

Quantum Unsupervised Learning on a Superconducting Processor

Rupak Chatterjee,^{*} Abhijat Sarma,[†] and Ting Yu[‡]

*Center for Quantum Science and Engineering
Department of Physics, Stevens Institute of Technology,
Castle Point on the Hudson, Hoboken, NJ 07030*

Various machine learning algorithms, namely K-means clustering and Support Vector Machines (SVMs), perform well on classifying and identifying patterns in wide ranges of datasets due to their versatility. However, as one increases the amount of data points and features, the computation time for training and using these statistical models grows intractably. The ability to store information compactly in quantum states suggests that quantum computers can be used to optimize such algorithms. Here, we propose and experimentally implement on superconducting qubits a quantum analogue to K-means clustering, and compare it to a previously developed quantum SVM. We apply the algorithm to the standard Wine and Iris datasets, and find that our quantum clustering algorithm can produce linear boundaries to a high degree of accuracy reflecting the true boundaries in the data.

I. Introduction

Machine learning offers solutions to several classes of problems unreachable through conventional computing means. For example, solutions to classification problems and regression of large datasets based on machine learning techniques are in general much more powerful than previously available solutions. These algorithms suffer in that they grow polynomially with the size and dimension of the data, which leads to substantial run times when dealing with large datasets, coined "big data". The ability of data to be more efficiently stored and manipulated in quantum states has recently lead to the proposal of several quantum algorithms for machine learning [1–9]. Building upon [10] and [11], in this paper, we develop a hybrid K -means clustering algorithm in order to identify clusters in data more quickly than possible with similar purely classical algorithms. The algorithm relies on a distance measure, here taken to be Euclidean square distance. This distance can be calculated efficiently on a quantum computer, as we will show, in order to speed up the algorithm as a whole. We compare our quantum K-means clustering algorithm to a classical one, in terms of accuracy in solving trinary classification problems on a real dataset. Additionally, we include a multiclass extension of the quantum SVM introduced in [12] in order to better understand differences between different quantum machine learning algorithms.

II. Hybrid K -means Clustering

In machine learning theory, it is often mathematically convenient to consider the data as encoded in a vector. Therefore, each data point with P different variables (fea-

tures), can be encoded as a P -dimensional feature vector. The total dataset is therefore a set of vectors in P -dimensional space, known as input space. K -means clustering is an unsupervised learning algorithm which considers the problem of partitioning N feature vectors \mathbf{X}^i into K subsets, or *clusters*. The algorithm seeks to find the K clusters which minimize the dissimilarity between each cluster's members. Local minimization can be iteratively found with the standard K -means algorithm, which relies on a distance measure, usually taken to be Euclidean square distance.

Sampling and estimating Euclidean distances between post-processed vectors on a classical computer is known to be exponentially hard. For big data sets, the algorithm becomes slow as convergence relies on repeated calculations of this distance measure. In the next section, we propose and implement a quantum algorithm for calculating the Euclidean distance, which can be shown to be faster than the classical counterpart [10]. Note that although we use Euclidean distance in our clustering algorithm, one could use any distance measure. The Euclidean distance is desirable for our purposes because it assigns linear boundaries between clusters, and is guaranteed to converge, assuming no error in calculation (INSERT CITATION). For non-linear boundaries, one could use a corresponding distance measure.

