OPTIC: A Practical Quantum Binary Classifier for Near-Term Quantum Computers

Tirthak Patel, Daniel Silver, and Devesh Tiwari
Northeastern University, Boston, MA, USA

***Abstract*—Quantum computers can theoretically speed up optimization workloads such as variational machine learning and classification workloads over classical computers. However, in practice, proposed variational algorithms have not been able to run on existing quantum computers for practical-scale problems owing to their error-prone hardware. We propose OPTIC, a framework to effectively execute quantum binary classification on real noisy intermediate-scale quantum (NISQ) computers.**

***Index Terms*—Quantum Computing, NISQ Computing, Quantum Machine Learning, Quantum Image Classification**

I. INTRODUCTION

Background. Quantum algorithms have been developed in the areas of chemistry, cryptography, drug-discovery, finance, and optimization [42], [28], [14], [46], [17], [19], [12], [32], [44]. A class of algorithms known as quantum variational algorithms are designed to optimize and execute quantum machine learning and classification workloads [39], [29], [23], [4], [6], [2], [25], [34], [37].

While theoretically-promising, existing quantum machine learning (ML) classifiers are designed for future, large-scale ideal quantum systems, and evaluated only on ideal quantum simulators. This is because loading data, training, and testing samples on existing noisy intermediate-scale quantum (NISQ) computers is challenging due to significant hardware errors [35], [7], [11], [31], [43], [30], [41]. Consequently, a classifier trained on a quantum simulator yields poor accuracy when run on existing NISQ computers because it does not mitigate the effect of errors on the real systems. *Currently, computer systems researchers do not have access to any practical and feasible capability to perform classification tasks on real quantum machines for exploration and improvement.*

Contributions. OPTIC specifically bridges this gap by proposing a binary classification framework for NISQ quantum machines. OPTIC builds upon three key ideas and insights to make quantum classification tasks practical on quantum computers:

(1) OPTIC uses a hybrid quantum-classical approach to mitigate the data loading challenge and erroneous nature of quantum machines. The training is partially done on a quantum simulator for faster training to overcome the challenge of loading large data. The training is also augmented using partial training on real error-prone quantum machines to account for the effects of errors and markedly improve the classification accuracy of the inference tasks on real quantum machines.

(2) OPTIC performs partial training and inference tasks on a pool of quantum machines instead of a single machine. While the idea of building a pool of classifiers has been applied for classical machine learning tasks, we found that applying this idea as-is in the quantum machine learning context does not provide any benefit without mitigating quantum-specific challenges that have no classical analogy. For example, to make the pool effective, OPTIC builds an optimizer to disperse the observed Pauli Z measurements for different quantum machines as real machines do not display ideal dispersion.

(3) OPTIC improves the effectiveness of the classification process by making each quantum computer's training and inference process accommodate for its inherent error characteristics using computer-optimal logistic functions. *As an outcome,* OPTIC *designs and implements a real quantum classifier circuit that works effectively on real IBM quantum machines.* OPTIC *is open-sourced at:* `https://doi.org/10.5281/zenodo.5758994`.

Limitations and Scope. While OPTIC achieves only moderate classification accuracy (≈70%) compared to its matured classical counterparts, these limitations are due to the NISQ technology capabilities. OPTIC's performance will only improve with the rapidly improving technology. OPTIC is the first step toward demonstrating how NISQ computers can be used to perform non-trivial ML classification tasks with reasonable accuracy by overcoming quantum roadblocks.

II. QUANTUM COMPUTING AND CLASSIFIER BACKGROUND

A Brief Overview of Quantum Computing. A qubit exists in a *superposition* of two basis states: $|0\rangle$ and $|1\rangle$, which can be represented as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where $|\psi\rangle$ is the qubit state, and α and β are amplitudes s.t. $\|\alpha\|^2 + \|\beta\|^2 = 1$. When a qubit is measured, its superposition collapses and it can be observed in state $|0\rangle$ with probability $\|\alpha\|^2$ and state $|1\rangle$ with probability $\|\beta\|^2$. Extending the one-qubit notation, the state of a system of two qubits is $|\psi\rangle = \alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \lambda|11\rangle$, where $\|\alpha\|^2 + \|\beta\|^2 + \|\gamma\|^2 + \|\lambda\|^2 = 1$. Each state's measurement probability is determined by the associated coefficient. Similarly, this notation can be extended to an n-qubit system.

