Training Hybrid Classical-Quantum Classifiers via Stochastic Variational Optimization

Ivana Nikoloska, *Graduate Student Member, IEEE,* and Osvaldo Simeone, *Fellow, IEEE.*

***Abstract*—Quantum machine learning has emerged as a potential practical application of near-term quantum devices. In this work, we study a two-layer hybrid classical-quantum classifier in which a first layer of quantum stochastic neurons implementing generalized linear models (QGLMs) is followed by a second classical combining layer. The input to the first, hidden, layer is obtained via amplitude encoding in order to leverage the exponential size of the fan-in of the quantum neurons in the number of qubits per neuron. To facilitate implementation of the QGLMs, all weights and activations are binary. While the state of the art on training strategies for this class of models is limited to exhaustive search and single-neuron perceptron-like bit-flip strategies, this letter introduces a stochastic variational optimization approach that enables the joint training of quantum and classical layers via stochastic gradient descent. Experiments show the advantages of the approach for a variety of activation functions implemented by QGLM neurons.**

***Index Terms*—Quantum Machine Learning, Quantum Computing, Probabilistic Machine Learning**

I. Introduction

Ever since Richard Feynman proposed the concept of quantum computers almost half a century ago, technology giants, startups and academic labs alike have competed against, and collaborated with each other, eager to make it a reality. Progress has had to contend with the stark limitations of current noisy-intermediate scale quantum (NISQ) systems providing 50-100 non-fault-tolerant qubits [1]. An emerging potential practical use of NISQ hardware is quantum machine learning, a hybrid research discipline that combines machine learning and quantum computing [2]–[4].

Many classical techniques, ranging from kernel methods [5] and Boltzmann machines [6] to deep learning models like convolutional [7] and graph neural networks [8], now have quantum counterparts [9]–[12], which can operate, at least on a small scale, on NISQ hardware. These methods apply classical optimization routines to select parameters that define the operation of a quantum circuit. Such parameters are most often continuous and optimized via gradient descent techniques that apply the so-called parameter-shift rule, an exact form of finite-difference differentiation [4]. Alternative approaches, which may be more promising in the short term, involve hybrid quantum-classical models, where classical computation, e.g., for feature extraction, is combined with quantum parametric circuits [13].

A notable example of hybrid quantum-classical models is the multi-layer artificial neural network introduced in [14], [15], which includes stochastic binary neurons implementing generalized linear models (GLMs). The key merits of this architecture are that: (*i*) quantum implementations of GLMs, referred to as QGLM, have an exponentially large number of inputs and number of synaptic weights in the number of physical resources – qubits – within the neuron; and (*ii*) unlike more conventional deterministic models, such as [16], the outcomes of the quantum neurons need not be averaged by performing multiple measurements to obtain the expectations of given observables.

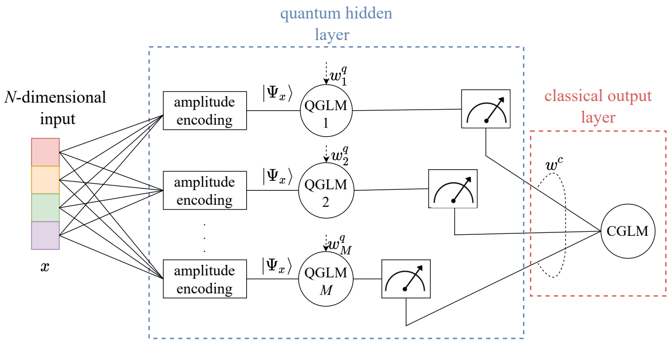

Fig. 1: In the studied hybrid classical-quantum classifier, a quantum hidden layer, fed via amplitude encoding and consisting of quantum generalized linear models (QGLMs), is followed by a classical combining output layer with a single classical GLM (CGLM) neuron. All weights and activations are binary.

The QGLM-based neural network considered in [14], [15] has binary synaptic weights, as well as binary, classical, activations, rendering standard optimization methods infeasible. For models comprised of one or two QGLM neurons, the weight vectors were optimized in [14], [15] via a single-neuron perceptron-like sign-flips strategy or via exhaustive search. Both approaches are inapplicable or infeasible for models comprised of an arbitrary number of neurons.

In this paper, we address this issue by focusing – as in the experiments presented in [15] – on the hybrid classical-quantum two-layer architecture illustrated in Fig. 1. In it, a first layer of QGLMs is followed by a second classical combining layer. The input to the first, hidden, layer is obtained via amplitude encoding (see, e.g., [4]). This letter introduces a stochastic variational optimization (SVO) approach [17] that enables the joint training of quantum and classical layers via stochastic gradient descent. The proposed SVO-based training strategy operates in a relaxed continuous space of variational classical parameters. Experiments via computer simulations show the advantages of the approach for a variety of activation functions implemented by the QGLM neurons.

