New Journal of Physics
The open access journal at the forefront of physics
Deutsche Physikalische Gesellschaft DPG
IOP Institute of Physics
Published in partnership with: Deutsche Physikalische Gesellschaft and the Institute of Physics

OPEN ACCESS

RECEIVED
27 May 2020

REVISED
30 July 2020

ACCEPTED FOR PUBLICATION
4 August 2020

PUBLISHED
7 September 2020

PERSPECTIVE

Key questions for the quantum machine learner to ask themselves

Nathan Wiebe
Department of Physics, University of Washington, Seattle, WA 18195, United States of America
Pacific Northwest National Laboratory, Richland, WA 99354, United States of America

Keywords: quantum computing, quantum machine learning, machine learning

Abstract
Within the last several years quantum machine learning (QML) has begun to mature; however, many open questions remain. Rather than review open questions, in this perspective piece I will discuss my view about how we should approach problems in QML. In particular I will list a series of questions that I think we should ask ourselves when developing quantum algorithms for machine learning. These questions focus on what the definition of quantum ML is, what is the proper quantum analogue of QML algorithms is, how one should compare QML to traditional ML and what fundamental limitations emerge when trying to build QML protocols. As an illustration of this process I also provide information theoretic arguments that show that amplitude encoding can require exponentially more queries to a quantum model to determine membership of a vector in a concept class than classical bit-encodings would require; however, if the correct analogue is chosen then both the quantum and classical complexities become polynomially equivalent. This example underscores the importance of asking ourselves the right questions when developing and benchmarking QML algorithms.

Quantum machine learning (QML) has taken the quantum computing community by storm [1–3] promising substantial performance improvements for classifying data in complex data sets and modeling quantum systems. While great strides have been made towards understanding the impacts that quantum technologies will have in this space, many deep questions remain. In particular, it is still not clear what QML even means. It is further not clear how much quantitative improvement quantum models can offer for classical datasets and what fundamental limitations are imposed on QML from quantum information theory and quantum query complexity. In this perspective piece, I will argue that in the community's desire to address the former issues many foundational questions have been overlooked and increased attention is needed to some of these more fundamental questions. Without a closer inspection of these key questions, it is my belief that it will be difficult to make continued progress towards finding the first killer application for QML.

There are of course a legion of important questions that we need to ask, especially in a field as new as QML. However, here I identify four main questions that I believe practitioners of QML need to ask themselves. I will go over these questions in detail and provide some of the partial answers that have been developed to respond to them. These answers should not be seen as unique or canonical. They are provided as merely a representation of how I, and others in the field, think about how we could answer such questions. It is my hope that by reflecting on these questions we as a community will not only find better answers, but also better questions.

1. Question 1: what do I mean by quantum machine learning?

Of the three questions that I will discuss this question is perhaps the one that I spend the most time struggling with. In classical systems, the definition of machine learning is somewhat more concrete. The field is the study the behavior of algorithms whose performance is derived from past experiences which are fed to it in the form of a training data set [4]. From this perspective, the antecedents of machine learning can be seen to stretch back to Gauss' invention of least-squares fitting and perhaps even older. In all such cases, the salient property is that a model is provided for data that is capable of predicting features or labels for the data and the quality of such models improves as more data is fed to it. From this perspective, modern approaches to machine learning such as convolutional neural networks [5] and generative adversarial networks [6, 7] are simply richer models for data than their ancestors.

Currently, there is no such concise widely agreed upon statement of what QML is. Nonetheless, when pressed most practitioners provide a variant of one of the two following responses.

Answer 1: QML involves using a quantum device to solve a machine learning task with greater speed or accuracy than its classical analogue would allow.

This answer is perhaps the most common answer given by the community. Many early quantum algorithms for machine learning implicitly used such a definition. For example, quantum algorithms that give polynomial speedups for clustering [8, 9], nearest-neighbor classification [9, 10], nearest-centroid classification [9, 11], recommendation systems [12], training Boltzmann machines [13–16] and support vector machines [17], perceptron training [18, 19] all follow this form of answer to the question.

This answer is also the least ambitious definition of what is meant by QML because it typically seeks an advantage from quantum computers in machine learning applications on data. When viewed from the perspective of a computer scientist, the aim of of QML is to reduce the complexity in terms of either samples (sample complexity) or the number of operations needed in order to train the model, classify a test vector or generate a novel example of a concept. The difference between these particular resources is made especially clear in the literature of quantum PAC learning [20–22] in which quantum and classical sample complexities are found to be equivalent but time-complexity is not. People who approach the problem from more of a perspective of a data scientist will often claim that the goal for quantum computer is to provide a richer class of models for the data, meaning that by incorporating quantum effects one can model correlations in a data set that would otherwise be difficult to describe using traditional models [23–25].

Typically those who approach the field from what I call the data science perspective, are most interested in optimizing a test loss function (such as the classification error rate for supervised learning problems). Questions about the sample, query or time complexity often are secondary from their perspective. The computer science perspective is much more commonly taken by those who approach QML from a quantum algorithms perspective and seldom focuses on the empirical performance of quantum classifiers. In such cases, the sample, query and time complexities are seen as being of central importance relative to the attainable values of the loss function.

While answer 1 captures much of the flavor of QML, it also falls short in important ways: such answers presuppose that a unique classical analogue exists. The need to problematize this point of view is a theme that underlies many of the questions raised in this article and will be further problematized throughout this perspective piece.

Answer 2: QML involves using a quantum device to classify or extract features from quantum states.

Answer 2 is becoming increasingly popular as a response from the QML community. This popularity stems, in part, from the challenges that appear in finding an exponential speedup on quantum computers for classically important machine learning tasks. Adherents to this way of thinking tend to focus on problems where the data itself is hard to generate or store classically. Examples of such work include, generative quantum Boltzmann machine training [16, 26–28], quantum neural networks that are used to identify quantum phase transitions [29] and algorithms such as quantum principal component analysis [30] and reinforcement learning [31].

While considering QML applications where the input is manifestly quantum can provide, in some cases, exponential advantages over their classical analogues in either sample or time complexity [32–34] it is difficult to compare the costs of quantum and classical machine learning problems in both cases. In particular, if the quantum states that define the training set were specified as bit strings then the classical analogue would require exponential input. Also there is a further issue about the input model. If one has access to a unitary process for preparing the input quantum states then this affords the algorithm possibilities that sometimes cannot be granted by models for QML where the training data is provided through a non-invertible procedure. Perhaps the most dramatic manifestation of this is in the PAC model of learning wherein only constant factor improvements in learning concepts from quantum states are possible from a non-invertible oracle [22]; whereas, if the oracle that yields the training data is invertible then quadratic reductions in the number of times the oracle that yields the data needs to be queried can be achieved [10, 35].