Qubits initialized to state $|0\rangle$ can be put in desired superpositions by applying quantum gates. A general one-qubit gate is a rotation-gate with three parameters, $R(\theta, \phi, \delta)$, where the three parameters determine the angles of rotation that the qubit experiences along the three axes in a representational Bloch sphere. For example, an X gate applies a rotation of π about the x-axis, which can transfer a qubit from state $|0\rangle$ to state $|1\rangle$. Two-qubit gates can $entangle$ two qubits into an interdependent system. For example, a CX gate applies the X gate to the "target" qubit if the "control" qubit is in state $|1\rangle$. A general quantum algorithm or "circuit" is a set of one- and two-qubit gates that are applied to qubits in the $|0\rangle$ state in sequential order to put the multi-qubit system in a desired superposition. At the end of the circuit execution, the qubits are measured and the output probabilities of the states are obtained.

Quantum Variational Classifiers (QVCs). QVCs are a class of quantum circuits that have tunable and differentiable parameters (e.g., rotation angles of single-qubit gates) and can therefore be optimized for a specific objective [28]. The steps of designing a QVC are as following. First, the tunable parameters are initialized. Then, the circuit is executed on a quantum computer for the training dataset. Using an optimization approach (e.g., simulated annealing, Adam [24]), the new values of the tunable parameters are calculated on a classical computer and the QVC is updated with those parameters. This is repeated until the parameters are reasonably tuned.

Quantum variational classifiers are a type of QVCs that are used for data classification. They typically have repeated "layers" of tunable sub-circuits known as quantum neural networks (QNNs). To work on NISQ computers and demonstrate training speedup, QNNs need to be much smaller than their classical counterparts. Once a QNN is tuned and optimized, its circuit can be directly run on a quantum machine for inference. The output of the circuit can be used to classify the inference sample. Inference on a real quantum machine is faster than using a classical simulation of a quantum machine. However, the accuracy of inference on quantum machines is reduced because of the hardware errors on current quantum machines.

Quantum Hardware Errors. NISQ machines are prone to a variety of errors [35], [7], [11], [31], [43], [30], [41]. These include: (1) state preparation and measurement (SPAM) errors related to the imprecision in preparing the initial state of the qubit as well measuring the qubit; (2) gate operation errors related to the qubit being put in a slightly-off superposition – these errors can accumulate over the course of applying multiple circuit gates, resulting in the final system state being much off the intended state; (3) amplitude and phase dampening errors known as T1 and T2 coherence errors, respectively. Overall, these errors make it challenging to produce accurate inference on NISQ computers. This is the fundamental challenge that OPTIC tackles to enable an effective quantum classifier.

III. Related Work and Motivation for OPTIC

Prior related work can broadly be classified as following:

Quantum Variational Algorithms. Quantum variational algorithms have been proposed to solve problems spanning several different fields [34], [42], [29], [18], [28], [25], [46], [17], [19]. For example, the Variational Quantum Eigensolver (VQE) algorithm [34] has been shown to solve for the eigenvalues of the Hamiltonian of a quantum system (e.g., a molecule).

Quantum Variational Machine Learners. Hybrid quantum-classical computing has applications in supervised machine learning [9], [3], [40], [20], [21], [38], [45], [27], [8]. For example, Wilson et al. [45] make use of quantum circuits to transform classical input features into quantum features in a non-linear manner. Tomesh et al. [40] use coresets to load data on a quantum computer to perform k-means clustering using the Quantum Approximate Optimization Algorithm (QAOA).

Quantum Variational Classifiers. Quantum variational classifiers, the most relevant category in the context of OPTIC, are used to predict the category of a data sample using input features [39], [23], [6], [2], [14], [37], [12]. The most recent effort is by Capelletti et al. [12], which uses small 2-qubit circuits to classify simple datasets with very few features such as the Iris flower dataset [22], but does not consider the effect of hardware errors (leading to approx. 50% accuracy on real machines even on a smaller dataset than used by OPTIC). As another example, Adhikary et al. [2] propose a single-run training algorithm that uses data inputs in the form of multi-level quantum system to train the variational circuit in a single run. While theoretically-promising for future large-scale and error-free quantum computers, previous works are neither suited for nor evaluated on current noisy quantum computers; they do not co-design training and inference for NISQ machines.

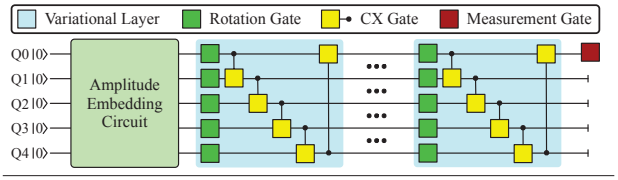

Fig. 1: OPTIC's classifier circuit structure. The horizontal lines show the qubits and the gates are executed from left to right.

