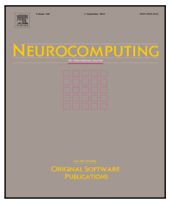
Neurocomputing 647 (2025) 130565



Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom





Quantum AI for Alzheimer's disease early screening

Giacomo Cappiello [a,*], Filippo Caruso [b,c,d,*]

[a] DiMaI "U. Dini", Università degli Studi di Firenze, Viale Morgagni 67/A, I-50134, Firenze, Italy
[b] Dept. of Physics and Astronomy, Florence Univ., Via Sansone 1, I-50019 Sesto Fiorentino, Italy
[c] LENS - European Laboratory for Non-Linear Spectroscopy, Florence Univ., Via Nello Carrara 1, I-50019 Sesto Fiorentino, Italy
[d] Istituto Nazionale di Ottica del Consiglio Nazionale delle Ricerche (CNR-INO), I-50019 Sesto Fiorentino, Italy

ARTICLE INFO

Communicated by K. Liu

Dataset link: https://archive.ics.uci.edu/dataset/732/darwin

Keywords:
Quantum machine learning
Alzheimer's disease
Supervised learning
Data classification
Parametrized quantum circuit

ABSTRACT

Quantum machine learning is a new research field combining quantum information science and machine learning. Quantum computing technologies appear to be particularly well-suited for addressing problems in the health sector efficiently. They have the potential to handle large datasets more effectively than classical models and offer greater transparency and interpretability for clinicians. Alzheimer's disease is a neurodegenerative brain disorder that mostly affects elderly people, causing important cognitive impairments. It is the most common cause of dementia and it has an effect on memory, thought, learning abilities and movement control. This type of disease has no cure, consequently an early diagnosis is fundamental for reducing its impact. The analysis of handwriting can be effective for diagnosing, as many researches have conjectured. The DARWIN (Diagnosis AlzheimeR WIth haNdwriting) dataset contains handwriting samples from people affected by Alzheimer's disease and a group of healthy people. Here we apply quantum AI to this use-case. In particular, we use this dataset to test classical methods for classification and compare their performances with the ones obtained via quantum machine learning methods. We find that quantum methods generally perform better than classical methods.

Our results pave the way for future new quantum machine learning applications in early-screening diagnostics in the healthcare domain.

1. Introduction

Neurodegenerative diseases are incurable conditions caused by the progressive degeneration of nerve cells [1]. Alzheimer's disease (AD) is one of the most common amongst them [2]. The predominant symptom in the early phase of AD is episodic memory impairment followed by progressive amnesia, a result of widespread brain damage. Since there is no cure for AD, it is critical to improve the approaches now used for diagnosis [3]. Early diagnosis can make patients eligible for clinical trials, which are essential for the development of new treatments. Participation in these trials can provide access to cutting-edge therapies that are not yet widely available [4] and it can help maintain cognitive function for a longer period, allowing individuals to live more independently and enjoy a better quality of life [5].

Many techniques have been used for this purpose. Neuroimaging techniques are commonly used to detect brain atrophy and other structural changes associated with AD; however their limitations include restricted availability and the inability to detect early-stage diseases [6]. Cognitive assessments, on the other hand, despite potentially being influenced by the patient's education level, measure abilities such as memory, attention, language, and problem-solving, while motor tests examine coordination, balance, and gait. This comprehensive approach aids in fully understanding the extent of the disease's impact [7]. Handwriting is a task that involves both cognitive and motor functions to plan and properly execute the movements that are required with high coordination [8]. The analysis of handwriting and drawing dynamics is a valid non-invasive way for evaluating AD progression. For this purpose a protocol of 25 tasks was defined in Ref. [9]. Each handwriting/drawing task is described by 18 features and the data collected amongst a group of healthy people and patients made up the DARWIN (Diagnosis AlzheimeR WIth haNdwriting) dataset. In Ref. [10] it is described how this dataset can be exploited by machine learning models to discriminate between AD patients and healthy people.

Machine learning (ML) is a subfield of artificial intelligence (AI) that focuses on the development of algorithms and statistical models that enable computers to learn and make predictions or decisions based on data [11]. These models can identify patterns, trends, and relationships within large datasets, allowing for tasks such as classification, regression, clustering, and anomaly detection [12]. One of the main obstacles

* Corresponding authors.
E-mail addresses: giacomo.cappiello@unifi.it (G. Cappiello), filippo.caruso@unifi.it (F. Caruso).

https://doi.org/10.1016/j.neucom.2025.130565
Received 19 July 2024; Received in revised form 6 April 2025; Accepted 21 May 2025
Available online 6 June 2025
0925-2312/© 2025 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

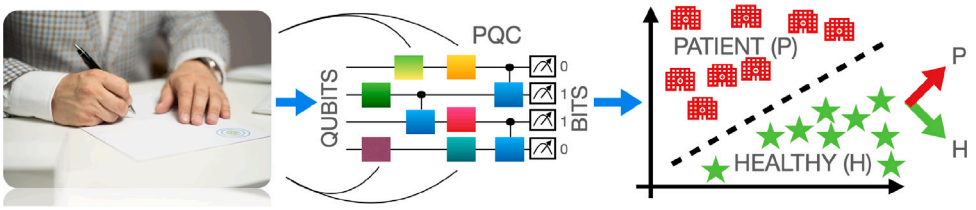


Fig. 1. The data obtained from the handwriting tasks are processed and mapped into quantum states through a parametrized quantum circuit (PQC). Then a classification algorithm is applied to distinguish the samples between 'patient' (P) and 'healthy' (H).

to the usage of AI in the healthcare field, and in particular for the diagnosis of diseases, is the "black box" nature of many AI models. Medical practitioners often find it challenging to trust AI predictions because they cannot easily understand how these models arrive at their conclusions [13]. Quantum AI can improve the transparency and interpretability of AI models by leveraging the principles of quantum mechanics and providing a clear visual representation of models [14], allowing clinicians to understand the decision-making process better. Additionally, it can handle large volumes of data more efficiently than classical AI, making it a powerful tool for data-intensive tasks [15]. In quantum machine learning (QML) inputs are encoded in quantum states and computation is done on a quantum computer [16,17]. Although quantum processing units (QPUs) are still subjected to relatively large noise rates, it seems that applications in ML can lead to remarkable results. For example, in Ref. [18] it is shown how variational quantum models, namely models based on updating parameters in a quantum circuit through classical optimization, can exhibit a significant learning advantage in solving a regression task with input data from the fashion-MNIST dataset. A review of the contributions of QML in medical image analysis is given in Ref. [19]. A previous study [20] approaches via a variational quantum classifier (VQC) the problem of classification defined by the same dataset we consider in our work.

Here we focus in particular on quantum kernel methods, i.e. kernel methods in which the feature space is a space of quantum states (see Fig. 1). We utilize these algorithms to classify the samples and evaluate their performance against a classical kernel algorithm, two additional classical methods and hybrid neural networks. The paper is organized as follows. In Section 2 we describe how the DARWIN dataset was created and we define the classical and quantum methods for classification. Then in Sections 3 and 4 we report their performances, we discuss the results and draw our conclusions.

2. Methods

2.1. Dataset

The DARWIN dataset is composed of handwriting data collected according to the protocol defined in Ref. [9]. This protocol includes 25 tasks that can be divided in the following categories: graphic tasks, like joining points and drawing geometrical figures; copy tasks, testing the ability of repeating more complex gestures to write letters, numbers and words; memory tasks, involving writing previously memorized words; and dictation tasks. The protocol was submitted to 174 participants: 89 AD patients and 85 healthy people, recruited so that the two groups would match in terms of age, education and gender. None of the participants was taking medications that influenced their cognitive abilities. The tasks were performed on paper sheets put on top of a tablet equipped with a pen that could both write in ink and sample coordinates of the tip and pressure exerted. The tablet was connected to a PC that recorded movements and displayed them in real-time. A software was used to extract 18 features for each task. Features and tasks are described in details in Appendix. The dataset includes also a feature for identification of participants and another one that indicates whether the sample is associated to an AD patient (P) or an healthy person (H). This feature is used as label in the classification problem via a (classical or quantum) supervised learning model.

