sizes of 20, 30, 50, 80, and 100–120 samples, with test sizes of 20, 30, or 50.

Classical RFM employs a Laplace kernel with AGOP-based updates implemented in PyTorch, running for 1–9 iterations. QRFM uses Qiskit's FidelityQuantumKernel [25]with a ZZFeatureMap over 4 qubits and 1000 shots, applying AGOP to update feature weights across 2–10 iterations. All quantum kernel evaluations are simulated classically.

All experiments were conducted on a dual-socket Intel Xeon Gold 6240 CPU machine with 72 threads and 187GiB of RAM. Quantum kernel evaluations were simulated using the Qiskit state vector backend in a conda environment and torch [26]. To accelerate fidelity computation, we implemented adaptive batching and parallel execution across CPU cores.

We report both test accuracy and mean squared error (MSE), using early stopping to prevent overfitting.

B. Experimental Results

Quantum Recursive Feature Machines (QRFM) represent a pioneering approach in quantum machine learning, integrating recursive feature learning with quantum computing to explore new paradigms in data processing. By leveraging quantum properties such as superposition and entanglement, QRFM has the potential to address challenges in quantum data analysis and high-dimensional feature learning, areas where classical methods may struggle. This subsection examines QRFM's behavior on the MNIST 4x4 dataset, highlighting its current capabilities and its promise for advancing quantum machine learning.

Table I presents the performance metrics of QRFM alongside Classical RFM across various configurations of the MNIST 4x4 dataset, offering insights into QRFM's behavior under current quantum hardware constraints.

QRFM achieves up to 90% accuracy in binary classification tasks (e.g., 30/20 split, 2 classes), demonstrating its potential for specific applications. However, its higher Mean Squared Error (MSE), such as 0.4895 in the same scenario, indicates that quantum kernel architectures require further refinement to enhance stability and class discriminability. These results reflect the challenges of implementing quantum-adapted models on current hardware, but they also highlight opportunities for optimizing QRFM to better leverage quantum computing's capabilities.

Figures 2, 3, and 4 provide a dynamic view of QRFM's performance trajectories. Figure 2 shows QRFM's accuracy fluctuations over training rounds, suggesting that iterative refinement could improve the stability of quantum feature learning, a key area for advancing quantum machine learning. Figure 4 illustrates QRFM's MSE trends, with higher variance underscoring the need for more robust quantum feature representations, which could unlock new applications in quantum data processing.

As shown in Table I, classical RFM in general outperforms QRFM, which is expected due to the limitation of the current quantum computers. As quantum computers improve, QRFM is expected to demonstrate advantages for applications. We are exploring ways to realize the advantages on the current quantum computers [27].

QRFM's ability to operate on quantum hardware, despite challenges like kernel concentration reducing class separability [2], marks a significant step forward.

[0]Experiments capped at 150 samples due to simulator limits.

TABLE I
PERFORMANCE METRICS OF RFM AND QRFM ON VARIATIONS OF THE MNIST 4X4 DATASET

Train/Test Size	Performance Metrics				
	Cls.	*RFM Acc.*	*QRFM Acc.*	*RFM MSE*	*QRFM MSE*
20/20	2	80.00%	75.00%	0.0371	0.4955
30/20	2	100.00%	90.00%	0.0033	0.4895
50/20	2	100.00%	85.00%	0.0031	0.4814
80/20	2	100.00%	85.00%	0.0009	0.4757
100/30	2	100.00%	87.00%	0.0001	0.4672
120/50	2	100.00%	78.00%	0.0059	0.4445
100/20	10	45.00%	45.00%	0.0629	0.0969

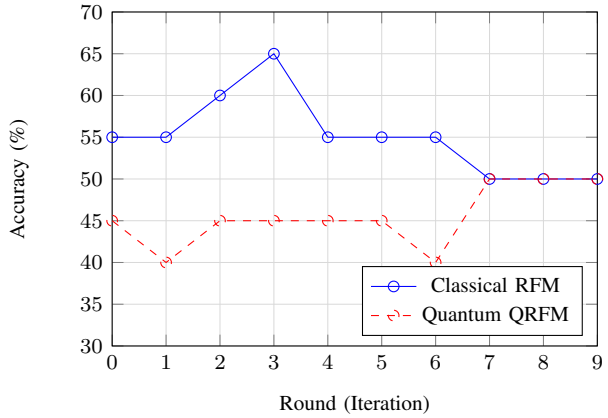

Fig. 2. Round-by-round test accuracy, highlighting QRFM's (dashed red) training dynamics and potential for iterative optimization.

