Exploring the State Vector Classification Algorithm and Its Quantum Equivalent

Ethan Hunt*, Hieu Nguyen†, and Tu N. Nguyen*
*Department of Computer Science, Kennesaw State University, Marietta, GA 30060, USA.
†Faculty of Information Technology, University of Science, VNU-HCM, Ho Chi Minh City, Vietnam.

***Abstract*—With the escalating demand for high-performance artificial intelligence, the domain of quantum machine learning emerges as a promising avenue for enhancing machine learning frameworks. This paper introduces two innovative algorithms: the *State Vector Classification Algorithm (SVCA),* inspired by quantum principles, and its hybrid counterpart, the *Quantum State Vector Classification Algorithm (QSVCA).* Both algorithms operate by mapping input data into a *classification space* and projecting it onto the basis vector with the largest component. This projection operation serves as the central for classifying elements within a dataset and introducing non-linearity without relying on an activation function. Specifically, the SVCA's classification process bears resemblance to measurements in quantum computing, facilitating a seamless transition of many SVCA processes into a quantum framework. Furthermore, extensive evaluation results show that the SVCA performs comparably with other classical classifiers and notably enhances accuracy on the tested elliptical dataset. Additionally, the QSVCA serves as a quantum counterpart to the classical SVCA, consistently demonstrating strong performance on simple relationships within datasets and even outperforming the Linear Support Vector Machine SVM and the Random Forest Classifier on a synthetic dataset. Both algorithms maintain consistent performance across multiple datasets, highlighting their reliability.**

***Index Terms*—Quantum machine learning and classifier.**

I. Introduction

Quantum machine learning. The emergence of advanced AI technologies, such as ChatGPT, has significantly influenced the applicability of artificial intelligence, driving a need for faster and more dependable machine learning methodologies. Quantum machine learning (QML) emerges as a promising solution to address these needs. By leveraging principles from quantum mechanics, QML aims to reduce computation times for complex tasks, making previously unsolvable problems manageable [1]. QML merges quantum computing with traditional machine learning, enhancing the processing and analysis of complex, high-dimensional datasets. This integration leverages quantum parallelism and superposition, achieving unprecedented data processing speeds where classical algorithms struggle.

Classification algorithm. In practical terms, QML encompasses the utilization of quantum algorithms for machine learning tasks [2], exploring both classical and quantum data. This opens up new avenues for research, including quantum adaptations of traditional models such as binary classifiers [3] and neural networks [4], as well as innovative quantum methods for state or transformation identification. At the core of QML's application spectrum are quantum classification algorithms (QCA), which enhance the efficiency of sorting data into defined categories by exploiting quantum properties like superposition and entanglement. These algorithms, such as the quantum classifier for binary classification and the quantum support vector machine [5], hope to surpass their classical counterparts in speed and efficiency. However, conflicting reports on both the theoretical rigor and practical results of QML algorithms are observed in papers across the field. While QML methods like the QSVM have been shown to do better on some datasets [6] often crediting the advantage to the quantum kernels, it has been shown that while powerful and have the potential practical QML it has been shown that a more nuanced approach is needed when evaluating QML methods [7].

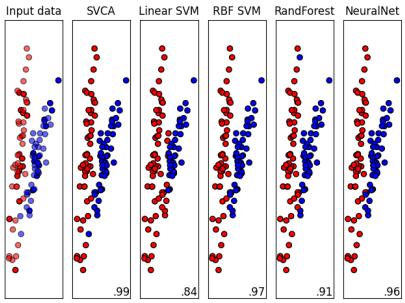


Fig. 1: The accuracy of the SVCA (99%), Linear SVM (84%), RBF SVM (97%), Random Forest Classifier (91%), and Neural Network (96%) on the given dataset.

Motivation. Classification algorithms are fundamental in ML, assigning labels to data points to discern underlying structures. Traditional algorithms use various similarity metrics to identify classes. However, within inner product spaces, the notion of similarity reveals complexities beyond mere numerical measures. This motivates us to delve into the intricate relationship between similarity and class within the realm of QCA. Through the lens of inner product spaces, we probe how both traditional and quantum computing approaches, such as the state vector classification algorithm (SVCA) and its quantum counterpart, the quantum SVCA (QSVCA), interpret and harness these concepts. We conduct empirical studies to assess the efficacy of SVCA compared to other conventional classifiers (Fig. 1).

We thank the anonymous reviewers for their suggestions and feedback. This research was in part supported by US NSF under Grants: AMPS-2229073 and CNS-2103405. Corresponding author: Tu N. Nguyen.

Our contributions. To tackle this issue, we first explore the notion of similarity within the framework of inner product spaces between two vector objects, laying the groundwork for a deeper analysis of class dynamics within a dataset. The conventional concept of similarity, defined as the inner product between two vectors, serves as an initial basis for comparing classes. However, this direct comparison falls short when scrutinizing the nuanced relationships that emerge within and across classes. The SVCA conceptualizes classes as orthogonal entities within a specially defined vector space – known as the classification space – ensuring clear and distinct class separations. This orthogonality guarantees that classes remain distinct and non-overlapping, a departure from traditional similarity-based classification methods. This perspective is further extended into the quantum realm by the QSVCA, leveraging quantum mechanics to project data vectors onto a high-dimensional classification space with enhanced efficiency. The main contributions of this paper are as follows:

- **Development of novel algorithms:** We introduce 1) the state vector classification algorithm (SVCA), which maps input data into a classification space and projects it onto the basis vector with the largest component, introducing non-linearity without the need for an activation function. We also extend the SVCA into the quantum realm, developing the quantum SVCA (QSVCA), leveraging quantum mechanics to efficiently project data vectors onto a high-dimensional classification space.
- **Theoretical analysis:** In analysis, we illustrate that both SVCA and QSVCA are capable of representing a linearly separable binary relationship in a dataset. This, combined with its intrinsic ability to handle nonlinear datasets, positions it as a versatile classifier for general purposes.
- **Assessment:** The evaluation shows that SVCA performed comparably with, and sometimes outperformed, traditional classifiers (e.g., SVM, random forest, neural networks), especially with non-linearly separable datasets like elliptical ones. QSVCA consistently performed well across multiple datasets, showcasing its reliability among QML algorithms. Across various synthetic and real datasets, both SVCA and QSVCA demonstrated effectiveness and robustness in classifying complex data structures. Specific instances highlight QSVCA's close approximation to classical classifiers, indicating promising practical applications.

The rest of this paper is structured as follows: In §II, we present an overview of QML algorithms and explore potential approaches to quantum-inspired classification. In §III, we delve into the theoretical foundations of SVCA, highlighting its potential to redefine classification in the quantum computing era. In §IV, we introduce and analyze the QSVCA algorithms' capability to examine the relationship between similarity and class, contributing to the broader discourse on integrating quantum computing with ML and paving the way for revolutionary advancements in data analysis and classification. Finally, we conclude with key remarks and propose potential future research directions in §VI.

II. Preliminary

With the surge in research interest, both in academia and industry, driven by the success of major artificial intelligence platforms, the demand for faster and more accurate machine learning models is escalating. To meet these demands, exploration into quantum-inspired machine learning algorithms and quantum/hybrid machine learning algorithms has intensified. While the latter can be easily validated both theoretically and empirically, the former often relies solely on theoretical evidence or small toy datasets for testing.