There are also strong connections between learning algorithms that take as input quantum data and quantum characterization verification and validation. Protocols such as quantum Hamiltonian learning [36], shadow tomography [37] and other protocols adjacent to these fields can also be viewed as QML protocols and in some cases can support exponential speedups (unless of course BQP= BPP, which is highly unlikely).

The wide range input models that can be considered in QML complicates the search for quantum advantages for machine learning tasks or finding new forms of machine learning that may not have clear classical analogues. Thinking critically about the definitions of the input for purely QML problems is absolutely vital both to understand an algorithm and contextualize its performance.

2. Question 2: what is the classical analogue of my quantum machine learning algorithm?

The importance of this question was raised recently in a very visible manner by Ewin Tang and others [9, 35, 38, 39] in a series of papers that criticized the claims of exponential speedups in algorithms such as quantum principal component analysis, nearest centroid classification, quantum recommendation systems and related algorithms. The central issue behind most of the claims refuted in these works stemmed from the fact that the exponential speedups were not justified using complexity theoretic, query complexity arguments or even the inability for the community to provide an efficient classical algorithm for the problem (as in the case of factoring); rather, they were justified by comparison to a classical algorithm that was deemed to be analogous to the quantum algorithm. The central problem with such a comparison is that there often are *multiple* classical analogues of a quantum algorithm. In all these cases, the claims of exponential speedups evaporated when the correct analogue of the classical algorithms is compared against.

I find that I struggle more giving clear answers to question 2 than I do for question 1. The struggle is very much akin to understanding the classical limit of quantum mechanics. There are two main correspondences that often are used to understand the classical limit of quantum mechanics: the Ehrenfest and the Liouville correspondence [40]. The former states that a single Newtonian trajectory should emerge from the correct classical limit of quantum dynamics. The Liouville correspondence states instead that the classical limit of quantum dynamics occurs when the quantum predictions are in practically indistinguishable from those of a classical probability distribution. This exact same issue arises in QML. Philosophically, we can either choose the classical analogues of a QML algorithm to either be deterministic or randomized classical algorithms. Both options need to be considered when comparing a quantum ML algorithm to a classical counterpart. I detail below some of the thought process that I have found useful for clarifying these issues in my own research.

2.1. Question 2.1: what is my input model?

An input model is an abstraction that describes how the data is both represented and fed into a quantum algorithm. Classically, there are a number of such input models. The first is a database model, wherein the data is assumed to be provided by a database that holds all the training data. Another popular model is the streaming model or online learning, wherein the algorithm does not have explicit access to the underlying training vectors but instead is fed the data one at a time. Examples include perceptron training and various forms of kernel learning [41, 42].

In either of the above cases, there are also multiple ways that the data can be represented. In quantum computing, however, there are further complications because the feature vectors that are used classically can be encoded directly as qubit strings that represent vectors or as a quantum state vector. These two encoding schemes are known as *bit-encodings* and *amplitude-encodings*. I will also discuss a third input model wherein the data fed into the system is an unknown quantum state and will refer to it as a *quantum input* model (despite the fact that all three input models are quantum).

2.1.1. Bit encodings

The most concise way to think of input models that use a bit encoding is in an oracular model. In order to understand this model, let us assume that the learning problem has a training set that is composed of training vectors of the form of the following form: $\{\mathbf{v}_i : i = 0, \ldots, N-1\}$. I will also denote a bit-string representation of the training vectors to be $[\mathbf{v}_i]$ in the following. These bit vectors can be accessed on demand through a self inverse quantum oracle of the form

$$O_{\text{data}}|i\rangle|b\rangle = |i\rangle|b \oplus [\mathbf{v}_i]\rangle. \tag{1}$$

I call this form of the input oracle a *coherent bit-oracle*. This is perhaps the most commonly used abstraction for how to get training data into a quantum algorithm. It is used in quantum algorithms for perceptron training, support vector machines, nearest neighbor classification and can be instantiated in low-depth using a data structure known as a QRAM [43, 44]. The salient feature of such oracles is that they allow us to perform superpositions of queries on the training data of the form

$$O_{\text{data}}\left(\sum_x a_x|x\rangle|0\rangle\right) = \sum_x a_x|x\rangle|\,[\mathbf{v}_i]\rangle, \tag{2}$$

which is necessary in order to achieve a quantum speedup over the size of the data set.

Another sub-class of a bit-oracle is the *incoherent bit-oracle*. In this case the input data is yielded by a subroutine that acts as

$$O_{\text{data}}(i)|b\rangle = |b \oplus [\mathbf{v}_i]\rangle. \tag{3}$$

This form of the input precludes the use of quantum superpositions over the training data and in turn the speedups that are associated with this. This feature is perhaps most dramatically demonstrated in the recent proof that at best constant factor improvements in sample (i.e. query) complexity relative to classical learning can be attained within the quantum PAC model, which is a model that precludes coherent accesses to the training data [22]. On the other hand, this input model does accurately reflect the use cases that emerge QML algorithms that do not depend on superpositions over the input data such as variational autoencoders [27, 28], feed-forward quantum neural networks [23, 24] and some implementations of quantum Boltzmann machines [15].

The main advantage of bit-encodings are that they provide the data in exactly the same manner that the data is provided to most classical machine learning algorithms. This means that a query to a subroutine that provides the data is analogous to an access to the classical training set.

The primary disadvantage of these approaches is that the space overheads needed to store the quantum state can be prohibitive. In particular, for the case of the MNIST example considered earlier the training images consist of 784 pixels, which means that a bit encoding of the input would require minimally 784 qubits. This makes most applications in vision out of reach of existing quantum computers, which typically have fewer than 100 quantum bits [45].

2.1.2. Amplitude encodings

With the exception of experiments performed on quantum annealers [13, 15], the vast majority of QML experiments on real-world data sets have been performed on amplitude encoded data. The advantage of amplitude encodings is that they represent their data using only logarithmically many quantum bits by embedding the vectors in the training set as unit vectors in the quantum computer [25, 46]. In particular, let $\{v_i : i = 1, \dots, N\}$ be a set of training vectors in $\mathbb{R}^{2^n}$ such that $\max_i |v_i| \leqslant \Lambda$. Furthermore, if $v_i = \sum_j v_{ij}\,|j\rangle$ then let $|v_i\rangle = \sum_j v_{ij}/|v_i||j\rangle$.

