Quantum machine learning in high energy physics

IOP Publishing Mach. Learn.: Sci. Technol. 2 (2021) 011003 https://doi.org/10.1088/2632-2153/abc17d

MACHINE LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
20 May 2020

REVISED
6 August 2020

ACCEPTED FOR PUBLICATION
15 October 2020

PUBLISHED
31 March 2021

TOPICAL REVIEW

Quantum machine learning in high energy physics

Wen Guan[1], **Gabriel Perdue**[2], **Arthur Pesah**[3], **Maria Schuld**[4], **Koji Terashi**[5], **Sofia Vallecorsa**[6] and **Jean-Roch Vlimant**[7]

[1] University of Wisconsin-Madison, Madison, WI, 53706, United States of America
[2] Fermi National Accelerator Laboratory, Fermilab Quantum Institute, PO Box 500, Batavia, IL, 60510-0500, United States of America
[3] Technical University of Denmark, DTU Compute, Lyngby, Denmark
[4] University of KwaZulu-Natal School of Chemistry and Physics, Durban, ZA 4000, South Africa
[5] ICEPP, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, JP 300-1153, Japan
[6] CERN IT, 1, Esplanade des Particules, Geneva, CH 1211, South Africa
[7] California Institute of Technology, PMA, Pasadena, CA, 91125-0002, United States of America

E-mail: jvlimant@caltech.edu

Keywords: particle physics, quantum machine learning, quantum annealing, quantum circuit, quantum variational circuit

Abstract

Machine learning has been used in high energy physics (HEP) for a long time, primarily at the analysis level with supervised classification. Quantum computing was postulated in the early 1980s as way to perform computations that would not be tractable with a classical computer. With the advent of noisy intermediate-scale quantum computing devices, more quantum algorithms are being developed with the aim at exploiting the capacity of the hardware for machine learning applications. An interesting question is whether there are ways to apply quantum machine learning to HEP. This paper reviews the first generation of ideas that use quantum machine learning on problems in HEP and provide an outlook on future applications.

1. Introduction

Particle physics is a branch of science aiming to understand the fundamental laws of nature by studying the most elementary components of matter and forces. This can be done in controlled environments with particle accelerators such as the Large Hadron Collider (LHC), or in uncontrolled environments such as cataclysmic events in the cosmos. The Standard Model of particle physics is the accomplishment of decades of theoretical work and experimentation. While it is an extremely successful effective theory, it does not allow the integration of gravity, and is known to have limitations. Experimentation in particle physics requires large and complex datasets, which poses specific challenges in data processing and analysis.

Recently, machine learning has been played a significant role in the physical sciences. In particular, we are observing an increasing number of applications of deep learning to various problems in particle physics and astrophysics. Beyond typical classical approaches [1] (boosted decision tree (BDT), support vector machine (SVM), etc), state-of-the-art deep learning techniques (convolutional neural networks, recurrent models, geometric deep learning, etc) are being successfully deployed in a variety of tasks [2, 3].

The ambitious high luminosity LHC (HL-LHC) program in the next two decades and beyond will require enormous computing resources. It is interesting to ask whether new technologies such as quantum machine learning could possibly help overcome this computational challenge. The recent development of quantum computing platforms and simulators available for public experimentation has lead to a general acceleration of research on quantum algorithms and applications. In particular, quantum algorithms have recently been proposed to tackle the computational challenges faced in particle physics data processing and analysis. Beyond explicitly writing quantum algorithms for specific tasks [4–8], quantum machine learning is a way to learn quantum algorithms to achieve a specific task, similarly to *classical* machine learning.

This review paper of how quantum machine learning is used in high energy physics (HEP) is organized as follows. An overview of the fields of quantum computing and quantum machine learning are first provided in sections 2 and 3. We review the applications of quantum machine learning algorithms for particle physics using quantum annealing QA in Sections 4 and quantum circuits in section 5. We provide a

field-wide view of unpublished work and upcoming results in section 6. We conclude with discussions on the future of quantum machine learning applications in HEP in section 7.

2. Quantum computing

More than three decades after Richard Feynman's proposal of performing simulations using quantum phenomena [9], the first practical quantum computers are finally being built. The scope of calculations has significantly expanded, with a range of promising applications emerging, including optimization [10–12], chemistry [13, 14], machine learning [15–17], particle physics [4–8], nuclear physics [18–20] and quantum field theory [21–24].

2.1. Quantum circuit model

Quantum computers were formally defined for the first time by David Deutsch in his 1985 seminal paper [25], where he introduced the notion of a *quantum Turing machine*, a universal quantum computer based on qubits and quantum circuits. In this paradigm, a typical algorithm consists of applying a finite number of quantum gates (unitary operations) to an initial quantum state, and measuring the expectation value of the final state in a given basis at the end. Deutsch found a simple task that would require a quantum computer less steps to solve than all classical algorithms, thereby showing that quantum Turing machines are fundamentally different and can be more powerful than classical Turing machines. Since then, many quantum algorithms with a lower computational complexity than all known classical algorithms have been discovered, the most well-known example being Shor's algorithm to factor integers exponentially faster than our best classical algorithm [26]. Other important algorithms include Grover's algorithm invented in 1996 to search an element in an unstructured database with a quadratic speed-up [27], and the Harrow-Hassidim-Lloyd algorithm, invented in 2008 to solve linear systems of equations [28].

However, all those algorithms require large-scale fault-tolerant quantum computers to be useful, while current and near-term quantum devices will be characterized by at least three major drawbacks:

(a) Noise: the *coherence time* (lifetime) of a qubit and the *fidelity* of each gate (accuracy of the computation) have increased significantly during the past years, but are still too low to use the devices for applications beyond small proof-of-principle experiments involving only a few qubits—even if tricks like error-mitigation are used (see for example [29]).
(b) Small number of qubits: current near-term quantum computers consist of between 5 and 100 qubits, which is not enough for traditional algorithms such as Shor's or Grover's to achieve a quantum advantage over classical algorithms. While steady improvements are made, increasing the number of qubits is not just a matter of scaling current solutions: Problems of connectivity, cross-talk, and the consistent quality of qubits require new engineering approaches for larger systems.
(c) Low connectivity: most current quantum devices have their qubits organized in a certain lattice, where only nearest-neighbors can interact. While it is theoretically possible to run any algorithm on a device with limited connectivity—by 'swapping' quantum states from qubit to qubit—the quantum advantage of some algorithms can be lost during the process [30].

Therefore, a new class of algorithms, the so-called near-term intermediate-scale quantum (NISQ) algorithms [31], have started to emerge, with the goal of achieving a quantum advantage with those small noisy devices. One of the main classes of NISQ algorithms is based on the concept of *variational circuits*: fixed-size circuits with variable parameters that can be optimized to solve a given task. They have shown promising results in quantum chemistry [13] and machine learning [32] and will be discussed in more detail in section 3.1.

2.2. Quantum annealing

Another paradigm of quantum computing, called *adiabatic quantum computing* (or *quantum annealing*, QA) was introduced several years after the gate model described above [33, 34] and has been implemented by the company D-Wave. In theory, this paradigm is computationally equivalent to the circuit model and Grover's algorithm can for instance be ported to QA [35]. It is based on the continuous evolution of quantum states to approximate the solution of *quadratic unconstrained binary optimization* (QUBO) problems, of the form

$$\min_{\boldsymbol{x}} E(\boldsymbol{x}) = \sum_{i,j=1}^{n} J_{ij} x_i x_j + \sum_{i=1}^{n} h_i x_i \tag{1}$$

where $x_i \in \{0, 1\}$ and J_{ij} and h_i are real numbers defining the problem.

This general problem belongs to the complexity class NP-Hard, meaning that it can probably not be solved exactly in polynomial time even by a quantum computer[8]. QA is a heuristic proposed to approximate the solution of a QUBO problem, or even solve it exactly when the input parameters J_{ij} and h_i have some particular structures [35].

More precisely, solving a QUBO instance is equivalent to finding the ground-state of the problem Hamiltonian:

$$H_{\mathrm{P}} = \sum_{i,j=1}^{n} J_{ij}\sigma_i^z\sigma_j^z + \sum_{i=1}^{n} h_i\sigma_i^z \tag{2}$$

where σ_i^z is the Z-Pauli matrix applied to the i^{th} qubit. QA consists of initializing the system in the ground-state of a simpler Hamiltonian, such as

$$H_{\mathrm{I}} = \sum_{i=1}^{n} \sigma_i^x \tag{3}$$

and slowly evolving the system from H_I to H_P during a total time T, for instance by changing the Hamiltonian along the trajectory:

$$H(t) = \left(1 - \frac{t}{T}\right) H_I + \frac{t}{T} H_{\mathrm{P}} \tag{4}$$

The quantum adiabatic theorem tells us that if the transition between the two Hamiltonians is 'slow enough', the system will stay in the ground-state during the whole trajectory, including at the end for our problem Hamiltonian. Measuring the final state will therefore give us the solution to our QUBO problem. The main caveat of this approach is that the maximum allowed speed of the evolution can fall rapidly with the system size (sometimes exponentially low), removing any potential advantage compared to classical algorithms. Knowing if a given problem (or class of problems) can take advantage of QA is an open research question, which is why research on QA applications has been driven largely by empirical studies.