2.2. Classical SVC

Support vector machines (SVMs) are supervised learning models for classification and regression [21]. Considering a binary classification problem in which the aim is to decide whether a set of samples belongs to one of two classes, each data point is viewed as a d-dimensional vector. The goal is to separate the points with a $(d-1)$-dimensional hyperplane such that the distance from the hyperplane to the nearest data points on each side, called support vectors, is maximized. This can be formulated as a constrained optimization problem in the following way:

$$\begin{aligned} &\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \\ &\text{subject to:} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \quad \forall i = 1, \dots, n \end{aligned} \tag{1}$$

where $\mathbf{w} \in \mathbb{R}^d$ is the weight vector, $b \in \mathbb{R}$ is the bias, $y_i \in \{-1, +1\}$ are the class labels and $\mathbf{x}_i \in \mathbb{R}^d$ are the feature vectors. The hyperplane equation is $\mathbf{w}^\top \mathbf{x} + b = 0$. A graphic representation with $d = 2$ is given in Fig. 2.

In the original space this problem has often no solution, i.e. the sets of point are not linearly separable. A SVM uses the kernel trick, the technique of mapping inputs into high-dimensional feature spaces, to solve this problem [22]. The problem then becomes:

$$\begin{aligned} &\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \\ &\text{subject to:} \quad y_i(\mathbf{w}^\top \phi(\mathbf{x}_i) + b) \geq 1 \quad \forall i = 1, \dots, n \end{aligned} \tag{2}$$

where the feature map $\phi : \mathbb{R}^d \to \mathcal{F}$ transforms the inputs into the feature space. The dot products of input pairs in the feature space are called kernels. Defining a map $\phi : \mathbb{R}^d \to \mathcal{F}$ from the is equivalent to defining a kernel function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ that maps couples of inputs into their dot products in the feature space. The kernels appear in the dual version of the problem, defined using the Lagrangian

The Lagrangian associated to the optimization problem (2) is

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(\mathbf{w}^\top \phi(\mathbf{x}_i) + b) - 1\right) \tag{3}$$

where $\alpha_i \geq 0$ are the Lagrange multipliers associated with each constraint. Differentiating the Lagrangian and setting the derivatives equal to zero, we obtain the dual form of (2):

$$\begin{aligned} &\max_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) \\ &\quad \text{subject to:} \quad \sum_{i=1}^{n} \alpha_i y_i = 0 \\ &\qquad\qquad\qquad\quad \alpha_i \geq 1, \quad \forall i. \end{aligned} \tag{4}$$

Here the constraints are simpler and the optimization is only over the Lagrangian multipliers.

For our classification task we consider different kernel functions [23]. The radial basis function (RBF) kernel maps a pair of inputs $(\mathbf{x}, \mathbf{x}')$ into $k_{rbf}(\mathbf{x}, \mathbf{x}') = \exp\left(-\gamma|\mathbf{x} - \mathbf{x}'|^2\right)$. This is one of the most common choices amongst kernel functions and it is used by default in the standard version of SVC implemented in the Scikit library in Python [24]. The linear kernel function is a simple dot product, $k_{lin}(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle$.

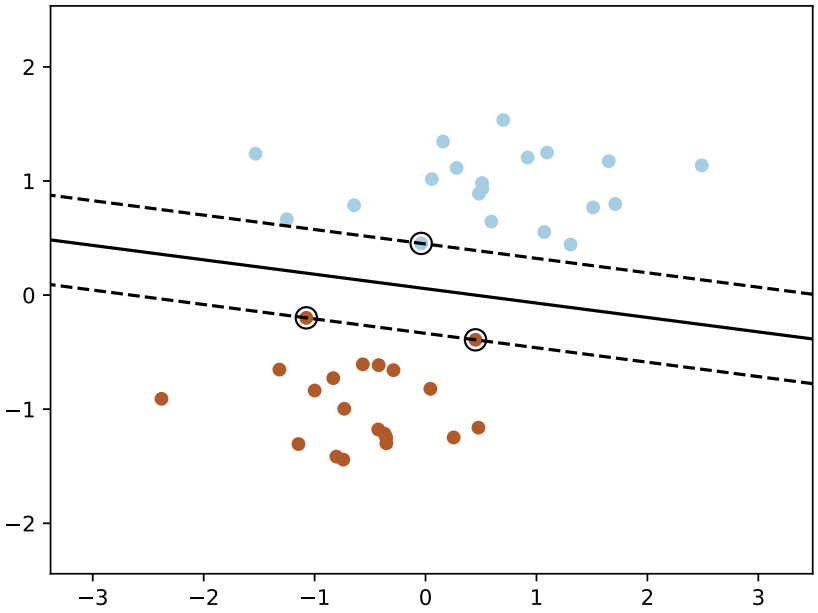


Fig. 2. Graphic representation of a classification model in dimension $d = 2$. The hyperplane is a straight line. Support vectors are circled.

The polynomial function is $k_{pol}(\mathbf{x}, \mathbf{x}') = (\gamma\langle \mathbf{x}, \mathbf{x}'\rangle + r)^d$. Lastly, the sigmoid kernel is given by $k_{sig}(\mathbf{x}, \mathbf{x}') = \tanh(\gamma\langle \mathbf{x}, \mathbf{x}'\rangle + r)$.

Given a kernel function k, the Gram matrix associated to it is the matrix whose entries are $G_{ij} = k(x_i, x_j)$, where x_i and x_j are two inputs. Its eigenvalue have to be all positive so that the kernel defines a valid inner product in the feature space. Moreover the eigenvalue curve of the matrix G is flat when the problem is difficult to learn, whereas a desirable situation is found when the eigenvalue curve is non-flat and decays, either polynomially or exponentially fast [25].

2.3. k-nearest neighbors

The k-Nearest Neighbors (kNN) model is a method used both for classification and regression [26]. With this technique, a label is assigned to a data point based on the most common label among its k nearest neighbors. In particular, the probability that the unknown sample $\mathbf{x}_0$ belongs to class j is given by

$$P(Y = j \mid X = \mathbf{x}_0) = \frac{1}{k} \sum_{i \in N_0} I(y_i = j), \tag{5}$$

where k is the number of nearest neighbors, N_0 is the set of the k nearest neighbors to $\mathbf{x}_0$ and $I(y_i = j)$ is an indicator function that equals 1 if the ith neighbor y_i has the label j, and 0 otherwise.

2.4. Decision tree

A Decision Tree (DT) is a decision support tool that visually represents decisions and their possible outcomes in a hierarchical, tree-like structure [27]. This structure is composed of:

- nodes, which are decision points where a specific feature is tested;
- branches, representing the possible outcomes of the test conducted at each node;
- leaf nodes where a final decision is made and a label is assigned.

The tree is constructed by recursively splitting the data based on the attribute that provides the best separation according to a chosen criterion (e.g., Gini impurity or information gain). This process continues until the stopping criteria are met, such as a maximum tree depth or a minimum number of samples per leaf node.

2.5. Quantum SVC

In the quantum version of SVMs, inputs are mapped into the space of quantum states [28]. The feature map is defined by a parametrized quantum circuit (PQC), whose parameters are given by the features of the inputs. A PQC is a sequence of parametrized unitary operators, called quantum logical gates, acting on a set of qubits, or quantum bits, which are two-state quantum systems [29]. To introduce the representation of qubit states, we need to define Dirac's notation for unit vectors in $\mathbb{C}^2$. The standard basis of $\mathbb{C}^2$ is given by the vectors $\{|0\rangle, |1\rangle\}$, where $|0\rangle = \begin{pmatrix}1\\0\end{pmatrix}$, and $|1\rangle = \begin{pmatrix}0\\1\end{pmatrix}$. A generic unit vector in $\mathbb{C}^2$ is $|\phi\rangle = \alpha|0\rangle + \beta|1\rangle$, where $\alpha, \beta \in \mathbb{C}$ are such that $|\alpha|^2 + |\beta|^2 = 1$. The inner product of two unit vectors $|\phi\rangle, |\psi\rangle \in \mathbb{C}^2$ is denoted by $\langle\phi|\psi\rangle$, while their outer product is $|\phi\rangle\langle\psi|$. The state of a qubit is defined by a hermitian, positive semi-definite matrix $\rho \in \mathbb{C}^{2\times 2}$, such that its trace is equal to 1. The state is pure when there exists a $|\phi\rangle \in \mathbb{C}^2$ such that $\rho = |\phi\rangle\langle\phi|$, otherwise it is mixed.