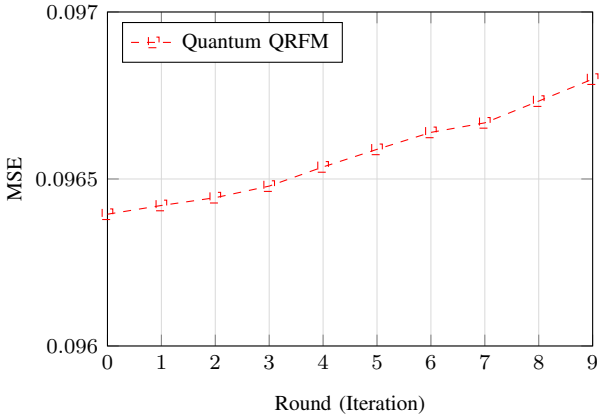

Fig. 4. Round-by-round test MSE for Quantum QRFM, illustrating variance and areas for improvement.

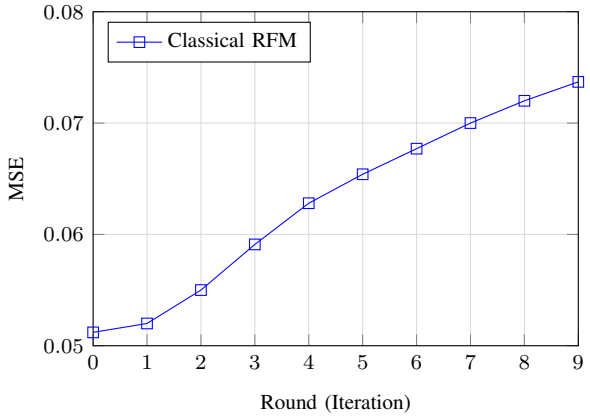

Fig. 3. Round-by-round test MSE for Classical RFM.

V. DISCUSSION

As discussed in Section II, quantum kernel methods [16] suffer from limitations such as kernel concentration and lack of task-aware feature adaptation. By leveraging feature metric updates via AGOP in Classical RFMs [3], [4], QRFM provides a solution. Here we discuss practical implementation considerations and challenges.

A. Comparison with Quantum Kernel Alignment (QKA)

Unlike Quantum Kernel Alignment (QKA) [16], which uses gradient ascent on an alignment objective with respect to circuit parameters, QRFM emphasizes **input-level** gradients. In some scenarios, this approach can capture **second-order information** more directly, similar in sprit to a Fisher Information method [28], though it may require more quantum evaluations than methods based solely on first-order parameter gradients.

B. Incorporating an Explicit Metric M

If one prefers to maintain an explicit matrix M (in classical space), the quantum circuit can remain fixed while the classical inputs are whitened by $M^{\frac{1}{2}}$. This two-step method lets classical RFM entirely determine M via AGOP; the transformed features are then fed into a static quantum kernel. Caution is needed to maintain consistency in the quantum feature space, especially if M involves complex mixing, in which case additional qubit rotations might be required.

C. Stopping Criteria and Considerations

In small data regimes, limited qubit counts, and noise in circuit evaluations may prevent QRFM from fully converging. Overfitting is possible if feature weights or entangling angles

grow excessively. As in classical RFM, one may dampen updates:

$$M_{\text{new}} = (1 - \lambda) M_{\text{old}} + \lambda G,$$

or adopt early stopping to prevent the model from over-adjusting to the training data.

D. Adaptive Batching and Parallel Execution Strategy

Integrating the fidelity-based quantum kernel into the RFM framework presented a major bottleneck due to the high computational cost of kernel evaluation. Unlike classical kernels, which rely on fast linear algebra operations and hardware acceleration, quantum kernels require circuit simulation for each data pair, making them considerably slower.

To mitigate this, we implemented a parallel batching strategy that distributes fixed-size fidelity computations across multiple CPU cores. While this reduced runtime, uneven load distribution in later stages caused idle cores and inefficiencies. To address this, we introduced a dynamic batching scheme that adaptively subdivides remaining tasks and reallocates them to available cores, maintaining a buffer of pending jobs to ensure balanced execution. This approach further improved runtime performance and resource utilization. Prior to implementing adaptive batching and parallel execution, the kernel evaluation for a matrix of size (50, 50) required approximately 23 minutes. With the proposed method, the runtime was reduced significantly to approximately 4 minutes. This computation corresponds to a single fidelity matrix. In our framework, the operation is repeated six times per iteration, resulting in a total runtime reduction of approximately two hours in this case for one iteration.