IV. OPTIC's Design and Experimental Methodology

Testbed. Our quantum inference runs on five quantum computers available via the IBM cloud: Ourense, Vigo, Rome, Bogota, and Santiago (each five-qubit wide) [13]. These machines have quantum volumes of 8, 16, 32, 32, and 32, respectively. Quantum volume (QV) is a metric that encapsulates the performance of a quantum computer by considering the number of qubits on the computer, the gate and SPAM errors of all the qubits on a computer, and the inter-qubit connectivity and cross-talk [15]. The higher the quantum volume of a computer, the better its performance. We also perform experiments on the 15-qubit machine Melbourne computer to understand the effect of having more qubits. We use a widely-used Pennylane quantum simulator [10] running on a classical machine for training to ensure that the training is not affected by real errors.

Dataset and Pre-processing. We evaluate the performance of our classifier using the MNIST dataset. The MNIST dataset is a compilation of images of digits 0-9 in a variety of different handwritten fonts [26]. We use all images of digits 1 and 5 for our binary classification, although the results are not affected by the choice of digits. Due to current limited capability of NISQ computers, it is not possible to load, train and test all combinations of the digits. Principle Component Analysis (PCA) [1] is used to reduce the feature size from 784 (image size is 28×28 pixels) to 32, while maintaining an explained variance of 95%. This allows us to build a five-qubit variational classifier circuit ($2^5 = 32$) for the five-qubit machines. $\approx$ 12k images are used for training and $\approx$ 2k are used for testing with approximately equal number of images for each class.

Programming Environment. We train the circuit using the Adam optimization algorithm on Pennylane [10]. The circuits are run on IBM QX systems using the Qiskit [5] interface of

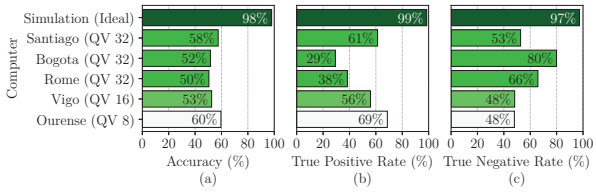

Fig. 2: Real computers achieve a lower accuracy and true positive/negative rates than ideal simulation. Higher quantum volumes (QV) do not necessarily lead to better performance.

Pennylane. Python-based scikit-learn library is used to optimize the parameters of the logistics functions via dual annealing [33].

Metrics. OPTIC is evaluated using standard classifier metrics: accuracy, true positive rate (TPR), true negative rate (TNR), F_1 score, and the absolute difference between TPR and TNR.

V. OPTIC'S DESIGN AND FRAMEWORK

Designing OPTIC's Quantum Variational Classifier. OPTIC takes a simplistic approach by developing a quantum variational circuit to use for classification. Prior to adding the variational layers to the circuit, the data (sample image) has to be loaded in the circuit (Fig. 1). To solve the data loading challenges, OPTIC employs amplitude embedding method, similar to [36]. Amplitude embedding embeds the sample in the form of the amplitudes of the quantum states. For example, a five-qubit quantum system has 32 states and each of these states has a corresponding amplitude: $|\psi\rangle = \alpha_1 |00\ldots0\rangle + \alpha_2 |00\ldots1\rangle + \cdots + \alpha_{32} |11\ldots1\rangle$. For such a system, up to 32 features can be embedded into the 32 amplitudes of the states. Note that all features have to be numericalized and normalized to ensure that the norm of the amplitudes adds up to one. Amplitude embedding is applied to qubits initialized to the $|0\rangle$ state. At the end of the embedding, the features manifest as a multi-qubit state. The variational logic can then be applied to this state.

All subsequent layers begin with a single rotational gate on each of the qubits. A rotational gate has three parameters: one angle corresponding to rotations about each of the three axes on the Bloch sphere. Next, qubit entanglements are established by performing CX operations on consecutive pairs of qubits. Following this step, the next layer (with the same structure) repeats for up to four layers. At the end of the circuit, one qubit is measured to classify the sample (Fig. 1).

The Pauli Z measurement measures the qubit state relative to the z-axis of the Bloch sphere. The expectation value of the measurement is obtained by running the circuit multiple times and getting the probabilities of the $|0\rangle$ state and the $|1\rangle$ state. The expectation value of the Pauli Z measurement is -1 for state $|0\rangle$ and 1 for state $|1\rangle$. Rest of the superpositions result in a value between -1 and 1. Thus, binary classification can be performed by counting all measurements below 0 as class -1 and above 0 as class 1. Due to timing constraints, data loading, and hardware errors, model training happens on a quantum simulator and inference on real quantum machines.