To compute kernels we measure the fidelity of pairs of quantum states. Fidelity evaluates the degree of similarity of a pair of quantum states [30]. Given a pair of pure quantum states $\rho = |\phi\rangle\langle\phi|$ and $\sigma = |\psi\rangle\langle\psi|$, their fidelity is defined by $\mathcal{F}(\rho, \sigma) = |\langle\phi|\psi\rangle|^2$. The most used generalization of this definition for a generic pair of mixed states ρ and σ is $\mathcal{F}(\rho, \sigma) = \left(\mathrm{Tr}(\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}})\right)^2$, where Tr denotes the trace function. The calculation of fidelities is the only computation requiring a quantum hardware. All other computations in quantum SVCs are handled by a classical hardware.

Quantum SVC is implemented using the standard version in Scikit library with precomputed kernels. We use different PQCs, scaling the number of qubits with similar ansätze. The ansätze are chosen considering the following aspects. The number of quantum gates needs to be small enough to make the time needed to perform the calculations as small as possible. Time of execution is a critical point in quantum computation, since it is often limited on real QPUs. An ansatz is expressive when it explores the space of unitaries as fully and uniformally as possible. Expressive ansätze are desirable when facing a problem with no prior knowledge of it, because of their ability to adapt to the task. However it is proved that an expressive anstatz is difficult to train [31], while an ansatz with low expressibility can achieve a better performance on the task it is specialized to solve [32]. One possible

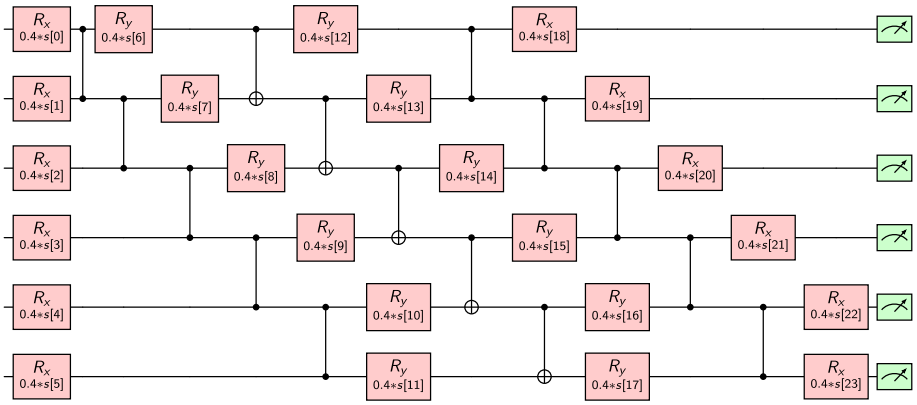


Fig. 3. Graphic representation of a 6-qubit PQC that maps a sample $\mathbf{s} = (s[0], s[1], \dots, s[23])$ into a quantum state. The parameters in the rotation gates are given by a bandwidth factor $b = 0.4$, chosen in order to maximize the accuracy, times the components of $\mathbf{s}$.

solution to reduce expressibility is introducing a bandwidth factor, a scaling hyperparameter that multiplies the parameters in the PQC. The bandwidth parameter limits the reach of the feature map [33].

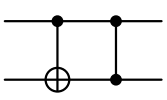
The PQCs we use are composed of R_x and R_y rotation gates, coontrolled-z and controlled-x gates. The gates R_x and R_y are 1-qubit parametrized unitaries that perform a rotation of an angle given by its parameter about the x-axis and the y-axis of the Bloch sphere, respectively. The Bloch sphere is a 3-dimensional geometrical representation of the states of a qubit. The matrix representations of the rotation gates are the following:

$$R_x(\theta) = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -i\sin\left(\frac{\theta}{2}\right) \\ -i\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix}, \quad R_y(\theta) = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix}. \tag{6}$$

A controlled gate is a quantum operation acting on multiple qubits. One or more qubits act as controls, i.e. based on their configuration, a certain gate is applied or not to the remaining qubit, called target. The controlled-x gate is a 2-qubit controlled gate. When the control is in the state $|1\rangle$ an X gate is applied to the target, where $X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. The controlled-z is also a 2-qubit controlled gate. It is symmetric, meaning that both qubits act as control, activated by the state $|1\rangle$, and target. The matrix representations of controlled-x and controlled-z gates are

$$CX = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad CZ = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}. \tag{7}$$

A control qubit activated by the state $|1\rangle$ is represented in a circuit as •. A target qubit to which an X gate is applied is denoted by $\oplus$. Therefore the graphic representations of controlled-x and controlled-z in the circuits are, respectively, the following.



In Fig. 3 we show the graphic representation of a 6-qubit circuit, composed by layers of R_x and R_y gates alternating with layers of controlled-z and controlled-x gates. On the right-hand side are symbols representing measurements applied to the qubits, converting the quantum states in a sequence of classical bits. The number of circuit parameters is 24. A similar structure with the same number of parameters is used to define also an 8-qubit and a 12-qubit circuit. The 8-qubit ansatz is built with three encoding layers of R_x, R_y and R_x again, alternating with two entangling layers of CZ and CX. Lastly the 12-qubit ansatz is defined by two encoding layers of R_x and R_y alternating with 2 entangling layers of CZ and CX.

The number of parameters must be equal to the number of features of the samples, so that each input feature corresponds to a parameter. Therefore the dataset needs to be processed through a principal component analysis (PCA) prior to applying the methods [34]. In order to reduce expressibility, the parameters in the rotation gates are multiplied by a tunable bandwidth hyperparameter b.

We consider an ansatz with n qubits. Given a sample $\mathbf{x}$, if $\rho_0 = (|0\rangle\langle 0|)^{\otimes n} = |0\rangle\langle 0| \otimes |0\rangle\langle 0| \otimes \cdots \otimes |0\rangle\langle 0|$ is the initial state, the quantum state given by the PQC considered is

$$\rho(\mathbf{x}; b) = U(\mathbf{x}; b)\rho_0 U^\dagger(\mathbf{x}; b) \tag{8}$$

where $U(\mathbf{x}; b)$ is the encoding unitary obtained as the composition of all gates in the circuit. Therefore the fidelity quantum kernel of a couple of sample vectors $(\mathbf{x}, \mathbf{x}')$ can be written as

$$k_{FQ}(\mathbf{x}, \mathbf{x}'; b) = \left[\mathrm{Tr}\left(\sqrt{\sqrt{\rho(\mathbf{x}; b)}\rho(\mathbf{x}'; b)\sqrt{\rho(\mathbf{x}; b)}}\right)\right]^2. \tag{9}$$

Obtaining a closed-form expression for the fidelity quantum kernel is extremely difficult due to the exponential growth of the Hilbert space and the complex interplay of entangling operations. This difficulty suggests that quantum kernel methods explore feature spaces that may not be efficiently accessible using classical methods.

The eigenvalue curves of the Gram matrices associated with the three PQCs, all computed with the same fixed bandwidth value of 0.5, are shown in Fig. 4. The eigenvalue curves are positive and exhibit a decaying trend. Meanwhile, Fig. 5 illustrates how the condition number, i.e. the ratio of the largest to the smallest eigenvalue, of the kernel matrix decreases as the bandwidth increases. For small bandwidths the condition number is extremely high, indicating that the kernel matrix is nearly singular. This suggests that small bandwidths cause the kernel to become highly sensitive to perturbations, potentially leading to numerical instability and overfitting. As the bandwidth increases, the condition number drops exponentially, showing that the kernel matrices become better conditioned and more numerically stable. The 6-qubit circuit consistently has the highest condition number. However the relationship between qubit count, bandwidth, and eigenvalue distribution appears to be complex, suggesting that other factors, such as entanglement structure, may play a significant role in determining numerical stability.

