Reexamining the Computational Efficiency of separable quantum support vector machine training oracles

Kamal Al-Ameri

August 2025

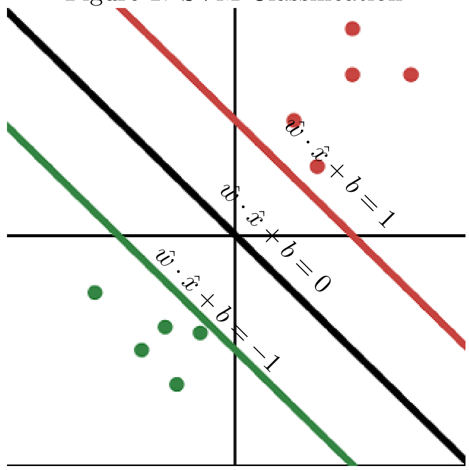
1 Abstract

The quantum support vector machine (QSVM) is one of the the simplest methods of data classification on quantum computers, but suffers from deep circuits which current noisy hardware can not simulate accurately. In this text, we examine the qubit scaling of the approach of kernel generation proposed in a 2019 paper by Jiaying Yang, Ahsan Javed Awan, and Gemma Vall-Llosera, mathematically proving the redundancy of the majority of the computation done in the training-data oracle proposed in the original paper, and demonstrate that we can achieve the same result as in the original paper using a single qubit and a single data point.

2 Introduction

Quantum machine learning is one of the most actively researched fields of quantum computation, and many algorithms for implementing machine learning algorithms on quantum computers have been proposed [2]. However, currently available quantum hardware, also known as Noisy Intermediate Scale Quantum (NISQ), suffers from increasingly inaccurate results as the complexity of the algorithms increases [3]. A previous paper by Yang et al. [4] proposed a Quantum Support Vector Machine (QSVM) algorithm, a crucial part of which was optimized to have as little circuit depth as possible, arguably providing a significant advantage for NISQ hardware. Additionally, their proposed method removes the need for the usage of the quantum tomography technique. We mathematically prove and numerically demonstrate that the same accuracy achivied in the paper could be achieved without the overwhelming majority of the computational overhead. In fact, we prove that almost all of the computational overhead is redundant.

Figure 1: SVM Classification



3 Background

3.1 The SVM Algorithm

The support vector machine algorithm is used to classify data into one of two classes, by finding a hyperplane which divides the points into 2 sections as accurately as possible. More formaly, the SVM tried to find the hyperplane that divides the data into two classes and has the maximum distance to the nearest point of each class.

3.2 Basic Quantum Computing Concepts

3.2.1 The Qubit

The building block of quantum computing is the qubit, which like a classical bit, can exist in two states: $|0\rangle$ and $|1\rangle$. However, unlike a classical bit, it can exist in a *superposition* of the two states, mathematically represented as $|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$, where β and α are complex amplitudes, and $|\psi\rangle$ is the state of the qubit. When the superposition *collapses*, the qubit is projected into either $|0\rangle$ or $|1\rangle$, with probabilities $|\alpha|^2$ and $|\beta|^2$ respectively.

3.2.2 Entanglement

A group of qubits may become *entangled*, meaning that their states are intrinsically linked. Measurement of one qubit instantaneously determines the state of its entangled partner.

3.2.3 Quantum Gates and Circuits

Quantum gates are operations that manipulate qubits, and can create superposition and entanglement. A set of quantum gates is called a *circuit*, which has the properties or width, which is the number of qubits used in the circuit, and depth, which is the count of time steps needed to execute all the gates in a circuit, where gates that can be executed in parallel count as one time step, while those that must be executed sequentially count as multiple time steps.

3.2.4 The Quantum Statevector

The *quantum statevector* for a system with n qubits is a vector of length 2^n, where each element is a complex amplitude of a quantum basis state of the system. In other words, for a quantum state $|\psi\rangle$, such that:

$$|\psi\rangle = \alpha_1 |000..0\rangle + \alpha_2 |000..1\rangle + ... + \alpha_n |111..1\rangle$$

the quantum state vector of $|\psi\rangle$ is the vector $\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ ... \\ \alpha_n \end{pmatrix}$

Note that because both the of the 2 equations above represent the same concept, $|\psi\rangle$ will refer to both the quantum state and the quantum state vector of itself.

