

Quantum Machine Learning with HQC Architectures using non-Classically Simulable Feature Maps

Syed Farhan Ahmad^{1,2},

1. Department of Electronics and
Communication Engineering, RVCE,
Bangalore, Karnataka, India
2. QPower Research, Salt Lake City,
Utah, United States
syedfarhana.ec18@rvce.edu.in

Raghav Rawat^{1,2},

1. Department of Electronics and
Communication Engineering, RVCE,
Bangalore, Karnataka, India
2. QPower Research, Salt Lake City,
Utah, United States
raghavrawat.ec18@rvce.edu.in

Minal Moharir

Department of Computer Science and
Engineering, RVCE, Bangalore,
Karnataka, India
minalmoharir@rvce.edu.in

Abstract— Hybrid Quantum-Classical (HQC) Architectures are used in near-term NISQ Quantum Computers for solving Quantum Machine Learning problems. The quantum advantage comes into picture due to the exponential speedup offered over classical computing. One of the major challenges in implementing such algorithms is the choice of quantum embeddings and the use of a functionally correct quantum variational circuit. In this paper, we present an application of QSVM (Quantum Support Vector Machines) to predict if a person will require mental health treatment in the tech world in the future using the dataset from OSMI Mental Health Tech Surveys. We achieve this with non-classically simulable feature maps and prove that NISQ HQC Architectures for Quantum Machine Learning can be used alternatively to create good performance models in near-term real-world applications.

Keywords— Quantum Machine Learning, Hybrid Quantum-Classical Architecture, QSVM, Quantum Feature Map, Quantum binary Classifiers

I. INTRODUCTION

Quantum Computing is the use of Quantum physics and Quantum phenomena for computation of tasks. It is a completely different paradigm and has shown great applications for problems that Classical Computers cannot solve. The hype and scepticism of Quantum Computers were positively justified when Google successfully demonstrated Quantum Supremacy with their 54-qubit Sycamore processor [1]. Quantum Computing has multiple real-life applications that include Drug discovery [2], Disease Risk Predictions [2], Routing and Optimization problems [3], Materials Discovery [4], and Machine Learning [19].

Quantum Computing can greatly help in creating better Machine Learning models [16] that train much faster and can encode more information than their classical counterparts [18]. They provide an exponential speedup due to the availability of an exponential state space in which the classical data can be mapped [5]. The field of Quantum Machine Learning has seen the use of 3 different architectures: Quantum Data with Classical processing, that can be used for phase estimations in matter [6], Quantum data with Quantum processing, that can be used for inherent quantum error correction [7], and the final architecture is the where classical data is mapped onto quantum systems and the power of quantum mechanics helps in solving the problem faster and with a better accuracy (in some cases) [3]. This architecture is known as the HQC (Hybrid Quantum-Classical) architecture. In this paper, we

will be discussing about a Quantum Classifier as a trainable quantum circuit which is a form of HQC architecture for quantum machine learning.

Today's generation of Quantum Computers fall under the NISQ (Noisy Intermediate-State Quantum) era, which makes them prone to errors, less reliable and not very usable [17]. Measurement errors, interactions of adjacent qubits and thermal fluctuations are the main causes by which fidelity of quantum gates and circuits are greatly reduced. The Hybrid Quantum-Classical Architecture for Machine Learning shows promising results in this NISQ era [8].

The quantum component of the HQC Architecture maps data features very effectively compared to Classical SVM (Support Vector Machines) Architecture and in this paper, we will be talking about a classical to quantum mapping that cannot be classically simulated and hence provides a quantum advantage due to quantum phenomena like entanglement. These HQC Architectures facilitate the processing of purely classical data while enjoying the benefits of a Quantum feature spaces [9]. This is achieved by non-linear mapping of the data into the quantum state $\Phi(\vec{x})$.

$$\Phi: \vec{x} \rightarrow |\Phi(\vec{x})\rangle\langle\Phi(\vec{x})|$$

The application we have put forth in this paper is of a supervised quantum binary classification.