For this reason, we now perform an entanglement analysis. Given a quantum system consisting of two subsystems A and B, the state of the subsystem A is described by the reduced density matrix $\rho_A = \mathrm{Tr}_B(\rho_{AB})$, where Tr_B is the partial trace function over B, and ρ_{AB} is the total density matrix [35]. The Von Neumann entropy computed on the

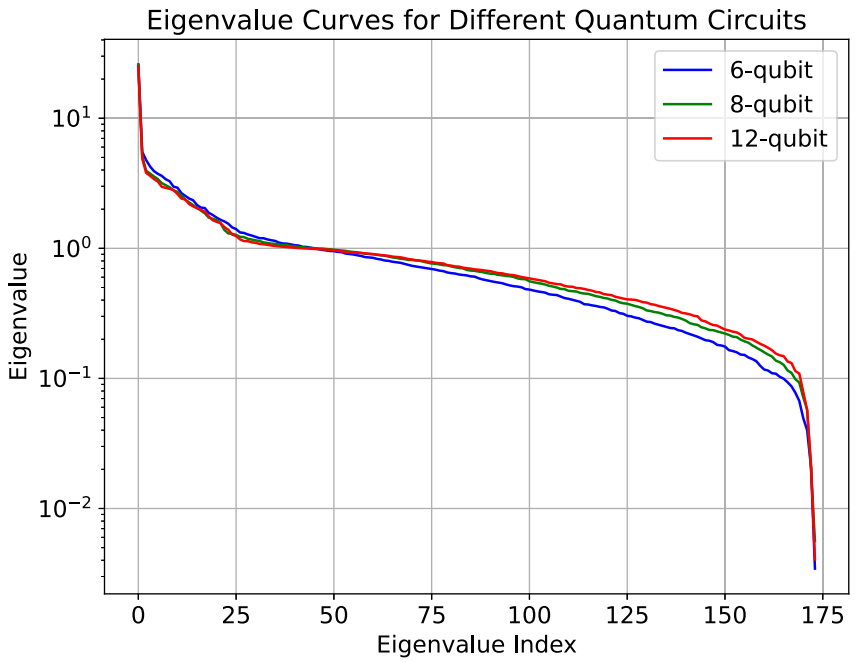


Fig. 4. Plot (in logarithmic scale) of the eigenvalue curves associated with the kernel matrices defined by the 6-qubit, 8-qubit and 12-qubit circuit described in the main text, computed with the same fixed bandwidth value of 0.5.

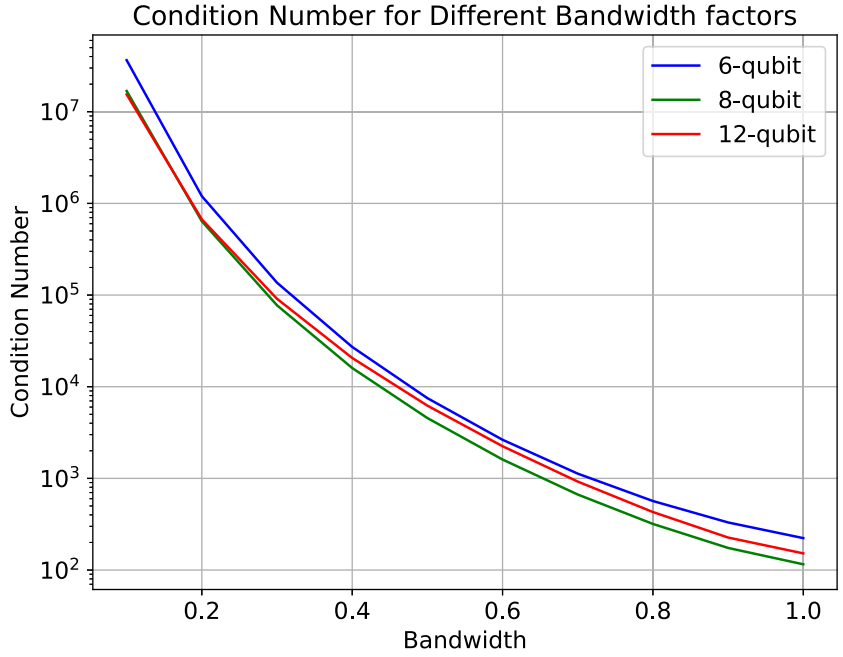


Fig. 5. Plot (in logarithmic scale) of condition number of the kernel matrix as the bandwidth increases, for the three different quantum circuits (6, 8, and 12 qubits).

subsystem A measures the degree of entanglement between A and B and is given by

$$S(\rho_A) = -\mathrm{Tr}(\rho_A \log_2 \rho_A). \tag{10}$$

For a pure quantum state, the Von Neumann entropy is zero, whereas for mixed states, it takes positive values. In the case of a system of n qubits, the maximum value is n.

In our case, we measure the Von Neumann entropy of the subsystems obtained tracing out half the qubits of the quantum states defined by the PQCs. For different values of the bandwidth, we initialize the PQCs with random parameters scaled by a bandwidth and compute the Von Neumann entropy. The process is repeated 20 times for each bandwidth value to average out the randomness of the initializations. As shown in Fig. 6, for all qubit sizes entropy increases as bandwidth increases up to a certain point and after reaching a peak, it stabilizes or fluctuates slightly. The 6-qubit system reaches a higher peak entropy, likely because it has three entangling layers, whereas the other circuits have only two. The fact that the 6-qubit circuit has both a higher entropy and a higher condition number across all bandwidths suggests that there may be a link between entanglement structure and numerical stability. Interestingly, the systems become more stable even as entanglement increases.

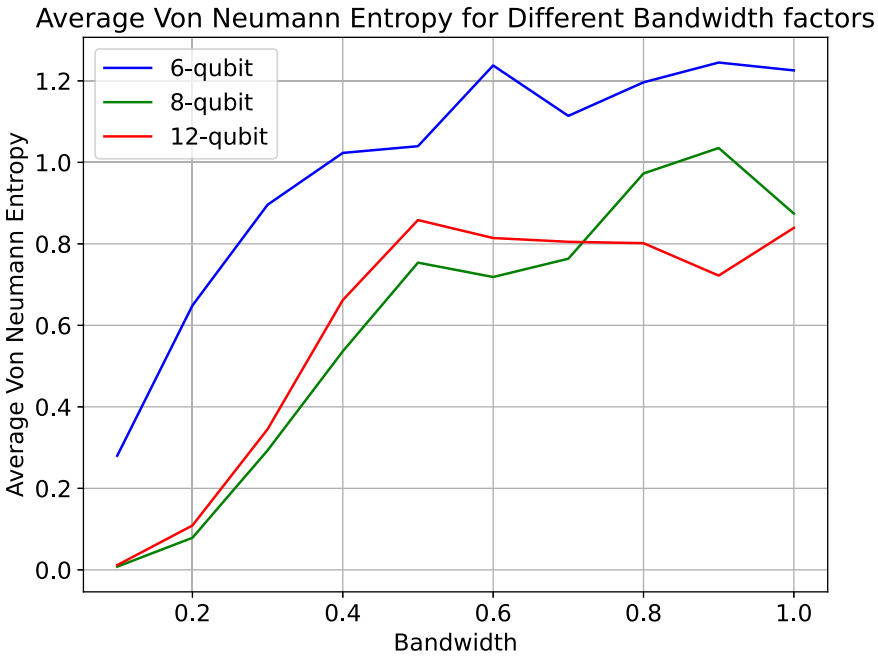


Fig. 6. Plot of average Von Neumann Entropy as the bandwidth increases, for the three different quantum circuits (6, 8, and 12 qubits).

2.6. Hybrid neural network

Neural networks are computational models inspired by the human brain, composed of interconnected layers of artificial neurons [36]. These models learn patterns from data using weight adjustments through backpropagation. They are widely used in tasks such as image recognition, natural language processing, and classification.

Hybrid neural networks (HNN) combine classical and quantum computing components by integrating quantum layers into classical architectures [37]. In our implementation, the model begins with an input layer of 24 nodes, corresponding to the 24 features. This is followed by a hidden layer with 16 nodes and a quantum layer, that is a quantum circuit. As with the feature map in quantum SVC, here we consider 3 different PQCs: 6-qubit, 8-qubit and 12-qubit circuit. Finally, the output layer consists of 2 nodes, which provide a probability distribution used for label assignment. Input and hidden layers are followed by a Rectified Linear Unit activation function, defined as $f(x) = \max(0, x)$, used to introduce non-linearity.