Why not make the circuit deeper and complex with more parameters? While OPTIC's trained model achieves nearly 98% accuracy on a quantum computing simulator, its accuracy drops to around 58% on real quantum machines (Fig. 2). In fact, the accuracy level is not a monotonic function of the quantum volume of the machine. A natural tendency in classical machine learning is to make networks more complex, bigger and deeper. Via our iterative design and experimentation, we learned that quantum circuits need to be relatively simpler. For example, we discovered that more layers and tunable parameters increased the accuracy to 99% on the simulator, but on the real system the accuracy dropped to around 52%. Developing a larger 15-qubit quantum circuit for the Melbourne machine also only achieved 52% accuracy (as shown in Sec. VI). Circuits with higher number of qubits and gates are more likely to suffer from decoherence and SPAM errors. In fact, even our simpler circuit achieves only ≈60% accuracy at maximum.

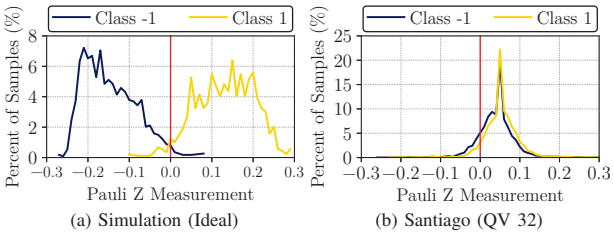

Fig. 3: PMFs of the Pauli Z measurements. The PMFs are well split when using a simulation, but indistinguishable on real devices due to the high degree of hardware errors.

To investigate this further, we analyzed the values of the Pauli Z measurements of the output qubit in simulation compared to a real computer. Fig. 3 shows the probability mass function (PMF) of the Pauli Z measurements of the test images. Recall that if the Pauli Z measurement is less than 0, the sample is classified as class -1, and if it is greater than 0, the sample is classified as 1. The figure shows that in the ideal simulation results (Fig. 3(a)), there is a distinctive split between the two classes: most of the class -1 samples fall below 0 and most of the class 1 samples fall above 0. On the other hand, Fig. 3(b) shows that on real computers (e.g., Santiago), there is no distinctive difference between the two distributions due to the errors of quantum hardware. This poor accuracy on error-prone quantum machines reveals the need for the model training process to be embedded with the effects of the hardware errors.

Key Ingredient 1: Hybrid Quantum-Classical Training. OPTIC employs a hybrid training process where a small subset of the data is used to trained on real quantum computers. Recall that due to time constraints and data load challenges, the full training cannot be performed on quantum computers at this stage of technological development. Partial training on quantum computers helps capture the effects of hardware errors. However, each quantum computer can have varying spatial and temporal error characteristics and training on a single quantum machine may introduce machine-specific bias in the model.

Key Ingredient 2: Pool of Classifiers. To address the above challenge, OPTIC implements a voting mechanism among a pool of classifiers where each machine in the pool participates

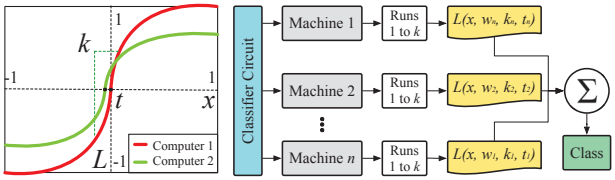

Fig. 4: (a) Parameters that determine the shape and properties of logistic functions. (b) Representation of how an inference sample is run on multiple machines, and how the machine-specific logistic functions are used to obtain the final class.

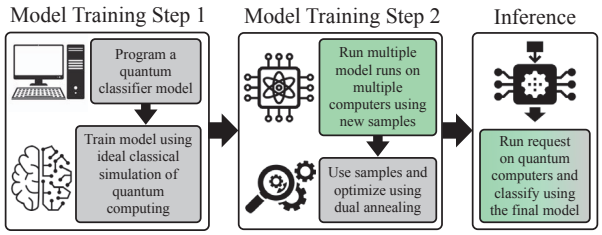

Fig. 5: Visual representation of the three-step training and inference procedure of OPTIC to execute classification workloads on real quantum machines in the current NISQ era.

in the partial training process. Then, it performs inference on all quantum machines and takes an average of the outputs. For instance, if one computer is biased toward classifying more images as class -1 (e.g., Bogota has a high TNR of 80% in Fig. 2 because it classifies most images as class -1 and therefore, has a very low TPR), then, getting the opinion of another computer that is less biased or biased in a different manner can be helpful (e.g., Ourense has a high TPR of 69%, which can help reduce the bias of Bogota). This is similar to the concept of ensembles of learners where a few learners participate and improve the accuracy of classification. However, in this case, the same variational classifier model is run on each computer. Individual computers act as different learners due to their unique error properties without the need to train different variational classifier models. Note that OPTIC also performs multiple runs on the same computer to get stable results.