Then we can construct an isomorphism between the training vectors and unit vectors in $\mathbb{R}^{2^{n+\lceil\log_2 N\rceil+1}}$:

$$\begin{aligned} v_i &\mapsto |v_i\rangle\frac{|v_i|}{\Lambda}|0\rangle|0\rangle + \sqrt{1-\frac{|v_i|^2}{\Lambda^2}}|0\rangle|i\rangle|0\rangle := |v_i'\rangle \\ v_i^\dagger &\mapsto |v_i\rangle\frac{|v_i|}{\Lambda}|0\rangle|0\rangle + \sqrt{1-\frac{|v_i|^2}{\Lambda^2}}|0\rangle|i\rangle|1\rangle := |v_i'^\dagger\rangle. \end{aligned} \tag{4}$$

Note that this is an isomorphism because the inner products between the vectors remain the same up to a factor of Λ^{-2} which is a known constant.

Additional qubits are needed in this mapping is to guarantee that the junk-space, which is used to pad the non-unit vector into a unit vector, will be unique and thus not have a projection onto any of the other encodings of the training vectors. These additional space requirements needed to maintain the correct inner products between amplitude encoded vectors can be prohibitive and so some in the QML community use such encodings without guaranteeing the orthogonality of the vectors. This, in effect, applies a feature map to the quantum vectors that are loaded in. This feature map means that the training data that the quantum algorithm is provided differs from that the classical algorithm is provided and care should be taken, especially if the error rate for the quantum model is higher than classical methods.

In direct analogy to the bit-encodings, there are two main classes of amplitude encoding: coherent and incoherent encodings. For the case of a coherent encoding

$$O_{\text{data}}|i\rangle|0\rangle = |i\rangle|v_i'\rangle, \tag{5}$$

and the incoherent encoding is defined in precisely the same manner:

$$O_{\text{data}}(i)|0\rangle = |v_i'\rangle. \tag{6}$$

The coherent encoding version of this is used, for example, in nearest neighbor classification algorithms and the incoherent version is more commonly used in quantum neural networks.

2.1.3. Quantum input

Quantum input models are the final category of inputs that are commonly used in QML algorithms and are most commonly used in unsupervised training examples, but they have also been proposed for other approaches such as quantum PCA [30], quantum Hamiltonian learning [36] and also generative training of quantum neural networks [26].

There are many ways such models can be formalized, but here we will consider two concrete versions of access models that generalize the models that are often implicitly used in QML algorithms that take quantum inputs. In the first such model, a subroutine exists that prepares allows the user to prepare one of an indexed family of density operators $\rho_i : i = 1, \ldots, N$. This procedure can be viewed as an explicit function Λ mapping the indices for the training set to copies of distinct density operators

$$\Lambda_{\text{data}} : i \mapsto \rho_i. \tag{7}$$

A further refinement of this model is sometimes used wherein the quantum data is provided in the form of a pure state. In particular, if we assume that the Hilbert space factorizes into a tensor product of space $\mathcal{A}$ and $\mathcal{B}$ wherein we define $\text{Tr}_{\mathcal{B}}(|\psi_i\rangle) = \rho_i$ and the purified state preparation algorithm can be written as

$$U_{\text{data}}(i)|0\rangle = |\psi_i\rangle. \tag{8}$$

This model is relevant in cases where density matrix exponentiation [30] is used as a tool in QML because the fundamental limits that the technique imposes on the query complexities can be alleviated by using a purified quantum input oracle [47].

Coherent analogues of the purified state preparation algorithm can also be considered:

$$U'_{\text{data}}|i\rangle|0\rangle = |i\rangle|\psi_i\rangle. \tag{9}$$

Such models are necessary for achieving quantum speedups over the size of the set and are often instantiated using a QRAM quantum data structure [44].

2.2. Question 2.2: what does my algorithm output?

In order to properly compare quantum and classical we also need to be able to clearly articulate what form the outputs of the quantum algorithm take. For example, in supervised learning tasks the output could be a class label or potentially an estimate for a gradient for the weights (i.e. the parameters) of the model. For other quantum algorithms, the output is often taken to be a quantum state [33, 48]. Such cases often arise when the QML algorithm to be a subroutine in a larger algorithm since there are no clear classical outputs for such algorithms. In order to fairly compare quantum to classical we therefore need to agree upon a classical task that both need to solve, or at the very least establish a correspondence between the outputs of the two models. Fortunately, this is typically much easier to do for the output of quantum algorithms than it is for their input.

Having an answer to question 2.2 is vital for practitioners of QML to have in hand for any algorithm that they propose since several of the most significant quantum algorithms in the field suffer from the problem of extracting classically meaningful data from the quantum output [30, 33]. This problem has been referred to in the literature as the output problem [1], wherein the cost of extracting desired information from the solution dominates the cost of a quantum algorithm. Addressing this problem, along with the issue of efficiently loading the data into the algorithm (the input problem) remain among the two most pressing issues when trying to devise quantum algorithms that offer exponential speedups for ML on classical datasets [49].

Answer 1: I am using a bit encoding so standard classical algorithms are a natural analogue to my quantum algorithm.

This is the simplest case that we can consider for addressing question 2. Here the inputs to the quantum algorithm are specified as bit strings which makes comparison between the quantum algorithm and its classical counterparts straight forward. Examples of quantum algorithms that take this tack are given in [10, 15, 24].

There are also several important properties that bit-encoded quantum data has that are infrequently remarked on. The first is that while we can have quantum superpositions over the input bit-vectors when we measure the result we will get a unique training vector back. Second, if we are provided a quantum bit string corresponding to a training vector then we can clone the vector to our heart's content in both the quantum and the classical case. The third property is that subtle differences between training vectors can be immediately deduced by looking at their binary representations. We will see that none of these properties hold for amplitude encoded data.

Answer 2: I am using an amplitude encoding so randomized classical ML algorithms are a natural analogue of my quantum algorithm.