Many optimization problems, including in machine learning, can be mapped to a QUBO instance, making QA an attractive platform for quantum machine learning, as developed in section 3.2.

3. Quantum machine learning

Quantum machine learning has evolved in recent years as a subdiscipline of quantum computing that investigates how quantum computers can be used for machine learning tasks—in other words, how quantum computers can learn from data [17, 36]. One can approach this question in three different ways, which reflect similar angles established in quantum computing:

- the foundational approach that reformulates learning theory in a quantum setting [37, 38];
- efforts to find quantum algorithms that speed up machine learning with regards to computational complexity measures [39–42];
- a near-term perspective that develops new machine learning applications tailor-made for NISQ devices [43].

Currently, classical machine learning is a distinctively empirical discipline, pioneered by research conducted in industry. It is therefore not surprising that quantum machine learning research is also dominated by the near-term perspective, a fact reflected in the selection of papers discussed in this review.

The near-term perspective of quantum machine learning starts from the quantum devices available today and asks how they can be used to solve a machine learning problem. Circuit-based quantum computers have been predominantly used to *compute the prediction* of a quantum machine learning model that can be trained classically [32, 44], while quantum annealers have been proposed to *optimize* classical models [45, 46].

3.1. Quantum circuits as trainable models

A machine learning model can often be written as a function $f(x, \theta)$ that depends on an input data point x—for example describing the pixels of an image or a vectorized text document—as well as trainable parameters θ. The result of the model, f, is interpreted as a prediction, e.g. revealing the label of x in a classification task. For simplicity, we will here assume a scalar output.

[8] While a proof is still to be found, complexity theorists believe that quantum computers will not lead to exponential speed-ups for NP-Complete or NP-Hard problems.

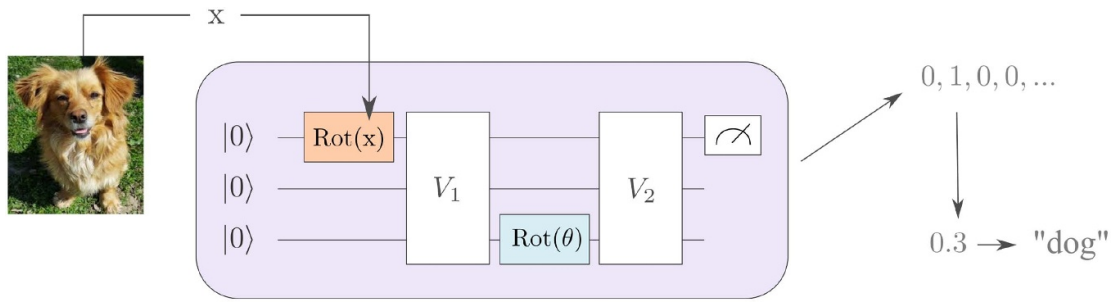

Figure 1. Simplified example of a variational quantum circuit used for classification. A feature x from the input data is loaded into the circuit by associating it with the angle of a rotation gate. The angle θ of another rotation gate is used as a variable parameter that can be trained to adjust the circuit. The three qubits are represented in the standard circuit notation as wires, and gates are represented by symbols acting on the wires. The unitaries V_1 and V_2 summarise arbitrary quantum operations applied to the qubits. The first qubit is measured in the end, and an expectation value is computed by averaging over measurement results. This expectation is interpreted as the prediction of a quantum model.

We know from the basics of quantum mechanics that the result of a quantum circuit is a measurement with a probabilistic outcome—for example, a qubit measured in state $|0\rangle$ or $|1\rangle$. However, the *expectation value* of a quantum observable—a central concept in quantum theory—is a deterministic value. In simple terms, the expectation value is the weighted average of a measurement result. For example, after taking 1000 measurements ('shots') of a qubit, of which 900 resulted in the outcome $|1\rangle$ an estimate of the expectation of the qubit state would be 0.9. We can interpret this expectation as a prediction, and the quantum circuit is thereby serving as a *quantum classifier* or *quantum machine learning model*.

But how do we make the output of the quantum model depend on inputs x and trainable parameters θ? The central idea is to associate physical control parameters with the input features and individual parameters. For instance, in most circuit-based quantum computers we have control over the rotation angle of qubits. Assuming for now that x is a single scalar, we can therefore rotate one qubit by an angle of exactly x to encode the input[9]. Using the same strategy for a parameter θ, considered to be a scalar as well for now, we can rotate another (or the same) qubit by an angle θ. Physically, there is no difference in how the inputs and free parameters are treated, but there are profound conceptual differences; see for example [47]. These rotations can be performed as part of a larger quantum algorithm that consists of other gates, and which is described by an overall unitary $U(x, \theta)$ that depends on the input and parameter (see figure 1). The crux is that now the expectation value of the circuit with respect to an observable M is formally given by

$$f_q(x, \theta) = \left\langle 0 \middle\| U(x, \theta)^\dagger M U(x, \theta) \middle\| 0 \right\rangle,$$

and can be interpreted as the prediction of x. In short, the quantum circuit is used as a machine learning model.

Of course, the heart of machine learning is to adapt a model to data. The circuit can be trained by adjusting the parameters θ by a classical optimization routine that minimizes a standard cost function comparing predictions with the correct target outputs, such as the mean square loss. Trainable circuits are also known as *variational* or *parametrized* circuits (or sometimes, a bit misleadingly, as *quantum neural networks*), and were initially proposed in the context of quantum chemistry [48]. The optimization can be performed by using the quantum computer to evaluate $f_q(x, \theta)$ at different values for θ, and using a classical co-processor to find better candidates for the parameter with respect to the cost function, using either gradient-free or finite-difference based optimization methods.

Inspired from quantum control, quantum machine learning has recently developed an elaborate framework of gradient-based optimization [49, 50] that has already been implemented in powerful software frameworks [51, 52], which may prove superior to gradient-free methods when quantum computers get bigger [53]. An essential result was to notice that in many cases used in practice, one can compute the analytic or exact gradient from $f_q(x, \theta + s)$ and $f_q(x, \theta - s)$, where s is a constant which depends on the way that θ enters the quantum circuit—in other words, which gate is used to encode the parameter. While this is reminiscent of a finite-difference rule, the important fact is that s is a macroscopic variable such as $\pi/2$, which makes estimating the two values by repeated measurements on a noisy device possible. Furthermore, the resulting gradient is not an approximation, but the true analytic gradient. The ability to compute

[9] Note that x has to be rescaled to lie in the interval $[0, 2\pi]$ for the encoding to be unique.

gradients of variational circuits has potential consequences that reach far beyond quantum machine learning, since it makes quantum computing amenable for the paradigm of *differentiable programming*.

Finally, it should be mentioned that there are many other ways that variational circuits are employed in quantum machine learning. For example, the genuinely probabilistic nature of quantum measurements suggests that variational circuits can be used as an ansatz for *generative models*. In the generative mode, the result of a quantum measurement is interpreted as a sample of a probabilistic machine learning model that defines a probability distribution over the data that may depend on parameters [54, 55]. This has amongst other proposals led to *quantum generative adversarial networks* [56, 57].

3.2. Quantum annealers as optimizers

Quantum annealers represent a different approach to quantum machine learning. As natural optimizers, they outsource the *training part* of machine learning to quantum computers, rather than the *prediction part*. Since quantum annealers solve very specific optimization problems, more precisely QUBO problems (see equation (1)), the central challenge is to rephrase the loss function of a (quantum) machine learning problem in this format.

For example, an interesting and very early proposal [46] recognized that the mean square loss of an ensemble of perceptrons—the simple building blocks of neural nets—can be written as a QUBO problem. A prerequisite is that the weights of the model have to be binary values—a condition that may even offer advantages for machine learning. The approach has been termed *QBoost* and tested in one of the first commercial quantum annealers, the D-Wave machine, as early as in 2009 [58]. Other proposals to use the QUBO structure of quantum annealers for machine learning have been proposed for anomaly detection, in particular software verification and validation [59].

Another, slightly different idea uses quantum annealers as samplers to support classical training of classical models [45]. In the training of so-called Restricted Boltzmann Machines (RBMs), samples from a Gibbs distribution are required to find better candidates for the parameters in every step. The intimate connections between RBMs and Ising-type models in many-body physics (see also [60] which reveals this connection through the language of tensor networks) suggest that quantum annealers, which are based on interacting spins, can produce samples from such Gibbs distribution. The details, especially when it comes to real hardware, are non-trivial, but successful quantum-assisted training has been demonstrated for small applications [45]. An important question raised as a result of this strategy was how samples from true quantum distributions, such as the Ising model with a transverse field, can be used to train *quantum RBMs* [61].

4. QA applications

For quantum annealers, the two most common approaches to machine learning involve mapping the problem into an optimization problem over the full dataset, and using the quantum device as a sampling engine to solve a difficult gradient calculation problem. In this section, we review papers that provide examples of these paradigms, we refer the reader to [62–64] for more in-depth reading.