We will not explain dirac notation in depth in this text, but it is worth mentioning that $\langle\psi|$ is the complex conjugate of $|\psi\rangle$, $\langle\alpha| |\beta\rangle$ is the inner product of $\langle\alpha|$ and $|\beta\rangle$, and that $|\alpha\rangle \langle\beta|$ is the outer product of $|\alpha\rangle$ and $\langle\beta|$.

4 Mathematical Proof of Redundancy

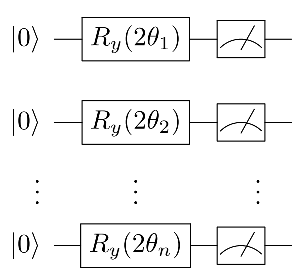


Figure 2: Training-data oracle proposed in the original paper

In this text, we refer to [4] as *the original paper*, as we will reference it quite often.

The original paper proposes a training-data oracle for generating the kernel of the support vector machine, whose width scales linearly with the number of training points, but has a depth of one (see fig. 2). This means that the circuit uses much more qubits (M qubits for M training points, rather than the $log_2 M+1$ qubits used by the training-data oracle in [5]), but the constant depth of the circuit arguably provides it with an advantage for near-term quantum computers, which can handle up to around 100 qubits, but suffer from too much noise to handle deep or dense circuits. [3]

The oracle proposed in the original paper is composed of one R_y gate being applied to each qubit. Then, the density operator of the final state, $|\psi\rangle$, is constructed, and then it's partial state is computed, which is passed to the HHL algorithm. Proving the redundancy of the oracle is quite simple:

The density operator of a separable (i.e. unentangled) quantum state vector is the tensor product of the density operators of it's qubits (see sec. 2.4.1 of [1]). Therefore, the density operator of $|\psi\rangle$ is:

$$\rho = \rho_1 \otimes \rho_2 \otimes ... \otimes \rho_m \tag{1}$$

Where ρ_i is the density operator of the i-th qubit.

The partial trace, $\mathrm{Tr_b}$, of a tensor product of quantum states, tracing out every state but the b-th state, is equal to ρ_b, up to some normalization factor (see sec. 8.3.1 of [1]) (for our data, normalized as in the original paper, the normalization factor is 2). Thus:

via Eq. (1)

$$\mathrm{Tr_b}(\rho) = \mathrm{Tr_b}(\rho_1 \otimes \rho_2 \otimes ... \otimes \rho_b \otimes ... \otimes \rho_m)$$

$$= 2\rho_b$$

$$= 2(|R_y(2\theta_b)(|0\rangle)\rangle \langle R_y(2\theta_b)(|0\rangle)|) \tag{2}$$

Where θ_b is the training theta for the b-th qubit.

Therefore, the training-data oracle can achieve the same results demonstrated in the paper using only 1 training point, 1 qubit, and one rotation gate.

Note that some quantum circuit frameworks use different conventions for rotation gates, most commonly, flipping the sign of the angle. to prevent such issues from affecting the kernel, we add an absolute:

$$= 2|(|R_y(2\theta_b)(|0\rangle)\rangle \langle R_y(2\theta_b)(|0\rangle)|)| \tag{3}$$

4.1 Generalization to similar oracles

The property of only one qubit having an effect on the final kernel is inherent to separable training-data oracles, assuming that the final goal of the oracle is to take the partial trace of the matrix in order to generate the kernel. This strongly suggests that entanglement and circuit depth play a crucial role in accurate training-data oracles for quantum support vector machines, implying potential limitations for NISQ hardware in machine learning applications.