While it may be tempting to think that the bit-encoded classical training vectors used in most ML protocols are analogous to amplitude encoded quantum state vectors, they are not. To see this, let v_k be a training vector in $\mathbb{R}_+^n$. We can represent this vector as a probability distribution using the exact same tricks employed in amplitude encoding. To see this, consider the following probability distribution (represented as a density operator)

$$\rho_{v_k} = \sum_{j=1}^{N} \frac{[v_k]_j}{\max_k \|v_k\|_1} |j\rangle\langle j| \otimes |0\rangle\langle 0| + \left(1 - \frac{\|v_k\|_1}{\max_k \|v_k\|_1}\right) |0\rangle\langle 0| \otimes |k\rangle\langle k|. \tag{10}$$

The corresponding amplitude encoded quantum state would be

$$|\psi_{v_k}\rangle = \sum_{j=1}^{N} \frac{[v_k]_j}{\max_k \|v_k\|_2} |j\rangle|0\rangle + \sqrt{1 - \left(\frac{\|v_k\|_2}{\max_k \|v_k\|_2}\right)^2} |0\rangle|k\rangle. \tag{11}$$

Thus when the classical analogue of an amplitude encoded state is measured it yields a single outcome that is distributed according to the training vector, just as the amplitude encoded state is. Similarly, given a quantum state the no-cloning theorem prevents one from generating two states from a single copy of $|\psi_{v_k}\rangle$ and the same restriction clearly holds for ρ_{v_k}. Thus amplitude encoded states have few of the salient features of bit encoding that are considered above and have a closer correspondence to the probabilistic classical encoding given in (10). While amplitude encodings are frequently used in QML [12, 23, 25, 30], this correspondence is seldom commented on in the literature.

Ewin Tang and others, however, explicitly invoke this correspondence in works such as [9, 38] to show that classical algorithms exist that are only polynomially slower than quantum algorithms for clustering and nearest neighbor classification. In turn, it has been noted that after this correspondence is used then the apparent exponential advantages of quantum algorithms no longer persists. This underscores the importance of considering classical randomized approaches to ML when comparing quantum to classical and also reinforces why we need to think clearly about what the classical analogues of our quantum algorithms are, especially in cases where the input data is amplitude encoded.

Answer 3: I am learning directly from quantum states and so classical analogues of my algorithm may exist in the quantum tomography literature.

Perhaps even more than with amplitude encoding, our aim to find a classical analogue for our QML protocols is stymied when the input itself is quantum. This is because classical machine learning algorithms take classical information as input and the quantum algorithms allow us to manipulate quantum information. It is difficult therefore to see how such quantum algorithms can ever truly have a classical analogue.

While I am not aware of a direct analogue for such methods, strong parallels with such algorithms exist with tomographic protocols [3, 26]. This point has been made a few times in the literature and explicit comparisons have been drawn with partial tomography, which is a procedure that attempts to reconstruct features of the quantum state (of which channel identification can be viewed as an important sub-problem) rather than the entire state.

As an example of a classical analogue for a quantum protocol, consider the following. If we assume that we are using a direct quantum access model and we assume that we are given an informationally complete POVM $F = \{F_j : j = 1, \ldots, M\}$ then in the classical setting we allow ourselves to draw samples from a distribution $P(j|i) = \mathrm{Tr}(F_j \Lambda(i))$. This means that the classical version of the learning algorithm can access quantum data and in principle reconstruct the entire training set. Giving the classical computer access to this information allows it to in principle simulate any quantum algorithm's action on the training data (at exponential cost unless $\mathsf{BQP} = \mathsf{BPP}$).

Once an approach such as this is taken, the inputs in the quantum and classical cases are analogous in that both algorithms are allowed to directly make queries of the quantum state and in principle both algorithms can learn a perfect model for the data. Work by Aaronson and others has also shown that this type of reasoning leads to a protocol called quantum shadow tomography [37], which gives a way of predicting the values of most quantum observables using a small number of samples from a quantum system.

The difference between quantum and classical machine learning methods given quantum inputs arises from the way in which we are allowed to interact with the quantum data. In the classical case, we can envision that we are allowed to sample from a training corpus that consists of measurements of the quantum states using the fixed POVM. In the quantum setting, we can apply arbitrary quantum transformations and entangle the data with ancillary quantum bits.

A challenge with this definition of a classical analogue to QML techniques is that the ability to measure an arbitrary POVM can in some cases obfuscate substantial quantum computational power because the POVM elements themselves may require a substantial number of quantum gates to implement. In order to fairly compare the quantum and the classical ML algorithms under this correspondence then the number of classical and quantum operations needed in both the quantum algorithm and the classical algorithm's implementation of $P(j|i)$ must also be compared.

3. Question 3: how well does my QML algorithm compare to corresponding classical approaches?

This question is perhaps the most important, but hardest to address question in QML. In QML there are a number of criteria that are relevant for benchmarking the performance of a protocol.

- The value of a loss function attainable through quantum or classical means, such as prediction loss, KL-divergence or intra-cluster variance for clustering algorithms.
- The time complexity of an algorithm specified by the number of operations needed to successfully execute the algorithm within a given error tolerance.
- The number of samples (for incoherent access models or streaming models) or the number of quantum queries to the training data needed by the training protocol.

All of these issues must be addressed in order to make a compelling argument for a QML protocol. After all, if a quantum model yields greater prediction accuracy than a classical analogue but requires exponentially more samples from the training set then the advantages yielded by the algorithm may be minimized. In order to assess the value of any quantum algorithm this question needs to be addressed and the above points should always be considered in a response.

Answer 1: my QML algorithm, which uses amplitude encoding, may be exponentially worse than classical algorithms that use a bit encoding.

Perhaps in light of question 2, this point should not be seen as a surprise since the natural analogue of amplitude encoding is a classical *probability encoding* and while a bit oracle can simulate a query to a probability oracle efficiently, the converse is not true.

In order to understand the power of these two methods, let us first focus on the classical case. In the classical case, given in (10), bounds on the distinguishability of distributions cause models that inherit this encoding to be intrinsically weaker than conventional models where the input is given by a bit string. Let us consider two concept classes C_+ and C_- for a binary classifier that are separated and compact. Specifically, Let v_+ and v_- be two vectors taken from these two classes such that $\|v_+ - v_-\|_1 = \min_{x\in C_+, y\in C_-}\|x-y\|_1 = \Delta > 0$. Given access to an oracle that yields bit strings encoding v_+ and v_- even these two nearby vectors can be distinguished in a single query. On the other hand, the total variational distance between ρ_{v_+} and ρ_{v_-} is (note that the k labels of both vectors cannot be assumed to be distinct in this context, otherwise that would hold information that could distinguish the classes)

$$\frac{1}{2}\|\rho_{v_+} - \rho_{v_-}\|_1 \in \Theta\left(\sum_j \frac{|[v_+]_j - [v_-]_j|}{\max_k \|v_k\|_1}\right) \subseteq \Omega\left(\frac{\Delta}{\max_k\|v_k\|_1}\right). \tag{12}$$

For fixed $\max_k\|v_k\|_1$ the number of samples needed to distinguish the two distributions using the optimal estimator scales as $O(1/\Delta)$. Thus the concept classes C_+ and C_- need $O(1/\Delta)$ samples to distinguish between them as opposed to 1 in cases where the entire training vector is provided. Conversely, a sample can be generated using the bitstring representing v_+ or v_- so this model is strictly weaker than the corresponding bit-encoded classical model.

