Quantum-Enhanced Fuzzy K-Nearest Neighbor

Srishak Dash
Department of Mathematics
South Asian University
New Delhi, India
srishakdash@students.sau.ac.in

Q. M. Danish Lohani
Department of Mathematics
South Asian University
New Delhi, India
danishlohani@cs.sau.ac.in

***Abstract*—This paper introduces a quantum-enhanced fuzzy k-nearest neighbor (QE-FKNN) algorithm that integrates quantum feature encoding with fuzzy decision-making to improve classification accuracy. Classical data are mapped into quantum states using angle embedding and entanglement, allowing complex nonlinear relationships to be represented in a high-dimensional Hilbert space. Each training sample is assigned a fuzzy membership value, and the class of a test sample is determined by a weighted combination of its k nearest neighbors. As a result, the proposed method produces smoother decision boundaries and shows greater robustness in regions where class boundaries overlap. Experimental results on ten benchmark datasets demonstrate that QE-FKNN outperforms related classical and quantum KNN-based methods in terms of classification accuracy.**

***Index Terms*—Fuzzy set, KNN, quantum circuit, quantum encoding, angle embedding, scalar quantum feature**

I. Introduction

K-nearest neighbors (KNN) [1], [2] is one of the popular and simplest supervised machine learning technique for classification and regression. It is an instance-based learning approach that does not require an explicit training phase. Instead, for a given test instance, the predicted label is determined by the labels of its k nearest neighbours in the training set. Although KNN has simple and broad applicability, this method often performs poorly when the data environment is characterized by uncertainty, including imprecise observations, noisy measurements, and incomplete samples.

Fuzzy machine learning (FML) [3] combines fuzzy techniques with machine learning to better handle uncertainty, ambiguity, and noise in real-world data by incorporating concepts such as fuzzy sets [4], fuzzy systems [5], fuzzy clustering [6], fuzzy relations [7], fuzzy measures [8], fuzzy matching [9], and fuzzy optimization [10]. A prime example is the fuzzy KNN (FKNN) algorithm, which is a generalization of the classical KNN algorithm, which was introduced in [11], that can handle uncertain data in practical applications and enhance robustness by incorporating fuzzy membership information into the classification process. However, FKNN [12] still relies on computing distances in the original feature space, which can become slow and less effective when the data is complex, noisy and large. Recently, quantum computing techniques such as angle encoding [13] offer a promising alternative that enables the nearest-neighbour search in a quantum feature space, which reduces the need for the explicit pairwise distance calculations typically required in FKNN.

Quantum machine learning (QML) [14]–[16] is the incorporation of quantum algorithms into machine learning applications. It has the potential to achieve exponential or quadratic speedups in quantum computing while also demonstrating effective handling of quantum data. In this context, quantum versions of classical machine learning algorithms have attracted significant interest, especially instance-based methods such as the KNN algorithm. By exploiting quantum parallelism and efficient distance evaluation, these approaches aim to enhance computational efficiency and scalability. An inner-product distance-based quantum KNN (QKNN) algorithm is proposed in [17] that achieved a quadratic speedup. Subsequently, alternative distances have been explored, including Euclidean [18], Hamming [19] and Mahalanobis [20] distances for the proposal of QKNN. Further, by incorporating polar distance, a QKNN algorithm is introduced, enabling the joint consideration of angular and magnitude information through an adjustable parameter [21]. To improve the practical efficiency of QKNN, a divide-and-conquer strategy is proposed in [22]. Moreover, a fidelity-based QKNN (FQKNN) algorithm is introduced in [23], which employs quantum-state fidelity as a similarity measure. Despite these advances, most existing QKNN methods rely on precise distance computations and hard neighbor assignments, which can lead to unstable classification near class boundaries.

In this work, we propose a hybrid extension of the classical FKNN, called quantum-enhanced fuzzy k-nearest neighbors, which encodes classical data into quantum states, enabling quantum circuits to map data into high-dimensional Hilbert spaces where more expressive feature representations and correlations can be captured, so that it can capture complex patterns between input data points. Unlike conventional KNN based approaches that employ hard neighbor assignments, QE-FKNN assigns fuzzy membership values to training samples. For a given test sample, the k nearest neighbors are identified from the training set using the Euclidean distance. The class membership degree of the test sample is then calculated as a weighted aggregation of the membership values of these neighbors, and the final class label is determined accordingly. This combination leads to smoother decision boundaries and improved robustness, particularly in regions where class distributions overlap, thereby reducing classification instability. The

performance of the QE-FKNN is validated by comparisons with well-known related methods using benchmark indexes.

The rest of the paper is structured as follows: Section II reviews the related work relevant to this study. The proposed QE-FKNN algorithm is formulated in Section III. Section IV provides a comparative experimental analysis conducted on datasets from the UCI and KEEL repositories. Finally, Section V concludes the paper and summarizes the key findings.

II. Preliminaries

In this section, we begin by presenting some fundamental definitions and concepts relevant to our work.