2.7. Implementation

Our algorithms are implemented in Python, using the functionalities of Scikit and Qiskit libraries [38]. First we perform some preprocessing on the dataset. In particular we drop the identification feature, while keeping the feature healthy/AD's patient (H/P) as a label. We standardize the samples, by subtracting the mean and scaling to unit variance. Then we perform a PCA with 24 components. Finally we standardize the set of samples derived from the PCA. In fact, from our analysis it emerges that normalizing twice improves the performance of both classical and quantum methods. The dataset is divided for cross-validation using ShuffleSplit, performing 20 splits: 60% of the data is used for training, 20% for validation and 20% as test set. For each method, a parameter grid is defined for hyperparameter tuning: in this phase the model is trained on the training set and evaluated on the validation set. After that, the model with the best hyperparameters is trained on the union of training and validation set and is evaluated on the test set. The mean and standard deviation of the accuracies across the splits are calculated.

Both classical and quantum SVC use Sequential Minimal Optimization (SMO) as optimizer. The large quadratic optimization problem (4) is broken into smaller subproblems: at each step SMO picks two Lagrangian multipliers to optimize while keeping others fixed. For classical SVC, the parameter grid includes different options for 4 hyperparameters:

- kernel function: 'linear', 'poly', 'rbf', 'sigmoid';
- 'C' (balances error minimization and model complexity): 0.1, 1, 10, 100;
- 'gamma' (influence range of training examples): 'scale' (scales with the number of samples), 1, 0.1, 0.01, 0.001;
- 'tol' (precision for stopping criteria): $10^{-3}, 10^{-4}, 10^{-5}$.

To simplify the computations, the only hyperparameter tuned during the validation phase is the bandwidth parameter, which takes values from the set $\{0.1, 0.2, 0.3, \ldots, 1\}$. Therefore, except for selecting the option of running the model with precomputed kernels, the other parameters of SVC() are left by default: C = 1.0, 'gamma' = 'scale' and 'tol' = 0.001. The quantum methods are simulated locally using Aer Simulator.

Two more classical methods for classifications are used for comparison, kNN and DT. In kNN no optimization problem is involved, because the algorithm simply stores the training data and computes distances to classify new test samples. Here the parameter grid includes 3 hyperparametrs:

- number of neighbors: 3, 5, 7 or 9;
- metric: 'uniform', 'distance';
- type of weights: 'euclidean', 'manhattan', 'minkowski'.

In DT, the optimization process involves finding, at each step, the feature and threshold that minimize an impurity function. The parameter grid includes 5 hyperparameters:

- impurity: 'gini', 'entropy';
- splitter: 'best', 'random';
- maximum depth: none, 10, 20, 30;
- minimum samples for split: 2, 5, 10;
- minimum samples for leaf: 1, 2, 4.

A hybrid neural network is also used for benchmarking. The quantum layer is implemented using Pennylane's AngleEmbedding and BasicEntanglerLayers. The model weights are updated using the Adam

Table 1
Mean accuracy and standard deviation (expressed in percentage) of classical SVC, k-Nearest Neighbors, decision tree, quantum SVCs and HNNs defined by 6-qubit, 8-qubit and 12-qubit PQCs.

Classical			
SVC		kNN	DT
85.28 ± 7.46		69.57 ± 8.20	73.57 ± 5.26
Quantum			
	6q	8q	12q
SVC	83.57 ± 4.59	83.14 ± 4.51	88.29 ± 4.69
HNN	81.21 ± 5.04	81.07 ± 5.29	79.64 ± 5.87

Table 2
Mean accuracy and standard deviation (expressed in percentage) of classical SVC and 12-qubit SVC models applied to the subset of features obtained through copy, graphic or memory and dictation tasks.

	Classical	12q
Copy	85.57 ± 8.00	85.71 ± 7.06
Graphic	78.14 ± 4.97	81.29 ± 5.68
Memory	79.28 ± 6.19	78.57 ± 4.99

optimizer with a learning rate of 0.001 and a weight decay of 10^{-5}. CrossEntropyLoss is used as the loss function. The dataset is split, reserving 40% of the samples for testing. Training runs for 200 epochs, using mini-batches of size 20. The entire process is repeated 20 times, storing the test accuracies to compute the mean and standard deviation for performance evaluation.

3. Results and discussion

In Table 1 we show the mean accuracy and standard deviation of classical and quantum methods. Amongst classical methods, SVC has the highest mean accuracy, but the standard deviation is relatively high, suggesting that its performance can vary significantly across different splits; kNN and DT are less effective in terms of average performance, although the latter offers more consistent results. The 6-qubit and 8-qubit SVCs perform comparably to the classical SVC, with slightly lower accuracy but a smaller standard deviation, indicating more consistent performance. The quantum SVC with 12 qubits achieves the highest accuracy and consistency, outperforming all other models, both classical and quantum. This suggests that increasing the number of qubits can enhance the performance of quantum kernel classifiers. However, unlike quantum SVCs, scaling up the number of qubits does not improve the performance of HNNs, which remains lower than both classical and quantum SVC models. This indicates that the kernel-based approach may be more effective in leveraging quantum feature maps with minimal hyperparameter tuning.

3.1. Data subsampling

Now we compare the performance of classical and quantum methods when restricted to subsets of features. For simplicity, we consider only the classical SVC and 12-qubit quantum SVC. To properly evaluate the performance of the methods on feature subsets, we use for both methods the hyperparameters that have been most frequently chosen in the previous analysis. First we use one category of features at a time, e.g. the features derived only from graphic tasks, then only copy tasks, and finally only memory and dictation tasks. The dataset is divided for cross-validation using ShuffleSplit, performing 20 splits: 80% of the data is used for training and 20% for testing. In Table 2 we show the mean accuracy and standard deviation of classical and quantum methods.

Subsequently we consider the features obtained from each of the 25 tasks individually. For each task we run classical SVC, store the predictions made by the 25 models and use the majority vote decision rule [39] to select the outputs; namely for each sample we select the output predicted by most of the models. Since each task is described by 18 features, we need to define a new PQC with 18 parameters. For this purpose we choose a 9-qubit circuit. After considering all 25 tasks, we select the 5 tasks that achieve singularly the best accuracies in the previous runs. For classical SVC the tasks are 21, 17, 16, 7, 23, while for quantum SVC they are 21, 17, 24, 14, 23. As before we use the majority vote decision rule to select the predictions amongst those made by the 5 models associated with the tasks. In Table 3 we show the mean accuracy and standard deviation of classical and quantum methods over 20 splittings.

Table 3
Mean accuracy and standard deviation (expressed in percentage) of classical SVC and 9-qubit SVC models achieved combining with the majority vote decision rule the predictions made considering all tasks individually and considering only the best 5 tasks.

	Classical	9q
All 25	85.71 ± 3.59	86.00 ± 4.30
Best 5	80.28 ± 4.23	81.35 ± 5.19

Table 4
Accuracy (expressed in percentage) of quantum SVCs run on Aer Simulator without noise and on the noisy backend Fake Melbourne; for the latter the accuracy is obtained combining the predictions made on 20 executions with the majority vote decision rule.

	6q	8q	12q
Aer Simulator	72.86	62.86	77.14
Fake Melbourne	77.14	64.29	77.14

Across all data subsampling scenarios, the quantum algorithm consistently demonstrates improved or comparable performance, underscoring its robustness and potential benefits in managing diverse data types.

3.2. Noise robustness

Due to limited accessibility to real quantum hardware, computation of kernels is simulated without noise on a classical hardware.

Now we compare the performance of the methods executed on a noiseless Aer Simulator with those run on a fake backend. A fake backend is a simulated quantum computing environment that mimics the behavior of real IBM Quantum systems, providing real hardware qubit properties to simulate realistic conditions. We select a splitting of the dataset in training and test set, leaving 20% of the samples for the latter, and we run the quantum SVCs on the Aer Simulator without noise, with number of shots equal to 256. This means that each circuit run is executed 256 times and the measurement results are aggregated to provide a more accurate representation of the quantum state. The default number of shots is usually set to 1024, but we choose to reduce this to speed up the computation, especially when using the fake backend. Then, with the same splitting and the same number of shots, we run the methods on Fake Melbourne backend, which simulates a 14-qubit quantum processor. The average T1 and T2 times of Fake Melbourne are 50 and 70 microseconds respectively, where T1 is a measure of how long a qubit can retain its state before it decays to ground state, and T2 indicates how long a qubit can stay in a superposition state before losing coherence, due to interactions with the environment [40]. A noiseless backend always yields the same result, whereas the performance of a fake backend, influenced by noise, is variable. Therefore we run the methods on the fake backend 20 times, storing their predictions and using the majority rule to yield the final prediction. In Fig. 7 we plot, for each of the 3 methods, the accuracies achieved in the noiseless run and the 20 noisy runs. In Table 4 we compare the performance of the noiseless models and the noisy models combined with the majority vote decision rule.