The data is not always linearly separable in the real-world setups [10], and hence, kernel methods are of immense importance in machine learning. For a state x_j , we need to compute the scalar product $x_j^T x_j$ which takes the form of $\Phi: x \rightarrow (x, x^2)$ after kernel embedding [11]. The prime benefit of using the "Kernel Trick" is that knowledge of the embedding type is not required if we know the scalar product. One such type is the Gaussian Kernels where data is embedded to higher dimensions, followed by mapping to Gaussians and then computation of scalar product systematically. A comparison of such a classical mapping is also drawn out with Quantum feature mapping in the paper which uses the RBF Kernel feature map [11].

For $\sigma = 1$ and $\sigma^2 = 1$;

$$K(X_1, X_2) = e^{-\frac{\|X_1 - X_2\|^2}{2}}$$

To achieve the feature mapping in quantum respect: First, we will have to translate the classical data points (\vec{x}) into quantum datapoints ($|\Phi x\rangle$) This is achievable by the circuit $\mathcal{U}_{\Phi(\vec{x})} |0\rangle$ as shown in eq.2 , where Φ is a kernel function applied on the classical data points. Second, we require a parametrized quantum circuit $W(\theta)$, that processes the data in a way which enables the use of the training of model with these parameters. Third, we need a classical optimization loop that tunes the orientation of hyperplane and returns +1 or -1 for the classical data points (\vec{x}) .

The paper has been divided in the following manner. Section 2 builds up the environment for Quantum Computing by talking about quantum circuits and entanglement. Section 3 talks about the Classical Architecture that has been used for contrasting the results of the quantum architecture in this paper. Section 4 talks about the Experimental Methodology in depth, covering the datasets used, Quantum Machine Learning model training, embeddings, feature maps and runs on an actual IBM Quantum Computer. Section 5 compares the results of the Classical Model, Quantum Model, and the execution on an actual Quantum hardware. The paper concludes with remarks and possible future work in Section 6.

II. PARAMETERIZED QUANTUM CIRCUIT

A quantum circuit is a computational routine consisting of Coherent Quantum operations on quantum data, such as qubits, and concurrent real-time classical computations. Qubits are analogous to classical bits that can be in a superposition of states at the same time, which gives an added advantage of using the exponential state space in the Hilbert Space.

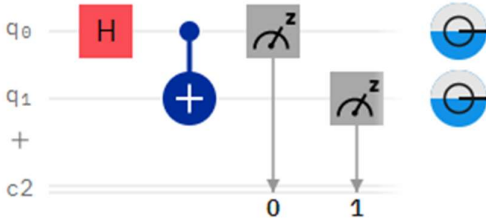


Fig. 1. Quantum Bell State circuit with Hadamard gate [pink], CNOT gate [dark blue], measurement operators [grey] and measurement outcomes of both qubits.

Quantum gates are unitary operations that alter the state of qubits. Quantum systems have a very unique property, known as entanglement, that allow pair or group of particles to interact, or share spatial proximity in a way such that the quantum state of each particle of the pair or group cannot be described independently of the state of the others. A Bell State is shown in figure 1 that entangles 2 qubits q0 and q1 using the Hadamard[H] and CNOT Gates. Eq.1 portrays the Quantum Bell State that has an absolute entanglement measure:

Eq 1.

$$\frac{1}{\sqrt{2}}(|00\rangle_{0,1} + |11\rangle_{0,1})$$

III. CLASSICAL MACHINE LEARNING ARCHITECTURE - SVM

In general, the objective of SVM (Support Vector Machine) algorithm is to find a hyperplane in any 'n' dimensional space that perfectly classifies the data. [10] Support vectors are the points which are closer to the separating plane and directly influence the width and orientation of the separating plane. The perpendicular distance of these points and the plane is called 'Margin' which is shown using the range of red lines in Fig 2. The aim is to calculate optimal parameters which minimizes $\|w\|$.