A. A Novel Approach to Quantum-Inspired Classification

The SVCA offers a new perspective on the fundamental query: "How do we define and measure the similarity between classes in a dataset?" Traditionally, similarity assessment relies on the inner product between vectors. By treating each vector as a representation of a class in our data, the inner product helps gauge the closeness between two classes. In this discussion, we delve into the nuanced question, "What is the relationship between similarity and class?" To explore this, we initially examine similarity within the framework of inner product spaces between two arbitrary vector objects o_1 and o_2: $\langle o_1|o_2\rangle = \vec{o_1} \cdot \vec{o_2}$. Pivoting from abstract objects to concrete representations, let us consider these vectors as distinct classes within a dataset, thus defining a set of classes $C \equiv \{c_i | i \in [0, |C|]\}$.

It is imperative to recognize that the relationship between a class and its elements diverges significantly. For example, if we are to classify a dataset into classes A and B, elements within one class might share similarities with elements from another without implying inherent class similarity. For instance, the vector $(2, 1)$ in class B – a linear amalgamation of vectors $(0, 1)$ from class A and $(1, 0)$ from class B – highlights this. Despite the zero similarity between vectors $(1, 0)$ and $(0, 1)$, vector $(2, 1)$ intertwines both. Consequently, elements of Class B display divergent similarity scores to an element of class A, obscuring a clear relationship without additional data.

B. The Classification Space $\mathbb{R}^C$

Departing from conventional classification algorithms, the SVCA introduces a groundbreaking concept where classes are viewed as orthogonal entities within a specialized vector space called the Classification Space. This conceptual departure necessitates that any two classes are orthogonal, expressed as $\langle c_1|c_2\rangle = \vec{c_1} \cdot \vec{c_2} = 0$. Despite potential similarities among classes, this orthogonality principle forms the basis for representing data within this unique vector space. For a dataset D encompassing c distinct classes, we establish the presence of a linear mapping $T_{\vec{d}} \in \mathbb{R}^C : \vec{d} \in D$, enabling the transformation of any data vector onto this classification dimension. This provides a geometric interpretation to the resulting vector in the Classification Space, where the relative magnitudes of each vector convey the score associated with the corresponding class.

C. Converting Measurements into Classes

Transitioning our discussion into the realm of Quantum Mechanics and the inherent computations within our Classification Space, we assign each class a natural basis vector. In a classification space $\mathbb{R}^2$, for instance, the first class may be denoted by the basis $(1, 0)$ and the second by $(0, 1)$. In figure 2 a visual example of transformation to the classification space $\mathbb{R}^2$ is given. Our objective is to map our data vector $\vec{x}n \times 1$ into $\mathbb{R}^C$ using the matrix $T_{m\times n}$, resulting in the prediction vector $\vec{p}_{m\times 1} = T_{m\times n}\vec{x}_{n\times 1}$. With $\vec{p} = |\vec{p}\rangle$ at our disposal, and acknowledging $\langle v_1|v_2\rangle = \vec{v}_1 \cdot \vec{v}2$, we compute $\langle e_i|\vec{p}\rangle = \sum_j \alpha_j \langle e_j|\vec{p}\rangle$ for each natural basis vector e_i. Due to the orthogonality of the basis vectors, $\langle e_j|\vec{p}\rangle = 0$ for $i \neq k$.

Unlike the stochastic nature of quantum measurements, we deterministically select the basis with the highest value for $\langle e_i|\vec{p}\rangle$. This selection collapses our measurement to that basis vector, effectively classifying $\vec{x}_{n\times 1}$ as the corresponding class. Since the process of choosing the class from the prediction vector is akin to the non-linear measurement in quantum mechanics, the SVCA does not require the use of activation functions like SoftMax or ReLu to express non-linear relationships (see figure 3 for an example). Thus, SVCA's ambition is to discover the optimal transformation $T_{m\times n}$ for data classification, which can be envisioned as transforming a vector $\vec{v} \in \mathbb{R}^n \rightarrow \vec{p} \in \mathbb{R}^C$.

III. The Classical SVCA

The State Vector Classification Algorithm (SVCA) stands out as a versatile classification tool capable of effectively handling complex datasets while requiring only a fraction of the training data typically needed. This remarkable capability is achieved through a process where the SVCA projects the input vector onto a basis vector corresponding to a specific classification dimension. Conceptually, SVCA bears some resemblance to a single-layer neural network, where the output dimension corresponds to the number of class labels. However, there are several key distinctions that set SVCA apart and contribute to its enhanced ability to generalize data compared to a typical single-layer neural network. Moreover, these distinctions facilitate the creation of an equivalent quantum model with relative ease.

A. SVCA Transformation and Dimensions

Given a matrix W and an input vector $\vec{x}$, with the dimensions defined as: $\dim(W) = m \times n, \dim(\vec{x}) = n \times 1, \dim(\vec{p}) = m \times 1, |\mathcal{C}| = m$. Then, the prediction vector $\vec{p}$ is calculated as follows:

$$\vec{p} = W\vec{x} = \sum_{i=1}^{n}(x_i\vec{w}_i) \quad (1)$$

This can also be expressed using the dot product and orthonormal basis vectors e_j as:

$$\vec{p} = \sum_{j=1}^{m}(\vec{w}_j \cdot \vec{x})\hat{e}_j = \sum_{j=1}^{m}\langle\vec{w}_j|\vec{x}\rangle|e_j\rangle \quad (2)$$

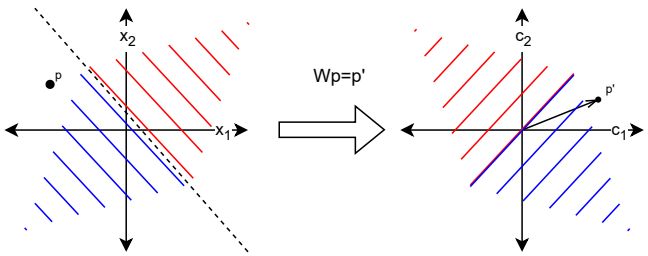


Fig. 2: An example of a data point p being mapped to the classification space by W.

B. SVCA Classification Process

The classification process is described by identifying the basis vector e_j that corresponds to the maximum coefficient in the prediction vector $\vec{p}$:

$$\arg\max_{\vec{p}}(\hat{e}_j) = \langle e_j|\vec{p}\rangle \quad (3)$$

This step classifies the input vector $\vec{x}$ into the class associated with the basis vector e_j that has the largest projection coefficient in $\vec{p}$. We put forth the following algorithm for defining the classification process in the SVCA.