A similar result holds for amplitude encoding. The distinguishability of two distributions is again given by the trace-distance (or Schatten 1-norm) which reads

$$\||\psi_{v_+}\rangle\langle\psi_{v_+}| - |\psi_{v_-}\rangle\langle\psi_{v-}|\|_1 = \sqrt{1 - |\langle\psi_+|\psi_-\rangle|^2} \in O\left(\frac{\Delta}{\max_k\|v_k\|_2}\right). \tag{13}$$

Thus the two concept classes C_+ and C_- cannot be distinguished easily from amplitude encoding if the margin between the two classes is sufficiently small. This shows that again, in principle, amplitude encodings that use a small number of can be weaker than classical classifiers for some problems. Note that the sample complexity in this model for learning the concept class scales as $O(1/\Delta)$ in this context as well. In principle, the number of samples needed (i.e. queries) can be quadratically boosted by amplitude amplification [50] but this does not change the fact that exponentially many state preparations will be needed.

As an example of a challenging concept to learn using an amplitude encoding that uses a small number of quantum bits let us consider learning the parity concept. Specifically, given n-bits the parity of the n bits is 0 if there is an even number of 1s in the string and is 1 otherwise. The most concise amplitude encoding uses only $O(\log_2(n))$ qubits (for a constant number of training vectors) to store the training vector. Consider the vectors 0 and 1 encoded in n qubits. The amplitude encoding of the states is

$$|\psi_0\rangle = |0\rangle|1\rangle \tag{14}$$

$$|\psi_1\rangle = \frac{1}{\sqrt{n}}|1\rangle|0\rangle + \sqrt{1-\frac{1}{n}}|0\rangle|1\rangle. \tag{15}$$

In this case the inner product between the two states is $1 - O(1/n)$ and therefore even if one is able to find an optimal classifier, an exponentially larger number of repetitions will be needed in order to decide on a class. Further, identifying an optimal classifier using a simple circuit remains a challenge. This can be addressed, in part, by simply using more qubits. If $O(n)$ qubits are used then the parity can be computed using a string of CNOT gates, which allows the class to be trivially determined. Thus a compact amplitude encoding may not always be desirable in terms of the sample complexity.

Despite these theoretical challenges, we note that the majority of the real-world problems that we consider are not as pathological as the parity problem. This suggests that relatively large margins exist when the problem is represented within the amplitude encodings that we use. However, it is important to keep these information theoretic restrictions in mind when considering how to use amplitude encoding when building classifier and also the potential impacts that choosing an improper amplitude encoding can have on the decision boundaries for a problem and in turn the sample complexity needed to classify data.

Answer 2: I really do not know, but I have uncompelling numerical/experimental results that may show advantages for an objective function I chose on a few small data sets.

The face of machine learning has changed within the last decade and rather than focusing on models that are based on a solid mathematical bedrock, machine learning (particularly in vision and speech) has begun to rely more heavily on models whose theoretical performance may be poorly understood but whose empirical performance often can be quite impressive. The success of this endeavor has been a result of several factors:

- The availability of sufficient computing power to train and validate models [51].
- The presence of standardized metrics by which to evaluate the performance of ML algorithms [52–54].
- Open source implementations through packages such as Tensorflow [55] or PyTorch [56] that allow (in principle) easy comparison to other state of the art ML methods.

This methodology has begun to influence QML as well, empowered in part by easy availability of quantum hardware in the form of the IBM quantum experience as well as a plethora of programming languages [57–59]. In contrast, less emphasis is being spent looking at QML from the perspective of a computer science and more emphasis is taken on examining QML from a data science perspective. In my opinion, this is perhaps a misstep because many of the features that make empirical approaches so successful for classical ML are absent from QML.

Present quantum computers are limited in scale and gate fidelity [60], which means that we often do not have sufficient quantum computing resources to test heuristic QML algorithms. Even simply evaluating the performance of the *final model* belies a much more substantial cost of learning reasonable parameters such as learning rates or architectures for the parameterized quantum gates that comprise the algorithms. This means that, even numerically, it is very difficult with existing computing resources to train models beyond 20 qubits and in experiment, Herculean efforts are needed to perform QML/variational algorithms on 12 or fewer quantum bits [61]. It should also be noted that these memory restrictions are not present in existing quantum annealers, although because of the absence of so-called non-stoquastic couplings such approaches can rigorously be simulated in polynomial time on classical computers [62, 63]. While these simulation methods are far from practical, they reveal that the advantages existing quantum annealers are expected to yield for typical ML problems are still unclear.

Table 1. Results of the benchmarking experiments taken from [25]. The cells are of the format 'training error/validation error'. The variance between the 50 repetition for each experiment was of the order of 0.01–0.001 for the training and test error. *For multilabel classification problems with *d* labels, the average of all *d* one-versus-all problems train and test errors were taken. Note that the quantum algorithm in question outperforms, in terms of test error, all other reference models only on the cancer set.

	CANCER	SONAR	WINE*	SEMEION*	MNIST256*
Quantum circuit	0.022/**0.058**	0.000/0.195	0.000/0.028	0.031/0.031	0.031/0.033
PERC	0.128/0.137	0.283/0.315	0.067/0.134	0.022/0.038	0.065/0.066
MLPlin	0.060/0.075	0.117/0.263	0.001/**0.039**	0.001/0.025	0.038/0.041
MLPshal	0.064/0.077	0.010/**0.174**	0.029/0.063	0.002/**0.024**	0.011/**0.018**
MLPdeep	0.056/0.076	0.001/**0.174**	0.010/0.063	0.001/0.026	0.014/0.021
SVMpoly1	0.373/0.367	0.452/0.477	0.430/0.466	0.101/0.100	0.092/0.092
SVMpoly2	0.169/0.169	0.334/0.383	0.090/0.099	0.100/0.101	0.091/0.092
Average	0.125/0.136	0.171/0.283	0.090/0.137	0.037/0.049	0.040/0.043

Quantum data sets have also yet to go through the standardization process that allowed the empirical performance of ML algorithms to be gauged. Excellent examples of such classical benchmarks include the MNIST dataset, the UCI repository and imagenet. These benchmarks, while appropriate for amplitude-encodings or applications on annealers, are often cannot be evaluated for bit encodings on existing quantum computers or classical simulators. This has meant that most comparisons are done on very small synthetic data sets that are easily scalable, such as the bars and stripes set [64]. This means that while small scale experiments can be performed at present, it is unclear whether any purported performance advantage will persist as we scale the problem size up.