Including multiple machines with different error characteristics helps improve the classification accuracy, but we observed that it may increase the bias in certain cases if multiple machines are making similar classification mistakes (increase in the difference between TPR and TNR, Sec. VI). To address this challenge, OPTIC implements an optimizer that disperses the observed Pauli Z measurements near 0 toward the extremes in a machine-specific way, as described next.

Key Ingredient 3: Tuning The Classifier For Each Machine. OPTIC uses a logistic function to transform the Pauli Z measurements near 0 to a value closer to -1 or 1 (Fig. 4(a)). A logistic function is defined as $L(x, w_i, k_i, t_i) = w_i \frac{2}{1+e^{-k(x-t_i)}}$. Here, x is the measurement value being transformed, w_i is the weight placed on the runs of the i^{th} computer, k_i is the parameter that determines the steepness of the "S" curve, and t_i is the parameter that determines the classification threshold; w_i, k_i, and t_i are computer-specific parameters. w_i has a value between -1 and 1. A computer with high prediction error would get a close-to-zero weight and a computer with low error would get a weight close to -1 or 1. k_i determines how far the Pauli Z measurement has to be from 0 for it to be pulled up to 1 or pulled down to -1. A higher k_i makes the curve more steep, thus indicating a higher confidence in classification even if the value is close to 0. Lastly, t_i determines where the inflection point of the logistic function should be. As an example, the hardware errors could shift all samples in the class 1 direction. In this case, t_i should be set to a slightly positive value to accommodate the shift. Once the measurements are transformed using the logistic functions for all runs across all computers, they can be summed up (Fig. 4(b)). If the sum is below 0, then class -1 is assigned; class 1 is assigned otherwise.

OPTIC determines the function parameter values by framing it as a black-box optimization problem: given the Pauli Z measurements of different image samples, determine the logistic function parameter values that maximize the training accuracy of the classification and minimize the difference between the TPR and the TNR: $minimize\ \big(|TPR - TNR| - accuracy\big)$. Note that the minimization term regarding the difference between TPR and TNR is added to ensure that the accuracy is not improved at the cost of large difference in the TPR and the TNR (as we observed experimentally with a naive solution of pool of multiple inferences). Note that we chose this metric for optimization instead of a metric like the F_1 score, because we found that it is more efficient to optimize and also improves the F_1 score by 12% points, as we show next in Sec. VI. We use dual annealing to perform this optimization [16].

Putting It All Together. As shown in Fig. 5, OPTIC begins by designing and programming a quantum variational classifier circuit consisting of data embedding followed by multiple layers of networks. This model is then trained using an ideal simulation of quantum computing. Training samples are then run on different quantum computers to make the training process embed the effects of hardware errors. These samples are used to train and optimize a logistic function for each quantum machine using a dual annealing approach. OPTIC weighs and sums the output of these logistic functions to predict the class. Once, the second stage of training is completed, the model can be used for inference.

VI. RESULTS AND ANALYSIS

OPTIC variants provide significant boost in the classification quality on real NISQ-era quantum machines. Fig. 6 shows the various figures of merit for OPTIC variants compared to the baseline (a quantum model trained on a simulator and run on the least erroneous machine, Santiago (QV 32)). We make several observations. First, OPTIC's approach of leveraging a pool of machines for inference is more effective than the baseline. This is true in terms of all the figures of merit (F_1-score, accuracy, and $|TPR - TNR|$). In particular, OPTIC's improves the classification accuracy by 12% points

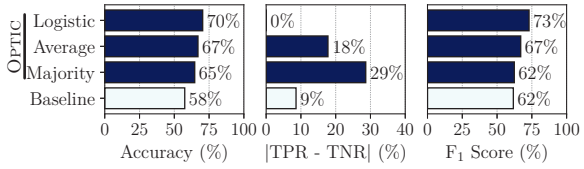

Fig. 6: OPTIC provides the highest accuracy and F_1 score, and the lowest $|TPR - TNR|$, as compared to other techniques.