Algorithm 1 Prediction and Classification Process

Require: Matrix W with dimensions $m \times n$, input vector $\vec{x}$ with dimensions $n \times 1$

Ensure: Classification of $\vec{x}$ into class associated with basis vector $\hat{e}_j$

1: $\vec{p} \leftarrow W\vec{x} = \sum_{i=1}^{n} x_i\vec{w}_i$
2: $\hat{y} \leftarrow \arg\max_{\vec{p}}(\hat{e}_j) = \langle e_j|\vec{p}\rangle$
3: **return** $\hat{y}$

C. SVCA Training process

The objective of training the SVCA is to determine the optimal transformation for our dataset. To achieve this, we employ the following approach for a given training sample $\vec{x}$:

1) First, calculate the prediction vector $\vec{p} = W\vec{x}$.
2) Next, compute $\hat{y} = \arg\max_{\vec{p}}(\hat{e}_j)$.
3) Compare the expected versus observed class $|\hat{y} - y|$.
4) Lastly, update the transformation matrix for the next epoch iteration k, with the transformation $\hat{W}_{k+1} = \nabla L_{\vec{x}}\eta(k) + \hat{W}$, where $L_{\vec{x}}$ is the loss for input vector $\vec{x}$ and $\eta(k)$ is the learning rate at iteration k.

This process is repeated for each sample in the training set per epoch iteration k. The learning rate schedule is defined by the function $\eta(k)$. To compute the gradient $\nabla L_{\vec{x}}$, we operate on its row vectors as $\vec{p} = W\vec{x}$. Thus, the gradient is computed as $\nabla L_{\vec{x}} = \sum \frac{\partial L_i}{\partial \vec{w}_i}$ where $\frac{\partial L_i}{\partial \vec{w}_i} = \vec{x}$ for simplification. To update the transformations per row vector, we use:

$$\hat{w}_i' = \begin{cases} -\frac{\partial \vec{L}_i}{\partial \vec{w}_i}\eta(k) + \hat{w}_i & : y \neq \hat{e}_j \\ \frac{\partial \vec{L}_i}{\partial \vec{w}_i}\eta(k) + \hat{w}_i & : y = \hat{e}_j \end{cases} \quad (4)$$

Algorithm 2 SVCA Training Algorithm

$\vec{p} \leftarrow W\vec{x}$ and $\hat{y} \leftarrow \arg\max_{\vec{p}}(\hat{e}_j)$
if $P \neq y$ **then**
  for $i = 0$ **to** $\text{len}(W) - 1$ **do**
    if $i \neq y$ **then**
      $W_i \leftarrow -\eta(k) \cdot \vec{x} + W_i$
    else
      $W_i \leftarrow \eta(k) \cdot \vec{x} + W_i$
    end if
  end for
end if
return W

This represents the update rule for the transformation matrix's row vectors, adjusting for the learning rate and the difference between expected and observed classes. Putting all this together, we provide the following implementation of the learning algorithm for the SVCA.

D. Linear Data and the SVCA

The SVCA, akin to the classical perceptron and SVM, can express linear relationships. However, unlike their default versions, it can effectively capture non-linear relationships. In this section, we'll explore how a linear relationship is expressed in an SVCA transformation T, realized through a weight matrix.

Theorem 1. *Consider the dataset $X = \mathbb{R}^n$ consisting of vectors $\vec{x} \in \mathbb{R}^n$, partitioned into two sets A and B based on some hyperplane in $\mathbb{R}^n$. Then there exists a $2 \times n$ real value matrix W where the values of entries in the vector $W\vec{x} = \vec{p}$ in $\mathbb{R}^2$ correspond to the "score" a given vector $\vec{x}$ has with a given class. Such that for every vector $\vec{v}$ in some arbitrary class C, the value of the entry p_C corresponding to class C in the prediction vector $W\vec{v} = \vec{p}$ is always greater than the value of the entry $p_{\neg C}$ corresponding to class $\neg C$.*

Proof. Let the data set $X = \mathbb{R}^n$ whose elements are vectors $\vec{x} \in \mathbb{R}^n$ be partitioned into two sets A and B based on the hyperplane H in $\mathbb{R}^n$. This hyperplane H can be represented by the equation:

$$\vec{m}^\intercal \vec{x} + b = 0$$

where:

- $\vec{m} \in \mathbb{R}^n$ is a weight vector normal to the hyperplane,
- $\vec{x} \in \mathbb{R}^n$ is an input feature vector,
- $b \in \mathbb{R}$ is a bias term,
- $\vec{v}^\intercal$ denotes the transpose operation, making $\vec{m}^\intercal\vec{x}$ a dot product between $\vec{m}$ and $\vec{x}$.

For a given vector $\vec{x}$, its classification is determined as follows:

- If $\vec{m}^\intercal\vec{x} + b \geq 0$, then $\vec{x}$ is classified into class A,
- If $\vec{m}^\intercal\vec{x} + b < 0$, then $\vec{x}$ is classified into class B.

Let the vector $\vec{z}$ be orthogonal to any vector $\vec{x}$ which is an element of H. It follows then that this can be expressed as:

$$\vec{z} \cdot \vec{x} = 0 : \vec{z} \cdot \vec{x} = \vec{m} \cdot \vec{x} + b$$

Let the vector $\vec{v} = (1, -1)$. Let the linear transformation U expressed as a $2 \times n$ matrix transform any vector $\vec{x} \in \mathbb{R}^n$ into a vector $\vec{p} \in \mathbb{R}^2$. Consider that:

$$\arg\max_{\vec{p}}(\hat{e}_j) \equiv \begin{cases} \vec{e_1} & : \vec{v} \cdot \vec{p} \geq 0 \\ \vec{e_2} & : \vec{v} \cdot \vec{p} \leq 0 \end{cases}$$

which implies that identifying the basis vector that corresponds to the maximum coefficient can be found using the linear functional $\vec{v} \cdot \vec{p}$ and a threshold of zero. Therefore it is possible to preserve the relationship between the two classes of vectors A and B be defined by the hyperplane H when applying the linear transformation U by solving for the entries of U that satisfy the following relationship:

$$\vec{v} \cdot U\vec{x} = \vec{z} \cdot \vec{x}$$

This can be rewritten by applying the property that $\vec{a} \cdot \vec{b} = \vec{a}^\intercal\vec{b}$ to get the expression:

$$(\vec{v}^\intercal U)^\intercal \cdot \vec{x} = \vec{z} \cdot \vec{x}$$

Since both sides of the equation are expressed as the dot product between $\vec{x}$ and some other vector in $\mathbb{R}^n$ the two vectors must be equal. This can then be expressed by the following equation:

$$(\vec{v}^\intercal U)^\intercal = \vec{z}$$

Therefore solving for the entries of the matrix U that satisfy the equation $(\vec{v}^\intercal U)^\intercal \cdot \vec{x} = 0$ will yield the linear transformation $W = U$. The solution W must exist since $\vec{z}$ exists and satisfies the equation $(\vec{v}^\intercal U)^\intercal = \vec{z}$.