$$\min \|w\|^2 \quad \text{or} \quad 1 = \min \frac{\|w\|^2}{2}$$

$$\text{such that } y_i(w \cdot x_i + b) - 1 \geq 0 \text{ for } i = 1 \dots l$$

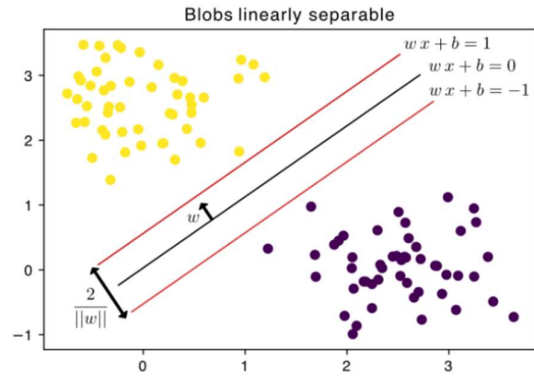


Fig 2. 2D Plot of sample blobs separated by a hyperplane where the support vectors define the margin orientation.

IV. EXPERIMENTAL METHODOLOGY

This section discusses the dataset used and the experimental steps of creating the Quantum Embeddings, the Quantum Variational Circuit, the Classical Loss Function, and the Optimizer for the HQC Architecture for Machine Learning. The overall architecture is shown in Fig 3.

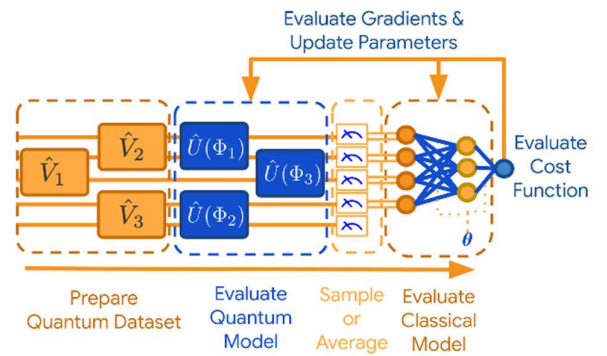


Fig 3. The HQC (Hybrid Quantum Classical) Architecture for Quantum machine learning with Quantum State Preparation, Quantum variational circuit, classical loss function and the optimizer

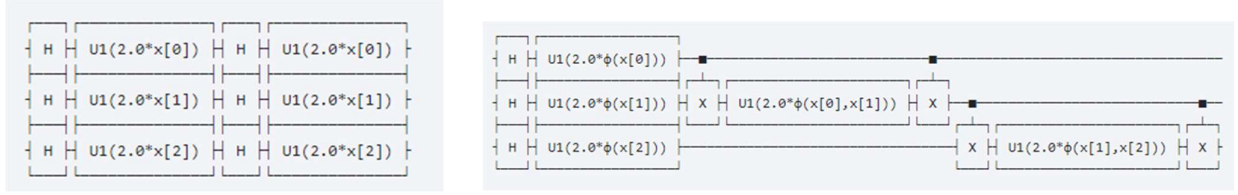


Fig 4a. ZFeatureMap Architecture(left), 4b. ZZFeatureMap(right)

Classification Algorithm in Quantum Variational Circuit

- | | |
|--------|--|
| Step 1 | : Input Labelled training datapoints, $(\vec{x} \in R^n) \times (y \in [+1, -1])$ |
| Step 2 | : Embedding into quantum feature space |
| Step 3 | : Entanglement using Variational Circuit, and measurement for classical optimization |
| Step 4 | : Classical Loss function computation for optimal Hyperplane |
| Step 5 | : Parameter Update — w_i and b (in this case, theta) |
| Step 6 | : Updating of VQC parameters |
| | : Repeating Steps 2-6 until convergence |

A. Dataset

We have used the Kaggle dataset for Mental Illness in the Tech World [12] in our Quantum Classification problem. This is a real-world dataset that can be analyzed with Classical Machine Learning techniques hence making it easier to benchmark the Quantum Machine Learning algorithm with the HQC architecture.

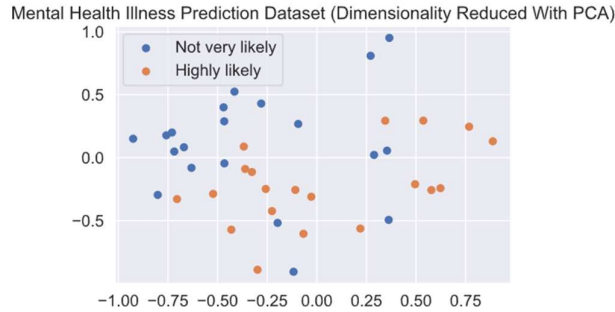


Fig 5. Mental Health in Tech World dataset class labels for 40 datapoints. The dataset is dimensionally reduced with computation of Principal Components.