If amplitude encoding is used then the above criticism do not apply. However, a new problem involving benchmarking the algorithm against its appropriate classical analogue emerges. In order to fairly compare the results, we therefore need to train our classical algorithms on the exact same training data provided to the quantum algorithm. If we do not do this, as demonstrated by the parity example above, this can result in the appearance of quantum approaches being much weaker than classical ones when in fact such differences may only arise from the use of a different input model.

A further challenge facing providing appropriate benchmarks arises from the lack of standardized metrics to compare. The lack of a common set of benchmarks as well as reference implementations means that the author is responsible for not only implementing and optimizing their algorithm but also its competition. The problem is that the author faces an incentive to show an advantage for the algorithm over the competition and thus it is difficult, even for the most high-minded researcher, to fairly compare their proposal to similarly optimized competitive approaches. Standardizing the benchmarks as well as providing open source reference implementations of competitive algorithms will be needed to meaningfully answer the question of whether a QML algorithm provides an advantage against its classical algorithm.

In order to publish a set of numerical experiments, it is often essential to show evidence of an advantage over analogous or competitive classical techniques. In the event that an advantage is not seen then the result will have a hard time being published in a prestigious venue. For this reason, the machine learner faces two main temptations. The first, is to selectively choose the test sets that the model is evaluated on in order to show an advantage. The second temptation is to spend more effort optimizing the quantum algorithm than is spent optimizing the classical algorithms.

A good example of a thorough effort to benchmark quantum classifiers is given in table 1 wherein the authors demonstrate the performance of the circuit centric classification algorithm of [25] is examined for a wide range of test cases (although the work does conflate amplitude encoding with bit encodings to some extent). In such cases a temptation exists to only publish the favorable data, in this case the performance of the algorithm on cancer data. Instead, by comparing the performance of the algorithm over a wide range of data sets a more nuanced view of the performance of the quantum algorithm is possible and in this case revealed that the advantage for the heuristic algorithm proposed was not in the accuracy attainable but rather in the dramatically smaller number of model parameters required to achieve comparable performance to the reference algorithms.

In short, if we are to provide compelling numerical studies of the performance of QML, we must continue to emulate the best practices of the classical machine learning community. Given the present excitement surrounding both quantum algorithms and machine learning we not only owe it to the scientific community to do our best to critically evaluate QML algorithms but also since the classical community does not test our protocols (yet) against their methods we cannot be sure that the reference examples that we test against are representative of the best that the community can provide. For this reason, until the field matures great effort will be needed in order to understand the true performance of existing and future heuristic quantum ML algorithms.

Answer 3: I have not done a solid empirical comparison but I see polynomial advantages in either query/sample/time complexity or have complexity theoretic arguments for why my scheme cannot be efficiently simulated classically.

This answer is another type of response that is common to this question and to me often represents a reply to give to yourself when considering a new heuristic quantum algorithm. An example of this involves heuristics where the underlying quantum circuits form a universal gate set [25] or result in a partial collapse of the polynomial hierarchy [65]. These solutions provide confidence that, regardless of the input model, the quantum algorithm is unlikely to be simulatable classically.

In the event that the quantum algorithm is an accelerated form of a classical protocol, such as the first Boltzmann training algorithms [14]; clustering [9, 17]; recommendation systems algorithms [12] and more, then query complexity separations can also be used to suggest that there may be a speedup to be gleaned from such an application.

Without one of these facts, or a related compelling justification for the algorithm, the impact that small scale implementations of the algorithm have are diminished. For this reason, it is important to consider this question well before any experiments are performed.

A nice side effect of trying to answer the question in this fashion forces you to think about what metrics you will be using to compare your quantum algorithm to its classical analogue. As mentioned, there are a host of different types of resources that emerge in such algorithms and by clearly articulating the tradeoffs between them it often becomes easier to contextualize any advantages once they become apparent or appreciate other advantages that may appear even if the quantum model affords no greater prediction accuracy than the classical method.

4. Question 4: what fundamental limitations exist for my quantum machine learning task?

Even if the previous questions are answered, an important last question to ask involves determining what fundamental obstacles stand in the way of QML algorithms. I have found this sort of question to be an excellent sanity check and challenge to see how much more a quantum algorithm can be improved.

Answer 1: there are complexity theoretic reasons why I cannot solve this QML task in polynomial time.

This answer arises frequently in QML. Many problems in machine learning can be recast as optimization problems that are NP-hard. K-means clustering is an excellent example of such a protocol [4], and as a result we strongly suspect that quantum computers will be unable to provide exponential speedups for such problems [9]. On the other hand, other protocols such as training quantum Boltzmann machines can reduce to problems that are QMA-hard [34]. Other tasks such as preparing Gibbs states using quantum rejection sampling involves implicitly estimating partition functions [66]. This task is #P-hard, which means that efficient exact algorithms for these protocols are highly unlikely to exist.

Answer 2: there are information theoretic limitations that impact the performance of my algorithm.

Even if there are no compelling reasons why improvements to an algorithm ought to be hard from complexity theoretic arguments, information theoretic arguments can also place limitations on QML protocols. This limitations can be especially pronounced in amplitude-encodings. An example of such limitations is given above when we showed that bounds on state discrimination can prevent vectors from being unambiguously classified using a sub-exponential number of measurements. Similarly, bounds on approximate cloning place limitations on the number of samples needed to be able to learn an accurate generative model for an unknown quantum state. Also results on channel compression have been used to argue about the maximum accuracy that could be attainable for quantum autoencoders [27].

I find information theoretic bounds particularly important because they not only provide simple arguments for the limitations of QML algorithms but also this line of thinking helps build bridges between the new field of QML and the much more mature field of quantum information theory. These bridges can, at times, allow us to leverage decades worth of development within that space and sometimes leads to easy answers to questions that we would otherwise be unable to even articulate.

Answer 3: there are arguments from random matrix theory that training my model will be generically hard.