Consider that for any vector $\vec{u}$ the value of $\vec{z} \cdot \vec{u} = f(\vec{u})$ is also given by the equation $f(\vec{u}) = \vec{v} \cdot W\vec{u}$. For every vector $\vec{x} \notin H$ in the dataset, consider a corresponding point $\vec{y}$, such that $\vec{y}$ is the reflection of $\vec{x}$ across the hyperplane, ensuring $\vec{x}$ is classified differently than $\vec{y}$. Let the vector $\vec{h}$ lying on the discriminating hyperplane between $\vec{x}$ and $\vec{y}$, governed by the function f that delineates the separation criterion: $f(\vec{y}) < f(\vec{h}) < f(\vec{x})$. It follows then that since $\vec{z} \cdot \vec{u} = f(\vec{u}) = \vec{v} \cdot W\vec{u}$ the W preserves the relationship of the hyperplane. Thus the values of entries in the vector $W\vec{x} = \vec{p}$ in $\mathbb{R}^2$ correspond to the "score" a given vector $\vec{x}$ has with a given class. Such that for every vector $\vec{u}$ in some arbitrary class C, the value of the entry p_C corresponding to class C in the prediction vector $W\vec{u} = \vec{p}$ is always greater than the value of the entry $p_{\neg C}$ corresponding to class $\neg C$. □

Based on Theorem 1, we observe that the SVCA can represent any binary relationship that is linearly separable. Unlike the perceptron or SVM, the SVCA can handle classification tasks with more than two classes without needing additional modifications. This is because the SVCA encodes the linear relationship(s) of the dataset into a new space, where separability depends on the transformed vector coordinates. This differs from the classification approach of the perceptron, which relies on a threshold of some linear functional.

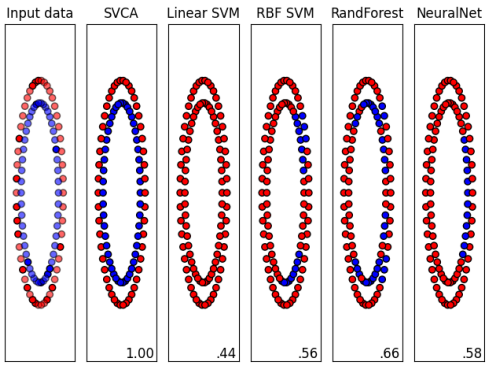
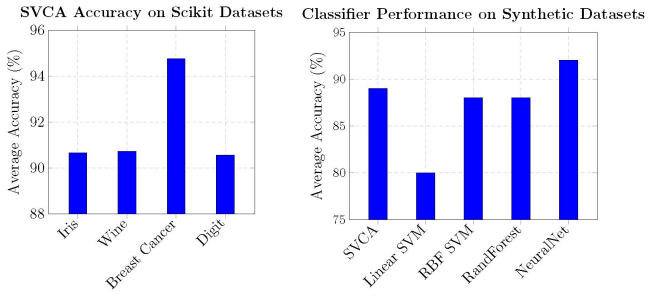


Fig. 3: Demonstrates the accuracy of various classifiers on a specific elliptical dataset: SVCA (100%), Linear SVM (44%), RBF SVM (56%), Random Forest Classifier (66%), and Neural Network (58%).

E. Results for the SVCA

While the SVCA can theoretically express linear relationships, in practice datasets often have noise in them making them more difficult to classify as well as not being restricted to only linear relationships. As such it is important to test the SVCA's ability to generalize various types of relationships across multiple types of data in addition to comparing its performance against other well studies techniques.

- **On the Elliptical Datasets.** One example of a nonlinear dataset involves classes described by inner and outer ellipses. This relationship type can be challenging to classify without specialized kernels to model it effectively. Utilizing the scikit-learn's method [8], we generated such datasets and then we extended the feature space by appending a new feature which is a polynomial of the original features. Finally we compared the performance of multiple classifiers against SVCA, as shown in Figure 3.
- **On the Scikit-Learn Toy Datasets.** Evaluating the validity of a machine learning paradigm often involves testing its performance on well-studied datasets. To accomplish this, we utilize several toy datasets from the scikit-learn library, encompassing a wide variety of classification scenarios. In Figure 4(a), we present the results of SVCA on these datasets. All datasets undergo the same preprocessing methods to highlight the consistency of SVCA performance without relying on preprocessing measures to enhance accuracy.
- **Comparisons across Synthetic Binary Datasets.** To assess the performance of SVCA against well-established machine learning methods, we compared the average performance of the Scikit-Learn implementations of SVCA, Linear SVM, RBF SVM, Random Forest Classifier, and a Neural Network across 1000 synthetic datasets. These datasets were generated using the *make_classification* function also from Scikit-Learn. The results are depicted in Figure 4(b), demonstrating that SVCA performed comparably to the other methods. Across a wide range of random binary classification tasks, SVCA consistently delivered scores on par with the other listed algorithms.



(a) (b)

Fig. 4: a) Presents the SVCA's average accuracy across multiple Scikit-Learn datasets, including Iris (90.66%), Wine (90.72%), Breast Cancer (94.76%), and Digit (90.56%) [8], based on 100 instances of a training-test split of 30:70 and b) presents the average accuracy of various classifiers, including SVCA (89%), Linear SVM (80%), RBF SVM (88%), Random Forest Classifier (88%), and Neural Network (92%), across 1000 synthetic binary datasets (training-test split of 75:25).

IV. Towards the Quantum SVCA

Given that the classical SVCA eschews the use of an activation function for classification tasks, opting instead for selecting the maximum value of an entry in a classification space – a concept reminiscent of measurements in quantum mechanics – it stands out as a promising candidate for implementation on quantum machines. This characteristic, coupled with the inherent representation of quantum states via matrix-vector multiplication, facilitates an intuitive integration of SVCA into quantum computing frameworks. Notably, a quantum SVCA offers the potential to combine novel quantum machine learning techniques with a robust machine learning paradigm, leveraging the inherent advantages of quantum computing.

A. The Quantum Data Encoding

The initial step in implementing SVCA on a quantum computer involves selecting a suitable Quantum Data Encoding method to represent the input vector and weight matrix. We observed that many encoding methods, designed for algorithms testable on near-term quantum computers, encounter difficulties in properly expressing vectors with entries from a real-number field. This observation is supported by Proposition 4.2 in [9], where the proposed encoding fails to represent the additive and multiplicative identity elements of a field. Such issues are non-trivial and can potentially impede the convergence of the QML algorithm to an optimal representation or lead to a representation that inadequately captures the dataset's underlying relationships.

To illustrate the impact of encoding schemes on representation, we introduce a simplified encoding scheme. For a given feature vector $\vec{x} \in \mathbb{R}^2$ within a dataset, where the first feature is denoted as a and the second feature as b, we define the toy encoding method as follows: the amplitudes of a two-dimensional quantum state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ represent the values of the features of $\vec{x}$, where $\alpha = \frac{a}{a+b}$ and $\beta = \frac{b}{a+b}$. This method vividly demonstrates how encoding influences dataset relationship representation.

Consider a dataset X comprising two-dimensional vectors, where a feature may lack correlation with a given class. In classical approaches, such features are typically assigned weights close to zero post-training to prevent undue influence on classification outcomes. However, the toy encoding scheme struggles to express null-valued entries, potentially causing classification errors. This limitation arises from the periodic nature of quantum state amplitudes during training, which can lead to "over-correction" of weight values. For example, when evolving amplitudes in $|\psi\rangle$ to represent weighted features, encountering an irrelevant feature necessitates nullification of its effect, ideally setting its value close to zero. However, achieving this nullification is impossible in a two-dimensional Hilbert space. When one state's amplitude approaches zero, the other necessarily approaches one, causing all informative feature values to become identical and impairing discrimination between different values of the same feature.