In summary, QRFM extends the principle of recursive feature adaptation to quantum settings, enabling the systematic identification of relevant quantum-encoded features and interactions while leveraging the benefits of quantum kernels. Its practical effectiveness depends on appropriate parameter selection and sufficient computational resources for iterative kernel evaluation. Future directions include optimizing quantum kernel architectures, investigating hybrid quantum-classical models, and benchmarking QRFM on quantum-native datasets, where its quantum characteristics may provide advantages over classical alternatives.

VI. Conclusion and Future Work

In this paper, we presented the Quantum Recursive Feature Machine (QRFM), a novel framework that adapts the classical Recursive Feature Machine (RFM) methodology to quantum kernel methods via an Average Gradient Outer Product (AGOP) update. Our experiments on a downsampled MNIST (4x4) dataset demonstrate that QRFM can refine quantum feature maps iteratively. However, our results also revealed several limitations. In particular, the computational overhead associated with repeated quantum kernel evaluations is significant, with simulation runtimes ranging from a few minutes to several hours per iteration. In contrast, classical RFM achieves high accuracy with much lower computational cost. Additionally, our current implementation shows kernel concentration effects that reduce class separability, and the lack of a rigorous convergence analysis raises concerns regarding the stability and robustness of the parameter update scheme.

These challenges suggest that, while QRFM offers a promising approach for adaptive feature learning in quantum-enhanced kernels, its practical impact is currently limited by both algorithmic and hardware constraints. Future studies are needed to address issues on several fronts:

- **Efficient Gradient Estimation and Improved Circuit Parameterization:** Develop more efficient methods to estimate the model's gradients with respect to input features and explore alternative circuit architectures that are less prone to kernel concentration.
- **Real Quantum Hardware Implementation:** Transition from classical simulators to experiments on actual quantum computers, which may reveal additional benefits such as intrinsic parallelism and provide more realistic insights into noise and decoherence effects.
- **Theoretical Analysis:** Investigate the convergence properties of the AGOP-based update rule, potentially deriving bounds or guarantees that would improve the theoretical foundations of QRFM.
- **Broader Dataset Evaluation:** Extend our empirical studies beyond the downsampled MNIST dataset to include more challenging and diverse datasets, thereby assessing the generalizability and scalability of the approach.
- **Hybrid Approaches:** Explore methods that combine explicit classical metric learning with quantum kernel evaluation to strike a better balance between performance and computational cost.

Overall, while QRFM marks an innovative step towards integrating adaptive feature learning with quantum kernels, addressing these limitations is essential before its full potential can be realized in practical, large-scale applications.

References

[1] 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*, vol. 567, no. 7747, pp. 209–212, 2019.

[2] M. Schuld and N. Killoran, "Machine learning with quantum kernels," *Physical Review A*, vol. 101, no. 3, p. 032308, 2020.

[3] A. Radhakrishnan, D. Beaglehole, P. Pandit, and M. Belkin, "Mechanism for feature learning in neural networks and backpropagation-free machine learning models," *Science*, vol. 383, no. 6690, pp. 1461–1467, 2024. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.adi5639

[4] A. Gupta, R. Mishra, W. Luu, and M. Bouassami, "On feature scaling of recursive feature machines," 2023. [Online]. Available: https://arxiv.org/abs/2303.15745

[5] A. Radhakrishnan, J. Yang, R. Y. Zhang, and J. Zou, "Feature engineering with outer products and gradients," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 5956–5967.

[6] S. J. Chacko, S. Biswas, C. M. Islam, F. T. Liza, and X. Liu, "Adversarial attacks on large language models using regularized relaxation," *arXiv preprint arXiv:2410.19160*, 2024. [Online]. Available: https://arxiv.org/abs/2410.19160

[7] R. Mengoni and A. D. Pierro, "Kernel methods in quantum machine learning," *Quantum Machine Intelligence*, vol. 1, pp. 65–71, 2019. [Online]. Available: https://doi.org/10.1007/s42484-019-00007-4

[8] J. Schnabel and M. Roth, "Quantum kernel methods under scrutiny: A benchmarking study," *arXiv preprint*, vol. arXiv:2409.04406, 2024, submitted on 6 Sep 2024, last revised 26 Sep 2024 (v2). [Online]. Available: https://doi.org/10.48550/arXiv.2409.04406

[9] Qiskit Machine Learning Team, "Fidelityquantumkernel class reference," https://qiskit.org/ecosystem/machine-learning/stubs/qiskit_machine_learning.kernels.FidelityQuantumKernel.html, 2024, accessed: 2025-04-14.