4.1. Di-photon event classification

The classification of collision events into signal or background categories is one of the main tasks in particle physics, and a frequent application for machine learning. The Higgs boson, until its discovery in 2012 [65, 66], was the missing piece of the standard model. The authors of [62] propose the use of QA to classify events between a Higgs decaying to a pair of photons and irreducible background events where two uncorrelated photons are produced. To this end, eight high level features are measured from the di-photon system. With a view to using the method proposed in [59]—so called *quantum adiabatic machine learning (QAML)*—a list of weak classifiers is computed from those eight features. Using the eight features and their products as input, $n = 36$ weak classifiers ($c_i(x_\tau)$) are computed. The weak classifiers assume values in the range $[-1, 1]$—the signal being represented by positive values. A strong classifier is then constructed from a binary linear combination of the weak classifiers (with parameter $w_i \in \{0, 1\}$ for each weak classifier index i).

The parameters w_i are then determined by the optimization of a carefully crafted QUBO:

$$E(\boldsymbol{w}) = \sum_{i,j=1}^{n=36} C_{ij} w_i w_j + \sum_{i=1}^{n=36} 2(\lambda - C_i) w_i \tag{5}$$

where $C_{ij} = \sum_\tau c_i(x_\tau) c_j(x_\tau)$ and $C_i = \sum_\tau c_i(x_\tau) y_\tau$ are computed from the values of the weak classifiers in the training set and their category ($c_i(x_\tau)$ and y_τ respectively). λ is a parameter penalizing solutions for too many weak classifiers participating. As described in section 2.2, the QUBO is transformed in a problem

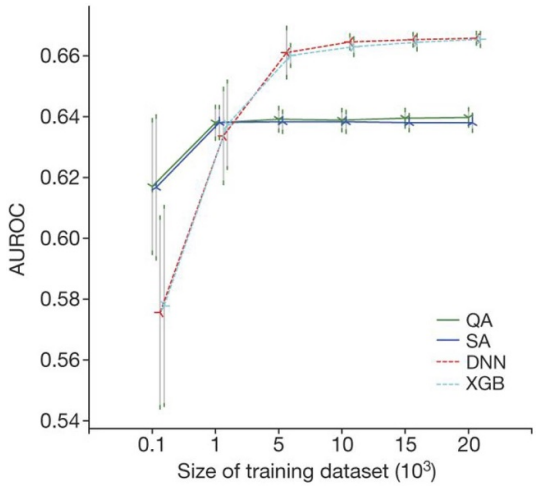

Figure 2. Area under the ROC curve (AUROC) of the strong classifier optimized on quantum annealer (QA) and simulated annealing (SA), together with the performance of boosted decision tree (BDT) and deep neural network (DNN) classifiers trained on the input features [62]. Reprinted by permission from Springer Nature Customer Service Centre GmbH: [Nature] [Nature] [62], (2017).

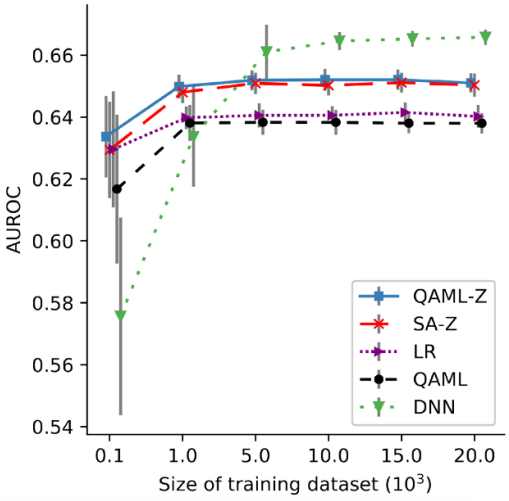

Figure 3. Area under the ROC curve (AUROC) of original QAML method, the continuous strong classifier optimized on quantum annealer (QAML-Z) and simulated annealing (SA-Z), together with the performance of a linear regression (LR) and deep neural network (DNN) classifiers [63]. Reprinted (figure) with permission from [63], Copyright (2019) by the American Physical Society.

Hamiltonian H_P (see equation (2)) with the change of variable $\sigma_i^z \leftarrow 2w_i - 1$, and further embedded in a machine Hamiltonian to be solved on the device. The set of parameters w_i^* obtained through this optimization defines an optimal strong classifier as constructed above.

The final performance of the strong classifier is compared with two classical machine learning methods: BDT and deep neural network (DNN). The authors note that importance ranking can be obtained among the weak classifiers, by varying the parameter λ. The optimization is both run on the D-Wave 2X quantum annealer system and performed with simulated annealing [67, 68] (SA) using variable fractions of the training dataset. While SA is accurately finding the same ground truth found by QA, it is unable to reproduce the excited states measured with QA. Therefore the inclusion of the excited states in the construction of the strong classifier with QA brings a slight, although not conclusive, difference in performance compared to the one derived with SA. SA and QA are typically *on par*, and not providing obvious classification advantage over BDT and DNN (see figure 2), although a slight advantage with a small training dataset is noted.

In [63], the binary linear combination ($w_i \in \{0, 1\}$) is extended to a continuous linear combination (denoted $\mu_i \in [0, 1]$ to avoid confusion) by running the optimization in an iterative manner. In order to take advantage of the continuous weights, additional weak classifiers, up to N_w in total, are derived from the existing ones. A new classifier is obtained from an existing one by shifting its value by a multiple of a given predefined shift, keeping the distribution clipped to the $[-1, 1]$ interval.

The real parameters μ_i are obtained using the iterative rule

$$\mu_i(0) = 0 \ ; \ \mu_i(t+1) = \mu_i(t) + \sigma_i^z(t)2^{-(t+1)} \tag{6}$$

where $\sigma_i^z(t)$ is the result of the optimization of the same Hamiltonian as in the binary case, evaluated under the change of variable

$$\sigma_i^z \leftarrow \mu_i(t) + \sigma_i^z(t)2^{-t}. \tag{7}$$

We refer the reader to [63] for more details. A bit flip heuristic is introduced between each iteration, with decreasing probability, as a regularization measure. The authors note that there might be other such heuristic that could provide a better final accuracy. The size of the problem Hamiltonian compared to the connectivity of the hardware is such that the authors prune cross-terms with low values and use a procedure provided by D-Wave to partially solve the optimization. The proposed hybrid algorithm (so called QAML-Z) outperforms QAML while remaining without an accuracy advantage over classical approaches (see figure 3). Here again results obtained using SA and QA are *on par*. The scheme under which a discrete optimization is used iteratively as an approximation of continuous optimization using quantum annealers opens new directions for future algorithms.

4.2. Classification in cosmology with quantum RBM

Quantum annealers do not provide identical answers every time they go through an annealing cycle. For some applications it would be ideal if, for example, they always returned the lowest energy configuration, but instead they produce a distribution of states. In principle, these states are Boltzmann-distributed with a characteristic temperature related to the physical device temperature. In practice, the actual distribution of states deviates from a Boltzmann distribution (on the D-Wave 2000Q, for example, it is 'colder' and tends to skew towards lower than expected energies). However, with some post-processing the sample distribution may be converted into a Boltzmann distribution. It may be also anecdotally observed that while the sampled distribution is not Boltzmann-distributed, simply applying the parameter update equations derived under the assumption of sampling from a Boltzmann distribution (see below, equations (9) through 11) will generally allow the model to converge anyway [69, 70].

Taken together, these observations mean that quantum annealers may also be used as sampling engines to fuel certain classes of machine learning algorithms. RBMs map well to modern quantum annealers for this purpose. They feature a bipartite connectivity graph that scales well in embedding algorithms as compared to a fully connected graph. The tunable couplings between qubits function as graph connection weights and the annealing process naturally samples from the graph configurations with clamped or unclamped values for the visible nodes in the graph as needed by the application.

RBMs are fundamentally generative models that approximate a target distribution over an array of *visible* binary variables ($\vec{v}$) as the marginal distribution of a bipartite graph that connects to a different set of *hidden* binary variables ($\vec{h}$). The distribution is described by

$$p(\vec{v}, \vec{h}) \propto \exp(-v^T W h + b^T v + c^T h) \tag{8}$$

for some parameter (bias) vectors $\vec{b}$, $\vec{c}$, and a connections weight matrix W.

RBMs are trained by maximizing the log-likelihood of a data distribution by updating the bias and weights parameters. With the loss (L) defined as the negative log-likelihood , the derivatives for the model parameters are

$$\frac{\partial L}{\partial b^i} = \langle v^i \rangle_{\text{data}} - \langle v^i \rangle_{\text{model}} \tag{9}$$

$$\frac{\partial L}{\partial c^i} = \langle h^i \rangle_{\text{data}} - \langle h^i \rangle_{\text{model}} \tag{10}$$

$$\frac{\partial L}{\partial W_i^j} = \langle v^i h^j \rangle_{\text{data}} - \langle v^i h^j \rangle_{\text{model}}. \tag{11}$$

These derivatives form a gradient for use in gradient descent for adjustments to $\vec{b}$, $\vec{c}$, and W. The expectations are computed over the data (the training set) with clamped values and over the model with unclamped

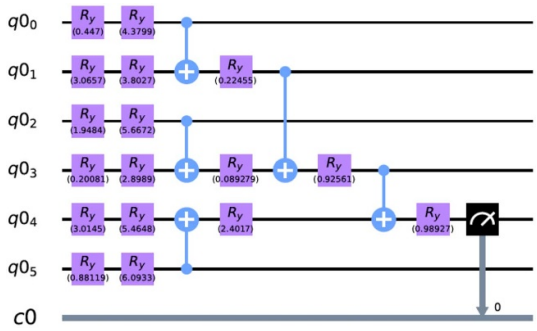

Figure 4. The quantum edge network implemented as a tree tensor network, a hierarchical quantum classifier [76]. The architecture uses R_y rotation gates and CNOT gates. A single output qubit is measured [72]. Reproduced with permission from [72]. (2020).

values. These steps are also referred to as the positive and negative phases. See [71] for a particularly clear explanation.