Consider a scenario during training where it is noted that the value α for feature one is too large, prompting its reduction. However, decreasing α results in an increase in β, creating a situation where mitigating the influence of one feature inadvertently boosts another's weight. While the outlined toy encoding scheme is inefficient, it effectively highlights the consequences of being unable to express all real number field values. To address these potential issues in encoding vectors with real-valued entries, this paper employs a method wherein one qubit is allocated for each entry in the input vector and weight matrix. This approach ensures the independence of each entry, effectively mitigating the aforementioned challenges. Specifically, the values of each entry are encoded as amplitudes of the $|1\rangle$ state in the corresponding qubit, employing a normalization function f.

$$|x_i\rangle = (1 - f(x_i))|0\rangle + f(x_i)|1\rangle \quad (4)$$

Given that the amplitudes of each qubit are independent of one another, each qubit's amplitude for the $|1\rangle$ state falls within the range $[0, 1]$. Conceptually, this process normalizes each entry onto the interval $[0, 1]$. To ensure proper representation on the quantum circuit, normalization is necessary. In this paper, we adopt the following normalization approach.

$$x' = [\frac{x_1}{\|\mathbf{x}\|}, \ldots, \frac{x_n}{\|\mathbf{x}\|}] : \|\mathbf{x}\| = \sum_{i=1}^{n} |x_i| \quad (5)$$

B. The Quantum Transformation Unitary U_T

The subsequent crucial step in constructing a quantum SVCA involves creating the Unitary U_T, which operates on the input state of the selected feature map to implement the transformation T on the quantum circuit. One approach, using the encoding scheme outlined in subsection IV-A, is to employ multiple swap tests [10] without measuring the control between a copy of the input state $|x\rangle$ and a row vector of the weight matrix $|w_i\rangle$. This is expressed as the following state:

$$\frac{1}{2}\left(|0\rangle\left(|x\rangle\,|w_i\rangle + |w_i\rangle\,|x\rangle\right) + |1\rangle\left(|x\rangle\,|w_i\rangle - |w_i\rangle\,|x\rangle\right)\right) \quad (6)$$

Using the following representation of the classical SVCA we can see that $W\vec{x}$ can be represented as the linear combination of the inner products between the input vector $\vec{x}$ and a corresponding row vector $\vec{w_i}$ as seen in Equation 2 in III-A:

$$\vec{p} = \sum_{j=1}^{m} (\vec{w_j} \cdot \vec{x})\hat{e_j} = \sum_{j=1}^{m} \langle \vec{w_j}|\vec{x}\rangle |e_j\rangle \quad (2)$$

The value of $\langle \vec{w_j}|\vec{x}\rangle$ is applied to the control qubit of the swap operation which corresponds to a basis vector $|e_j\rangle \in \mathbb{R}^C$. Since the classical swap test gives the probability of the jth control qubit being in the state $|0\rangle$ is given by

$$P(|e_j\rangle = |0\rangle) = \frac{1}{2} + \frac{1}{2}|\langle \vec{w_j}|\vec{x}\rangle|^2 \quad (8)$$

The probability of the control qubit being in a particular state describes some form of inner product between the input and row vectors. Consequently, the state represented by the tensor of the control qubits corresponds to a prediction vector $|p\rangle \in \mathcal{H}^C$, which serves as the quantum equivalent of $\vec{p} \in \mathbb{R}^C$.

C. The Classifier Unitary U_s

The computation of $\hat{y} = \arg\max_{|p\rangle}(|e_j\rangle)$ is feasible by measuring instances obtained from sampling the quantum state of the corresponding swap test, with an additive error of $O(\frac{1}{\varepsilon^2})$ samples [11]. Subsequently, comparing the inner product estimates from each swap test facilitates prediction. However, as these estimates are derived from the squares of inner-product values, the distinction between negative and positive values is eliminated. This issue can be rectified by implementing relative comparison via entanglement between qubits associated with the classification space's basis vectors.

Consider two independent swap tests A and B, where the states represent the controlled qubits' state for each test, denoted as $|\phi\rangle_A$ and $|\varphi\rangle_B$ respectively. Let P_A and P_B represent the probabilities of the control qubit in swap tests A and B being in the state $|1\rangle$, corresponding to basis vectors $|e_A\rangle$ and $|e_B\rangle$ respectively. We can define a unitary operator that, when applied to the state $|\phi\rangle_A|\varphi\rangle_B$, divides the probabilities of measuring the two qubits in either $|11\rangle_{AB}$ or $|00\rangle_{AB}$ by half and adds them to the probability of measuring the two qubits in $|10\rangle_{AB}$ or $|01\rangle_{AB}$. This renders the measured classes mutually exclusive, allowing only one control qubit to be in the $|1\rangle$ state at a time.

This concept can be extended to the scenario with m classes by constructing a series of controlled unitaries. With the use of auxiliary qubits, these unitaries enable a one-hot encoding scheme of the classes, akin to how the index with the maximum value is chosen in the classical SVCA for classification tasks. For any initial state $|\psi\rangle|0\rangle = \sum^{N-1} j = 0 a_j|j\rangle|0\rangle$, where $a_x|x\rangle|0\rangle$ represents a specific basis vector and S denotes a subset of basis vectors, each equal to a power of 2, there exists a Unitary transformation U. $U|\psi\rangle|0\rangle = \sum^{N-1} j \neq x a_i|j\rangle|0\rangle + \frac{1}{\sqrt{|S|}} \sum_{k\in S} a_k|k\rangle|1\rangle$.

Lemma 1. *Given a integer $n \geq 2$ and the state $|0\rangle^{\otimes n}$ there exists the Unitary $U_S|0\rangle^{\otimes n} = \frac{1}{\sqrt{|S|}}\sum_{k\in S} a_k|k\rangle$ such that $S = \{\ |i\rangle \in S\ |i \wedge j \in \mathbb{R}, \exists j < n : 2^j = i\}$*

Proof. Given the integer $n \geq 2$ and the state $|\Psi_0\rangle = |0\rangle^{\otimes n} \in \mathcal{H}^{\otimes n}$ assume that the unitary U_S takes the state $|0\rangle^{\otimes n}$ to $\frac{1}{\sqrt{|S|}}\sum_{k\in S} a_k|k\rangle$. Let $|\Psi_t\rangle$ denote the state described by the tth iteration in the sequence operations on the starting state. Let the $|\psi_l\rangle$ denote the subspace for the lth two-dimensional Hilbert space of the total Hilbert space $\mathcal{H}^{\otimes n}$. Let $U_S = U_2U_1(\Theta)$ where Θ is a set of angles. Consider the following unitary matrix:

$$U_1(\Theta) = U(\Theta)_{1b}U(\Theta)_{1a}$$

$$U(\Theta)_{1a} = \left(R_y(\theta_0)_0 \otimes \bigotimes_{j=1}^{n-1} I_j\right)$$

$$U_{1b}(\Theta) = \left(\prod_{k=1}^{n-2}\left(C_{k-1}R_y(\theta_k)_k \otimes \bigotimes_{j\notin\{k-1,k\}} I_j\right)\right)$$

Such that $|\Psi_1\rangle = U_1(\Theta)|\Psi_0\rangle$ which can be expressed as some superposition of the basis vectors:

$$\{|0\rangle^{\otimes n}, |1\rangle|0\rangle^{\otimes n-1}, |11\rangle|0\rangle^{\otimes n-2}, \ldots, |1\rangle^{\otimes n-1}|0\rangle\}$$