I. Nikoloska and O. Simeone are with KCLIP, CTR, Department of King's College London. e-mails: {ivana.nikoloska, osvaldo.simeone}@kcl.ac.uk.
This work was supported by the European Research Council (ERC) under the European Union's Horizon 2020 Research and Innovation Program (Grant Agreement No. 725731).

II. Model

In this section, we describe the hybrid classical-quantum classifier depicted in Fig. 1.

A. Hybrid Classical-Quantum Binary Classifier

As illustrated in Fig. 1, we consider a two-layer architecture implementing a binary classifier on an input, classical, vector x. The first, hidden, layer consists of M QGLM-based neurons, and the last layer of a classical GLM (CGLM) neuron producing the final classification decision.

The input vector $x = [x_0, ..., x_{N-1}]^T$, which is assumed to be binary with $x_n \in \{-1, +1\}$, is mapped to the amplitude vector of a pure quantum state consisting of $\log_2(N)$ qubits (assumed to be an integer) as

$$|\Psi_x\rangle = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x_n |n\rangle , \tag{1}$$

where $|n\rangle$, with $n \in \{0, ..., N-1\}$, denotes the nth vector of the computational basis (see, e.g., [18]). We note that preparing the quantum state (1) from a general binary input vector x entails a minimal complexity of $N/\log_2(N)$ computational steps, most of which are two-qubit gates, unless vector x has a specific structure (e.g., it is a sample from a smooth probability distribution) [4].

As seen in Fig. 1, the quantum state $|\Psi_x\rangle$ is prepared M times and input to each of the QGLM neurons in the first layer. Each mth QGLM neuron produces a stochastic binary output $y_m \in \{-1, +1\}$ as a function of the input state $|\Psi_x\rangle$ and of an N-dimensional vector $w^q_m = [w^q_{m,0}, ..., w^q_{m,N-1}]^T$ of binary weights, with $w^q_{m,n} \in \{-1, +1\}$.

The M binary digits $y = [y_1, ..., y_M]^T$ produced by the first layer are then fed to a CGLM neuron, which outputs a final, stochastic, decision $z \in \{-1, +1\}$ for binary classification as a function of its own vector of binary weights $w^c = [w^c_1, ..., w^c_M]^T$.

We emphasize that, unlike deterministic models such as [16], the outputs of the quantum neurons need not be averaged to obtain the expectation of an observable over multiple runs. Rather, the outcomes of the quantum neurons are obtained via a single measurement, producing a stochastic binary output.

B. Classical GLM (CGLM) Neuron

In order to describe the operation of the neurons, let us start with the classifying CGLM neuron in the last layer. As seen, the CGLM neuron receives an M-dimensional binary input vector y and produces a final binary decision $z \in \{-1, +1\}$. The probability of the binary output given the input vector y is given by

$$p_{w^c}(z = 1 \,|y) = \mathrm{g}^c \left(\sum_{m=1}^{M} w^c_m y_m \right) = \mathrm{g}^c \left(y^T w^c \right), \tag{2}$$

where $\mathrm{g}^c(\cdot)$ denotes the response function of the CGLM and $(\cdot)^T$ denotes transposition. Typical examples of the response function $\mathrm{g}^c(\cdot)$, which must be invertible, of GLMs include the sigmoid function and the Gaussian cumulative distribution function (see, e.g., [19]).

C. Quantum GLM (QGLM) Neuron

Quantum procedures closely mimicking the operation (2) of the CGLM neuron can be designed using (approximately) $\log_2(N)$ qubits (see [4] for an overview). The fact that $\log_2(N)$ qubits are sufficient to process an exponentially larger input x, of size N, illustrates the potential computational advantage of quantum information processing in this context. We refer to quantum circuits implementing the stochastic mapping defined by the conditional distribution

$$p_{w^q_m}(y_m = 1 \,|x) = \mathrm{g}^q \left(\sum_{n=0}^{N-1} w^q_{m,n} x_n \right) = \mathrm{g}^q \left(x^T w^q_m \right), \tag{3}$$

for some (invertible) response function $\mathrm{g}^q(\cdot)$, given binary weight vector w^q_m, as QGLM neurons. The outputs of the M QGLMs are independent given the input x, and hence we have $p_{w^q}(y \,|\, x) = \prod_{m=1}^{M} p_{w^q_m}(y_m \,|x)$, where $w^q = \{w^q_m\}_{m=1}^{M}$.