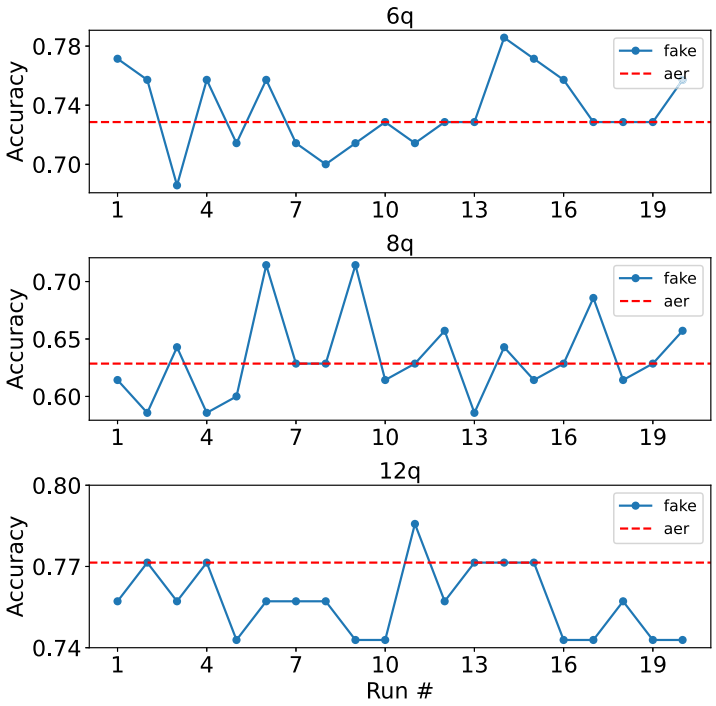


Fig. 7. Accuracies of the quantum SVCs run on a noiseless simulator and on a noisy fake backend.

The comparable or slightly better accuracy of the Fake Melbourne backend indicates that running multiple executions and using majority voting is a good strategy to mitigate variability caused by noise, achieving more consistent and accurate outcomes.

4. Conclusion

The aim of this work is comparing the performances of classical methods and quantum methods for classifying samples of AD patients and healthy people subjected to handwriting tests. The classical models used are Support Vector Classification, k-Nearest Neighbors and Decision Tree; quantum SVCs are implemented using ansätze with 6, 8, and 12 qubits; hybrid neural networks maintain the same classical architecture but incorporate a quantum layer constructed with circuits of 6, 8, and 12 qubits. Among the classical methods, the SVC delivers the highest performance, albeit with significant inconsistency: average accuracy is 85.28% and standard deviation 7.46%. Both kNN and DT are significantly less accurate. Evaluating the quantum methods we find that 6 and 8-qubit SVCs perform similarly to the classical SVC with slightly lower accuracy but better consistency. The quantum SVC with 12 qubits emerges as the most effective model, demonstrating superior accuracy and consistency. This indicates that increasing the number of qubits can significantly boost the performance of quantum kernel classifiers, potentially unlocking more advanced capabilities and better results. The HNN's higher complexity does not translate into better performance, possibly due to overfitting or difficulties in training deep hybrid models.

In addition to running the methods considering all features together we have also defined methods based on data subsampling. We have considered separately the features associated with the 3 categories of tasks, namely features derived from graphic tasks, copy tasks or memory and dictation tasks. In this case we just compare the performances of the 2 best models, i.e. classical SVC and 12-qubit SVC. The quantum algorithm generally shows improved or comparable performance across different feature subsets, highlighting its robustness and potential advantages in handling diverse types of data. Then we have examined the features obtained from each of the 25 task individually, defining a model for each task, and we have used the majority vote decision rule to combine the predictions made by the 25 models; after that we have considered just the best 5 tasks, i.e. the tasks that have achieved singularly the best accuracies in the previous runs. The quantum SVC demonstrates an advantage in accuracy over the classical SVC, whether considering all tasks or just the best ones, though it tends to have higher variability in its predictions.

Kernels in quantum methods have been calculated simulating quantum computation without noise. We have also investigated how noise affects the performance of quantum methods using a fake backend, which mimics the behavior of a real noisy backend. Averaging the predictions from several executions on the noisy backend, with the majority rule, leads to a slightly superior accuracy, counteracting the variability introduced by noise. Several studies in classical machine learning have explored how introducing randomness and noise can enhance the performance of kernel methods [41,42]. This concept holds significant promise when applied to quantum devices, which inherently operate in noisy environments. Noise may facilitate a broader exploration of the solution space, potentially leading to better solutions that might be missed in a noise-free environment. Regarding the use of real quantum hardware, currently, only IBM Quantum systems with 127 or more qubits are accessible, and each run for our computations takes 2 to 3 s. Similar considerations can be made for other quantum device providers. Despite computing multiple kernels in a single run, the execution of the methods exceeds the few minutes allocated in the free plan. Additionally, the number of kernels to compute increases quadratically with the number of samples. This makes quantum kernel methods more prone to time constraints compared to other quantum classifiers that can assign a label to a sample in a single run. Exploring optimizations, such as more efficient kernel computation techniques or hybrid quantum–classical approaches, could help mitigate these time constraints and make quantum kernel methods more feasible for real-world applications.

While our experimental results are competitive with the current state of the art, there remains potential for further enhancement. In the quantum circuits we have used, only one hyperparameter, the bandwidth, is tuned in the validation phase. Enhancing the results could be achieved by incorporating one or more layers of parametric

Table A.1
List of tasks performed. The tasks are divided in the categories memory and dictation (M), graphic (G), and copy (C).

#	Description	Category
1	Signature drawing	M
2	Join two points with a horizontal line, continuously for four times	G
3	Join two points with a vertical line, continuously for four times	G
4	Retrace a circle (6 cm of diameter) continuously for four times	G
5	Retrace a circle (3 cm of diameter) continuously for four times	G
6	Copy the letters 'l', 'm' and 'p'	C
7	Copy the letters on the adjacent rows	C
8	Write cursively a sequence of four lowercase letter 'l', in a single smooth movement	C
9	Write cursively a sequence of four lowercase cursive bigram 'le', in a single smooth movement	C
10	Copy the word "sheet"	C
11	Copy the word "sheet" above a line	C
12	Copy the word "mum"	C
13	Copy the word "mum" above a line	C
14	Memorize the words "telephone", "dog", and "shop" and rewrite them	M
15	Copy in reverse the word "bottle"	C
16	Copy in reverse the word "house"	C
17	Copy six words (regular, non regular, non words) in the appropriate boxes	C
18	Write the name of the object shown in a picture (a chair)	M
19	Copy the fields of a postal order	C
20	Write a simple sentence under dictation	M
21	Retrace a complex form	G
22	Copy a telephone number	C
23	Write a telephone number under dictation	M
24	Draw a clock, with all hours and put hands at 11:05 (Clock Drawing Test)	G
25	Copy a paragraph	C

gates with tunable hyperparameters for optimization. Moreover, parameters 'C', 'gamma' and 'tol' could be also added in the parameter grid of quantum methods. To enhance the user-friendliness of quantum methods, particularly in the clinical diagnostic process, we aim to automate the implementation of the methods we have developed. By simplifying the process of choice of the ansatz, optimization of parameters and execution on fake or real backends, we wish to make quantum kernel methods more accessible and usable for a wide range of applications, improving the performance of classical ML models for similar problems of early screening, e.g. detection of autism in children [43], or identification of various types of cancer [44,45].