While computing the expectations over the data is easy, computing the expectations over the model is costly, as that scales like $2^{\min(n_v,n_h)}$, with n_v and n_h equal to the number of visible and hidden units, respectively. There are a number of mitigation strategies to avoid this difficult computation, all discussed in [64]. Of particular relevance here, the expectations for a given set of model parameters using unclamped variables on a D-Wave can be computed, where each computed configuration is sample from the machine's output distribution. For small graphs this approach is impractical but it may eventually offer some computational advantage for very large graphs.

In practice, the distribution of states returned by the D-Wave 2000Q is not Boltzmann distributed, and significant post-processing is required to achieve a Boltzmann distribution. As observed in [64], the D-Wave offers essentially no sampling advantage over random string initial states if using only Boltzmann distributions for the optimization. However, it has been observed that RBMs may be optimized with imperfect gradients [69]. Therefore, it is possible to greatly reduce the amount of required post-processing and still train effective models.

For the task of galaxy morphology classification, in [64] it was observed that RBMs, regardless of the training methods, were less effective than gradient boosted trees (likely the best classical algorithm for structured data like the dimensionality reduced galaxy images). Additionally, the best classical methods for discriminative training outperformed the quantum, generative training. However, regardless of training strategy, RBMs offered a performance advantage for very small datasets that gradient boosted trees and logistic regression tended to badly overfit. Furthermore, early in the small dataset training runs, the quantum generative training outperformed the classical discriminative training.

5. Quantum circuit applications

As introduced in section 3.1, circuits with varying parameters can be optimized to perform a specific task, e.g. classification. The parameters of these circuits can be determined with gradient-based optimization method. The following papers are following this approach for HEP specific classification tasks. We refer the reader to [72–74] for more in-depth reading.

5.1. Quantum GNNs for particle track reconstruction

Quantum computers promise to greatly speed-up search in large parameter spaces. Charge particle tracking — *tracking* in short—is the task of associating sparse detector measurements (a.k.a 'hits') to the particle trajectory they belong to. Tracking is the cornerstone of event reconstruction in particle physics. Because of their ability to evaluate a very large number of states simultaneously, they may play an important role in the future of track reconstruction in particle physics experiments. Reconstructing particle trajectories with high accuracy will be one of the major challenges in the HL-LHC experiments [75].

Increase in the expected number of simultaneous collisions and the high detector occupancy will make tracking extremely demanding in terms of computing resources. State-of-the-art algorithms rely, today, on a Kalman filter-based approach: they are robust and provide good physics performance, however they are expected to scale worse than quadratically with the increasing number of simultaneous collisions [75]. The high energy physics community is investigating several possibilities to speed up this process [77–79] including deep learning-based techniques. For instance, introducing an image-based interpretation of the

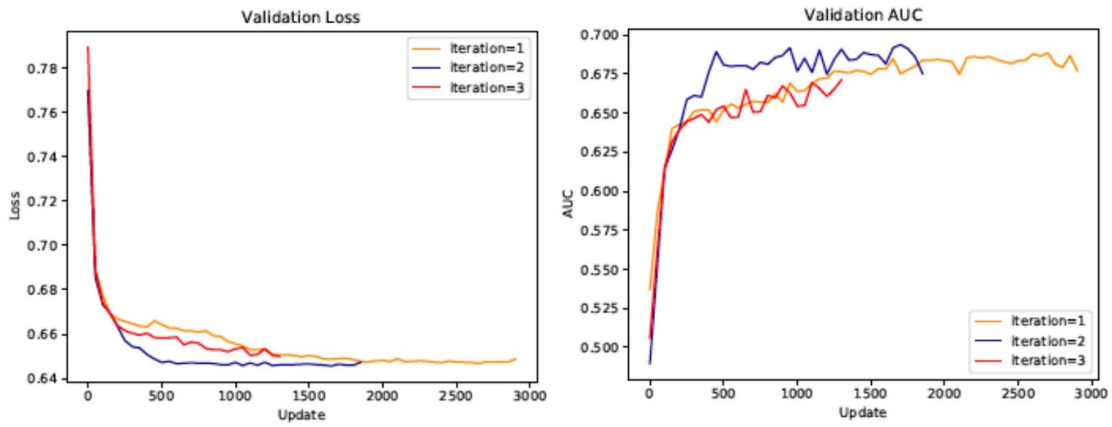

Figure 5. QGNN performance. The validation loss (on the left) decreases smoothly. Consistently, the validation accuracy (on the right) increases with the number of iterations. Results are shown for two epochs corresponding to 2900 steps (1 epoch = 1450 updates) [72]. Reproduced with permission from [72]. (2020).

detector data and using convolutional neural networks can lead to high-accuracy results [80]. At the same time, a representation based on space-points arranged in connected graphs could have an advantage given high dimensionality and sparsity of the tracking data. The HEPtrkX project [80] followed this approach and successfully developed a set of graph neural networks (GNNs) to perform hits and segments classification. In this approach, graphs of connected hits are built, features of the graph nodes and edges are computed and, finally, relevant hit connections are predicted. The dataset, designed for the TrackML challenge [81] contains precise locations of hits, and the corresponding particles. The classical GNN architecture consists of three networks organised in cascade: an *input network* encodes the hits information as node features, an *edge network* outputs edge features, using the start and end nodes, and a *node network*, that calculate hidden nodes features taking into account all connected nodes on the previous and next layers. The edge and node networks are applied iteratively after the input network (see [82] for more details). The work in [72] represents an exploratory look at this GNN architecture from a quantum computing perspective: it re-implements the input, edge and node networks as quantum circuits.

In particular, the edge and node networks are implemented as tree tensor networks (TTN) — hierarchical quantum classifiers originally designed to represent quantum many body states described as high-order tensors [76]. The data points are encoded (see figure 4) as parameters of R_y rotation gates:

$$R_y(\theta)\,|0\rangle = \cos(\theta/2)\,|0\rangle + \sin(\theta/2)\,|1\rangle. \tag{12}$$

The TTN network consists of R_y rotations and CNOT gates (see figure 4) and its output is the measurement from a single qubit. The TTN has 11 parameters which are the angles of rotations in Y direction on the Bloch sphere. These parameters are optimized using the ADAM optimiser and a binary cross entropy loss function using Pennylane [51] and Tensorflow [83]. The model is trained on 1450 subgraphs extracted from the TrackML dataset.

Although preliminary, the obtained performance (see figure 5) is promising: the validation losses decrease smoothly and the accuracy increases with the number of iterations. At convergence, the accuracy value is still lower than for the classical case. This is, however, expected as the number of hidden features, and iteration are reduced compared to the GNN, because of computation issues.

5.2. Classification using variational quantum circuits

The method used in [73] and [74] is based on variational quantum algorithms for machine learning (VQML). The VQML approach exploits the mapping of input data to an exponentially large quantum state space to enhance the ability to find an optimal solution. The data encoding circuit $U_{\Phi(\vec{x})}$ maps the data $\vec{x} \in \Omega$ to the quantum state $|\Phi(\vec{x})\rangle = U_{\Phi(\vec{x})}|0\rangle$. The quantum state with encoded input data is processed by applying quantum gates to create an ansatz state, which is then measured to produce the output. The variational quantum circuit $W(\vec{\theta})$ parameterized by $\vec{\theta}$ is applied [84]

$$W(\vec{\theta}) = U^{(l)}_{\rm loc}(\theta_l)\; U_{\rm ent} \ldots U^{(2)}_{\rm loc}(\theta_2)\; U_{\rm ent}\; U^{(1)}_{\rm loc}(\theta_1) \tag{13}$$

The probability of outcome y is obtained through

$$p_y(\vec{x}) \leftarrow \langle \Phi(\vec{x})|W^{\dagger}(\vec{\theta})M_y W(\vec{\theta})|\Phi(\vec{x})\rangle \tag{14}$$

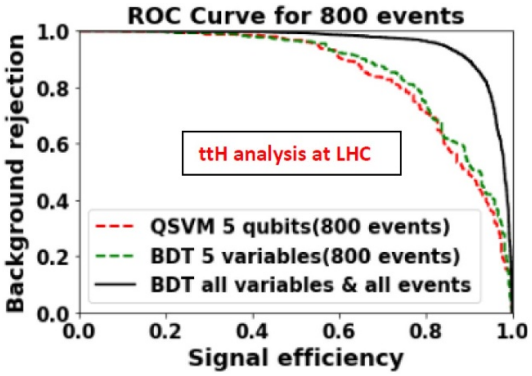

Figure 6. ROC curve of VQML and BDT methods. With 800 events and 5 qubits, the VQML have obtained very close performance to the one obtained using the classical machine learning method BDT [73]. Reproduced from [73]. CC BY-NC-ND 4.0.