Several implementations of QGLM neurons have been proposed in the literature using different quantum circuits. The main goal of these circuits is to produce a stochastic binary output with probabilities which are a function of the inner product $\langle \Psi_{w^q} | \Psi_x \rangle = x^T w^q_m$ between the input state $|\Psi_x\rangle$ and the amplitude-encoded binary weight vector $|\Psi_{w^q}\rangle = 1/\sqrt{N} \sum_{n=0}^{N-1} w^q_n |n\rangle$. Different solutions, along with the resulting response functions are listed in Table I, accompanied by relevant references. In the experiments provided in Sec. IV we consider all these options.

III. Stochastic Variational Optimization-Based Training

In this section, we first define the problem of training the hybrid classical-quantum classifier defined in the previous section, and then introduce an SVO-based training procedure.

A. Problem Definition

We assume to have access to a training set $\mathcal{D}$ of input-output samples (x, z), with $x \in \{-1, 1\}^N$ and $z \in \{-1, 1\}$. Given the model described in the previous section, we define the log-loss on an example (x, z) as

$$\mathcal{L}_{(x,z)}(w) = -\ln \left(\mathbb{E}_{p_{w^q}(y \,|\, x)} \left[p_{w^c} \left(z \,|\, y \right) \right] \right), \tag{4}$$

which is a function of the weights $w = \{\{w^q_m\}_{m=1}^{M}, w^c\}$ of the M QGLMs and of the CGLM. Note that computing the log-loss requires averaging over the outputs y of the hidden QGLMs. Using Jensen's inequality, the loss in (4) can be bounded as

$$\mathcal{L}_{(x,z)}(w) \leq \mathbb{E}_{p_{w^q}(y \,|\, x)} \left[\ell \left(z, \mathrm{g}(y^T w^c) \right) \right] := L_{(x,z)}(w), \tag{5}$$

where we have defined the negative log-probability of the output of the CGLM as

$$-\ln p_{w^c}(z \,|\, y) = \ell \left(z, \mathrm{g}(y^T w^c) \right), \tag{6}$$

with $\ell(a, b) = -\ln\Big(b^{\frac{1+a}{2}} (1-b)^{\frac{1-a}{2}} \Big)$.

The training objective is to minimize the loss bound in (5) over the $NM+M$ binary weights w by addressing the problem

$$\min_{w \in \{0,1\}^{NM+M}} \sum_{(x,z) \in \mathcal{D}} L_{(x,z)}(w). \tag{7}$$

TABLE I: List of response functions for QGLM neurons

Name	g$(\cdot)$	Routine
Quadratic (Q)	$\mathrm{g}(x^T w^q_m) = \lvert x^T w^q_m\rvert^2$	Sign-flip blocks [14], [15]
Biased quadratic (BQ)	$\mathrm{g}(x^T w^q_m) = 1/2 + 1/2\lvert x^T w^q_m\rvert^2$	Swap test [20]
Biased centered quadratic (BCQ)	$\mathrm{g}(x^T w^q_m) = 1/2 + 1/2\lvert x^T w^q_m - 1/2\rvert^2$	Swap test on extra dimensions [21]
Linear (L)	$\mathrm{g}(x^T w^q_m) = 1/2 + 1/2 x^T w^q_m$	Interference circuit [22]

B. SVO-Based Training Algorithm

The direct optimization of problem (7) is problematic due to the discrete nature of its domain. In this paper, we propose to address this problem via SVO. SVO is a generalization of stochastic optimization via simultaneous perturbation [23] that has the key merit of being applicable also to discrete models. It is noted that stochastic optimization via simultaneous perturbation was found to be effective in [3] to optimize a parameterized quantum circuit with continuous model parameters.

SVO makes use of the observation that the minimum of a collection of values cannot be larger than an arbitrary average of such values, i.e., [17]

$$\min_{w} \sum_{(x,z)\in\mathcal{D}} L_{(x,z)}(w) \\ \leq \min_{q(w\,|\,\phi)} \mathbb{E}_{q(w\,|\,\phi)}\left[\sum_{(x,z)\in\mathcal{D}} L_{(x,z)}(w)\right], \quad (8)$$

where $q(w\,|\,\phi)$ represents a parametric distribution on the space of the model parameters w that depends on a continuous-valued vector of parameters ϕ. Furthermore, if the family of distributions is sufficiently large to include all distributions concentrated at the possible values of w, the relation in (8) holds with equality.