A. Fuzzy Set

Let Ω denote the universe of discourse. A fuzzy set [24] $\mathcal{F}$ defined on Ω is characterized by a membership function

$$\mu_{\mathcal{F}} : \Omega \to [0, 1],$$

where $\mu_{\mathcal{F}}(a)$ represents the degree to which an element $a \in \Omega$ belongs to the fuzzy set $\mathcal{F}$.

B. KNN

The KNN algorithm [1], [2] is a supervised learning method that classifies a query sample by assigning the most frequent class among its k nearest neighbors in the training set. Given a labelled training dataset $X'_{\text{train}} = [a_1, \ldots, a_{p_1}]$ and an unlabeled testing set $X'_{\text{test}} = [b_1, \ldots, b_{p_2}]$, where p_1 and p_2 denote the number of training and testing samples and d' is the feature dimension, KNN determines the class of each test sample based on distance comparisons.

For a test sample $b_y \in \mathbb{R}^{d'}$, $y = 1, 2, \ldots, p_2$, the Euclidean distance to each training sample $a_x \in \mathbb{R}^{d'}$, $x = 1, 2, \ldots, p_1$, is computed as

$$d(b_y, a_x) = \sqrt{\sum_{i=1}^{d'} (b_{yi} - a_{xi})^2} \tag{1}$$

The final classification is obtained by majority voting among the k-nearest neighbors, and performance is commonly evaluated using accuracy with stratified K-fold cross-validation.

C. FKNN

The fuzzy k-nearest neighbor method [12] considers a set of training samples $X'_{\text{train}} = [a_1, \ldots, a_{p_1}]$ with known class labels, and a set of testing samples $X'_{\text{test}} = [b_1, \ldots, b_{p_2}]$ with unknown labels, where p_1 and p_2 denote the numbers of training and testing samples, respectively. The method is governed by the number of nearest neighbors k and the fuzziness index λ. The objective is to assign a class label to each testing sample $b_y \in \mathbb{R}^{d'}$, where $y \in \{1, 2, \ldots, p_2\}$ and d' is the dimension.

For each training sample $a_x \in \mathbb{R}^{d'}$, where $x \in \{1, 2, \ldots, p_1\}$, let $\overline{\mu}_{l'x}$ denote the membership degree of a_x to the l'-th class. For a specified number of nearest neighbors k, the FKNN algorithm identifies the k nearest neighbors of the x-th training sample by computing its distances to all other training samples. Based on the class distribution of these neighbors, the class membership value $\overline{\mu}_{l'x}$ for each training sample is then computed as

$$\overline{\mu}_{l'x} = \begin{cases} 0.51 + 0.49 \cdot (\frac{p_x^{l'}}{k}), & \text{if } x = l', \\ 0.49 \cdot (\frac{p_x^{l'}}{k}), & \text{if } x \neq l' \end{cases} \tag{2}$$

where $p_x^{l'}, (x \in \{1, 2, \ldots, k\})$ is the number of neighbors belonging to the l'th (where, $l' \in \{1, 2, \ldots, c'\}$) class among the k-nearest neighbors and c' is the total number of classes among the k-nearest neighbors.

Subsequently, for each testing sample b_y, $y \in \{1, 2, \ldots, p_2\}$, the k-nearest neighbors are searched within the training dataset using the Euclidean distance. The class membership degree $\mu^{l'}(b_y)$ of the testing sample is then computed as a weighted average of the membership values of its neighbors, given by

$$\mu^{l'}(b_y) = \frac{\sum_{x=1}^{k} \overline{\mu}_{l'x} \left(\frac{1}{\|b_y - a_x\|^{2/(\lambda-1)}} \right)}{\sum_{x=1}^{k} \left(\frac{1}{\|b_y - a_x\|^{2/(\lambda-1)}} \right)} \tag{3}$$

After calculating all the membership values of b_y its class is assigned as the class with the highest membership value which is given by

$$C'(b_y) = \arg \max_{l'=1,\ldots,c'} \left(\mu^{l'}(b_y) \right). \tag{4}$$

where C' represents the class label of the sample.

D. Quantum Encoding

Quantum encoding [13] is the process of mapping classical data into a quantum state so that it can be processed by a quantum circuit. Some of the techniques are listed below.

- **Angle encoding:** Angle encoding is a method where classical data are encoded into quantum states by adjusting the phases of the quantum states according to the classical data values. This encoding approach is especially effective for expressing classical data with phase information.
- **Feature maps:** Feature maps are trainable quantum circuits that map classical inputs to quantum states via nonlinear transformations. These maps might capture complex interactions in classical data, potentially boosting the expressiveness of quantum circuits.