A final limitation that often emerges involves insights gleaned from random matrix theory. Random matrix theory has a long history of being used to model the distributions found by complex quantum dynamics that emerge in quantum chaos as well as in quantum supremacy experiments and finally in training quantum neural networks. These arguments, which are used to great effect in [65], can be used to show that with high probability the gradients of training objective functions in large unstructured quantum circuits are concentrated exponentially closely to zero. Given that even quantum algorithms used to estimate the gradient of a model with D parameters acting on n-qubits require $\widetilde{O}(\sqrt{D}/\epsilon)$ queries to estimate the gradient within error ϵ (in the infinity-norm) [48], this implies that with high probability the gradient will require $O(\text{poly}(2^n))$ applications of the model to even learn the sign of any component of the gradient accurately.

Such problems it should be noted do not usually arise in classical bit encodings because of the presence of back propagation, which can rely on double precision to apply the chain rule implicitly and compute gradients of the training objective function. In quantum settings the development of quantum backpropagation algorithms does not address this issue in most cases (and in particular in quantum or amplitude-encoded input models) because of the precision requirements on the outputs. For these reasons, it is my belief that older approaches to the gradient decay problem, such as generative pre-training, may play a larger role in QML than it presently plays in classical machine learning. An examples of such generative pre-training would be to use a model like a Boltzmann machine wherein the gradients of the cross-entropy term in the KL-divergence (or more generally the quantum relative entropy) [16, 26] before switching over to a loss function appropriate for a supervised task. This can yield substantial advantages because the objective function is logarithmic and can dramatically amplify the vanishing gradient problem that such barren plateaus impose.

While generative pre-training may sometimes be used to surmount these problems, more commonly insight into the form of the data set is used to try to ensure the model weights are chosen so as to avoid the ‘chaotic regime’ where the gradients vanish [65]. Nonetheless, recognizing that many approaches to quantum neural networks may have gradients that are hard to evaluate is vitally important and stresses the significance of beginning the training process with an intelligently chosen weight distribution.

5. Conclusion

I have given a short list of the questions that I often ask myself when designing a quantum algorithm for a machine learning task. The key message behind these questions should be seen as a call for increased self-criticality. We need to know more than just how to execute our quantum algorithms: we need to know why to execute them. We need to know what the natural classical analogues are to our algorithms and we need to know what limitations physics places against us in the face of our attempts to bend the laws of nature to strengthen our machine intelligences. While QML may provide a path towards hitherto unrealized possibilities for understanding both quantum and classical datasets, we need to critically engage with ourselves to help understand the role that quantum will play in this future and provide the information needed to help those in the broader machine learning community both assess the significance of our developments and contextualize them with respect to existing knowledge. Until a deeper back and forth dialogue begins between these two camps, we will have to be our own gatekeepers and to that end ask ourselves the hard questions that will not only strengthen the science but help propel us closer to achieving the grand dreams of QML.

Acknowledgments

I would like to thank M Schuld, A Bocharov, K M Svore and A Kapoor for many useful conversations over the years regarding these topics. I would also like to thank H Neven and E Farhi for great discussions about the differences between bit encodings and amplitude encodings and the challenges faced by learning parity. This work is supported by the Google Quantum Research Award as well as the QUASAR initiative and LDRD program at Pacific Northwest National Laboratory.

References

[1] Biamonte J, Wittek P, Pancotti N, Rebentrost P, Nathan W and Lloyd S 2017 Quantum machine learning *Nature* **549** 195
[2] Schuld M, Sinayskiy I and Petruccione F 2015 An introduction to quantum machine learning *Contemp. Phys.* **56** 172–85
[3] Wittek P 2014 *Quantum Machine Learning: What Quantum Computing Means to Data Mining* (New York: Academic)
[4] Bishop C M 2006 *Pattern Recognition and Machine Learning* (Berlin: Springer)
[5] Krizhevsky A, Sutskever I and Hinton G E 2012 Imagenet classification with deep convolutional neural networks *Advances in Neural Information Processing Systems* pp 1097–105
[6] Radford A, Metz L and Chintala S 2015 Unsupervised representation learning with deep convolutional generative adversarial networks arXiv:1511.06434
[7] Goodfellow I 2016 NIPS 2016 tutorial: generative adversarial networks arXiv:1701.00160
[8] Pudenz K L and Lidar D A 2013 Quantum adiabatic machine learning *Quantum Inf. Process.* **12** 2027–70
[9] Wiebe N, Kapoor A and Svore K M 2018 Quantum nearest-neighbor algorithms for machine learning *Quantum Inf. Comput.* **15** 2027–70

[10] Aïmeur E, Brassard G and Gambs S 2006 Machine learning in a quantum world *Conf. of the Canadian Society for Computational Studies of Intelligence* (Berlin: Springer) pp 431–42

[11] Lloyd S, Mohseni M and Rebentrost P 2013 Quantum algorithms for supervised and unsupervised machine learning arXiv:1307.0411

[12] Kerenidis I and Prakash A 2016 Quantum recommendation systems arXiv:1603.08675

[13] Denil M and De Freitas N 2011 Toward the implementation of a quantum rbm *Neural Information Processing Systems (NIPS) Conf. on Deep Learning and Unsupervised Feature Learning Workshop* vol 5

[14] Wiebe N, Kapoor A and Svore K M 2016 Quantum deep learning *Quantum Inf. Comput.* **16** 541–87

[15] Amin M H, Andriyash E, Rolfe J, Kulchytskyy B and Melko R 2018 Quantum Boltzmann machine *Phys. Rev.* X **8** 021050

[16] Wiebe N and Wossnig L 2019 Generative training of quantum Boltzmann machines with hidden units arXiv:1905.09902

[17] Rebentrost P, Mohseni M and Lloyd S 2014 Quantum support vector machine for big data classification *Phys. Rev. Lett.* **113** 130503

[18] Schuld M, Sinayskiy I and Petruccione F 2015 Simulating a perceptron on a quantum computer *Phys. Lett.* A **379** 660–3

[19] Kapoor A, Wiebe N and Svore K 2016 Quantum perceptron models *Advances in Neural Information Processing Systems* pp 3999–4007

[20] Atici A and Servedio R A 2005 Improved bounds on quantum learning algorithms *Quantum Inf. Process.* **4** 355–86

[21] Aaronson S 2007 The learnability of quantum states *Proc. R. Soc.* A **463** 3089–114

[22] Arunachalam S and De Wolf R 2018 Optimal quantum sample complexity of learning algorithms *J. Mach. Learn. Res.* **19** 2878–9

[23] Schuld M, Fingerhuth M and Petruccione F 2017 Implementing a distance-based classifier with a quantum interference circuit arXiv:1703.10793

[24] Farhi E and Neven H 2018 Classification with quantum neural networks on near term processors arXiv:1802.06002