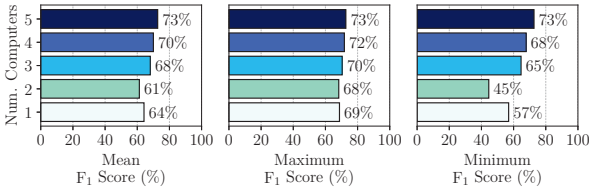

Fig. 7: OPTIC becomes more effective with inclusion of more machines and outperforms the best combination of smaller sets.

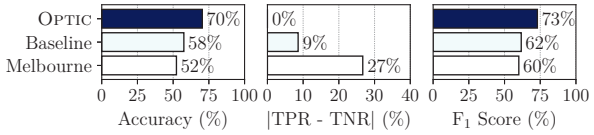

Fig. 8: The 15-qubit Melbourne computer performs worse than the 5-qubit baseline computer due to the higher hardware error.

and F_1 score by 11% points over the baseline. Second, using a pool of machines is useful, but its effectiveness is further improved by engaging the quantum machine in training and tuning each machine to accommodate the inherent effect of its error characteristics on the classification process (logistic function optimization). In particular, using the logistic function optimization improves the accuracy and removes the bias in the classification (the absolute difference between TPR and TNR is near zero) as expected because of the construction of the optimization function. Third, we note that the classification quality cannot be simply improved by taking the majority vote since it does not capture the contribution of the individual machines, and may lead to higher difference between TPR and TNR. This is why, it is critical to tune each machine individually using the logistic function optimizer.

OPTIC becomes increasingly effective with inclusion of more machines, and outperforms the best combination of smaller set of machines. To better understand the impact of real computers on OPTIC, we look at the performance of OPTIC with up to five computers (Fig. 7). Choosing fewer than five machines provides the opportunity to select different combinations of computers. For example, if two computers are selected, these computers can be Ourense and Vigo or Ourense and Rome. Because it is not clear which combination is the best before performing the experiments (recall that a computer with a higher quantum volume does not necessarily perform better), we consider all combinations and present the mean F_1-score of the combinations in Fig. 7(a), the maximum F_1-score in Fig. 7(b), and the minimum F_1-score in Fig. 7(c).

The results show that using all the five machines results in the best classification quality (F_1-score 73%) compared to smaller subsets of machines. In general, the results are worse with fewer number of machines. For example, for a three-computer selection, the mean F_1-score is 68%. The maximum F_1-score for the best combination of three machines is 69%, but this combination is not known apriori and some combinations have an F_1-score as low as 65%. In fact, the set of best and worst performing combination changes significantly as the sizes of the combinations change. For example, for a single-computer selection, Bogota performs the worst. But, for a four-computer selection, the best combination includes the worst-performing computer Bogota. Overall, OPTIC's approach of including all machines is more productive than guessing which machines are best or attempting to form a smaller set. This is because more machines provide higher chances of removing the bias from individual error characteristics.

Effect of larger circuits and more samples. Finally, we briefly discuss two important results. First, we found that running a 15-qubit version of OPTIC on the Melbourne machine, does not increase the classification quality, as shown in Fig. 8. The 15-qubit variational circuit was trained in the same manner and using the same variational structure as our 5-qubit variational circuit, using the Adam optimizer. This model also has an ideal simulation accuracy of 98%. But, when run on a real 15-qubit computer, the accuracy drops to 52% because variational circuits running on larger number of qubits are likely to suffer from more errors as the degree of error increases proportionally to the number of qubits. This offsets the benefit of running on larger machines at this point in the evolution of NISQ technology. We experimented with an increasingly larger circuit size on the 15-qubit machine and found that the accuracy drops with increasing circuit size. This shows another aspect of OPTIC's usability: in terms of the training overhead, building a diverse and tuned pool of smaller-size models is linearly scalable and faster than training a single 15-qubit model, which has an exponentially-scaling training time on the simulator. Our results also show that there is no benefit to undertaking this overhead as the accuracy does not improve; it only drops. Using smaller-sized models is more beneficial in terms of both the training overhead and the model's predictive performance.

Second, we found that increasing the number of runs beyond five per computer for each tested sample does not increase the classification quality of OPTIC. This is because five runs already amount to a large number of execution samples (5120 execution samples; 1024 per run) and further runs do not remove the error bias in the results.

VII. CONCLUSION

We proposed OPTIC to design quantum circuits that run on real NISQ computers to perform quantum binary classification tasks. OPTIC leverages simple variational circuit structures, divergent error characteristics of different computers, and logistic function optimization to achieve high accuracy. OPTIC improves classification accuracy by 12% points beyond running on just one computer, all the while ensuring that the true positive rate and the true negative rate are similar.