E. Quantum circuit

Quantum algorithms construct quantum circuits using a sequence of quantum gates acting on single and multiple qubits [25]. A set of quantum mechanical principles, such as superposition and entanglement, is used for execution and classical outcome extractions. Different gates fulfilled different purposes. The Pauli-X gate flips the bit and Pauli-Z gate flips the phases. On contrary, the Pauli-Y gate flip both bit and phase at a same time. To create quantum superposition Hadamard gates are widely used. While the controlled-NOT

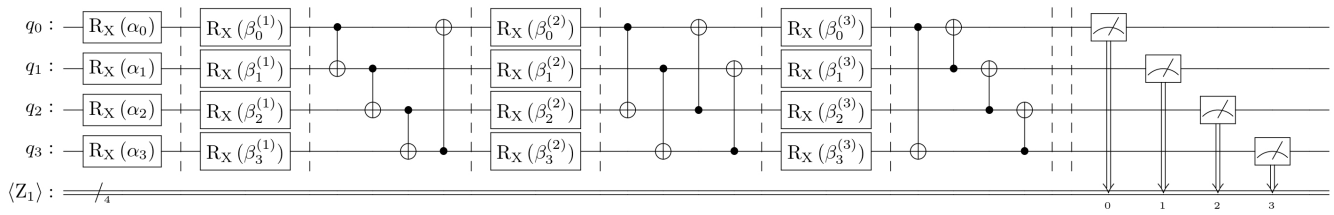

Fig. 1. Schematic representation of the 4-qubits, 3-layer parameterized quantum circuit, consisting of angle embedding, strongly entangling layers, and Pauli-Z measurement

$(CNOT)$ gates add conditional entanglement between qubits. circuits enable quantum computing beyond classical systems.

III. PROPOSED METHOD: QE-FKNN

In this section, we will formulate the proposed QE-FKNN by utilizing a quantum environment in the FKNN algorithm to enhance distance computation and feature representation. QE-FKNN exploits quantum state properties to compute distances more efficiently in a high-dimensional Hilbert space. Classical input data are first mapped into quantum states using angle embedding V_{emb}. This encoding enables a direct and continuous transformation of classical features into quantum states. First, the algorithm initializes the number of qubits and layers, along with the margin parameter ν, and defines a quantum circuit composed of parameterized gates (see Fig. 1). The dataset is then preprocessed by splitting it into training and testing sets and scaling the features using a standard scaler, followed by min–max normalization [26]. Each training sample is subsequently normalized using min-max scaling and encoded into the quantum circuit via angle embedding [13]. Specifically, the input to the angle-embedding layer is the normalized classical feature vector, where each feature value is mapped to the rotation angle of a single-qubit rotation-X (R_X) gate.

$$V_{\text{emb}}(\boldsymbol{\alpha}) = \prod_{q=0}^{m-1} R_X(\alpha_q), \quad \boldsymbol{\alpha} = [\alpha_0, \alpha_1, \ldots, \alpha_{m-1}] \tag{5}$$

where m denotes the total number of qubits and α_q is the rotation angle associated with the qth qubit. Initialize trainable circuit parameters $\boldsymbol{\beta} = \{\beta_q^{(r)}, \beta_{qs}^{(r)}\}$ randomly.

Then, to capture complex feature correlations, the encoded quantum state is further processed using strongly entangled layers V_{ent} composed of parameterized single-qubit rotation gates R_X and multi-qubit entangling operations. These layers introduce quantum entanglement, allowing the circuit to represent nonlinear relationships that are difficult to model in classical spaces [25].

$$V_{\text{ent}}(\boldsymbol{\beta}) = \prod_{r=0}^{R-1} \left(\prod_{q=0}^{m-1} R_X\left(\beta_q^{(r)}\right) \prod_{(q,s)\in\mathcal{E}} \text{CNOT}\left(\beta_{qs}^{(r)}\right) \right) \tag{6}$$

where, R specifies the number of entangling layers and $\mathcal{E}$ defines the qubit connectivity through a set of edges. The parameter $\beta_q^{(r)}$ controls the single-qubit rotation applied to qubit q in layer r, whereas $\beta_{qs}^{(r)}$ governs the controlled-NOT interaction between qubits q and s within the same layer. The circuit outputs [13] give the expectation value of the Pauli-Z operator acting on the first qubit, corresponding to the average spin projection along the z-axis, is given below:

$$\langle Z_1 \rangle = \langle \varphi(\boldsymbol{\alpha}, \boldsymbol{\beta}) \,|\, Z_1 \,|\, \varphi(\boldsymbol{\alpha}, \boldsymbol{\beta}) \rangle \tag{7}$$

where Z_1 denotes the Pauli-Z operator acting on the first qubit, and $|\varphi(\boldsymbol{\alpha}, \boldsymbol{\beta})\rangle$ represents the quantum state obtained after angle encoding and entangling transformations. The following Equation (8) shows the general expression of the parametrized quantum circuit used for the QE-FKNN to calculate the Euclidean distance:

$$\mathcal{Q}(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \left(\prod_{q=0}^{m-1} R_X(\alpha_q) \right) \left(\prod_{r=0}^{R-1} \left(\prod_{q=0}^{m-1} R_X\left(\beta_q^{(r)}\right) \prod_{(q,s)\in\mathcal{E}} \text{CNOT}\left(\beta_{qs}^{(r)}\right) \right) \right) \langle Z_1 \rangle \tag{8}$$