[10] Qiskit Community, "Zzfeaturemap documentation," https://docs.quantum.ibm.com/api/qiskit/qiskit.circuit.library.ZZFeatureMap, 2024, accessed: 2025-04-14.

[11] D. Alvarez-Estevez, "Benchmarking quantum machine learning kernel training for classification tasks," *arXiv preprint*, vol. arXiv:2408.10274, 2024, submitted on 17 Aug 2024, revised 16 Jan 2025. [Online]. Available: https://doi.org/10.48550/arXiv.2408.10274

[12] P. Rebentrost, M. Mohseni, and S. Lloyd, "Quantum support vector machine for big data classification," *Physical Review Letters*, vol. 113, no. 13, p. 130503, 2014. [Online]. Available: https://arxiv.org/abs/1307.0471

[13] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[14] M. Kelly, R. Longjohn, and K. Nottingham, "The uci machine learning repository," https://archive.ics.uci.edu, 2024, accessed: April 2025.

[15] A. Shastry, A. Jayakumar, A. Patel, and C. Bhattacharyya, "Shot-frugal and robust quantum kernel classifiers," *arXiv preprint*, vol. arXiv:2210.06971, 2022, last revised 31 Dec 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2210.06971

[16] G. Gentinetta, D. Sutter, C. Zoufal, B. Fuller, and S. Woerner, "Quantum kernel alignment with stochastic gradient descent," in *Proceedings of the IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, 2023, pp. 256–266. [Online]. Available: https://arxiv.org/abs/2304.09899

[17] C. Lei, Y. Du, P. Mi, J. Yu, and T. Liu, "Neural auto-designer for enhanced quantum kernels," *arXiv preprint*, vol. arXiv:2401.11098, 2024, submitted to ICLR 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2401.11098

[18] S. Altares-López, A. Ribeiro, and J. J. García-Ripoll, "Automatic design of quantum feature maps," *Quantum Science and Technology*, vol. 6, no. 4, p. 045015, 2021. [Online]. Available: https://doi.org/10.1088/2058-9565/ac1ab1

[19] D. T. Chang, "Variational quantum kernels with task-specific quantum metric learning," *arXiv preprint*, vol. arXiv:2211.05225, 2022, last revised 26 Nov 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2211.05225

[20] S. Lloyd, M. Schuld, A. Ijaz, J. Izaac, and N. Killoran, "Quantum embeddings for machine learning," *arXiv preprint arXiv:2001.03622*, 2020. [Online]. Available: https://arxiv.org/abs/2001.03622

[21] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2005. [Online]. Available: https://doi.org/10.7551/mitpress/3206.001.0001

[22] D. Wipf and S. Nagarajan, "A new view of automatic relevance determination," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 20. Curran Associates, Inc., 2007, pp. 1625–1632. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2007/file/6a2273a140a21bc48c56f5f7589b2f07-Paper.pdf

[23] ReneD, "Coarse-grained 4x4 pixel mnist dataset," 2020, accessed: April 14, 2025. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/79811-coarse-grained-4x4-pixel-mnist-dataset

[24] J. Redmon, "Mnist in csv," 2016, accessed: April 14, 2025. [Online]. Available: https://www.kaggle.com/datasets/oddrationale/mnist-in-csv

[25] Q. D. Team, *qiskit.quantum_info.Statevector*, IBM Quantum, 2025, accessed: 2025-04-14. [Online]. Available: https://docs.quantum.ibm.com/api/qiskit/qiskit.quantum_info.Statevector

[26] H. Wang, Y. Ding, J. Gu, Z. Li, Y. Lin, D. Z. Pan, F. T. Chong, and S. Han, "Quantumnas: Noise-adaptive search for robust quantum circuits," in *The 28th IEEE International Symposium on High-Performance Computer Architecture (HPCA-28)*, 2022.

[27] S. Jerbi, L. J. Fiderer, H. Poulsen Nautrup, J. M. Kübler, H. J. Briegel, and V. Dunjko, "Quantum machine learning beyond kernel methods," *Nature Communications*, vol. 14, no. 1, p. 517, 2023.

[28] A. Ly, M. Marsman, and E.-J. Wagenmakers, "A tutorial on fisher information," *arXiv preprint*, vol. arXiv:1705.01064, 2017. [Online]. Available: https://arxiv.org/abs/1705.01064