5 Numerical Method

In the context of classical statevector simulation of a single qubit system,

$$R_y(2\theta_b)(|0\rangle) = \begin{pmatrix} cos(\theta_b), -sin(\theta_b) \\ sin(\theta_b), cos(\theta_b) \end{pmatrix} \cdot \begin{pmatrix} 1.0 \\ 0.0 \end{pmatrix} = \begin{pmatrix} cos(\theta_b) \\ sin(\theta_b) \end{pmatrix} \tag{4}$$

Thus, (3) becomes:

$$\mathrm{Tr_b}(\rho) = 2 \left(\left| \begin{pmatrix} cos(\theta_b) \\ sin(\theta_b) \end{pmatrix} \otimes \begin{pmatrix} cos(\theta_b) \\ sin(\theta_b) \end{pmatrix} \right| \right) \tag{5}$$

You may input any generated training angle θ_b into this equation, and the result will be a valid kernel matrix, which you can input into the 4-qubit HHL circuit, with the rest of the inputs, resulting in a state $|\psi\rangle$.

The binary output of the QSVM is:

$$sgn(\langle\psi|O|\psi\rangle)$$

where

$$O = |0000\rangle \langle 0000| \otimes |1\rangle \langle 0|$$

with the arbitrary assumption that $\mathrm{sgn}(0) = 1$.

In our implementation used for the demonstrations, we implement the above equations directly and without optimizations, using the python programming language and the numpy library. The simulation of the quantum circuits will use a simple, ad-hoc, dense statevector simulator.

6 Demonstration on Datasets

In this subsection, we will train the QSVM using the single-qubit training-data oracle using different datasets, and demonstrate the accuracy.

6.1 MNIST Database

We will use the MNIST database of handwritten digits, choosing two arbitrary digits and training the QSVM to differentiate between them, comparing our method, simulated as described in chapter 5, to the method of the original paper, simulated with 12 data points (note that the accuracy of the original method does not change with the number of data points, the choice of 12 is arbitrary).

To keep the comparison fair, the partial trace of both methods will trace out all qubits except the fist qubit (i.e. $b = 1$), which, for clarification, does not include any actual partial tracing in our method, as we only take the b-th theta. However, by empirical evidence, we saw no significant difference in accuracy between keeping the first qubit or any other qubit.

first digit	second digit	accuracy (ours)	accuracy (original)
2	7	0.71	0.71
6	9	0.91	0.91
9	5	0.77	0.77

6.2 Boston Housing Data

We will use the keras Boston housing database, choosing two arbitrary feature indices and training the QSVM to differentiate between them, comparing our method, simulated as described in 5, to the method of the original paper, simulated with 12 data points.

first index	second index	accuracy (ours)	accuracy (original)
10	12	0.717	0.717
4	7	0.565	0.565
2	11	0.619	0.619

As you can see, there is no difference between the two methods, further confirming the redundancy of the method described in the original paper.

7 Conclusion

In this text, we mathematically proved the inefficiency of the training-data oracle proposed in the original paper, and numerically demonstrated that a single qubit oracle is equivalent to the oracle proposed in the original paper, by demonstrating that both oracles have exactly equivalent accuracies. Furthermore, we demonstrated that for simple datasets, like the MNIST optical character recognition dataset, the single-qubit oracle can reach up to 91% accuracy on certain inputs.

References

[1] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010.

[2] Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. "Quantum Support Vector Machine for Big Data Classification". In: *Phys. Rev. Lett.* 113.13 (2014), p. 130503. DOI: `10.1103/physrevlett.113.130503`. arXiv: `1307.0471 [quant-ph]`.

[3] John Preskill. "Quantum Computing in the NISQ era and beyond". In: *Quantum* 2 (2018), p. 79. DOI: `10.22331/q-2018-08-06-79`. arXiv: `1801.00862 [quant-ph]`.

[4] Jialei Yang, Ahsan Javed Awan, and Gemma Vall-llosera. "Support Vector Machines on Noisy Intermediate Scale Quantum Computers". In: *ArXiv* abs/1909.11988 (2019). URL: https://api.semanticscholar.org/CorpusID:202889170.

[5] Qing-Yuan Wu et al. "Experimental realization of a quantum classification: Bell state measurement via machine learning". In: *APL Machine Learning* 1.3 (Sept. 2023), p. 036111. ISSN: 2770-9019. DOI: 10.1063/5.0149414. eprint: https://pubs.aip.org/aip/aml/article-pdf/doi/10.1063/5.0149414/18126436/036111_1_5.0149414.pdf. URL: https://doi.org/10.1063/5.0149414.