where $\boldsymbol{\alpha}$ denotes the set of rotation angles of a single-qubit R_X gate used for angle embedding, $\boldsymbol{\beta}$ represents the trainable circuit parameters of the strongly entangled layers. Compute scalar quantum feature $f_{\alpha_q} = \mathcal{Q}(\alpha_q, \boldsymbol{\beta}) \in \mathbb{R}$ where, $q = 0, 1, 2, \ldots, m-1$ and $\mathcal{Q}$ represents the parametrized circuit function (see Eq. 8). Then the total loss function is formulated as:

$$\mathcal{L}(\boldsymbol{\beta}) = \frac{1}{|\mathcal{C}'_{\text{intra}}|} \sum_{\{\alpha_u, \alpha_v\} \in \mathcal{C}'_{\text{intra}}} \mathcal{J}_{\text{RoBoSS}}(d) + \frac{1}{|\mathcal{C}'_{\text{inter}}|} \sum_{\{\alpha_u, \alpha_v\} \in \mathcal{C}'_{\text{inter}}} \mathcal{J}_{\text{RoBoSS}}(\nu - d) \tag{9}$$

where $u \neq v$ and $u, v \in \{0, 1, \ldots, m-1\}$, $\mathcal{C}'_{\text{intra}}$ and $\mathcal{C}'_{\text{inter}}$ denote the sets of same-class and different-class sample pairs, respectively. d denotes the Euclidean distance between the scalar quantum features f_{α_u} and f_{α_v}, while $\mathcal{J}_{\text{RoBoSS}}(d)$ represents the RoBoSS loss function [27], and ν is the margin parameter. The quantum circuit parameters are optimized using the Adam optimizer [28] in conjunction with Eq. 9. Finally, both the training and test samples are encoded using the optimized quantum circuit, i.e., the circuit parameterized by the learned values of $\boldsymbol{\beta}$.

For each encoded training sample, fuzzy membership values are computed using its k nearest neighbors in the encoded

feature space, as defined in Eq. 2. For each encoded test sample, the k-nearest neighbours are identified from the encoded training set using Eq. 1. The corresponding fuzzy membership degrees are then evaluated using Eq. 3, and the final class label is assigned based on the maximum membership value as specified in Eq. 4. The overall workflow of the proposed QE-FKNN approach is summarized in Algorithm 1.

Algorithm 1 QE-FKNN

Input: Training features U'_{train}, training labels X'_{train}, test features U'_{test}, number of nearest neighbors k
Output: Predicted class labels for test samples X'_{test}
Initialize the number of qubits m, the number of layers R, margin parameter ν, trainable circuit parameters $\boldsymbol{\beta}$=$\{\beta_q^{(r)},\beta_{qs}^{(r)}\}$ randomly and define a quantum circuit with parametrized gates (see Eq. 8).
Load dataset and encode categorical features
Split dataset into training and testing sets
Apply standard scaling followed by min–max normalization [26]
for each training epoch **do**
    Encode training samples using angle embedding $V_{\text{emb}}(\boldsymbol{\alpha})$ (see Eq. 5)
    Apply strongly entangled quantum layers for every layer using trainable circuit parameters $\boldsymbol{\beta}$=$\{\beta_q^{(r)},\beta_{qs}^{(r)}\}$ (see Eq. 6)
    Measure Pauli-Z expectation value to obtain scalar quantum features (see Eq. 7)
    Update quantum circuit parameters using the Adam optimizer [28] with total loss function $\mathcal{L}(\boldsymbol{\beta})$ (see Eq. 9)
end for
Encode all training and testing samples using the optimized quantum circuit
for each training sample **do**
    Compute fuzzy membership values (see Eq. 2) based on its k nearest neighbors in the encoded feature space
end for
for each test sample **do**
    Identify k nearest neighbors from the encoded training samples using Eq. 1.
    Compute fuzzy membership degrees using Eq. 3
    Assign the class label corresponding to the maximum membership degree (see Eq. 4)
end for

IV. EXPERIMENTAL ANALYSIS

A. Experimental Setup

The performance of QE-FKNN is compared against KNN [1], FKNN [11] and QKNN [13] with 10 UCI and KEEL benchmark datasets. All experiments were implemented in Python 3.13.3 using the Pennylane 0.43.1. The quantum circuits were simulated on a classical backend. Training and evaluation of QE-FKNN and implementation of all baseline models are conducted on a system equipped with a Mac M4 Silicon chip with 16 GB RAM. In each QE-FKNN experiment, the quantum circuit has six strongly entangled layers. The bounding parameter of the RoBoSS loss function was set to unity. The shape parameter was varied within the range -1.9 to -1.5 with an increment of 0.1. The parameters for Adam optimizer are set as in [28]. The parameters for the baseline papers are set according to their original papers. For uniform and better performance we apply five fold cross validation with a 75:25 training and testing sample split. We analyze the mean accuracy, mean precision and mean F1-score along with the mean standard deviation.