CRediT authorship contribution statement

Giacomo Cappiello: Writing – original draft, Visualization, Software. **Filippo Caruso:** Writing – review & editing, Validation, Supervision, Methodology, Investigation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the European Commission's Horizon Europe Framework Programme under the Research and Innovation Action GA n. 101070546–MUQUABIS, by the European Union's Horizon 2020 research and innovation programme under FET-OPEN GA n. 828946–PATHOS, by the European Defence Agency under the project Q-LAMPS Contract No B PRJ- RT-989, and by the MUR Progetti di Ricerca di Rilevante Interesse Nazionale (PRIN) Bando 2022 - project n. 20227HSE83 – ThAI-MIA funded by the European Union - Next Generation EU. G. C. is a member of INFM (INdAM).

Appendix. Dataset details

The 25 handwriting/drawing tasks can be grouped in 3 categories: memory and dictation, graphic, copy. The tasks are described in Table A.1.

Each task is performed on a different paper sheet and the pile of 25 sheets is put on top of a Wacom's Bamboo tablet equipped with a pen that can both write in ink on paper and allow the tablet to sample x-y coordinate on paper and on air within a maximum distance of 3 cm. The tablet is connected to a PC that processes the raw data to obtain for each task the following features:

1. Total Time (TT): Total time spent to perform the entire task.
2. Air Time (AT): Time spent to perform in-air movements.
3. Paper Time (PT): Time spent to perform on-paper movements.
4. Mean Speed on-paper (MSP): Average speed of on-paper movements.
5. Mean Speed in-air (MSA): Average speed of in-air movements.
6. Mean Acceleration on-paper (MAP): Average acceleration of on-paper movements.
7. Mean Acceleration in-air (MAA): Average acceleration of in-air movements.
8. Mean Jerk on-paper (MJP): Average jerk of on-paper movements. Jerk is the variation of acceleration with respect to time.
9. Mean Jerk in-air (MJA): Average jerk of in-air movements.
10. Pressure Mean (PM): Average of the pressure levels exerted by the pen tip.
11. Pressure Var (PV): Variance of the pressure levels exerted by the pen tip.
12. GMRT on-paper (GMRTP): Generalization of the Mean Relative Tremor (MRT) as defined by Pereira et al. (2015). MRT measures the amount of tremor in drawing spirals and meanders.
13. GMRT in-air (GMRTA): Generalization of the Mean Relative Tremor computed on in air movements.
14. Mean GMRT (GMRT): Average of GMRTP and GMRTA.
15. Pendowns Number (PWN): Counts the total number of pendowns recorded during the execution of the entire task.
16. Max X Extension (XE): Maximum extension recorded along the X axis.

17. Max Y Extension (YE): Maximum extension recorded along the Y axis.
18. Dispersion Index (DI): The Dispersion Index measures how the handwritten trace is "dispersed" on the entire piece of paper. To calculate the index the sheet is ideally divided into TB boxes of 3×3 pixels, then the number CB of boxes containing a fragment of handwriting/drawing is computed. DI is given by the ratio between CB and TB.

Data availability

The DARWIN dataset is available at https://archive.ics.uci.edu/dataset/732/darwin.

References

[1] B.N. Dugger, D.W. Dickson, Pathology of neurodegenerative diseases, Cold Spring Harb. Perspect. Biol. 9 (7) (2017) a028035, http://dx.doi.org/10.1101/cshperspect.a028035.

[2] L.S. Honig, R. Mayeux, Natural history of alzheimer's disease, Aging Clin. Exp. Res. 13 (3) (2001) 171–182, http://dx.doi.org/10.1007/BF03351476.

[3] K. Jellinger, The neuropathological diagnosis of alzheimer disease, Ageing Dement. (1998) 97–118, http://dx.doi.org/10.1007/978-3-7091-6467-9_9.

[4] V. Leroy, W. Gana, A. Aïdoud, J.-A. N'kodo, A.-C. Balageas, P. Blanc, D. Bomia, C. Debacq, B. Fougère, Digital health technologies and alzheimer's disease clinical trials: might decentralized clinical trials increase participation by people with cognitive impairment? Alzheimer' s Res. Ther. 15 (1) (2023) 87, http://dx.doi.org/10.1186/s13195-023-01227-4.

[5] A. Rostamzadeh, A. Kahlert, F. Kalthegener, F. Jessen, Psychotherapeutic interventions in individuals at risk for alzheimer's dementia: a systematic review, Alzheimer' s Res. Ther. 14 (1) (2022) 18, http://dx.doi.org/10.1186/s13195-021-00956-8.

[6] K. Ding, M. Chetty, A. Noori Hoshyar, T. Bhattacharya, B. Klein, Speech based detection of alzheimer's disease: a survey of AI techniques, datasets and challenges, Artif. Intell. Rev. 57 (12) (2024) 1–43, http://dx.doi.org/10.1007/s10462-024-10961-6.

[7] T.L. McIsaac, N.E. Fritz, L. Quinn, L.M. Muratori, Cognitive-motor interference in neurodegenerative disease: a narrative review and implications for clinical management, Front. Psychol. 9 (2018) 2061, http://dx.doi.org/10.3389/fpsyg.2018.02061.

[8] E.H. Dooijes, Analysis of handwriting movements, Acta Psychol. 54 (1–3) (1983) 99–114, http://dx.doi.org/10.1016/0001-6918(83)90026-4.

[9] N.D. Cilia, C.D. Stefano, F. Fontanella, A.S. Di Freca, An experimental protocol to support cognitive impairment diagnosis by using handwriting analysis, Procedia Comput. Sci. 141 (2018) 466–471, http://dx.doi.org/10.1016/j.procs.2018.10.141, URL https://www.sciencedirect.com/science/article/pii/S1877050918317903, The 9th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN-2018) / The 8th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (ICTH-2018) / Affiliated Workshops.

[10] N.D. Cilia, G. De Gregorio, C. De Stefano, F. Fontanella, A. Marcelli, A. Parziale, Diagnosing alzheimer's disease from on-line handwriting: A novel dataset and performance benchmarking, Eng. Appl. Artif. Intell. 111 (2022) 104822, http://dx.doi.org/10.1016/j.engappai.2022.104822, URL https://www.sciencedirect.com/science/article/pii/S0952197622000902.

[11] J.M. Helm, A.M. Swiergosz, H.S. Haeberle, J.M. Karnuta, J.L. Schaffer, V.E. Krebs, A.I. Spitzer, P.N. Ramkumar, Machine learning and artificial intelligence: Definitions, applications, and future directions, Curr. Rev. Musculoskelet. Med. 13 (2020) 69–76, http://dx.doi.org/10.1007/s12178-020-09600-8.

[12] B.C. Love, Comparing supervised and unsupervised category learning, Psychon. Bull. Rev. 9 (4) (2002) 829–835, http://dx.doi.org/10.3758/BF03196342.

[13] V. Viswan, N. Shaffi, M. Mahmud, K. Subramanian, F. Hajamohideen, Explainable artificial intelligence in alzheimer's disease classification: A systematic review, Cogn. Comput. 16 (1) (2024) 1–44, http://dx.doi.org/10.1007/s12559-023-10192-x.

[14] Y. Zhu, K. Yu, Artificial intelligence (AI) for quantum and quantum for AI, Opt. Quantum Electron. 55 (8) (2023) 697, http://dx.doi.org/10.1007/s11082-023-04914-6.

[15] C. Ciliberto, M. Herbster, A.D. Ialongo, M. Pontil, A. Rocchetto, S. Severini, L. Wossnig, Quantum machine learning: a classical perspective, Proc. R. Soc. A: Math. Phys. Eng. Sci. 474 (2209) (2018) 20170551, http://dx.doi.org/10.1098/rspa.2017.0551.

[16] M. Schuld, I. Sinayskiy, F. Petruccione, An introduction to quantum machine learning, Contemp. Phys. 56 (2) (2015) 172–185, http://dx.doi.org/10.1080/00107514.2014.964942.

[17] M. Schuld, F. Petruccione, Machine learning with quantum computers, second ed., in: Quantum Science and Technology, Springer Cham, 2021, p. XIV, 312, http://dx.doi.org/10.1007/978-3-030-83098-4, 978-3-030-83097-7.