Here we propose a hybrid algorithm which calculates cluster centroids and assigns features classically, but computes Euclidean (square) distance with a quantum circuit. To develop our quantum algorithm for estimation of Euclidean distance, we first must introduce the *swap test*. (INSERT CITATION) Consider a state

$$|0\rangle \otimes |\psi\rangle \otimes |\varphi\rangle, \quad (1)$$

consisting of an ancillary qubit and two equal-qubit states which we wish to find the overlap of. Perform a Hadamard transformation \mathbf{H} on the ancillary followed by a **FREDKIN** gate (also known as a *controlled swap*)

^{*} Rupak.Chatterjee@stevens.edu

[†] absarma@ctemc.org

[‡] ting.yu@stevens.edu

on this state,

$$\begin{aligned}
& \text{FREDKIN}(\mathbf{H} \otimes \mathbf{I} \otimes \mathbf{I}) |0\rangle \otimes |\psi\rangle \otimes |\varphi\rangle \\
&= (|0\rangle \langle 0| \otimes \mathbf{I} \otimes \mathbf{I} + |1\rangle \langle 1| \text{SWAP}) \frac{1}{\sqrt{2}} [|0\rangle \otimes |\psi\rangle \otimes |\varphi\rangle \\
&\quad + |1\rangle \otimes |\psi\rangle \otimes |\varphi\rangle] \\
&= \frac{1}{\sqrt{2}} [|0\rangle \otimes |\psi\rangle \otimes |\varphi\rangle + |1\rangle \otimes |\varphi\rangle \otimes |\psi\rangle]
\end{aligned} \tag{2}$$

Applying the Hadamard transformation once more to the ancillary qubit gives

$$\begin{aligned}
|\Psi\rangle &= (\mathbf{H} \otimes \mathbf{I} \otimes \mathbf{I}) \frac{1}{\sqrt{2}} [|0\rangle \otimes |\psi\rangle \otimes |\varphi\rangle \\
&\quad + |1\rangle \otimes |\varphi\rangle \otimes |\psi\rangle] \\
&= \frac{1}{2} [(|0\rangle + |1\rangle) \otimes |\psi\rangle \otimes |\varphi\rangle \\
&\quad + (|0\rangle - |1\rangle) \otimes |\varphi\rangle \otimes |\psi\rangle] \\
&= \frac{1}{2} |0\rangle \otimes [|\psi\rangle \otimes |\varphi\rangle + |\varphi\rangle \otimes |\psi\rangle] \\
&\quad + \frac{1}{2} |1\rangle \otimes [|\psi\rangle \otimes |\varphi\rangle - |\varphi\rangle \otimes |\psi\rangle]
\end{aligned} \tag{3}$$

Finally, measure the state of the ancillary qubit. The probability of measuring $|0\rangle$ is given by

$$\begin{aligned}
& \langle \Psi | (|0\rangle \langle 0| \otimes \mathbf{I} \otimes \mathbf{I}) | \Psi \rangle \\
&= \frac{1}{4} \{ \langle \psi | \otimes \langle \varphi | + \langle \varphi | \otimes \langle \psi | \} [|\psi\rangle \otimes |\varphi\rangle + |\varphi\rangle \otimes |\psi\rangle] \\
&= \frac{1}{2} + \frac{1}{4} [(\langle \psi | \otimes \langle \varphi |)(|\varphi\rangle \otimes |\psi\rangle) \\
&\quad + (\langle \varphi | \otimes \langle \psi |)(|\psi\rangle \otimes |\varphi\rangle)] \\
&= \frac{1}{2} + \frac{1}{4} [\langle \psi | \varphi \rangle \langle \varphi | \psi \rangle + \langle \varphi | \psi \rangle \langle \psi | \varphi \rangle]
\end{aligned} \tag{4}$$

Therefore, as first shown in [13],

$$P[|0\rangle] = \langle \Psi | (|0\rangle \langle 0| \otimes \mathbf{I} \otimes \mathbf{I}) | \Psi \rangle = \frac{1}{2} + \frac{1}{2} |\langle \psi | \varphi \rangle|^2 \tag{5}$$

The swap test therefore allows us to experimentally determine the overlap between two states $|\psi\rangle$ and $|\varphi\rangle$. This will be integral in calculating the Euclidean distance. In order to calculate Euclidean distance, we must first encode our feature vectors into Hilbert Space. Using the base-2 *bit string configuration* $|p\rangle = |p_{n-1}p_{n-2}\dots p_1p_0\rangle$, $p = 2^0p_0 + 2^1p_1 + \dots 2^{n-1}p_{n-1}$, $P = 2^n$,

$$|\mathbf{X}^i\rangle = \frac{1}{|\mathbf{X}^i|} \sum_{p=1}^P X_p^{(i)} |p\rangle \tag{6}$$