The result of applying U_1 can be represented by the equation:

$$|\Psi_1\rangle = U_1|0\rangle^{\otimes n} = \sum_{k=1}^{n-1} a_k \left|2^n - 1 - \sum_{j=0}^{k} 2^j\right\rangle$$

This expression denotes the transformation of the initial state into a superposition of states with each subsequent state having one more qubit flipped to $|1\rangle$ than the previous, up to $n-2$ flips, represented in a reversed order. Let $m = n-2$ and consider the following unitary matrix:

$$U_2(\Theta) = U(\Theta)_{2b}U(\Theta)_{2a}$$

$$U_{2a} = \left(\prod_{k=0}^{n-2}\left(C_{m-k}X_{m-k+1} \otimes \bigotimes_{j\notin\{m-k,m-k+1\}} I_j\right)\right)$$

$$U_{2b} = \left(X_0 \otimes \bigotimes_{j=1}^{n-1} I_j\right)$$

Such that $|\Psi_2\rangle = U_2|\Psi_1\rangle$ which can be expressed as some superposition of the basis vectors in $S = \{\ |i\rangle \in S\ |i \wedge j \in \mathbb{R}, \exists j < n : 2^j = i\}$. This can be expressed as:

$$|\Psi_2\rangle = U_2|\Psi_1\rangle = \sum_{k=0}^{n-1} a_k|2^k\rangle$$

The state $|\Psi_{2n-2}\rangle \in \mathcal{H}^{\otimes n}$ can described by the recursive equation f where $x = n$:

$$f(x) = cos\left(\frac{\theta_{n-1-x}}{2}\right)|2^{n-1-x}\rangle + sin\left(\frac{\theta_{n-1-x}}{2}\right)f(x-1)$$

$$f(0) = |2^{n-1}\rangle$$

Consider that if $|\Psi_{2n-2}\rangle$ was in a state of equal supposition of the coefficients in f can be rewritten as follows:

$$f(x) = \sqrt{\frac{1}{x}}|2^{n-1-x}\rangle + \sqrt{\frac{x-1}{x}}f(x-1)$$

$$f(0) = |2^{n-1}\rangle$$

such that $\theta_k = 2 * cos^{-1}\left(\sqrt{\frac{1}{k}}\right)$ and $P(\langle 2^k|\Psi_2\rangle) = \frac{1}{n}$. Then let the set $\Theta = \{\theta_k|\sqrt{\frac{1}{k}} = cos\left(\frac{\theta_k}{2}\right)\}$. It follows that $|\Psi_2\rangle = \frac{1}{\sqrt{|S|}}\sum_{k\in S} a_k|k\rangle$. Since $U_S|\Psi_0\rangle = U_2U_1(\Theta) = |\Psi_2\rangle = \frac{1}{\sqrt{|S|}}\sum_{k\in S} a_k|k\rangle$ and the choice of $n \geq 2$ is arbitrary, therefore there exists the Unitary matrix U_S for any integer $n \geq 2$ such that $U_S|0\rangle^{\otimes n} = \frac{1}{\sqrt{|S|}}\sum_{k\in S} a_k|k\rangle$ where $S = \{\ |i\rangle \in S\ |i \wedge j \in \mathbb{R}, \exists j < n : 2^j = i\}$.

□

With the classification unitary defined, given a set of auxiliary qubits denoted as Q_A in combination with the qubits corresponding to the basis vectors in the classification space Q_C, every basis vector $|j\rangle$ in the Hilbert space of the classification qubits Q_C, which is not a power of 2, can be mapped to the state $|j\rangle_C|j\rangle_A$ by some unitary $U_j : U_j|j\rangle_C|0...0\rangle_A = |j\rangle_C|j\rangle_A$. Subsequently, applying the unitary $U_{j\to 0}$ to the state $|j\rangle_C|j\rangle_A$ results in $|0...0\rangle_C|j\rangle_A$, enabling the application of a controlled version of U_S on the Hilbert space of the classification qubits whenever the state vector of the auxiliary qubits is $|j\rangle$. Repeating this process for each of the possible bias states of $|j\rangle$ ensures that only basis states corresponding to powers of 2 are observed when measuring qubits in Q_C.

D. Quantum SVCA

With the necessary groundwork laid, we can proceed to define one possible method of implementing SVCA on a quantum computer. This implementation utilizes the modified swap test and multiple one-hot basis encoding unitary U_S to execute the algorithm on a quantum circuit.

E. The Cost of Implementing the QSVCA

The cost of QSVCA in terms of circuit depth and breadth depends on the encoding scheme employed. Broadly speaking, we can associate the cost of QSVCA as a function of variables n (size of the input vector) and m (number of classes).

- Let $E(n)$ represent the function outputting the number of qubits required to encode the input vector.
- Let $S(n)$ represent the function outputting the number of controlled swap operations needed to compute the product of a row vector in the weight matrix with the input vector.
- Let $A(m)$ represent the function outputting the number of qubits needed for the prediction vector to be properly created.
- Let $B(m)$ represent the function outputting the number of operations needed to compute the prediction vector.

Algorithm 3 Quantum SVSC (QSVCA)

Require: Number of input qubits n, number of output qubits m, input vector $\vec{x}$, weight matrix W

Ensure: Quantum Circuit for the classification operation

1: Initialize an array of m quantum registers R_Q with $2n$ qubits each, output quantum register O with m qubits, and auxiliary quantum register A with m qubits
2: Prepare the m copies of the state $|x\rangle$ representing the input vector
3: **for** each row vector $\vec{w}_i$ in W **do**
4: Prepare the state $|w_i\rangle_A$ representing the i-th row vector
5: **end for**
6: **for** each row vector $\vec{w}_i$ in W **do**
7: Perform a controlled swap that creates the state:

$$\frac{1}{2}\left(|0\rangle_O\left(|x\rangle|w_i\rangle+|w_i\rangle|x\rangle\right)+|1\rangle_O\left(|x\rangle|w_i\rangle-|w_i\rangle|x\rangle\right)\right)$$

8: **end for**
9: Define $S \leftarrow \{|i\rangle | i \wedge j \in \mathbb{R}, \exists j < n : 2^j = i\}$
10: Define function $f(i)$ to return a control index for a quantum state not in S
11: **for** $|i\rangle_O \in S$ **do**
12: Apply a controlled Not Gate to the ith auxiliary qubit when the output register is in the state $|i\rangle_O$
13: Apply the controlled Unitary $CU_i(A_i, O)$ to $|i\rangle_O|1\rangle_A$ to $|0...0\rangle_O|1\rangle_{A_i}$
14: Apply Not gate to the i^{th} auxiliary qubit
15: Apply controlled ME gate $CU_S(A_i, O)$
16: Apply Not gate to the ith auxiliary qubit
17: **end for**
18: Measure the output register O and store the result
19: **return** the measurement result

The number of qubits needed for an implementation is given by $m(2E(n)+1)+A(m)$. For the implementation in the paper, we need $2m(n+1)$ qubits. The total depth of the circuit is given by the expression $S(n)+B(m)$. For the implementation in this paper, the depth of the circuit is given by the expression $n \cdot m + 2m + 3$. Therefore it follows that the number of qubits required grows by the order of $O(n \cdot m)$ and the depth can be expressed as $O(m * n)$. This means that the number of qubits and circuit depth grows linearly with the number of features as a function of either m or n.