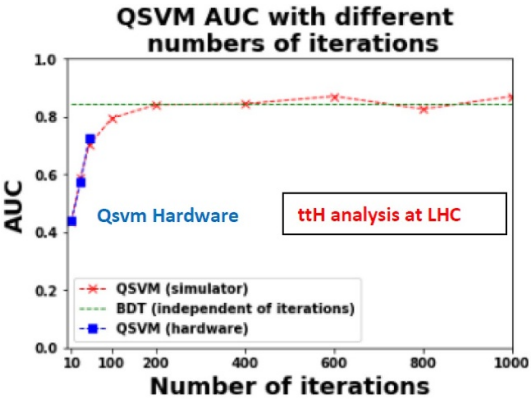

Figure 7. AUC with number of iterations. Within the limited testing iterations,the performance of the IBM Q quantum computer is compatible with the one from the quantum simulator [73]. Reproduced from [73]. CC BY-NC-ND 4.0.

whereas $\{M_y\}$ is the binary measurement. The optimization process consists in learning $\vec{\theta}$ to minimize the loss quantified as a difference between the predicted $p_y(\vec{x})$ and the known classification label y. Different optimizers, such as COBYLA [85] and SPSA [86, 87], can be applied.

In [73], the authors made some promising progress by obtaining preliminary results in the application of IBM quantum simulators and IBM Q quantum computer to ttH (Higgs coupling to top quark pairs) data analysis. The authors have measured the AUC (area under the ROC curve) with different numbers of events in the training dataset. With 5 qubits and 800 events, the VQML have obtained very close performance to the one obtained using the classical machine learning method BDT (see figure 6). A preliminary test was to perform VQML on the IBM Q quantum computer with 5 qubits, 100 training events and 100 test events. Within the limited testing iterations, the performance of the IBM Q quantum computer is compatible with the one from the quantum simulator, which reaches a performance similar to the BDT method with enough iterations (see figure 7).

In [74], the authors have attempted to use the VQML algorithm for the classification of a new physics signal predicted in a theory of supersymmetry. Two implementations of the VQML algorithm are tested, the first one called quantum circuit learning (QCL) [88], which is used with the Qulacs simulator [89], and the second called variational quantum classification (VQC) [84], which is used with the QASM simulator and real quantum computing devices. The QCL (VQC) uses the combination of R_Y and R_Z (Hadamard and R_Z) gates for encoding the input data. For the creation of an ansatz state, the combination of an entangling gate and single-qubit rotation gates are used for both implementations. The QCL uses the time-evolution gate e^{-iHt} with the Hamiltonian H of an Ising model with random coefficients as an entangling gate while the VQC uses the Hadamard and CNOT gates for that. The rotation angles used to create the ansatz are parameters to be tuned, and the number of parameters is chosen to be 27, 45 and 63 for the QCL and 12, 20 and 28 for the VQC using 3, 5 and 7 variables, respectively.

The experimental test of the quantum algorithm is performed in [74] with the SUSY data set in the UC Irvine Machine Learning Repositiory [90] using cloud Linux servers for the QCL and a local machine and

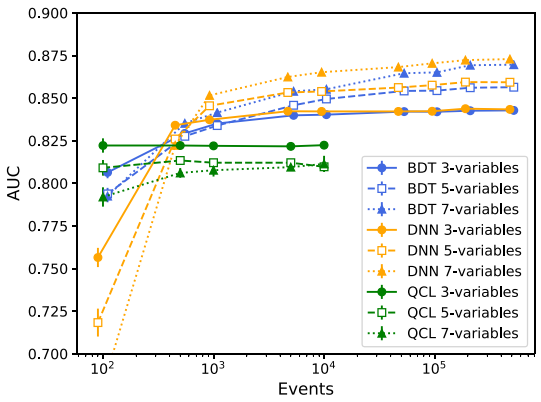

Figure 8. Average AUC values as a function of the training sample size for the BDT, DNN and QCL algorithms with 3, 5 and 7 variables [74]. Reprinted by permission from Springer Nature Customer Service Centre GmbH: [Springer] [high-energy physics] [74] (2020).

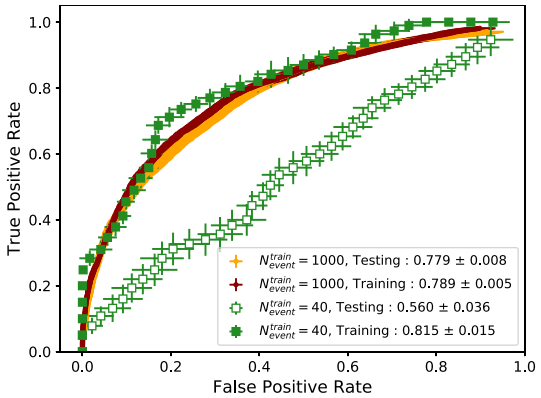

Figure 9. ROC curves in the training and testing phases of the VQC algorithm with 3 variables and the training sets of 40 and 1 000 events, obtained using QSM simulator [74]. Reproduced from [74]. CC BY-NC-ND 4.0.

Table 1. AUC values in a training phase for the VQC algorithm running on quantum computers and QASM simulator. The training condition is fixed to three variables, 40 training events and the number of iterations of 100 [74]. Reproduced from [74]. CC BY-NC-ND 4.0.

Device/Condition	AUC
Quantum Computer (Johannesburg)	0.799 ± 0.020
Quantum Computer (Boeblingen)	0.807 ± 0.010
QASM simulator	0.815 ± 0.015

the IBM Q quantum computer for the VQC. The performance of the quantum algorithm is compared with BDT and DNN optimized to avoid over-training at each training set. The QCL performance is relatively flat in the training size (see figure 8) while the performance of the BDT and DNN improves with the size. The computational resource needed to simulate QCL with 10 000 events or more is beyond the capacity used in [74]. According to these simulation studies, the three algorithms appear to have a comparable discriminating power when restricting the training set to be less than ~10 000 events, with an indication that the quantum algorithm might have an advantage with a small sample of $\mathcal{O}(100)$ events. Figure 8 shows ROC curves obtained using the 3-variable VQC algorithm on the QASM simulator with different numbers of events in the training set. The over-training is clearly visible if the training set contains only 40 events while it is largely gone when the training set is increased to 1 000. The small sample of 40 events is used to train the VQC model with IBM Q quantum computers as well. The AUC values from the QASM simulator and quantum computers are given in table 1. The results from the quantum computers appear to be slightly worse than those from the simulator, though they are consistent within the uncertainties (defined as the standard deviations of five measurements). The authors of [74] conclude that the variational quantum circuit can learn the properties of the input data with real quantum device, acquiring classification power for physics events of interest.

6. Applications coming soon

An interesting line of research concerns generative models, such as Boltzmann machines, variational auto-encoders and generative adversarial networks, and their quantum counterparts. Classical generative models are being investigated by the HEP community as solutions to speed up Monte-Carlo simulation, because of their ability to model complex probability distributions, and the relative lower computation cost during the prediction phase. Training those models is, however, a difficult task, and computing intensive. Coverage is one of the major issues when training or validating generative models performance and it is related to their representational power and how it maps to the original probability distribution. From this point of view quantum generative models might show an advantage, while relieving the computational cost [91].

Quantum SVMs offer an attractive approach not fully exploited in HEP. An SVM [92] is a supervised machine learning method which outputs an optimal hyperplane to categorize samples between two classes to classify data points. A quantum-enhanced kernel for SVM [84] can map the input vectors to an exponential Hilbert space, which could make it easier to construct an optimal hyperplane and increase the classification performance. Additionally, to calculate the quantum-enhanced kernel, the number of circuits is a function of the square of the number of input vectors, which may not be a good selection for classifying huge number of events. Multiple groups are actively exploring quantum kernel methods with gate-based quantum computers for event classification. Currently, these methods are limited by the dimensionality reduction required to make data compatible with modern hardware. Studying these algorithms provides new and different insights into the performance of modern computing platforms though. For example, they compute data element overlaps in Hilbert space, and the outcome state distributions are sensitive to device noise in different ways than variational algorithms like VQE or QAOA. New schemes for approaching *quantum feature map* in particular [47] are interesting directions.

7. Discussion and outlook

When considering applications of quantum machine learning for a field such as HEP, the immediate question is whether we have reason to believe that quantum machine learning—for near-term *or* universal quantum computers—is particularly suited to this type of application. The truth is that it is simply too early to tell, and only further investigation of the methods will provide the answers.