Note that the overlap between two of these states recovers the usual vector dot product, namely

$$\langle \mathbf{X}^i | \mathbf{X}^j \rangle = \frac{1}{|\mathbf{X}^i| |\mathbf{X}^j|} \sum_{p=1}^P X_p^{(i)} X_p^{(j)} = \frac{1}{|\mathbf{X}^i| |\mathbf{X}^j|} \mathbf{X}^i \cdot \mathbf{X}^j \tag{7}$$

Next, we construct the following states

$$\begin{aligned}
|\psi\rangle &= \frac{1}{\sqrt{2}} [|0\rangle \otimes |\mathbf{X}^i\rangle + |1\rangle \otimes |\mathbf{X}^j\rangle] \\
|\varphi\rangle &= \frac{1}{\sqrt{Z}} [|\mathbf{X}^i\rangle |0\rangle - |\mathbf{X}^j\rangle |1\rangle]
\end{aligned} \tag{8}$$

where $Z = |\mathbf{X}^i|^2 + |\mathbf{X}^j|^2$ is a normalization constant. Performing a swap test between the first qubit of $|\psi\rangle$ and the state $|\varphi\rangle$ will give the overlap $|\langle \psi | \varphi \rangle|^2$ of the two states. To calculate this overlap, we first calculate the partial overlaps $\langle \psi | \varphi \rangle$ and $\langle \varphi | \psi \rangle$, which reduce to

$$\begin{aligned}
\langle \psi | \varphi \rangle &= \frac{1}{\sqrt{2Z}} [|\mathbf{X}^i\rangle \langle \mathbf{X}^i| - |\mathbf{X}^j\rangle \langle \mathbf{X}^j|] \\
\langle \varphi | \psi \rangle &= \frac{1}{\sqrt{2Z}} [|\mathbf{X}^i\rangle |\mathbf{X}^i\rangle - |\mathbf{X}^j\rangle |\mathbf{X}^j\rangle]
\end{aligned} \tag{9}$$

Therefore, the complete overlap between these states $|\langle \psi | \varphi \rangle|^2 = \langle \psi | \varphi \rangle \langle \varphi | \psi \rangle$ is

$$\begin{aligned}
|\langle \psi | \varphi \rangle|^2 &= \langle \psi | \varphi \rangle \langle \varphi | \psi \rangle \\
&= \frac{1}{2Z} \{ |\mathbf{X}^i|^2 + |\mathbf{X}^j|^2 \\
&\quad - |\mathbf{X}^i\rangle |\mathbf{X}^j\rangle \langle \mathbf{X}^i | \mathbf{X}^j\rangle - |\mathbf{X}^j\rangle |\mathbf{X}^i\rangle \langle \mathbf{X}^j | \mathbf{X}^i\rangle \} \\
&= \frac{1}{2Z} \{ |\mathbf{X}^i|^2 + |\mathbf{X}^j|^2 - 2\mathbf{X}^i \cdot \mathbf{X}^j \} \\
&= \frac{1}{2Z} \{ |\mathbf{X}^i - \mathbf{X}^j|^2 \}
\end{aligned} \tag{10}$$