B. Data Characteristics and Evaluation Criteria

The performance of the proposed model and the comparative models is evaluated on 10 UCI and KEEL datasets listed in Table I using standard evaluation metrics, namely accuracy, precision, and F1 score, as commonly defined in the literature [13], [29]. Classification accuracy is reported as a percentage (%).

TABLE I

UCI and KEEL Data			
Dataset	Number of Classes	Number of elements	Number of features
Appendicitis	2	106	7
Cryotherapy	2	90	6
Haberman	2	306	3
Hayes–roth	3	160	4
Hepatitis	2	80	19
SIRTUIN6	2	100	6
Tae	3	151	5
Vertebral column	2	310	6
Wine	3	178	13
Zoo	7	101	16

C. Results and Discussion

The experimental results comparing KNN, QKNN, FKNN, and QE-FKNN for k= 3, 5, 7 and 9 are summarized in Table II. Overall, the experimental results demonstrate that the proposed QE-FKNN model shows a consistent performance in terms of classification accuracy over considered benchmark datasets, particularly for problems where complex feature relationships benefit from quantum-enhanced representations (see Appendicitis, Cryotherapy, Hayes–Roth, Hepatitis, Tae, and Wine datasets). However, the results also show that the performance of QE-FKNN depends on the choice of neighbor size. For datasets such as Vertebral Column and Haberman, QE-FKNN performs well only for specific values of k, whereas classical or quantum baseline methods achieve higher accuracy for other neighbor sizes. In addition, the lower performance on the Zoo dataset suggests that the proposed approach may not be universally effective, especially for datasets with distinct structural or categorical characteristics.

Moreover, the proposed QE-FKNN algorithm achieves a higher mean accuracy than the other methods across all considered datasets for k= 3, 5, 7, and 9. As shown in Fig. 2, the mean accuracy across different values of k exhibits less fluctuation compared to the other methods, indicating that

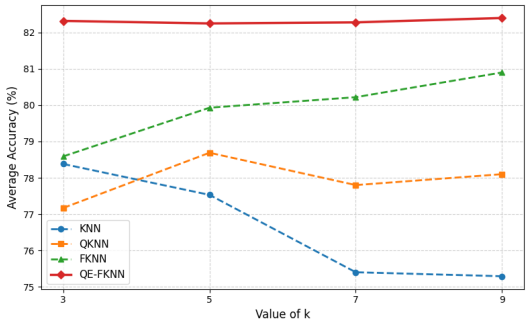

Fig. 2. Average classification accuracy comparison of KNN, QKNN, FKNN and our QE-FKNN across 10 benchmark datasets, evaluated for varying values of k (i.e k= 3, 5, 7, 9)

the proposed approach is more stable than the comparative models.

V. Conclusion

This paper proposed a quantum-enhanced fuzzy k-nearest neighbor algorithm that combines quantum feature encoding with fuzzy decision-making to improve classification performance. By mapping classical data into high-dimensional quantum states and using fuzzy neighbor memberships, the method produces smoother decision boundaries and greater robustness in overlapping class regions. Experimental results on benchmark datasets show that QE-FKNN achieves higher and more stable accuracy than classical and quantum KNN-based methods across multiple values of k.

In future work, we aim to enhance the QE-FKNN framework by incorporating dimensionality reduction in the quantum feature space to improve scalability for high-dimensional data, as the current version requires extensive computational resources.

References

[1] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer, "Knn model-based approach in classification," in *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*. Springer, 2003, pp. 986–996.

[2] S. Zhang, "Nearest neighbor selection for iteratively knn imputation," *Journal of Systems and Software*, vol. 85, no. 11, pp. 2541–2552, 2012.

[3] J. Lu, G. Ma, and G. Zhang, "Fuzzy machine learning: A comprehensive framework and systematic review," *IEEE Transactions on Fuzzy Systems*, vol. 32, no. 7, pp. 3861–3878, 2024.

[4] H.-C. Wu, "Probability density functions of fuzzy random variables," *Fuzzy sets and systems*, vol. 105, no. 1, pp. 139–158, 1999.

[5] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE transactions on systems, man, and cybernetics*, no. 1, pp. 116–132, 1985.

[6] L. Zhu, F.-L. Chung, and S. Wang, "Generalized fuzzy c-means clustering algorithm with improved fuzzy partitions," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 3, pp. 578–591, 2009.

[7] E. Czogala, J. Drewniak, and W. Pedrycz, "Fuzzy relation equations on a finite set," *Fuzzy Sets and systems*, vol. 7, no. 1, pp. 89–101, 1982.

[8] M. Sugeno, "Fuzzy measure and fuzzy integral," *Transactions of the Society of Instrument and Control Engineers*, vol. 8, no. 2, pp. 218–226, 1972.