[25] Schuld M, Bocharov A, Svore K M and Wiebe N 2020 Circuit-centric quantum classifiers *Phys. Rev.* A **101** 032308

[26] Kieferová M and Wiebe N 2017 Tomography and generative training with quantum boltzmann machines *Phys. Rev.* A **96** 062327

[27] Romero J, Olson J P and Aspuru-Guzik A 2017 Quantum autoencoders for efficient compression of quantum data *Quantum Science and Technology* **2** 045001

[28] Khoshaman A, Vinci W, Denis B, Andriyash E, Sadeghi H and Amin M H 2018 Quantum variational autoencoder *Quantum Science and Technology* **4** 014001

[29] Gardas B, Rams M M and Dziarmaga J 2018 Quantum neural networks to simulate many-body quantum systems *Phys. Rev.* B **98** 184304

[30] Lloyd S, Mohseni M and Rebentrost P 2014 Quantum principal component analysis *Nat. Phys.* **10** 631–3

[31] Jerbi S, Nautrup H P, Trenkwalder L M, Briegel H J and Dunjko V 2019 A framework for deep energy-based reinforcement learning with quantum speed-up arXiv:1910.12760

[32] Servedio R A and Gortler S J 2004 Equivalences and separations between quantum and classical learnability *SIAM J. Comput.* **33** 1067–92

[33] Harrow A W, Hassidim A and Lloyd S 2009 Quantum algorithm for linear systems of equations *Phys. Rev. Lett.* **103** 150502

[34] Wiebe N, Bocharov A, Smolensky P, Troyer M and Svore K M 2019 Quantum language processing arXiv:1902.05162

[35] Wiebe N, Kapoor A, Granade C and Svore K M 2015 Quantum inspired training for Boltzmann machines arXiv:1507.02642

[36] Wiebe N, Granade C, Ferrie C and Cory D G 2014 Hamiltonian learning and certification using quantum resources *Phys. Rev. Lett.* **112** 190501

[37] Aaronson S 2018 Shadow tomography of quantum states *Proc. of the 50th Annual ACM SIGACT Symp. on Theory of Computing* pp 325–38

[38] Tang E 2018 Quantum-inspired classical algorithms for principal component analysis and supervised clustering arXiv:1811.00414

[39] Tang E 2019 A quantum-inspired classical algorithm for recommendation systems *Proc. of the 51st Annual ACM SIGACT Symp. on Theory of Computing* pp 217–28

[40] Ballentine L E, Yang Y and Zibin J P 1994 Inadequacy of Ehrenfest's theorem to characterize the classical regime *Phys. Rev.* A **50** 2854

[41] Freund Y and Schapire R E 1999 Large margin classification using the perceptron algorithm *Mach. Learn.* **37** 277–96

[42] Jin R, Hoi S C H and Yang T 2010 Online multiple kernel learning: algorithms and mistake bounds *Int. Conf. on Algorithmic Learning Theory* (Berlin: Springer) pp 390–404

[43] Giovannetti V, Lloyd S and Maccone L 2008 Quantum random access memory *Phys. Rev. Lett.* **100** 160501

[44] Arunachalam S, Gheorghiu V, Jochym-O'Connor T, Mosca M and Srinivasan P V 2015 On the robustness of bucket brigade quantum RAM *New J. Phys.* **17** 123010

[45] Arute F *et al* 2019 Quantum supremacy using a programmable superconducting processor *Nature* **574** 505–10

[46] Lloyd S, Schuld M, Aroosa I, Izaac J and Nathan K 2020 Quantum embeddings for machine learning arXiv:2001.03622

[47] Low G H and Chuang I L 2017 Hamiltonian simulation by uniform spectral amplification arXiv:1707.05391

[48] Gilyén A, Su Y, Low G H and Wiebe N 2019 Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics *Proc. of the 51st Annual ACM SIGACT Symp. on Theory of Computing* pp 193–204

[49] Aaronson S 2015 Read the fine print *Nat. Phys.* **11** 291–3

[50] Brassard G, Hoyer P, Mosca M and Tapp A 2002 Quantum amplitude amplification and estimation *Contemp. Math.* **305** 53–74

[51] LeCun Y, Bengio Y and Hinton G 2015 Deep learning *Nature* **521** 436–44

[52] Deng L 2012 The mnist database of handwritten digit images for machine learning research [best of the web] *IEEE Signal Process. Mag.* **29** 141–2

[53] Deng J, Dong W, Socher R, Li L-J, Li K and Fei-Fei L 2009 Imagenet: a large-scale hierarchical image database *2009 IEEE Conf. on Computer Vision and Pattern Recognition* (IEEE) pp 248–55

[54] Asuncion A and Newman D 2007 *Uci machine learning repository*

[55] Abadi M *et al* 2016 Tensorflow: a system for large-scale machine learning *12th USENIX Symp. on Operating Systems Design and Implementation (OSDI 16)* pp 265–83

[56] Paszke A *et al* 2019 Pytorch: an imperative style, high-performance deep learning library *Advances in Neural Information Processing Systems* pp 8026–37

[57] Green A S, Lumsdaine P L, Ross N J, Selinger P and Valiron B 2013 Quipper: a scalable quantum programming language *Proc. of the 34th ACM SIGPLAN Conf. on Programming Language Design and Implementation* pp 333–42

[58] Svore K *et al* 2018 Q# enabling scalable quantum computing and development with a high-level dsl *2018 Proc. of the Real World Domain Specific Languages Workshop* pp 1–10

[59] McKay D C *et al* 2018 Qiskit backend specifications for openQASM and openpulse experiments arXiv:1809.03452
[60] Preskill J 2018 Quantum computing in the nisq era and beyond *Quantum* **2** 79
[61] Arute F *et al* 2020 Hartree–Fock on a superconducting qubit quantum computer arXiv:2004.04174
[62] Bravyi S 2014 Monte Carlo simulation of stoquastic Hamiltonians arXiv:1402.2295
[63] Childs A M, Su Y, Tran M C, Wiebe N and Zhu S 2019 A theory of Trotter error arXiv:1912.08854
[64] Liu J-G and Wang L 2018 Differentiable learning of quantum circuit born machines *Phys. Rev.* A **98** 062324
[65] McClean J R, Boixo S, Smelyanskiy V N, Babbush R and Neven H 2018 Barren plateaus in quantum neural network training landscapes *Nat. Commun.* **9** 1–6
[66] Poulin D and Wocjan P 2009 Sampling from the thermal quantum gibbs state and evaluating partition functions with a quantum computer *Phys. Rev. Lett.* **103** 220502