The classical Euclidean distance is therefore proportional to the overlap of the states specified in (9). Following [10], the quantum algorithm for calculating the Euclidean distance is to perform a swap test between those two states such that

$$\begin{aligned}
P[|0\rangle] &= \frac{1}{2} + \frac{1}{2} |\langle \psi | \varphi \rangle|^2 \\
&= \frac{1}{2} + \frac{1}{4Z} |\mathbf{X}^i - \mathbf{X}^j|^2
\end{aligned} \tag{11}$$

or

$$|\mathbf{X}^i - \mathbf{X}^j|^2 = Z(4P[|0\rangle] - 2). \tag{12}$$

III. Experimental Implementation of Hybrid Clustering Algorithm

We implement this formula for calculation of the Euclidean distance between arbitrary-dimensional feature vectors in our clustering algorithm in order to achieve a speed-up over similar classical algorithms using the open-source IBMQ software for creating and running quantum circuits. We achieved this by utilizing the Python framework *Qiskit* developed by IBMQ, coding a modular quantum algorithm for distance calculation and injecting it into a rudimentary *K*-means algorithm we developed. The *Qiskit* module offers the ability to run circuits on

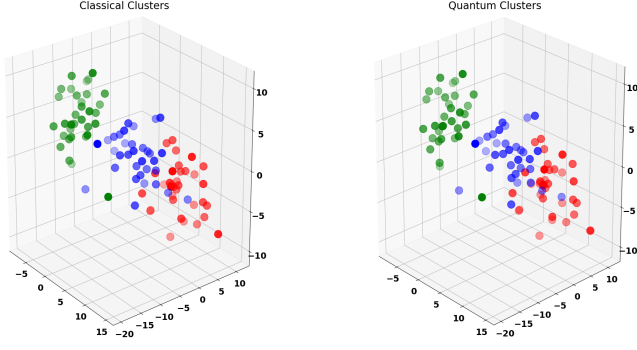


FIG. 1. Identified clusters by both algorithms in an ad-hoc dataset. The dataset was constructed by randomly assigning three cluster centroids with feature means ranging from $[-10, 10]$. The datapoints were then randomly generated using a standard deviation of 3.0 away from the cluster centroid, corresponding to a moderately noisy model. Note that the data contains 5 feature dimensions, while only 3 are pictured. Both algorithms perfectly cluster the data, as shown above.

quantum computers operated by IBM based on superconducting transmon qubits and Josephson junctions, as well as IBM’s quantum circuit simulator designed to accurately simulate typical noisy transmons. We ran our algorithm on the simulator as the real machines have limitations on how many executions can be conducted as well as the number of available qubits.

A. Clustering

To test our algorithm we first use the standard *Scikit* function *make_blobs* to generate clustered data. We test the algorithm on 4 datasets of 100 5-dimensional feature vectors with different levels of noise and clustering, represented numerically by increasing standard deviations in datapoints from the cluster centroids. We also run the classical K-means algorithm on the same data for the sake of comparison. Figure 1 shows the clustering of both algorithms on one dataset, while Figure 2 shows accuracy vs. standard deviation values for each algorithm. It is evident that the quantum algorithm performs very similarly to the classical analogue, only suffering a slight performance dropoff for highly noisy data. All simulations were run with *Qiskit Aer* on a Macbook Pro.

B. Classification

Here, we apply our hybrid algorithm to real datasets. We consider a trinary classification problem on the standard Wine and Iris datasets provided by the *Scikit* module for machine learning on Python. We feed the test data into our clustering algorithm, and then compare the generated clusters with the true class boundaries. We run

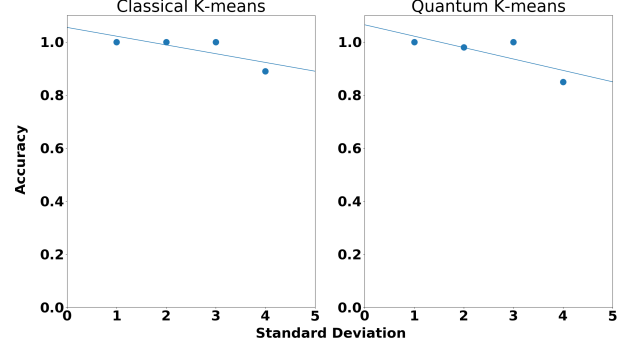


FIG. 2. Cluster accuracy vs. standard deviation of data. As the standard deviation of each generated datapoint from its cluster centroid increases, the model becomes more and more noisy. Each algorithm was ran on four datasets of standard deviations ranging from 1.0 to 4.0. A line of best fit for each algorithm is also shown. Both algorithms perform near perfectly below standard deviations of 3.0, and begin to suffer a performance dropoff above that.