TABLE II
Classification Accuracy (%), Precision and F1-score Comparison of KNN, QKNN, FKNN and QE-FKNN for Different Values of k Across Ten Datasets

Dataset			KNN				QKNN				FKNN				QE-FKNN			
			$k=3$	$k=5$	$k=7$	$k=9$	$k=3$	$k=5$	$k=7$	$k=9$	$k=3$	$k=5$	$k=7$	$k=9$	$k=3$	$k=5$	$k=7$	$k=9$
Appendicitis	ACC		85.84±5.84	88.66±4.31	87.71±5.49	87.71±5.49	88.66±5.47	89.57±6.26	89.57±6.26	88.66±5.47	84.85±5.34	86.75±7.12	87.71±5.49	86.75±6.28	**89.61** ±3.91	**89.61** ±3.91	**89.61** ±3.91	**90.52**±6.70
	Precision		0.870±0.053	0.892±0.041	0.879±0.056	0.879±0.056	0.802±0.121	0.836±0.136	0.897±0.081	0.846±0.126	0.856±0.048	0.864±0.081	0.872±0.067	0.858±0.079	0.897±0.078	0.897±0.078	0.897±0.078	0.871±0.134
	F1		0.852±0.057	0.878±0.051	0.867±0.060	0.867±0.060	0.775±0.120	0.788±0.119	0.808±0.097	0.798±0.116	0.851±0.051	0.863±0.075	0.870±0.063	0.857±0.074	0.808±0.089	0.808±0.089	0.808±0.089	0.832±0.134
Cryotherapy	ACC		83.45±7.45	77.78±6.80	77.78±6.80	83.33±4.75	74.44±6.33	80.00±6.33	81.11±7.45	80.00±8.43	87.78±6.09	86.67±3.04	86.67±4.97	88.89±3.93	**91.30**±3.70	**91.30**±3.70	**91.30**±3.70	**91.30**±3.70
	Precision		0.877±0.070	0.897±0.067	0.897±0.064	0.895±0.043	0.744±0.066	0.804±0.071	0.819±0.074	0.803±0.090	0.893±0.035	0.880±0.026	0.885±0.020	0.902±0.028	0.907±0.072	0.907±0.072	0.907±0.072	0.907±0.072
	F1		0.865±0.075	0.887±0.070	0.887±0.069	0.875±0.048	0.743±0.064	0.798±0.063	0.805±0.081	0.798±0.086	0.876±0.063	0.865±0.030	0.865±0.051	0.888±0.039	0.897±0.071	0.897±0.071	0.897±0.071	0.897±0.071
Haberman	ACC		67.30±7.13	69.93±5.91	68.29±5.35	71.88±6.13	67.95±7.03	**70.58**±6.17	**70.56**±8.52	**72.53**±5.92	66.00±5.47	66.98±5.22	66.65±5.38	67.96±4.83	**68.94**±5.16	69.92±4.22	69.59±3.90	69.92±3.89
	Precision		0.654±0.063	0.675±0.062	0.642±0.053	0.690±0.082	0.567±0.102	0.599±0.103	0.610±0.157	0.632±0.159	0.655±0.017	0.657±0.031	0.649±0.041	0.664±0.029	0.585±0.057	0.588±0.044	0.581±0.034	0.587±0.034
	F1		0.656±0.063	0.676±0.060	0.650±0.046	0.685±0.055	0.543±0.074	0.555±0.064	0.548±0.092	0.542±0.078	0.650±0.030	0.655±0.032	0.649±0.037	0.662±0.028	0.571±0.055	0.565±0.044	0.558±0.035	0.564±0.035
Hayes–Roth	ACC		65.62±8.56	56.88±5.13	43.75±5.41	39.38±2.80	69.38±7.46	72.50±4.64	69.38±4.94	66.88±6.48	65.62±10.60	68.12±7.78	68.12±6.77	68.12±8.39	**71.88**±8.08	**73.12**±6.08	**73.75**±6.76	**73.75**±6.76
	Precision		0.700±0.104	0.608±0.056	0.481±0.046	0.454±0.019	0.775±0.142	0.773±0.123	0.743±0.097	0.724±0.105	0.691±0.111	0.727±0.081	0.744±0.056	0.738±0.066	0.783±0.089	0.790±0.078	0.793±0.078	0.793±0.078
	F1		0.651±0.086	0.568±0.044	0.436±0.064	0.392±0.012	0.688±0.121	0.718±0.126	0.693±0.107	0.669±0.104	0.648±0.105	0.671±0.078	0.668±0.072	0.671±0.086	0.710±0.078	0.725±0.055	0.734±0.055	0.734±0.055
Hepatitis	ACC		88.75±5.23	86.25±5.23	85.00±9.48	86.25±6.85	87.50±7.65	88.75±6.85	88.75±6.85	88.75±6.85	80.00±5.23	82.50±5.23	85.00±7.13	87.50±7.65	**91.25**±5.59	**91.25**±5.59	**91.25**±5.59	**91.25**±5.59
	Precision		0.891±0.066	0.882±0.040	0.861±0.088	0.890±0.038	0.841±0.162	0.858±0.137	0.865±0.128	0.865±0.128	0.832±0.051	0.845±0.039	0.869±0.063	0.881±0.076	0.915±0.034	0.915±0.034	0.915±0.034	0.915±0.034
	F1		0.878±0.059	0.854±0.042	0.844±0.090	0.851±0.051	0.760±0.138	0.774±0.127	0.790±0.122	0.790±0.122	0.806±0.039	0.824±0.037	0.849±0.064	0.869±0.078	0.832±0.055	0.832±0.055	0.832±0.055	0.832±0.055