V. QSVCA Results

With the formulation of the QSVCA complete, it is easy to see that the given representation QSVCA corresponds to its classical counterpart with differing only in that the weighted sum operations in the QSVCA are returned via a relative probability distribution instead of explicitly being compared. It is still important to test the QSVCA as potential issues may still be present that are not immediately obvious. To confirm that the QSVCA correctly encompasses the classification process defined by the classical SVCA we tested the QSVCA on several different data sets.

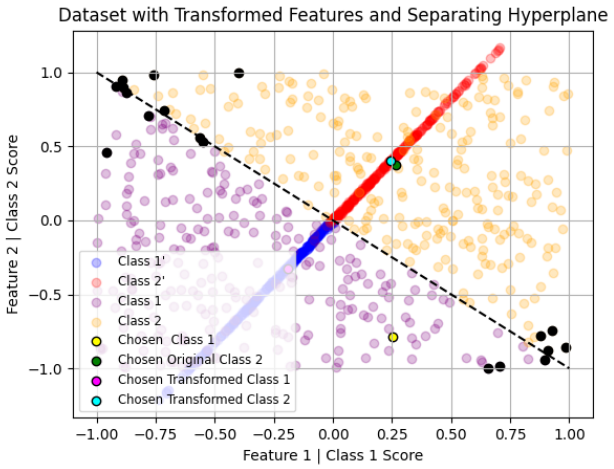


Fig. 5: The results of the QSVCA transformation on the Linear Dataset. The black data points represent the points in the Linear Dataset that are misclassified by the QSVCA.

A. QSVCA Results on Linearly Separable Datasets

We conducted a test on a simple toy linear dataset to evaluate the QSVCA's capability in representing linear relationships, a task easily handled by the classical SVCA. In Figure 5, the transformation of the dataset by the weight matrix is depicted. While the SVCA appears to capture the essence of the relationship, there are small sections where it misclassifies points. This discrepancy may stem from the choice of normalization used during data encoding onto the quantum circuit. It suggests that the quantum circuit is highly sensitive to minor changes in data encoding, highlighting the importance of preprocessing for achieving satisfactory results.

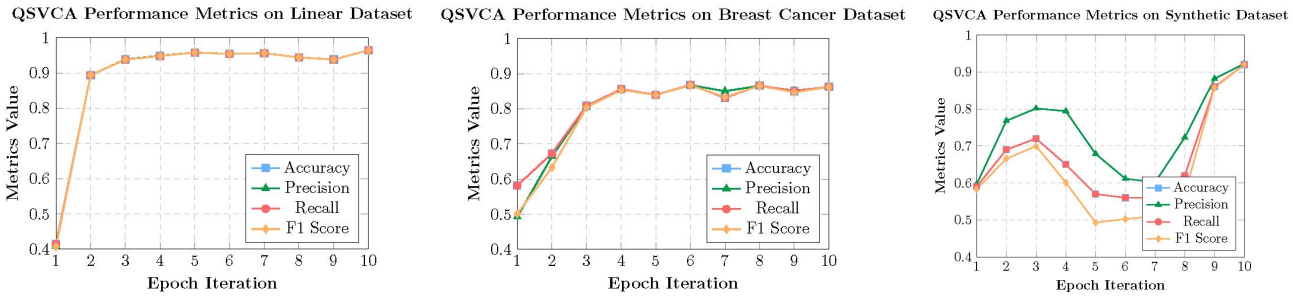
Furthermore, in Figure 6(a), we illustrate how well the weight matrix classifies the linear dataset at any given iteration. This visualization provides insights into the model's improvement after each training epoch, offering valuable information about the effectiveness of the training methods.

B. QSVCA Results on the Breast Cancer Dataset

In addition to testing on a linear dataset, we evaluate the QSVCA on the Scikit-Learn Breast Cancer Dataset [8], which consists of 30-dimensional input vectors. To mitigate the computational cost of simulating the QSVCA, we apply PCA to the dataset and selected the two most highly correlated features. Despite the reduced feature set, the algorithm performed admirably, demonstrating that the QSVCA operates as expected.

C. Other QSVCA Results

In addition to the previous performances of the QSVCA, we show how well the QSVCA performs compared to the SVCA on the dataset in Figure 1, using the same test and training data for both. The result of the QSVCA is shown in Figure 6(c), where it achieves a 92.0% accuracy. This is only a 7%



(a) (b) (c)

Fig. 6: a) Demonstrates the precise classification capability of the QSVCA transformation at each iteration, achieving a maximum accuracy of 96.4% on the linear dataset and b) illustrates the effectiveness of the QSVCA transformation in classifying the Breast Cancer Dataset, achieving a maximum accuracy of 86.3% and c) the QSVCA Results on the Synthetic Dataset seen in fig. 1 with a finally accuracy of 92.0%.

difference between the quantum and the classical version of the SVCA. In addition, the QSVCA performs 1% better the Random Forest classifier and 8% better than the Linear SVM on the same training and testing data.

D. QSVCA Results Overview

Based on the results of the QSVCA on the tested datasets, we observe that it can learn both complex and simple relationships within the dataset, achieving over 85% accuracy on the datasets tested. While the results of the classical SVCA are slightly higher compared to those of the QSVCA, we observe that the QSVCA still converges onto a model that decently approximates the given relationship. The fact that the QSVCA can achieve an accuracy that is less than 9% lower than the classical counterpart suggests that further research into optimizing the feature scaling and learning rates for encoding the entries of a given data point would help close the performance gap.

VI. Conclusion and Future Works

As the demand for new and more efficient machine learning algorithms increases, leveraging the unique computational aspects of quantum computers holds promise for potential performance improvements. While much of the work in QML is theoretical, as many mathematical advantages require more advanced quantum hardware, attempts based on exploring ansatz or variational circuits (like the VQC) [3], [12] have not consistently outperformed classical counterparts. In response to the limitations of existing quantum classifiers, we proposed the SVCA and its quantum equivalent, the QSVCA, aiming to develop a well-motivated quantum classifier that behaves consistently across various classification tasks. We benchmark the proposed algorithms from two main perspectives: 1) Measure how well the algorithm individually performs across a variety of datasets 2) Compare the algorithm against classical peers on the same data set using the same testing training split. The evaluation results demonstrate the high potential and promising outcomes of both SVCA and QSVCA.

- Our observations indicate that the QSVCA is functioning as intended and, with further refinement, should be competitive with its classical counterpart. This suggests the potential for appending additional quantum routines, such as new feature maps, to enhance performance beyond classical methods.
- Unlike some QML classifiers that perform well on specific datasets but struggle with others, the QSVCA demonstrates consistency. It successfully captures linear relationships and performs well on datasets with few features, such as the Breast Cancer dataset.
- The SVCA and its quantum counterpart offer a novel approach to classification tasks in machine learning. By leveraging classical analogies to create a quantum-inside machine learning algorithm, the SVCA can be readily translated onto a quantum circuit, making the QSVCA a reliable quantum classification algorithm akin to those used in classical machine learning.

Future work. This work lays the foundation for further research on quantum classifiers based on the SVCA and similar algorithms. We look forward to extending future research topics that hinge on the QML problem with the following directions: 1) using different encoding methods to help better preserve the underlying relationship in a real number-based metric space, 2) extending the QSVCA or SVCA to allow for the integration of kernels, and 3) testing the SVCA and QSVCA on a wider range of datasets.