the algorithm on each dataset with 30 feature vectors. For the Wine dataset, we find a classification success of 100% in 5 feature dimensions. For Iris, we find a success rate of 70% in 4 dimensions. In both cases, the algorithm performs similarly to the classical analogue in terms of accuracy. We also ran the algorithm on the Wine dataset for 100 feature vectors in 2 dimensions, finding an accuracy of 98%, showing the success of the algorithm for high number and dimension of feature vectors. Figures 3 and 4 graphically depict one run of the algorithm on each dataset in 2 feature dimensions. See section 5 for further discussion and comparison to other algorithms.

C. Limitations

Some limitations exist on experimental applications of this algorithm for quantum distance measure. For low shots, there is a high variance in calculating the probability distributions needed for (12). We deal with this by running each circuits for 100000 shots, for which we empirically find the coefficient of variance to be 1.6%. As noise mitigations are expectedly improved, this variance will rapidly decrease, allowing for the algorithm to be ran with less number of shots.

Additionally, when the optimal cluster centroids for the dataset are located very close together, the algorithm tends to misclassify (assign to a wrong cluster) a low number datapoints near the boundaries due to randomness in the distance measure, which can sometimes cause failure of the algorithm to terminate. Manual switches can be put in to the algorithm to force it to terminate, or else one can declare the algorithm to have converged if some number of features less than a threshold value change clusters, as opposed to the generally used value

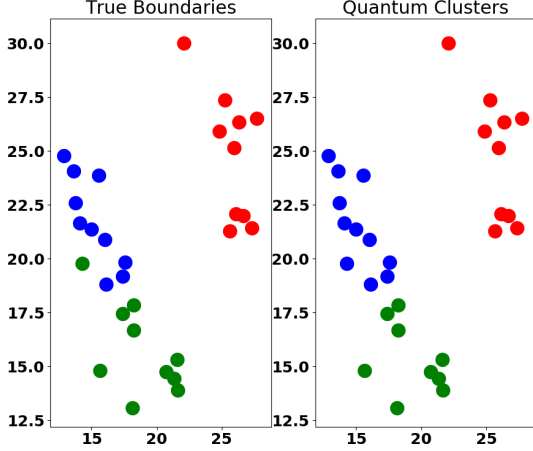


FIG. 3. Wine dataset - True and predicted classifications with Quantum K-means algorithm. Predicted clusters almost exactly reflect true boundaries in the data - only one datapoint which lies near the boundary between two clusters is misclassified. The natural clustering in the Wine dataset is very accurately detected by our algorithm, showing that it tends to detect clustering at a level comparable to similar classical algorithms.

of zero. We found that in nearly all cases where the algorithm does not converge, forcing the algorithm to terminate after a low number of iterations (< 8) is sufficient to get near-optimal results.

IV. Quantum Support Vector Machine

The support vector machine algorithm attempts to find a separating hyperplane - informally, an $M - 1$ dimensional generalization of a line - in M -dimensional *feature space* (a higher dimensional vector space of nonlinearly transformed feature vectors) from which all of the data points in one class will lie on one side of the hyperplane, and all of the data points in the other class will lie on the other. Let us consider a dataset consisting of N different feature vectors $\mathbf{X}^i = (X_1^i, X_2^i, \dots, X_P^i)$. The problem boils down to the convex optimization problem of finding the factors α^i such that

$$f(\mathbf{X}) = \sum_{i=1}^N \alpha^i y^i K(\mathbf{X}^i, \mathbf{X}) + b \quad (13)$$

is a decision function acting on a datapoint \mathbf{X} whose sign correctly classifies it - namely, positive values of $f(\mathbf{X})$ correspond to one classification for \mathbf{X} , and negative values correspond to the other. See [14] for more information. $K(\mathbf{X}, \mathbf{X}^i)$ is a positive-definite *kernel* function, equal to