SIRTUIN6	ACC		76.00±13.87	78.00±10.37	78.00±15.25	78.00±12.04	81.00±11.40	**83.00**±12.25	80.00±11.25	82.00±11.51	76.00±8.94	76.00±11.40	76.00±13.87	79.00±11.40	**84.00**±11.40	**83.00**±12.55	**83.00**±12.55	**83.00**±12.55
	Precision		0.783±0.135	0.809±0.074	0.809±0.074	0.820±0.105	0.840±0.146	0.818±0.120	0.837±0.108	0.790±0.066	0.790±0.095	0.790±0.122	0.820±0.081	0.849±0.109	0.842±0.116	0.842±0.116	0.842±0.116	0.842±0.116
	F1		0.753±0.142	0.770±0.120	0.770±0.164	0.769±0.134	0.807±0.118	0.827±0.155	0.796±0.125	0.186±0.816	0.750±0.105	0.749±0.127	0.749±0.150	0.779±0.130	0.838±0.116	0.827±0.129	0.827±0.129	0.827±0.129
Tae	ACC		50.99±9.54	50.32±7.35	50.30±9.87	46.95±8.28	51.63±13.32	55.55±10.20	52.28±9.21	52.95±6.83	60.97±8.97	62.95±9.92	63.59±8.39	61.59±5.53	**67.55**±4.33	**65.61**±8.69	**65.61**±8.69	**65.61**±8.69
	Precision		0.510±0.097	0.494±0.085	0.507±0.112	0.472±0.115	0.523±0.157	0.559±0.103	0.531±0.100	0.527±0.065	0.619±0.098	0.639±0.105	0.646±0.093	0.623±0.062	0.688±0.049	0.674±0.106	0.674±0.106	0.674±0.106
	F1		0.501±0.099	0.488±0.081	0.485±0.094	0.445±0.078	0.512±0.137	0.533±0.104	0.522±0.095	0.520±0.070	0.607±0.096	0.626±0.103	0.632±0.089	0.610±0.057	0.674±0.045	0.652±0.089	0.652±0.089	0.652±0.089
Vertebral Column	ACC		77.42±4.27	**80.65**±4.70	79.35±3.68	**80.32**±3.10	76.45±3.71	74.84±7.27	75.81±5.47	78.06±4.65	75.48±4.02	77.42±3.23	78.71±3.10	79.35±2.65	**79.49**±4.04	79.49±3.54	**79.49**±2.67	79.39±1.97
	Precision		0.780±0.038	0.810±0.047	0.798±0.031	0.807±0.028	0.730±0.044	0.707±0.098	0.724±0.069	0.754±0.058	0.761±0.050	0.780±0.032	0.795±0.031	0.798±0.022	0.782±0.043	0.782±0.031	0.782±0.029	0.782±0.020
	F1		0.775±0.040	0.806±0.046	0.794±0.034	0.803±0.029	0.725±0.054	0.695±0.104	0.712±0.068	0.735±0.064	0.754±0.043	0.774±0.031	0.788±0.029	0.794±0.023	0.787±0.078	0.787±0.065	0.787±0.054	0.787±0.043
Wine	ACC		94.38±1.97	93.84±3.01	92.71±2.43	91.03±2.99	94.40±1.94	93.84±2.29	93.29±3.13	94.40±1.94	93.24±2.57	94.94±2.34	93.79±3.13	93.81±2.37	**94.95**±2.32	**94.95**±2.32	**94.95**±2.32	**94.95**±2.32
	Precision		0.952±0.014	0.944±0.029	0.939±0.014	0.927±0.016	0.948±0.019	0.944±0.020	0.939±0.019	0.946±0.015	0.943±0.018	0.956±0.017	0.948±0.023	0.949±0.016	0.952±0.023	0.952±0.023	0.952±0.023	0.952±0.023
	F1		0.943±0.019	0.938±0.030	0.926±0.024	0.908±0.030	0.946±0.020	0.940±0.022	0.935±0.028	0.946±0.018	0.932±0.025	0.949±0.023	0.937±0.031	0.938±0.023	0.951±0.024	0.951±0.024	0.951±0.024	0.951±0.024
Zoo	ACC		94.00±5.48	93.00±5.70	91.14±6.36	88.14±4.36	80.29±7.47	78.29±5.28	77.29±6.84	76.80±4.67	**96.00**±6.32	**97.00**±4.47	**96.00**±6.32	**96.00**±6.52	84.29±10.28	84.29±9.05	84.29±9.05	84.29±9.05
	Precision		0.938±0.063	0.921±0.066	0.912±0.068	0.851±0.048	0.731±0.122	0.638±0.107	0.616±0.122	0.537±0.034	0.949±0.074	0.959±0.056	0.954±0.065	0.954±0.065	0.744±0.111	0.761±0.167	0.761±0.167	0.761±0.167
	F1		0.930±0.063	0.916±0.066	0.900±0.071	0.858±0.047	0.712±0.121	0.624±0.103	0.601±0.132	0.547±0.060	0.951±0.075	0.961±0.056	0.954±0.068	0.954±0.068	0.735±0.134	0.755±0.145	0.755±0.145	0.755±0.145
Mean ACC(%)			78.38	77.53	75.40	75.29	77.17	78.69	77.80	78.10	78.59	79.93	80.22	80.90	**82.32**	**82.25**	**82.28**	**82.40**