B. Quantum Embeddings

The first step of training quantum circuits is to encode the classical data into the quantum computer. We achieve this using the quantum feature map that embeds data into a higher-dimension Hilbert space. Many feature maps have been proposed; the two notable ones that give highest gate fidelities are the ZFeatureMap and the ZZFeatureMap, shown in figures 4a. and 4b. respectively.

ZZFeatureMap outperforms the ZFeatureMap in terms of the Entangling Capability and Expressivity as mentioned in this paper [13]. The ZZFeatureMap is a Second-order Pauli-Z evolution circuit, where φ is a classical non-linear function, which defaults to $\varphi(x) = x$ if and $\varphi(x,y) = (\pi - x)(\pi - y)$.

The quantum mapping function is shown in Eq2. below:

Eq 2.

$$K(\vec{x}, \vec{z}) = |\langle \Phi(\vec{x}) | \Phi(\vec{z}) \rangle|^2 = \langle 0^n | \mathcal{U}_{\Phi(\vec{x})}^\dagger \mathcal{U}_{\Phi(\vec{z})} | 0^n \rangle$$

Fig 6a. is used for evaluating and estimating the value of the quantum kernel to enable using the data in the classical loop.

C. Quantum Kernels and Feature Maps

For the quantum kernel, we take the inner product as per Eq. 2 above, but now using quantum feature mapping $\mathcal{U}_{\Phi(\vec{x})}$. By using the quantum feature maps which are relatively difficult to be simulated on classical computers, we are using advantages of quantum mechanics to solve the classical problems.

D. Classical Loss Function and Optimization Loop

For the Optimization loop after the quantum feature mapping is complete, the QSVM algorithm from qiskit Aqua has been used. This algorithm works well for classification tasks that need quantum feature mapping and for which kernel computation is not efficient with a classical approach. QSVM uses an ideal quantum simulator which directly estimates kernel in the feature mapping. Furthermore, in the training phase, it uses the kernel estimations to obtain support vectors. Now in the testing phase, new data is efficiently classified according to the hyperplane of the support vectors. The steps followed in this optimization loop can be seen equivalent to the most popular ones of classical SVM which are as follows [15]:

1. Parameter C of our SVM algorithm is set, which is responsible for the trade-off between training error minimization and margin width maximization.
2. Select appropriate kernel function and its parameters as per the given problem. For instance, in our case,

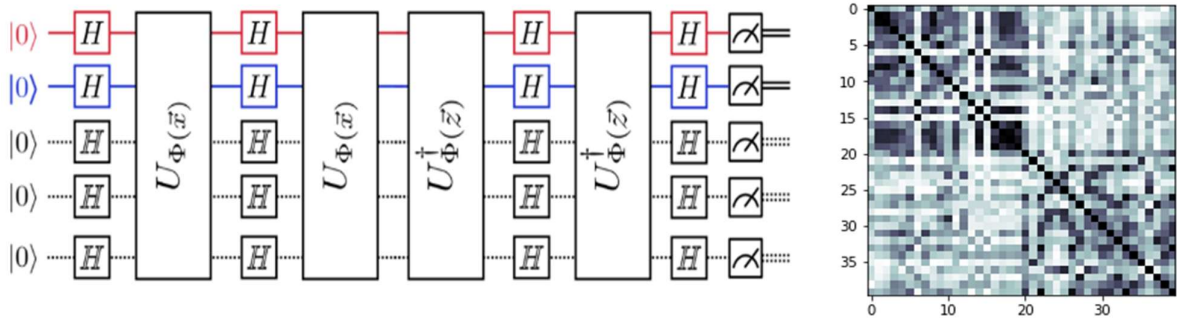


Fig 6a. Circuit to estimate the Quantum kernel(left); 6b. Kernel Matrix during training of QSVM on qasm simulator