One feature of HEP data sets is that they are notoriously large. In principle, this makes quantum speed-ups attractive, as they could be crucial to analyse big amounts of data. But significant (that is, exponential) speed-ups in quantum machine learning are still controversial as to their scope [93] and in some cases, their true quantum nature [94]. They often rely on special properties of the data such as sparsity [95], or a special oracle or device that can load the data in superposition [41]. The appeal of near-term approaches to quantum machine learning is without doubt that ideas can be easily tested on a small scale, using the rich landscape of quantum programming languages, cloud-based quantum computers, and quantum machine learning software packages. Even so, to encode large data sets into a quantum system to sufficient precision and to measure the outputs for every events in the dataset is a physical challenge that is significantly out of the scope of near-term quantum computing. Of course, in the age of Big Data, the large size of the data sets are not unique to HEP, and it needs to be further established whether the intersection discussed in this review poses any *particular* challenge to machine learning which would motivate the use of quantum computers.

7.1. Experimenting with quantum annealers

Despite continuous improvement of quantum annealers they remain noisy, with limited number of qubits, and limited connectivity.

7.1.1. Solver heuristics

A huge challenge is to map the reformulated problem to an actual device with a limited connectivity [96], and it is often necessary to include connectivity constraints already into the loss itself. One alternative available in the D-Wave software stack is *qbsolve* [97], a heuristic that allows to split large problems in several smaller ones that in turn can be solve on the available hardware. This allows one to experiment with much larger QUBO than the one directly solvable with existing hardware, but in return requires additional computing resources. It also prevents us from directly probing the stand-alone capabilities of the device.

7.1.2. Digital devices

Digital annealers [98] offer the potential to prototype algorithms with large numbers of *digital qubits*. Using custom ASICs, digital annealers are capable of simulating fully-connected quantum annealers with 4 096

qubits (with 64 bit precision) or as many as 8 192 qubits (with 16 bit precision). In principle, a digital annealer cluster could offer up to 1 000 000 qubits using multi-chip support. While in the very long-run fully quantum annealers should be able to overtake digital simulators, in the near-term, these machines are exciting application test-beds and may even be able to deliver competitive results.

7.2. Experimenting with quantum circuits

Applying quantum algorithms on quantum hardware is the core aim at any research on quantum computing. But the scale of even state-of-the-art studies quickly reveals the limitations of current-day hardware. Typical implementations use only a few qubits and datasets of four features (for example, [55, 99, 100]). The limited number of qubits, connectivity and short decoherence time of the current quantum hardware make it difficult to experiment with large and long variational circuits.

7.2.1. Circuit architecture

In the papers reviewed above, the quantum circuit architecture (the types and numbers of gates) is fixed and only the parameters of the gates are optimized. In combination of this approach, search for optimal gate assembly is also possible. In [101], reinforcement learning is used to derive circuits to solve combinatorial problems. This technique might provide further handle at developing well performing quantum machine learning models.

7.2.2. Error mitigation

Practically, circuit-based qubit devices allow only a few gates to be performed before a signal is drowned in noise. The fidelity of measurements on quantum device can be improved via error mitigation strategies [102]. Various techniques allow experiments with an increased number of gates or better qubit connectivity. In addition to techniques making explicit assumption on the form and origin of the noise, machine learning approaches can be used to learn directly from the device-dependent noise. The integration of such noise-modelling-cancelling technique of circuit compiler would help with experimenting on quantum device, at the cost of increased resources.

7.2.3. Circuit simulation

Prototyping quantum algorithms with a classical simulator is an important step in the development and testing of new algorithms. The classical simulator used for the VQML study in [74] has enabled the authors to test the QCL algorithm with up to seven variables or ~10 000 events for the training set size. The simulation time and memory usage increases exponentially with the number of input variables in the creation of variational quantum states with $W(\vec{\theta})$. Despite continuous improvement in the simulator [89], the experimentation with circuits with large number of qubits is still hampered by this computation requirement. Of course, it is expected that the simulation of a quantum device will be classically hard. Because of this, it may be better when possible to experiment on smaller numbers of qubits—where circuits can be run—and study the time to solution or complexity, as a function of the number of qubits.

7.2.4. Optimization in quantum machine learning

There are two types of optimizer: gradient-based and derivative-free. For some derivative-free optimizers, it may require many iterations to achieve good training performance as the number of variational parameters increases. For the gradient-based optimizer, fewer iterations may be required. However, to calculate the gradient is also difficult [103] and numerical differentiation requires the circuit to be run additional times as the number of variational parameters increases. Changing a single circuit parameter for the evaluation of gradients through a cloud-based service can take of the order of many seconds, which quickly makes optimization of even a small system a matter of hours and days.

7.3. Quantum data

All the algorithms described in this review so far made use of a classical machine learning dataset, embedded into a quantum device. However, quantum machine learning algorithms have the unique property to be usable with a dataset made of quantum states, or *quantum data* [104, 105]. Those input quantum states are usually the output of some quantum circuits (e.g. circuits that extract the ground state of different Hamiltonians [106]) and are then processed by a variational circuit that has learned a desired quantum function (e.g. a property of this ground-state). However, one could also imagine directly feeding the quantum objects resulting from a HEP, dark matter, or gravitational wave detection experiment into the QML algorithm. Several application of machine learning on quantum data have been developed, including clustering of quantum states [104], detecting anomalies on a quantum device [107], learning algorithms to estimate the fidelity or the purity of a state [108, 109], learning phases of matter [106] and classifying quantum states [110, 111].

The question of how to exploit the quantum nature of the systems generating HEP data has not been prominent in the literature, but there are two interesting outlooks.

The first is to do quantum machine learning *directly* on the quantum objects measured in HEP. As an example, instead of processing the classical signal formed in photonic sensors, one could direct the photons into a *photonic quantum computer* and apply a variational circuit before conducting the final measurement. The circuit could be trained to extract important information from the quantum state, or to classify the state. Applying this process to axion dark matter experiments [112] or to neutrino detectors [18] could be promising research directions.

The second path follows the idea of *quantum simulations* [113, 114], an important use of quantum computers in simulating complex quantum systems to determine their properties. If a HEP experiment could be simulated on a quantum computer [18, 19, 21], the simulation could be followed by a quantum machine learning routine executed on the very same device, and analysing the quantum states produced by the simulation. Instead of costly state tomography to characterise the results, the wave function is directly accessed and important information extracted.

In both cases, an important insight from quantum machine learning—possibly the one with the highest future impact on other quantum disciplines—is the ability to differentiate through quantum computations. This includes a wealth of knowledge and practical methods to get partial derivatives of a measurement result with respect to (classical) physical parameters of the experiment, such as a magnetic field strength or pulse length. Quantum differentiation opens a door to design experiments by adaptively optimizing some cost functions, which is crucial for quantum data analysis.

7.4. Concluding remarks

Overall, we are just at the beginning of exploring the intersection between quantum machine learning and HEP. The papers presented in this review therefore have to be understood as exploratory studies that propose angles to approach the problem of how to use quantum machine learning algorithms to understand fundamental particles.

We presented papers on performing classification using quantum machine learning with QA, restrictive Boltzmann machines, quantum graph networks and variational quantum circuits. The capacity of quantum annealers to perform classification is limited due to the restrictive formulation of the problem. Quantum-circuit-based machine learning is yet of limited performance due to the necessary down-scaling of the problems, so as to fit on the quantum device, or to be amenable in simulation.

In the outlook we discussed practical considerations of experimenting with quantum machine learning and the prospect of analysing quantum data. These challenges put quantum machine learning into a particularly difficult spot. The quality of a machine learning algorithm is usually estimated through empirical benchmarks on pseudo-realistic datasets. Evidence from deep learning suggests that machine learning on large datasets behaves very differently from the small-data regime. And while consistently improving, the theory of machine learning is currently unable to explain the performance of algorithms such as neural networks. The challenges for practical experiments as well as fundamental limits of classical simulations restrict quantum machine learning benchmarks to small proof-of-principle investigations that may only say very little about their performance in realistic settings.

As the technology develops, more theory work is needed to understand the power of near-term quantum machine learning. While the current performance of quantum machine learning on high energy physics data is limited, there is hope that future advances on both quantum devices and quantum algorithms will help with the computation challenges of particle physics.

Acknowledgments

The authors wish to thank Alexander Zlokapa, Joshua Job, Cenk Tuysuz and Shaojun Sun for sharing material reproduced here.

JRV is partially supported by DOE/HEP QuantISED program grant, Quantum Machine Learning and Quantum Computation Frameworks (QMLQCF) for HEP, award number DE-SC0019227. MS acknowledges support by the Big Data and Informatics Flagship and BDSS initiative of the University of KwaZulu-Natal. WG is partially supported by DOE/HEP QuantISED program grant, Application of Quantum Machine Learning to High Energy Physics Analysis at the LHC using IBM Quantum Computer Simulator and Hardware, award number DE-SC0020416. GP is partially supported by the DOE/HEP QuantISED program grant HEP Machine Learning and Optimization Go Quantum, identification number 0000240323. This manuscript has been authored by Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics.

Data sharing is not applicable to this article as no new data were created or analysed in this study.