VII. The State-of-the-Art

Given the nascent nature of QML, numerous ideas and proposals have emerged to address similar problems. Consequently, there exists a wide array of methods for evaluating the performance of these models, making it challenging to assert their efficiency definitively. However, a recent survey [13] of over 5,000 QML papers provides valuable insights. The survey presents results for several QML algorithms on benchmark datasets, as summarized in Table I. Notably, the observed performance of QML algorithms appears promising, with none achieving less than 87% accuracy. This suggests that QML holds significant potential to compete with classical methods in practical applications.

TABLE I: Quantum algorithms and their accuracy.

Dataset	Algorithm	Testing Precision (%)
MNIST [14]	Deep QCNN	98.97
GTSRB [14]	Deep QCNN	91.40
MNIST [15]	QGAN	87.50
Fashion-MNIST [15]	QGAN	93.16
IRIS [16]	QMC	92.10
BNA [16]	QMC	89.50
WIL [16]	QMC	91.73
IRIS [17]	QNN	96.66
PlanePoint [17]	QNN	96.82

Despite the promising advancements in QML, the lack of reliable quantum computers has hindered thorough testing of various methods across diverse benchmarks, leading to uncertainty regarding their anticipated performance. For instance, multiple proposed algorithms for the quantum perceptron (QP) employ vastly different approaches to data and weight encoding/manipulation [9], [18]–[21], complicating comparison and evaluation. Moreover, while even basic forms of classical machine learning remain unimplemented on quantum computers, these methods often operate on boolean or pseudo-boolean values, lacking intuitive extensions to real-valued numbers.

Furthermore, QML algorithms frequently prioritize performance on *harder* datasets as a measure of increased expressibility, neglecting exploration of their efficacy on *easier or medium* difficulty datasets. This oversimplification regarding the ability of QML algorithms to capture dataset relationships overshadows classical options. While some instances demonstrate QML methods outperforming classical ones, this does not imply a consistent advantage. The generalizability of performance is just one of the factors contributing to the delay in QML applications for real-world problems. Several surveys of QML methods [6], [13] have highlighted potential limitations common quantum classification methods may encounter, as summarized in Table II.

Developing theoretical QML algorithms capable of harnessing quantum properties for a general advantage necessitates further investigation. Tackling these challenges presents substantial theoretical and practical obstacles. Nonetheless, surmounting them is crucial to unlocking the complete potential of QML. One strategy to accomplish this is to meticulously adapt established classical algorithms for implementation on quantum computers. This adaptation allows for their integration with novel quantum techniques.

TABLE II: The state-of-the-art QML classification algorithms.

Model	Description
QP [4], [6], [9], [18]–[21]	No Standardized or definitive version, often are not equivalent to other quantum perceptions and can also not be equivalent to classical perceptron
HQC [6], [22], [23]	Works efficiently with large numbers of feature vectors in a large dataset.
VQC [3], [6], [12], [24]	High cost of implementation.
QSVM [5]–[7]	High complexity.

References

[1] Esma Aïmeur, Gilles Brassard, and Sébastien Gambs. Quantum speed-up for unsupervised learning. *Machine Learning*, 90(2):261–287, 2013.

[2] Dan Ventura. Quantum associative memory. *Information Sciences*, 124(1-4):273–296, 2000.

[3] Danyal Maheshwari, Daniel Sierra-Sosa, and Begonya Garcia-Zapirain. Variational quantum classifier for binary classification: Real vs synthetic dataset. *IEEE Access*, 10:3705–3715, 2022.

[4] Nathan Wiebe, Ashish Kapoor, and Krysta M. Svore. Quantum perceptron models. In *Advances in Neural Information Processing Systems*, volume 29, pages 3999–4007, 2016.

[5] Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. Quantum support vector machine for big data classification. *Physical Review Letters*, 113(13):130503, 2014.

[6] Zainab Abohashima, Mohamed Elhosen, Essam H. Houssein, and Waleed M. Mohamed. Classification with quantum machine learning: A survey, 2020.

[7] H. Y. Huang, M. Broughton, M. Mohseni, et al. Power of data in quantum machine learning. *Nature Communications*, 12:2631, 2021.

[8] 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, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[9] Adenilton José da Silva, Teresa Bernarda Ludermir, and Wilson Rosa de Oliveira. Quantum perceptron over a field and neural network architecture selection in a quantum computer. *Neural Networks*, 76:55–64, 2016.

[10] Kang Min-Sung, Heo Jino, Choi Seong-Gon, Moon Sung, and Han Sang-Wook. Implementation of swap test for two unknown states in photons via cross-kerr nonlinearities under decoherence effect. *Scientific Reports*, 9(1):6167, 2019.

[11] Ronald de Wolf. Quantum computing: Lecture notes, 01 2021.

[12] Marcello Benedetti, Edward Lloyd, Stefan Sack, and Mattia Fiorentini. Parameterized quantum circuits as machine learning models. *Quantum Science and Technology*, 4(4):043001, 2019.

[13] David Peral-García, Juan Cruz-Benito, and Francisco José García-Peñalvo. Systematic literature review: Quantum machine learning and its applications. *Computer Science Review*, 51:100619, 2024.

[14] Y. Li, R.G. Zhou, R. Xu, J. Luo, and W. Hu. A quantum deep convolutional neural network for image recognition. *Quantum Science and Technology*, 5(4), 2020.

[15] M. Alam and S. Ghosh. Qnet: A scalable and noise-resilient quantum neural network architecture for noisy intermediate-scale quantum computers. *Frontiers in Physics*, 9, 2022.

[16] A. Chalumuri, R. Kune, and B.S. Manoj. A hybrid classical-quantum approach for multi-class classification. *Quantum Information Processing*, 20(3), 2021.

[17] P. Li and B. Wang. Quantum neural networks model based on swap test and phase estimation. *Neural Networks*, 130:152–164, 2020.

[18] Samuel A Wilkinson and Michael J Hartmann. Evaluating the performance of sigmoid quantum perceptrons in quantum neural networks, 2022.

[19] E. Torrontegui and J. J. García-Ripoll. Unitary quantum perceptron as efficient universal approximator. *Europhysics Letters (EPL)*, 125(3):30004, 2019. The Physics of Quantum Engineering and Quantum Technologies.

[20] Rodrigo Araiza Bravo, Khadijeh Najafi, Taylor L. Patti, Xun Gao, and Susanne F. Yelin. Expressive quantum perceptrons for quantum neuromorphic computing, 2024.

[21] Y. Freund and R. E. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296, 1999.

[22] Giuseppe Sergioli, Roberto Giuntini, and Hector Freytes. A new quantum approach to binary classification. *PloS one*, 14(5):e0216224, 2019.

[23] Abdullah M. Iliyasu and Chastine Fatichah. A quantum hybrid pso combined with fuzzy k-nn approach to feature selection and cell classification in cervical cancer detection. *Sensors*, 17(12):2935, 2017.

[24] Shi-Yu Chen and Hsi-Sheng Goan. Variational quantum circuits and deep reinforcement learning. *arXiv preprint arXiv:190700397*, 2019.