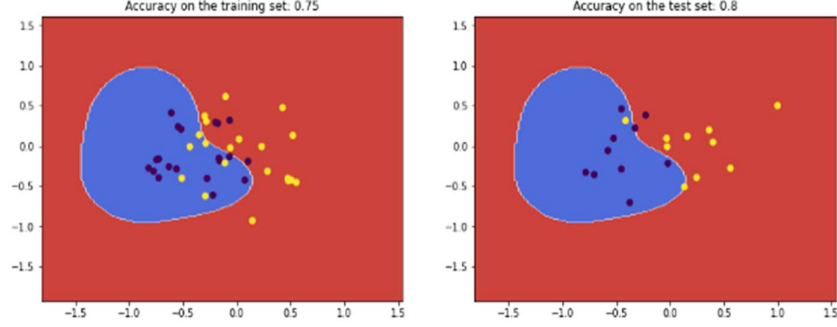


Fig 7: Accuracy of the model for the training set(left) and the test set(right) respectively

radial basis function kernel requires selection of gaussian width σ .

3. The SVM formulation employed is solved using linear programming or polynomial programming algorithms.
4. Obtaining the threshold parameter b using the support vectors.
5. Classification of the new data points as per the following signum function

$$f(x) = \text{sign} \left(\sum_i y_i \alpha_i K(x, x_i) - b \right)$$

E. Execution on Real Quantum hardware

Quantum Circuits can be executed on a real quantum hardware via IBM's IQX, an online platform to manage the tasks given to the IBM's Quantum Computers. In our experiments, we have used 2 hardware architectures: IBM Yorktown device architecture and the IBM Ourense device architectures. [14]

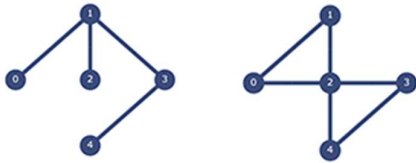


Fig 8. IBM Quantum Device Architectures: Ourense device with T architecture(left), Yorktown device with bowtie architecture(right)

V. RESULTS AND DISCUSSION

In this section, the findings and analysis with possible conclusions from training and inference of the classical and quantum models are discussed, on both classical hardware and on IBM Quantum hardware.

A. Classical SVM Model

Following the methodology mentioned in Section 3, the training accuracy of the model rises from 0.65 to 0.75 and the test accuracy rises from 0.75 to 0.8.

B. Quantum SVM Model

Following the methodology mentioned in Section 4 B-D, the training accuracy of the model rises from 0.65 to 0.69 and the test accuracy rises from 0.65 to 0.70 when the dataset size is very small. The Qiskit qasm_simulator backend [14] was used for this purpose.

C. IBMQ Quantum Computer

Following the methodology mentioned in Section 4E, running the QSVM model circuit on the actual IBM Quantum Hardware gives a train accuracy of 0.63 and test accuracy of 0.69 for the Ourense device. The accuracies observed for the Yorktown device are 0.61 for training and 0.69 for test when the dataset size is very less.

D. Discussions

Table 1: Comparison of training and test parameters

Accuracy	Device/Architecture Type			
	Classical SVM	QSVM qasm Simulator	IBM Ourense	IBM Yorktown
Train	0.75	0.69	0.63	0.61
Test	0.8	0.70	0.69	0.69

Table 1 compares the training and test accuracies of the model when run with classical architecture and quantum architecture with both classical simulations and actual quantum IBM hardware. The kernel matrix during training is shown in Fig 6b. justifies the effective performance of matrix operations in higher dimensional vector space, i.e., the Hilbert space.

We were able to successfully compare the model's performance on 2 quantum hardware architectures, as shown in table 1 and Fig 8., the bowtie-architecture from Yorktown device and the T-architecture from the Ourense device. The T-architecture outperforms the bowtie-architecture for our application.