ORCID iDs

Gabriel Perdue https://orcid.org/0000-0001-6785-8720
Arthur Pesah https://orcid.org/0000-0002-5759-6314
Maria Schuld https://orcid.org/0000-0001-8626-168X
Koji Terashi https://orcid.org/0000-0001-6520-8070
Sofia Vallecorsa https://orcid.org/0000-0002-7003-5765
Jean-Roch Vlimant https://orcid.org/0000-0002-9705-101X

References

[1] Radovic A, Williams M, Rousseau D, Kagan M, Bonacorsi D, Himmel A, Aurisano A, Terao K and Wongjirad T 2018 Machine learning at the energy and intensity frontiers of particle physics *Nature* **560** 41–8
[2] Albertsson K *et al* 2018 Machine learning in high energy physics community white paper *J. Phys. Conf. Ser.* **1085** 022008
[3] Guest D, Cranmer K and Whiteson D 2018 Deep Learning and its application to LHC physics *Ann. Rev. Nucl. Part. Sci.* **68** 161–81
[4] Shapoval I and Calafiura P 2019 Quantum associative memory in HEP track pattern recognition *EPJ Conf.* **214** 01012
[5] Bapst F, Bhimji W, Calafiura P, Gray H, Lavrijsen W and Linder L 2020 A pattern recognition algorithm for quantum annealers *Comput. Softw. Big Sci.* **4** 1
[6] Bauer C W, Jong W A D, Nachman B and Provasoli D 2019 A quantum algorithm for high energy physics simulations p 4
[7] Zlokapa A, Anand A, Vlimant J-R, Duarte J M, Job J, Lidar D and Spiropulu M 2019 Charged particle tracking with quantum annealing-inspired optimization **8**
[8] Cormier K, Sipio R Di and Wittek P 2019 Unfolding measurement distributions via quantum annealing *JHEP* **11** 128
[9] Feynman R P 1982 Simulating physics with computers *Int. J. Theor. Phys.* **21** 467–88
[10] Farhi E, Goldstone J and Gutmann S 2014 A quantum approximate optimization algorithm
[11] Brandao F G S L and Svore K M 2017 Quantum speed-ups for solving semidefinite programs *2017 IEEE 58th Symp. on Foundations of Computer Science (FOCS)* pp 415–26
[12] Chakrabarti S, Childs A M, Li T and Wu X 2020 Quantum algorithms and lower bounds for convex optimization *Quantum* **4** 221
[13] Cao Y *et al* 2019 Quantum chemistry in the age of quantum computing *Chem. Rev.* **119** 10856–915
[14] McArdle S, Endo S, Aspuru-Guzik A, Benjamin S, and Yuan X 2018 Quantum computational chemistry
[15] Biamonte J, Wittek P, Pancotti N, Rebentrost P, Wiebe N and Lloyd S 2017 Quantum machine learning *Nature* **549** 195 EP
[16] Wittek P 2014 *Quantum Machine Learning: What Quantum Computing Means to Data Mining* (New York: Academic)
[17] Schuld M and Petruccione F 2018 *Supervised Learning With Quantum Computers* (Berlin: Springer Int. Publishing)
[18] Roggero A and Carlson J 2019 Dynamic linear response quantum algorithm *Phys. Rev.* C **100** 034610
[19] Roggero A, Li A C Y, Carlson J, Gupta R and Perdue G N 2020 Quantum computing for neutrino-nucleus scattering *Phys. Rev.* D **101** 074038
[20] Dumitrescu E F, McCaskey A J, Hagen G, Jansen G R, Morris T D, Papenbrock T, Pooser R C, Dean D J and Lougovski P 2018 Cloud quantum computing of an atomic nucleus *Phys. Rev. Lett.* **120** 210501
[21] Preskill J 2018 Simulating quantum field theory with a quantum computer (arXiv:1811.10085)
[22] Martinez E A *et al* 2016 Real-time dynamics of lattice gauge theories with a few-qubit quantum computer *Nature* **534** 516–19
[23] Lu H-H *et al* 2019 Simulations of subatomic many-body physics on a quantum frequency processor *Phys. Rev.* A **100** 012320
[24] Klco N, Savage M J and Stryker J R 2020 Su (2) non-abelian gauge field theory in one dimension on digital quantum computers *Phys. Rev.* D **101** 074512
[25] Deutsch D 1985 Quantum theory, the church–turing principle and the universal quantum computer *Proc. R. Soc.* A **400** 97–117
[26] Shor P W 1997 Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer *SIAM J. Comput.* **26** 1484–509
[27] Grover L K 1996 A fast quantum mechanical algorithm for database search (*Proc. Twenty-Eighth Annual Symp. on Theory of Computing, STOC '96*) (Association for Computing Machinery) *New York, NY, USA* pp 212–9
[28] Harrow A W, Hassidim A and Lloyd S 2009 Quantum algorithm for linear systems of equations *Phys. Rev. Lett.* **103** 150502
[29] Córcoles A D, Kandala A, Javadi-Abhari A, McClure D T, Cross A W, Temme K, Nation P D, Steffen M and Gambetta J M 2019 Challenges and opportunities of near-term quantum computing systems (arXiv:1910.02894)
[30] Bravyi S, Gosset D and Movassagh R 2019 Classical algorithms for quantum mean values
[31] Preskill J 2018 Quantum computing in the NISQ era and beyond *Quantum* **2** 79
[32] Schuld M, Bocharov A, Svore K and Wiebe N 2018 Circuit-centric quantum classifiers (arXiv:1804.00633)
[33] Kadowaki T and Nishimori H 1998 Quantum annealing in the transverse ising model *Phys. Rev.* E **58** 5355–63
[34] Farhi E, Goldstone J, Gutmann S and Sipser M 2000 Quantum computation by adiabatic evolution
[35] Roland Jémie and Cerf N J 2002 Quantum search by local adiabatic evolution *Phys. Rev.* A **65** 042308
[36] Biamonte J, Wittek P, Pancotti N, Rebentrost P, Wiebe N and Lloyd S 2017 Quantum machine learning *Nature* **549** 195
[37] Arunachalam S and Ronald de W 2017 Guest column: A survey of quantum learning theory *ACM SIGACT News* **48** 41–67
[38] Ciliberto C, Rocchetto A, Rudi A and Wossnig L 2020 Fast quantum learning with statistical guarantees (arXiv:2001.10477)
[39] Lloyd S, Mohseni M and Rebentrost P 2014 Quantum principal component analysis *Nat. Phys.* **10** 631–3
[40] Rebentrost P, Mohseni M and Lloyd S 2014 Quantum support vector machine for big data classification *Phys. Rev. Lett.* **113** 130503
[41] Kerenedis I and Prakash A 2017 Quantum recommendation systems *Kerenidis, Iordanis and Anupam Prakash. 'Quantum Recommendation Systems.' LIPIcs-Leibniz Int. Proc. in Informatics* vol 67
[42] Ciliberto C, Herbster M, Ialongo A D, Pontil M, Rocchetto A, Severini S and Wossnig L 2018 Quantum machine learning: a classical perspective *Proc. R. Soc.* A **474** 20170551
[43] Perdomo-Ortiz A, Benedetti M, Realpe-Gómez J and Biswas R 2018 Opportunities and challenges for quantum-assisted machine learning in near-term quantum computers *Quantum Sci. Technol.* **3** 030502
[44] Farhi E and Neven H 2018 Classification with quantum neural networks on near term processors (arXiv:1802.06002)
[45] Benedetti M, Realpe-Gómez J, Biswas R and Perdomo-Ortiz A 2017 Quantum-assisted learning of hardware-embedded probabilistic graphical models *Phys. Rev.* X **7** 041052