[9] W.-Y. Wen, J.-C. Li, S.-Y. Lin, J.-Y. Chen, and S.-C. Chang, "A fuzzy-matching model with grid reduction for lithography hotspot detection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 11, pp. 1671–1680, 2014.

[10] M. Luhandjula, "Fuzzy optimization: an appraisal," *Fuzzy sets and systems*, vol. 30, no. 3, pp. 257–282, 1989.

[11] J. M. Keller, M. R. Gray, and J. A. Givens, "A fuzzy k-nearest neighbor algorithm," *IEEE transactions on systems, man, and cybernetics*, no. 4, pp. 580–585, 1985.

[12] Z. Bian, C. M. Vong, P. K. Wong, and S. Wang, "Fuzzy knn method with adaptive nearest neighbors," *IEEE transactions on cybernetics*, vol. 52, no. 6, pp. 5380–5393, 2020.

[13] P. Bhaskaran and S. Prasanna, "An accuracy analysis of classical and quantum-enhanced k-nearest neighbor algorithm using canberra distance metric," *Knowledge and Information Systems*, vol. 67, no. 1, pp. 767–788, 2025.

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

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

[16] J. D. Martín-Guerrero and L. Lamata, "Quantum machine learning: A tutorial," *Neurocomputing*, vol. 470, pp. 457–461, 2022.

[17] N. Wiebe, A. Kapoor, and K. Svore, "Quantum algorithms for nearest-neighbor methods for supervised and unsupervised learning," *arXiv preprint arXiv:1401.2142*, 2014.

[18] E. Zardini, E. Blanzieri, and D. Pastorello, "A quantum k-nearest neighbors algorithm based on the euclidean distance estimation," *Quantum Machine Intelligence*, vol. 6, no. 1, p. 23, 2024.

[19] Y. Ruan, X. Xue, H. Liu, J. Tan, and X. Li, "Quantum algorithm for k-nearest neighbors classification based on the metric of hamming distance," *International Journal of Theoretical Physics*, vol. 56, no. 11, pp. 3496–3507, 2017.

[20] L.-Z. Gao, C.-Y. Lu, G.-D. Guo, X. Zhang, and S. Lin, "Quantum k-nearest neighbors classification algorithm based on mahalanobis distance," *Frontiers in Physics*, vol. 10, p. 1047466, 2022.

[21] C. Feng, B. Zhao, X. Zhou, X. Ding, and Z. Shan, "An enhanced quantum k-nearest neighbor classification algorithm based on polar distance," *Entropy*, vol. 25, no. 1, p. 127, 2023.

[22] L.-H. Gong, W. Ding, Z. Li, Y.-Z. Wang, and N.-R. Zhou, "Quantum k-nearest neighbor classification algorithm via a divide-and-conquer strategy," *Advanced Quantum Technologies*, vol. 7, no. 6, p. 2300221, 2024.

[23] A. Zeguendry, Z. Jarir, and M. Quafafou, "A fidelity-based quantum k-nearest neighbors (fqknn) for enhanced classification," in *2025 11th International Conference on Optimization and Applications (ICOA)*. IEEE, 2025, pp. 1–6.

[24] L. A. Zadeh, "Fuzzy sets," *Information and control*, vol. 8, no. 3, pp. 338–353, 1965.

[25] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*. Cambridge university press, 2010.

[26] A. Jacquier, O. Kondratyev, A. Lipton, and M. López de Prado, "Quantum machine learning and optimisation in finance," *Birmingham: Packt Publishing Ltd*, 2022.

[27] M. Akhtar, M. Tanveer, and M. Arshad, "Roboss: A robust, bounded, sparse, and smooth loss function for supervised learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 47, no. 1, pp. 149–160, 2025.

[28] Z. Zhang, "Improved adam optimizer for deep neural networks," in *2018 IEEE/ACM 26th international symposium on quality of service (IWQoS)*. Ieee, 2018, pp. 1–2.

[29] O. Rainio, J. Teuho, and R. Klén, "Evaluation metrics and statistical tests for machine learning," *Scientific Reports*, vol. 14, no. 1, p. 6086, 2024.