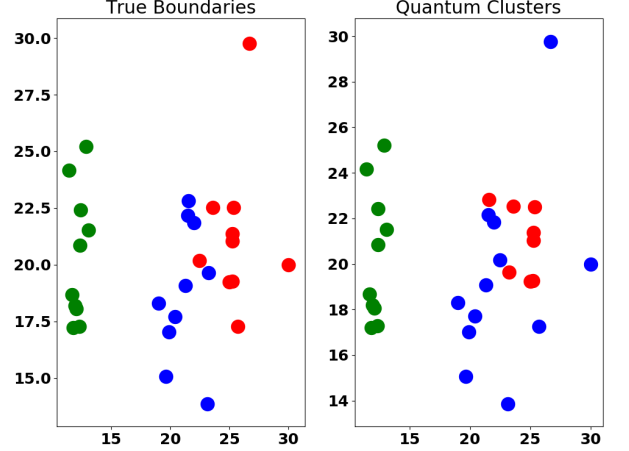


FIG. 4. Iris dataset - True and predicted classifications with Quantum K-means algorithm. Two of the three predicted clusters stray significantly from the true data boundaries. This is because there is little natural cluster separation between the two classes represented on the right side of the graphs, and as such, clustering algorithms do not perform well at classifying the data.

an inner product in some high dimensional vector space.

$$K(\mathbf{X}, \mathbf{X}^i) = \sum_j^\infty \varphi_j(\mathbf{X}^i) \varphi_j(\mathbf{X}). \quad (14)$$

Here, φ is some non-linear transformation into higher dimensional vector space. The power of the SVM stems from the fact that the kernel function can be calculated without explicitly calculating φ . Different kernels give rise to different decision boundaries, but many kernels are classically intractable. However, some can be more efficiently calculated on a quantum computer.

To achieve calculation of a classically intractable kernel, we must encode the feature vectors \mathbf{X}^i into quantum states that can be manipulated to compute our desired kernel. Here, we explore one such algorithm. Note that the algorithm presented here was developed by Havlicek et al. in [12]. We define the unitary gate

$$\mathbf{U}_{\Phi(\mathbf{X}^i)} = \exp(i \sum_{S \subseteq [n]} \phi_S(\mathbf{X}^i) \prod_{j \in S} Z_j), \quad (15)$$

as well as the n -qubit gate

$$\mathbf{M}_{\Phi(\vec{x})} = \mathbf{U}_{\Phi(\mathbf{X}^i)} \mathbf{H}^{\otimes n} \mathbf{U}_{\Phi(\mathbf{X}^i)} \mathbf{H}^{\otimes n} \quad (16)$$

where \mathbf{H} is the usual Hadamard gate. Here we take $n = 2$. The data is encoded through the coefficients $\phi_S(\mathbf{X}^i)$, such that

$$\phi_1(\mathbf{X}^i) = X_1^i, \phi_{1,2}(\mathbf{X}^i) = (\pi - X_1^i)(\pi - X_2^i) \quad (17)$$

We define our kernel as follows

$$K(\vec{x}, \vec{z}) = |\langle \Phi(\vec{x}) | \Phi(\vec{z}) \rangle|^2 = |\langle 0^n | \mathbf{M}_{\Phi(\vec{x}^i)}^\dagger \mathbf{M}_{\Phi(\vec{x}^j)} | 0^n \rangle|^2. \quad (18)$$

This kernel is thought to be classically intractable. However, it can be easily calculated by applying the gate $\mathbf{M}_{\Phi(\vec{x})}$ followed by the gate $\mathbf{M}_{\Phi(\vec{z})}^\dagger$ to an initial state $|0\rangle^n$ and experimentally calculating the frequency of getting the zero string 0^n as a result from the circuit.