[46] Neven H, Denchev V S, Rose G and Macready W G 2008 Training a binary classifier with the quantum adiabatic algorithm (arXiv:0811.0416)
[47] Lloyd S, Schuld M, Ijaz A, Izaac J and Killoran N 2020 Quantum embeddings for machine learning (arXiv:2001.03622)
[48] McClean J R, Romero J, Babbush R and Aspuru-Guzik Aan 2016 The theory of variational hybrid quantum-classical algorithms *New J. Phys.* **18** 023023
[49] Mitarai K, Negoro M, Kitagawa M and Fujii K 2018 Quantum circuit learning *Phys. Rev.* A **98** 032309
[50] Schuld M, Bergholm V, Gogolin C, Izaac J and Killoran N 2019 Evaluating analytic gradients on quantum hardware *Phys. Rev.* A **99** 032331
[51] Bergholm V, Izaac J, Schuld M, Gogolin C and Killoran N 2018 Pennylane: Automatic differentiation of hybrid quantum-classical computations (arXiv:1811.04968)
[52] Broughton M *et al* 2020 Tensorflow quantum: A software framework for quantum machine learning (arXiv:2003.02989)
[53] Harrow A and Napp J 2019 Low-depth gradient measurements can improve convergence in variational hybrid quantum-classical algorithms (arXiv:1901.05374)
[54] Liu J-G and Wang L 2018 Differentiable learning of quantum circuit born machines *Phys. Rev.* A **98** 062324
[55] Benedetti M, Garcia-Pintos D, Perdomo O, Leyton-Ortega V, Nam Y and Perdomo-Ortiz A 2019 A generative modeling approach for benchmarking and training shallow quantum circuits *npj Quantum Inf.* **5** 1–9
[56] Lloyd S and Weedbrook C 2018 Quantum generative adversarial learning *Phys. Rev. Lett.* **121** 040502
[57] Dallaire-Demers P-L and Killoran N 2018 Quantum generative adversarial networks *Phys. Rev.* A **98** 012324
[58] Neven H, Denchev V S, Drew-Brook M, Zhang J, Macready W G and Geordie R 2009 Nips 2009 demonstration: Binary classification using hardware implementation of quantum annealing *Quantum* pp 1–17
[59] Pudenz K L and Lidar D A 2013 Quantum adiabatic machine learning *Quant. Inf. Proc.* **12** 2027–70
[60] Glasser I, Pancotti N, August M, Rodriguez I D and Ignacio Cirac J 2018 Neural-network quantum states, string-bond states and chiral topological states *Phys. Rev.* X **8** 011006
[61] Amin M H, Andriyash E, Rolfe J, Kulchytskyy B and Melko R 2018 Quantum boltzmann machine *Phys. Rev.* X **8** 021050
[62] Mott A, Job J, Vlimant J-R, Lidar D and Spiropulu M 10 2017 Solving a higgs optimization problem with quantum annealing for machine learning *Nature* **550** 375–9
[63] Zlokapa A, Mott A, Job J, Vlimant J-R, Lidar D and Spiropulu M 2019 Quantum adiabatic machine learning with zooming
[64] Caldeira Jão, Job J, Adachi S H, Nord B and Perdue G N 2019 Restricted Boltzmann Machines for galaxy morphology classification with a quantum annealer **11**
[65] Chatrchyan S *et al* 2012 Observation of a new boson at a mass of 125 gev with the cms experiment at the lhc *Phys. Lett.* B **716** 30–61
[66] Aad G *et al* 2012 Observation of a new particle in the search for the standard model Higgs boson with the ATLAS detector at the LHC *Phys. Lett.* B **716** 1–29
[67] Kirkpatrick S, Gelatt C D and Vecchi M P 1983 Optimization by simulated annealing *Science* **220** 671–80
[68] Katzgraber H G, Trebst S, Huse D A and Troyer M 2006 Feedback-optimized parallel tempering monte carlo *J. Stat. Mech.: Theory Experiment* **2006** P03018–P03018
[69] Hinton G E 2002 Training products of experts by minimizing contrastive divergence *Neural Comput.* **14** 1771–800
[70] Adachi S H and Henderson M P 2015 Application of quantum annealing to training of deep neural networks
[71] Adachi S H and Henderson M P 2015 Application of quantum annealing to training of deep neural networks arXiv e-print
[72] Dermikoz B Dobos D Fracas F Novotny K Potamianos K Vallecorsa S Vlimant J Tuysuz C and Carminati F 2020 Particle track reconstruction with quantum algorithms
[73] Chan J, Guan W, Sun S, Wang A Z, Wu S L, Zhou C, Livny M, Carminati F and Meglio A Di 2019 Application of quantum machine learning to high energy physics analysis at lhc using ibm quantum computer simulators and ibm quantum computer hardware *PoS* **LeptonPhoton2019** 049
[74] Terashi K, Kaneda M, Kishimoto T, Saito M, Sawada R and Tanaka J 2020 Event classification with quantum machine learning in high-energy physics (arXiv:2002.09935)
[75] Nakamoto T, Rossi L, Apollinari G and Bruening O 2017 High luminosity large hadron collider HL-LHC (arXiv:1705.08830)
[76] Grant E, Benedetti M, Cao S, Hallam A, Lockhart J, Stojevic V, Green A G and Severini S 2018 Hierarchical quantum classifiers *npj Quantum Inf.* **4** 17–9
[77] Gumpert C, Salzburger A, Kiehn M, Hrdinka J and Calace N 2017 ACTS: from ATLAS software towards a common track reconstruction software *J. Phys.: Conf. Series* **898** 042011
[78] Fast Track Reconstruction for HL-LHC. Technical Report ATL-PHYS-PUB-2019-041, CERN, Geneva 2019
[79] Sioni S and Andrew R 2019 Kalman filter track reconstruction on FPGAs for acceleration of the high level trigger of the CMS experiment at the HL-LHC *EPJ Conf.* **214** 01003
[80] Farrell S *et al* 2017 The hep.trkx project: deep neural networks for hl-lhc online and offline tracking *EPJ Conf.* **150** 00003
[81] Amrouche S *et al* 2019 The tracking machine learning challenge: Accuracy phase *The Springer Series on Challenges in Machine Learning* pp 231–64
[82] Farrell S *et al* 2018 Novel deep learning methods for track reconstruction
[83] Abadi M *et al* 2015 TensorFlow: Large-scale machine learning on heterogeneous systems Software available from tensorflow.org
[84] Havlícek V, Córcoles A D, Temme K, Harrow A W, Kandala A, Chow J M and Gambetta J M 2019 Supervised learning with quantum-enhanced feature spaces *Nature* **567** 209–12
[85] Powell M J D 1994 A direct search optimization method that models the objective and constraint functions by linear interpolation *Math. Its Appl.* **275**
[86] Spall J C 1997 A one-measurement form of simultaneous perturbation stochastic approximation *Automatica* **33**
[87] Spall J C 2000 Adaptive stochastic approximation by the simultaneous perturbation method *IEEE Transaction on Automatic Control* p 45
[88] Mitarai K, Negoro M, Kitagawa M and Fujii K 2018 Quantum circuit learning *Phys. Rev.* A **98**
[89] Qulacs 2018
[90] Dua D and Graff C 2017 UCI machine learning repository
[91] Gao X, Zhang Z-Y and Duan L-M 2018 A quantum machine learning algorithm based on generative models *Sci. Adv.* **4**
[92] Boser B E, Guyon I M and Vapnik V N 1992 A training algorithm for optimal margin classifiers *COLT 92: Proc. of the Fifth Annual Workshop on Computational Learning Theory* p 144
[93] Aaronson S 2015 Read the fine print *Nat. Phys.* **11** 291–3

[94] Tang E 2019 A quantum-inspired classical algorithm for recommendation systems *Proc. of the 51st Annual ACM Symp. on Theory of Computing* pp 217–28
[95] Arrazola J M, Delgado A, Bardhan B R and Lloyd S 2019 Quantum-inspired algorithms in practice (arXiv:1905.10415)
[96] Benedetti M, Realpe-Gómez J, Biswas R and Perdomo-Ortiz A 2016 Estimation of effective temperatures in quantum annealers for sampling applications: A case study with possible applications in deep learning *Phys. Rev.* A **94** 022308
[97] Booth M, Reinhardt S P and Roy A 2017 Partitioning optimization problems for hybrid classical/quantum execution
[98] Fujitsu 2020 Digital annealer (https://www.fujitsu.com/global/services/business-services/digital-annealer/)
[99] Havlíček Věch, Córcoles A D, Temme K, Harrow A W, Kandala A, Chow J M and Gambetta J M 2019 Supervised learning with quantum-enhanced feature spaces *Nature* **567** 209
[100] Schuld M, Fingerhuth M and Petruccione F 2017 Implementing a distance-based classifier with a quantum interference circuit *EPL (Europhys. Lett.)* **119** 60002
[101] McKiernan K A, Davis E, Sohaib Alam M and Rigetti C 2019 Automated quantum programming via reinforcement learning for combinatorial optimization
[102] Temme K, Bravyi S and Gambetta J M 2017 Error mitigation for short-depth quantum circuits *Phys. Rev. Lett.* **119** 180509
[103] Schuld M, Bergholm V, Gogolin C, Izaac J and Killoran N 2019 Evaluating analytic gradients on quantum hardware *Phys. Rev.* A **99**
[104] Sentís G, Monràs A, Muñoz-Tapia R, Calsamiglia J and Bagan E 2019 Unsupervised classification of quantum data *Phys. Rev.* X **9** 041029
[105] Poland K, Beer K and Osborne T J 2020 No free lunch for quantum machine learning (arXiv:2003.14103)
[106] Cong I, Choi S and Lukin M D 2019 Quantum convolutional neural networks *Nat. Phys.* **15** 1273–8
[107] Liu N and Rebentrost P 2018 Quantum machine learning for quantum anomaly detection *Phys. Rev.* A **97** 042315
[108] Cincio L, Subaş Yğit, Sornborger A T and Coles P J 2018 Learning the quantum algorithm for state overlap *New J. Phys.* **20** 113022
[109] Cerezo M, Poremba A, Cincio L and Coles P J 2020 Variational quantum fidelity estimation *Quantum* **4** 248
[110] Shahi F and Rezakhani A T 2017 Binary classification of quantum states: Supervised and unsupervised learning (arXiv:1704.01965)
[111] Sentís G, Guţă Mădălin and Adesso G 2015 Quantum learning of coherent states *EPJ Quantum Technol.* **2** 1–22
[112] van Bibber K, Lehnert K and Chou A 2019 Putting the squeeze on axions *Phys. Today* **72** 48–55
[113] Lloyd S 1996 Universal quantum simulators *Science* 1073–8
[114] Georgescu I M, Ashhab S and Nori F 2014 Quantum simulation *Rev. Mod. Phys.* **86** 153–85