ACKNOWLEDGEMENT

We are thankful for the thoughtful feedback from our anonymous reviewers and the support from Northeastern University, NSF Award 1910601, and the Massachusetts Green High Performance Computing Center (MGHPCC) facility. We acknowledge the use of the IBM Q for this work. The views expressed are those of the authors and do not reflect the official policy or position of IBM or the IBM Q team.

REFERENCES

[1] Hervé Abdi and Lynne J Williams. Principal Component Analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459, 2010.

[2] Soumik Adhikary, Siddharth Dangwal, and Debanjan Bhowmik. Supervised Learning with a Quantum Classifier using Multi-Level Systems. *Quantum Information Processing*, 19(3):89, 2020.

[3] Mahabubul Alam, Abdullah Ash-Saki, and Swaroop Ghosh. Accelerating Quantum Approximate Optimization Algorithm using Machine Learning. In *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 686–689. IEEE, 2020.

[4] Mahabubul Alam, Abdullah Ash-Saki, and Swaroop Ghosh. Circuit Compilation Methodologies for Quantum Approximate Optimization Algorithm. In *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pages 215–228. IEEE, 2020.

[5] Gadi Aleksandrowicz et al. Qiskit: An Open-Source Framework for Quantum Computing. 16, 2019.

[6] Ismael CS Araujo and Adenilton J da Silva. Quantum Ensemble of Trained Classifiers. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

[7] Abdullah Ash-Saki, Mahabubul Alam, and Swaroop Ghosh. QURE: Qubit Re-Allocation in Noisy Intermediate-Scale Quantum Computers. In *Proceedings of the 56th Annual Design Automation Conference 2019*, pages 1–6, 2019.

[8] Johannes Bausch. Recurrent Quantum Neural Networks. *Advances in Neural Information Processing Systems*, 33, 2020.

[9] Marcello Benedetti, Erika Lloyd, Stefan Sack, and Mattia Fiorentini. Parameterized Quantum Circuits as Machine Learning Models. *Quantum Science and Technology*, 4(4):043001, 2019.

[10] Ville Bergholm et al. Pennylane: Automatic Differentiation of Hybrid Quantum-Classical Computations. *arXiv preprint arXiv:1811.04968*, 2018.

[11] Debjyoti Bhattacharjee, Abdullah Ash Saki, Mahabubul Alam, Anupam Chattopadhyay, and Swaroop Ghosh. MUQUT: Multi-Constraint Quantum Circuit Mapping on NISQ Computers. In *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 1–7. IEEE, 2019.

[12] William Cappelletti, Rebecca Erbanni, and Joaquín Keller. Polyadic Quantum Classifier. In *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pages 22–29. IEEE, 2020.

[13] Davide Castelvecchi. IBM's Quantum Cloud Computer Goes Commercial. *Nature News*, 543(7644):159, 2017.

[14] Samuel Yen-Chi Chen, Chih-Min Huang, Chia-Wei Hsing, and Ying-Jer Kao. Hybrid Quantum-Classical Classifier based on Tensor Network and Variational Quantum Circuit. *arXiv preprint arXiv:2011.14651*, 2020.

[15] Andrew W Cross, Lev S Bishop, Sarah Sheldon, Paul D Nation, and Jay M Gambetta. Validating Quantum Computers using Randomized Model Circuits. *Physical Review A*, 100(3):032328, 2019.

[16] Kathryn Anne Dowsland and Jonathan Thompson. Simulated Annealing. *Handbook of natural computing*, pages 1623–1655, 2012.

[17] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A Quantum Approximate Optimization Algorithm. *arXiv preprint arXiv:1411.4028*, 2014.

[18] Pranav Gokhale et al. Partial Compilation of Variational Algorithms for Noisy Intermediate-Scale Quantum Machines. In *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, pages 266–278, 2019.

[19] Stuart Hadfield, Zhihui Wang, Bryan O'Gorman, Eleanor G Rieffel, Davide Venturelli, and Rupak Biswas. From the Quantum Approximate Optimization Algorithm to a Quantum Alternating Operator Ansatz. *Algorithms*, 12(2):34, 2019.

[20] Aram W Harrow. Small Quantum Computers and Large Classical Data Sets. *arXiv preprint arXiv:2004.00026*, 2020.

[21] Vojtěch Havlíček, Antonio D Córcoles, Kristan Temme, Aram W Harrow, Abhinav Kandala, Jerry M Chow, and Jay M Gambetta. Supervised Learning with Quantum-Enhanced Feature Spaces. *Nature*, 567(7747):209–212, 2019.