[18] S. Jerbi, L.J. Fiderer, H. Poulsen Nautrup, J.M. Kübler, H.J. Briegel, V. Dunjko, Quantum machine learning beyond kernel methods, Nat. Commun. 14 (1) (2023) 517, http://dx.doi.org/10.1038/s41467-023-36159-y.

[19] L. Wei, H. Liu, J. Xu, L. Shi, Z. Shan, B. Zhao, Y. Gao, Quantum machine learning in medical image analysis: A survey, Neurocomputing 525 (2023) 42–53, http://dx.doi.org/10.1016/j.neucom.2023.01.049.

[20] E. Akpinar, Quantum machine learning in the cognitive domain: Alzheimer's disease study, 2023, http://dx.doi.org/10.48550/arXiv.2401.06697, arXiv preprint arXiv:2401.06697.

[21] V. Jakkula, Tutorial on support vector machine (svm), Sch. EECS Wash. State Univ. 37 (2.5) (2006) 3.

[22] T. Hofmann, B. Schölkopf, A.J. Smola, Kernel methods in machine learning, Ann. Statist. 36 (3) (2008) 1171–1220, http://dx.doi.org/10.1214/009053607000000677.

[23] T. Kavzoglu, I. Colkesen, A kernel functions analysis for support vector machines for land cover classification, Int. J. Appl. Earth Obs. Geoinf. 11 (5) (2009) 352–359, http://dx.doi.org/10.1016/j.jag.2009.06.002.

[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in python, J. Mach. Learn. Res. 12 (2011) 2825–2830, http://dx.doi.org/10.48550/arXiv.1201.0490.

[25] S. Goel, A. Klivans, Eigenvalue decay implies polynomial-time learnability for neural networks, Adv. Neural Inf. Process. Syst. 30 (2017) http://dx.doi.org/10.48550/arXiv.1708.03708.

[26] A. Kataria, M. Singh, A review of data classification using k-nearest neighbour algorithm, Int. J. Emerg. Technol. Adv. Eng. 3 (6) (2013) 354–360.

[27] S.B. Kotsiantis, Decision trees: a recent overview, Artif. Intell. Rev. 39 (2013) 261–283, http://dx.doi.org/10.1007/s10462-011-9272-4.

[28] M. Schuld, Supervised quantum machine learning models are kernel methods, 2021, http://dx.doi.org/10.48550/arXiv.2101.11020, arXiv preprint arXiv:2101.11020.

[29] M.A. Nielsen, I.L. Chuang, Quantum Computation and Quantum Information: 10th Anniversary Edition, Cambridge University Press, 2010, http://dx.doi.org/10.1017/CBO9780511976667, 9781107002173.

[30] Y.-C. Liang, Y.-H. Yeh, P.E. Mendonça, R.Y. Teh, M.D. Reid, P.D. Drummond, Quantum fidelity measures for mixed states, Rep. Progr. Phys. 82 (7) (2019) 076001, http://dx.doi.org/10.1088/1361-6633/ab1ca4.

[31] Z. Holmes, K. Sharma, M. Cerezo, P.J. Coles, Connecting ansatz expressibility to gradient magnitudes and barren plateaus, PRX Quantum 3 (1) (2022) 010313, http://dx.doi.org/10.1103/PRXQuantum.3.010313.

[32] J. Kübler, S. Buchholz, B. Schölkopf, The inductive bias of quantum kernels, Adv. Neural Inf. Process. Syst. 34 (2021) 12661–12673, http://dx.doi.org/10.48550/arXiv.2106.03747.

[33] A. Canatar, E. Peters, C. Pehlevan, S.M. Wild, R. Shaydulin, Bandwidth enables generalization in quantum kernel models, 2022, http://dx.doi.org/10.48550/arXiv.2206.06686, arXiv preprint arXiv:2206.06686.

[34] T. Howley, M.G. Madden, M.-L. O'Connell, A.G. Ryder, The effect of principal component analysis on machine learning accuracy with high dimensional spectral data, in: International Conference on Innovative Techniques and Applications of Artificial Intelligence, Springer, 2005, pp. 209–222, http://dx.doi.org/10.1007/1-84628-224-1_16.

[35] M.P. Fisher, V. Khemani, A. Nahum, S. Vijay, Random quantum circuits, Annu. Rev. Condens. Matter Phys. 14 (1) (2023) 335–379.

[36] M. Islam, G. Chen, S. Jin, An overview of neural network, Am. J. Neural Netw. Appl. 5 (1) (2019) 7–11.

[37] D. Arthur, P. Date, Hybrid quantum-classical neural networks, in: 2022 IEEE International Conference on Quantum Computing and Engineering, QCE, IEEE, 2022, pp. 49–55.

[38] Q. contributors, Qiskit: An open-source framework for quantum computing, 2023, http://dx.doi.org/10.5281/zenodo.2573505.

[39] J. Kittler, M. Hatef, R.P. Duin, J. Matas, On combining classifiers, IEEE Trans. Pattern Anal. Mach. Intell. 20 (3) (1998) 226–239, http://dx.doi.org/10.1109/34.667881.

[40] A.A. Houck, J. Schreier, B. Johnson, J. Chow, J. Koch, J. Gambetta, D. Schuster, L. Frunzio, M. Devoret, S. Girvin, et al., Controlling the spontaneous emission of a superconducting transmon qubit, Phys. Rev. Lett. 101 (8) (2008) 080502, http://dx.doi.org/10.1103/PhysRevLett.101.080502.

[41] J.E. Johnson, V. Laparra, A. Pérez-Suay, M.D. Mahecha, G. Camps-Valls, Kernel methods and their derivatives: Concept and perspectives for the earth system sciences, Plos One 15 (10) (2020) e0235885, http://dx.doi.org/10.1371/journal.pone.0235885.

[42] K.-L. Du, M.N.S. Swamy, Kernel methods, in: Neural networks and statistical learning, in: Neural Networks and Statistical Learning, Springer London, London, 2019, pp. 569–592, http://dx.doi.org/10.1007/978-1-4471-7452-3_20.

[43] H. Abbas, F. Garberson, E. Glover, D.P. Wall, Machine learning approach for early detection of autism by combining questionnaire and home video screening, J. Am. Med. Informatics Assoc. 25 (8) (2018) 1000–1007, http://dx.doi.org/10.1093/jamia/ocy039.

[44] M. Tahmooresi, A. Afshar, B.B. Rad, K. Nowshath, M. Bamiah, Early detection of breast cancer using machine learning techniques, J. Telecommun. Electron. Comput. Engineering (JTEC) 10 (3–2) (2018) 21–27, URL https://jtec.utem.edu.my/jtec/article/view/4706.

[45] M.K. Gould, B.Z. Huang, M.C. Tammemagi, Y. Kinar, R. Shiff, Machine learning for early lung cancer identification using routine clinical and laboratory data, Am. J. Respir. Crit. Care Med. 204 (4) (2021) 445–453, http://dx.doi.org/10.1164/rccm.202007-2791OC.

Giacomo Cappiello received his M.Sc. degree in Mathematics from the University of Bari, Italy, in 2022. He is currently a Ph.D. student at the University of Florence, Italy. His research interests include quantum machine learning and applications in healthcare.

Filippo Caruso received M.Sc. degree in Physics from Catania University, Italy, in 2005 and Ph.D. degree in Physics from Scuola Normale Superiore, Pisa, Italy, in 2008. He was a postdoctoral researcher at Imperial College of London (UK) and a Marie-Curie Fellow at Ulm University (Germany) in 2008–2011. He became assistant professor in Florence University (Italy) in 2012 and later was awarded with a tenure-track position in 2015. In 2018 he started a new position as associate (permanent) professor in theoretical condensed matter physics, and was finally awarded as full professor in 2024 at the Physics Department of Florence University. In the last 20 years, he got a consolidated, internationally recognized, experience in quantum information theory and quantum technologies, going from quantum communication to quantum sensing, from quantum computing to quantum AI, more from the algorithmic side but also towards practical technological applications to solve very hard optimization and AI problems. He is also leading a (more than 10 years old) Quantum Information & AI research group, has published around 100 papers in international peer-review journals (including very high-impact factor journals as Nature Communications, Nature Materials, Review of Modern Physics, Physical Review Letters, etc.) and coordinated several, national and EU, very prestigious research projects and interdisciplinary consortia.