VI. CONCLUSION

In this research, an extremely efficient quantum binary classifier using QSVM (Quantum Support Vector Machine) with non-classically Simulable quantum feature map has been implemented on the following devices: qasm simulator, IBM Quantum Ourense device and IBM Quantum Yorktown device. The accuracy of quantum model comes out to be almost comparable to the classical SVM counterpart which proves that NISQ quantum computers can be used for solving near-term real-world applications. The quantum system gave out a training accuracy of 69% and a test accuracy of 70% which is a great achievement in terms of quantum metrics. The method proposed in the paper is highly generalizable and can be applicable to any binary classification task with real-world data. The model can further be extended to multi-class classification and clustering tasks by altering the variational quantum circuit architecture. Selecting the right quantum feature map and quantum variational circuit is an open area of research in the quantum community. Gradient based HQC machine learning architectures are yet to be analyzed at full capacity due to the inability of properly understanding how information is scrambled due to entanglement in quantum circuits. A future direction would be to analyze the loss landscape of the quantum machine learning model to better understand correlations between data and model training, enabling us to create better HQC architectures.

The research thus makes HQC Architecture based Quantum Machine Learning more accessible and demonstrates the use of this algorithm to solve a near-term real-world problem.

VII. REFERENCES

- [1] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell, et al., "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, no. 7779, pp. 505–510, 2019
- [2] Carlos Outeiral, Martin Strahm, Jiye Shi, Garrett M. Morris, Simon C. Benjamin, Charlotte M. Deane, "The prospects of quantum computing in computational molecular biology", arXiv:2005.12792 [quant-ph]
- [3] Davide Castaldo, Marta Rosa, Stefano Corni, "Quantum optimal control with quantum computers: an hybrid algorithm featuring machine learning optimization", arXiv:2007.00368 [quant-ph]
- [4] Panagiotis Kl. Barkoutsos, Fotios Gkritis, Pauline J. Ollitrault, Igor O. Sokolov, Stefan Woerner, Ivano Tavernelli, "Quantum algorithm for alchemical optimization in material design", arXiv:2008.06449 [quant-ph]
- [5] Carlton M. Caves, Ivan H. Deutsch, Robin Blume-Kohout, "Physical-resource demands for scalable quantum computation", arXiv:quant-ph/0304083
- [6] P. M. Q. Cruz, G. Catarina, R. Gautier, J. Fernández-Rossier, "Optimizing quantum phase estimation for the simulation of Hamiltonian eigenstates", arXiv:1910.06265 [quant-ph]
- [7] Todd A. Brun, "Quantum Error Correction", arXiv:1910.03672 [quant-ph]
- [8] Giacomo Torlai, Roger G. Melko, "Machine learning quantum states in the NISQ era", arXiv:1905.04312 [quant-ph]
- [9] K. Bertels, A. Sarkar, A. Mouedenne, T. Hubregtsen, A. Yadav, A. Krol, I. Ashraf, "Quantum Computer Architecture: Towards Full-Stack Quantum Accelerators", arXiv:1903.09575v3
- [10] Yichuan Tang, "Deep Learning using Linear Support Vector Machines", arXiv:1306.0239 [cs.LG]
- [11] Vojtech Havlicek, Antonio D. Córcoles, Kristan Temme, Aram W. Harrow, Abhinav Kandala, Jerry M. Chow, Jay M. Gambetta, "Supervised learning with quantum enhanced feature spaces", arXiv:1804.11326 [quant-ph]
- [12] Mental Health in Tech Survey: Survey on Mental Health in the Tech Workplace <https://www.kaggle.com/osmi/mental-health-in-tech-survey> [online]
- [13] Sukin Sim, Peter D. Johnson, Alan Aspuru-Guzik, "Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms", arXiv:1905.10876 [quant-ph]
- [14] IBM Quantum systems overview, <https://quantum-computing.ibm.com/docs/manage/backends/> [online]
- [15] Kristin P. Bennett and Colin Campbell. 2000. "Support vector machines: hype or hallelujah?" (Dec. 2000), 1–13. DOI:https://doi.org/10.1145/380995.380999
- [16] Report for: Machine learning & artificial intelligence in the quantum domain: A review of recent progress. (n.d.).
- [17] Beer, K., Bondarenko, D., Farrelly, T., Osborne, T., Salzmann, R., Scheiermann, D., & Wolf, R. (2020, February 10). Training deep quantum neural networks.
- [18] Dunjko, V., & Wittek, P. (2020, March 17). A non-review of quantum machine learning: Trends and explorations.
- [19] Schuld, M. (2019, March 13). Machine learning in quantum spaces.