V. Comparison of the Algorithms

In this section, we compare the various introduced quantum machine learning algorithms on real datasets in terms of runtime and accuracy. While K -means clustering and the support vector machine belong to different classes of machine learning algorithms, with the former being a clustering algorithm and the latter being a supervised binary classification algorithm, we can compare them under certain constraints. For the K -means clustering algorithms, we input the unlabeled test data and then compare the generated clusters with the true labels of the data. Then, we calculate the accuracy by calculating the percentage of correctly classified features. For the Support Vector Machine, we utilize the *One Against Rest* multiclass extension provided by *Qiskit* to extend the SVM to more than two classes. The *One Against Rest* extension constructs a number of SVMs equal to the number of classes, each of which compare one class against all the others. See [15] for more details. We include a classical K -means clustering algorithm as well as a classical RBF kernel SVM as controls. We run five experiments on each classifier, with 2 feature dimensions and 30 test inputs, as well as 30 training inputs for the SVMs. Note that the clustering algorithms are not provided with any training vectors and corresponding labels, while the SVMs are. Results of the simulations are shown in Tables 1 and 2.

As expected, the SVM algorithms are much more consistent in terms of time and accuracy than the clustering methods. This is because the SVM is a supervised algorithm, and the amount of time that the clustering algorithm takes depends on how many iterations it must undergo, which consequently depends on the random initial seeding of the clusters. On the Wine dataset, the hybrid clustering algorithm performs similarly to the classical SVM, with the quantum SVM falling far behind in terms of accuracy. For the Iris dataset, which contains little natural clustering between two of the three classes, the SVM algorithms perform better as their kernels allows them to classify non-linear boundaries. Surprisingly, the quantum K -means clustering algorithm is more accurate than its classical counterpart on both datasets. The quantum K -means algorithm takes several orders of magnitude more time to execute than its classical counterpart, but this is a consequence of simulating the quantum circuits as opposed to running them on a real ma-

Wine	Accuracy	Time (s)
Classical SVM	96.7%	0.0052
Quantum SVM by Havlicek et al.	63.3%	104.7
Classical K-means	88.7%	0.0170
Quantum K-means	96.7%	114.9

TABLE I. Trinary Classification on Wine Dataset. Each row depicts one algorithm, showing the average of all 5 trials' accuracy and runtime. As the Wine dataset contains natural clustering, both classical and quantum K-means clustering algorithms perform very well at classifying the data. Surprisingly, the quantum clustering algorithm performs better than its classical analogue.

Iris	Accuracy	Time (s)
Classical SVM	93.3%	0.0064
Quantum SVM by Havlicek et al.	80.0%	89.41
Classical K-means	72.7%	0.0059
Quantum K-means	75.3%	107.7

TABLE II. Trinary Classification on Iris Dataset. Each row depicts one algorithm, showing the average of all 5 trials' accuracy and runtime. As the Iris dataset does not contain much natural clustering, both classical and quantum K-means clustering algorithms suffer in performance at classifying the data. Still, the quantum clustering algorithm performs better than its classical analogue.

chine. See section 6 for time complexity analysis of our algorithm.

VI. Time Complexity Analysis

ELABORATE HERE. In general, as the dimension and size of the data increases, we strongly suspect that the hybrid algorithm will become less computationally expensive than the classical one to implement. We base this assertion off previous time complexity analysis done by Lloyd et al. [10] and Kerenidis et al. [11]. Additionally, it is well known that K -means clustering algorithms are much less computationally expensive than SVMs on average, and we suspect that this holds for the quantum analogues of each algorithm (INSERT CITATION).