[22] Patrick S Hoey. Statistical Analysis of the Iris Flower Dataset. *University of Massachusetts At Lowell, Massachusetts*, 2004.

[23] Iordanis Kerenidis and Alessandro Luongo. Classification of MNIST Dataset with Quantum Slow Feature Analysis.

[24] Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[25] Jonas M Kübler, Andrew Arrasmith, Lukasz Cincio, and Patrick J Coles. An Adaptive Optimizer for Measurement-Frugal Variational Algorithms. *Quantum*, 4:263, 2020.

[26] Yann LeCun. The MNIST Database of Handwritten Digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[27] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum Principal Component Analysis. *Nature Physics*, 10(9):631–633, Jul 2014.

[28] Jarrod R McClean, Jonathan Romero, Ryan Babbush, and Alán Aspuru-Guzik. The Theory of Variational Hybrid Quantum-Classical Algorithms. *New Journal of Physics*, 18(2):023023, 2016.

[29] Nikolaj Moll et al. Quantum Optimization using Variational Algorithms on Near-Term Quantum Devices. *Quantum Science and Technology*, 3(3):030503, 2018.

[30] Tirthak Patel, Baolin Li, Rohan Basu Roy, and Devesh Tiwari. UREQA: Leveraging Operation-Aware Error Rates for Effective Quantum Circuit Mapping on NISQ-Era Quantum Computers. In *2020 USENIX Annual Technical Conference (USENIX ATC 20)*, pages 705–711, 2020.

[31] Tirthak Patel, Abhay Potharaju, Baolin Li, Rohan Roy, and Devesh Tiwari. Experimental Evaluation of NISQ Quantum Computers: Error Measurement, Characterization, and Implications. In *2020 SC20: International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, pages 636–650. IEEE Computer Society, 2020.

[32] Tirthak Patel and Devesh Tiwari. DisQ: A Novel Quantum Output State Classification Method on IBM Quantum Computers using Openpulse. In *Proceedings of the 39th International Conference on Computer-Aided Design*, pages 1–9, 2020.

[33] Fabian Pedregosa et al. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[34] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J Love, Alán Aspuru-Guzik, and Jeremy L O'brien. A Variational Eigenvalue Solver on a Photonic Quantum Processor. *Nature communications*, 5:4213, 2014.

[35] John Preskill. Quantum Computing in the NISQ Era and Beyond. *Quantum*, 2:79, 2018.

[36] Maria Schuld. *Supervised Learning with Quantum Computers*. Springer, 2018.

[37] Maria Schuld et al. Circuit-Centric Quantum Classifiers. *Physical Review A*, 101(3):032308, 2020.

[38] Maria Schuld and Nathan Killoran. Quantum Machine Learning in Feature Hilbert Spaces. *Physical review letters*, 122(4):040504, 2019.

[39] Maria Schuld and Francesco Petruccione. Quantum Ensembles of Quantum Classifiers. *Scientific reports*, 8(1):1–12, 2018.

[40] Teague Tomesh, Pranav Gokhale, Eric R Anschuetz, and Frederic T Chong. Coreset Clustering on Small Quantum Computers. *arXiv preprint arXiv:2004.14970*, 2020.

[41] Alvaro Velasquez, Sumit Kumar Jha, Rickard Ewetz, and Susmit Jha. Automated Synthesis of Quantum Circuits using Symbolic Abstractions and Decision Procedures. In *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5. IEEE, 2021.

[42] Dave Wecker, Matthew B Hastings, and Matthias Troyer. Progress Towards Practical Quantum Variational Algorithms. *Physical Review A*, 92(4):042303, 2015.

[43] Robert Wille, Lukas Burgholzer, and Alwin Zulehner. Mapping Quantum Circuits to IBM QX Architectures using the Minimal Mumber of SWAP and H Operations. In *2019 56th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2019.

[44] Robert Wille, Stefan Hillmich, and Lukas Burgholzer. JKQ: JKU Tools for Quantum Computing. In *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pages 1–5. IEEE, 2020.

[45] CM Wilson, JS Otterbach, Nikolas Tezak, RS Smith, AM Polloreno, Peter J Karalekas, S Heidel, M Sohaib Alam, GE Crooks, and MP da Silva. Quantum Kitchen Sinks: An Algorithm for Machine Learning on Near-Term Quantum Computers. *arXiv preprint arXiv:1806.08321*, 2018.

[46] Xiao Yuan, Suguru Endo, Qi Zhao, Ying Li, and Simon C Benjamin. Theory of Variational Quantum Simulation. *Quantum*, 3:191, 2019.