VII. Conclusions

In this paper we introduced a hybrid algorithm for K -means clustering, based on the standard classical clustering algorithm and a novel technique of computation of Euclidean distance with quantum states. We find that the algorithm is successful in detecting natural clustering patterns in real data, and can reduce computation time of distance for large feature dimensions and many feature vectors when ran on a superconducting processor. This suggests that when one reasonably suspects there

to be clustering in a large dataset, our algorithm can be useful in quickly finding those clusters for various data analysis uses. As described in [10], there exists a purely adiabatic analogue to our hybrid algorithm which can save even more computational time in which cluster clas-

sification is conducted by perturbing a single quantum state which can then be sampled to return the clustering assignments with high probability. In the future we intend to attempt to construct and test such a clustering algorithm experimentally on an adiabatic quantum computer such as D-Wave.

-
- [1] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, Quantum circuit learning, *Phys. Rev. A* **98**, 032309 (2018), arXiv:1803.00745 [quant-ph].
 - [2] E. Farhi and H. Neven, Classification with Quantum Neural Networks on Near Term Processors, arXiv e-prints , arXiv:1802.06002 (2018), arXiv:1802.06002 [quant-ph].
 - [3] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, *Nature (London)* **549**, 195 (2017), arXiv:1611.09347 [quant-ph].
 - [4] J. Romero, J. P. Olson, and A. Aspuru-Guzik, Quantum autoencoders for efficient compression of quantum data, *Quantum Science and Technology* **2**, 045001 (2017), arXiv:1612.02806 [quant-ph].
 - [5] K. H. Wan, O. Dahlsten, H. Kristjánsson, R. Gardner, and M. S. Kim, Quantum generalisation of feedforward neural networks, *npj Quantum Information* **3**, 36 (2017), arXiv:1612.01045 [quant-ph].
 - [6] R. Chatterjee and T. Yu, Generalized Coherent States, Reproducing Kernels, and Quantum Support Vector Machines, arXiv e-prints , arXiv:1612.03713 (2016), arXiv:1612.03713 [quant-ph].
 - [7] M. Schuld and N. Killoran, Quantum machine learning in feature Hilbert spaces, arXiv e-prints , arXiv:1803.07128 (2018), arXiv:1803.07128 [quant-ph].
 - [8] M. Schuld, A. Bocharov, K. Svore, and N. Wiebe, Circuit-centric quantum classifiers, arXiv e-prints , arXiv:1804.00633 (2018), arXiv:1804.00633 [quant-ph].
 - [9] P. Rebentrost, M. Mohseni, and S. Lloyd, Quantum Support Vector Machine for Big Data Classification, *Phys. Rev. Lett.* **113**, 130503 (2014), arXiv:1307.0471 [quant-ph].
 - [10] S. Lloyd, M. Mohseni, and P. Rebentrost, Quantum algorithms for supervised and unsupervised machine learning, arXiv e-prints , arXiv:1307.0411 (2013), arXiv:1307.0411 [quant-ph].
 - [11] I. Kerenidis, J. Landman, A. Luongo, and A. Prakash, q-means: A quantum algorithm for unsupervised machine learning, arXiv e-prints , arXiv:1812.03584 (2018), arXiv:1812.03584 [quant-ph].
 - [12] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, Supervised learning with quantum-enhanced feature spaces, *Nature* **567**, 209–212 (2019).
 - [13] E. Aïmeur, G. Brassard, and S. Gambs, Machine learning in a quantum world, in *Advances in Artificial Intelligence*, edited by L. Lamontagne and M. Marchand (Springer Berlin Heidelberg, Berlin, Heidelberg, 2006) pp. 431–442.
 - [14] A. J. Smola and B. Schölkopf, A tutorial on support vector regression, *Statistics and Computing* **14**, 199 (2004).
 - [15] Multiclass extension, https://qiskit.org/documentation/aqua/multiclass_extensions.html, accessed: 2019-8-6.