Given that the model parameters are discrete, we define the variational distribution $q(w\,|\,\phi)$ as the product-Bernoulli probability mass function

$$q(w\,|\,\phi) = \prod_{m=1}^{M}\prod_{n=0}^{N-1} \sigma(\phi^q_{m,n})^{\frac{1+w^q_{m,n}}{2}}(1-\sigma(\phi^q_{m,n}))^{\frac{1-w^q_{m,n}}{2}} \\ \times \prod_{m=1}^{M} \sigma(\phi^c_m)^{\frac{1+w^c_m}{2}}(1-\sigma(\phi^c_m))^{\frac{1-w^c_m}{2}}, \quad (9)$$

where $\sigma(a) = (1+\exp(-a))$ denotes the sigmoid function and we have defined the $NM + M$ variational parameters $\phi = \{\{\{\phi^q_{m,n}\}^M_{m=1}\}^{N-1}_{n=0}, \{\phi^c_m\}^M_{m=1}\}$. These are the natural parameters defining the probabilities that the corresponding weights equal 1. Specifically, the real parameter $\phi^q_{m,n}$ yields the probability $\sigma(\phi^q_{m,n})$ that we have $w^q_{m,n} = 1$, and ϕ^c_m gives the probability $\sigma(\phi^c_m)$ of event $w^c_m = 1$ under the variational distribution $q(w\,|\,\phi)$. With this choice, we address problem (7) by minimizing the upper bound in (8) over the $NM+M$-dimensional real vector ϕ.

To this end, we aim at approximating the gradient-based update

$$\phi \leftarrow \phi + \eta\nabla_\phi\, \mathbb{E}_{q(w\,|\,\phi)}\left[\sum_{(x,z)\in\mathcal{D}} L_{(x,z)}(w)\right] \\ = \phi + \eta\, \mathbb{E}_{q(w\,|\,\phi)}\left[\left(\sum_{(x,z)\in\mathcal{D}} L_{(x,z)}(w) - b\right)\nabla_\phi \ln q(w\,|\,\phi)\right], \quad (10)$$

where $\eta > 0$ is the learning rate and b is an arbitrary vector of the same dimensions as ϕ. The equality in (10) follows from the standard REINFORCE expression [24]. The gradient of the log-variational distribution in (10) has the simple form

$$\nabla_\phi \ln q(w\,|\,\phi) = \frac{1+w}{2} - \sigma(\phi), \quad (11)$$

where the functions are evaluated entry-wise.

In order to obtain a practical update rule, the expectation over the variational distribution $q(w|\phi)$ and the sum over the data set in (10) are estimated using samples from both the variational distribution $q(w|\phi)$ and from the data set $\mathcal{D}$. With a single sample $w \sim q(w|\phi)$, we obtain the doubly stochastic estimate [24]

$$\phi \leftarrow \phi + \eta\left[\left(\frac{|\mathcal{D}|}{|\mathcal{D}_b|}\sum_{(x,z)\in\mathcal{D}_b} L_{(x,z)}(w) - b\right)\nabla_\phi \ln q(w\,|\,\phi)\right], \quad (12)$$

where $|\cdot|$ is the cardinality of the argument set and $\mathcal{D}_b$ represents a mini-batch of the data set $\mathcal{D}$. The estimate (12) can be readily extended to any number of samples from $q(w|\phi)$. Importantly, the update (12) is local: Each entry of vector ϕ can be updated separately based on knowledge of the global loss $L_{(x,z)}(w)$ for the batch of samples $\mathcal{D}_b$. In fact, the gradient (11) can be computed entry-wise and hence separately for each parameter in vector ϕ.

The update (12) can be interpreted in a manner similar to stochastic optimization via simultaneous perturbations [23] (which is also related to evolution-based strategies [25]). At each iteration, the random sampling of the weights from the variational distribution $q(w|\phi)$ explores the space of the binary model parameters "around" the current variational probability vector ϕ.

It is important to choose a baseline vector b in an adaptive manner so as to minimize the variance of the gradient estimate [24]. This can be done as derived in (10) as [26]

$$b = \frac{\mathbb{E}\left[\left(\sum_{(x,z)\in\mathcal{D}} L_{(x,z)}(w)\nabla_\phi \ln q(w\,|\,\phi)\right)^2\right]}{\mathbb{E}\left[(\nabla_\phi \ln q(w\,|\,\phi))^2\right]}, \quad (13)$$

where again operations are carried out entry-wise. In order to estimate the expectations in (13), we use two moving averages, estimated using an exponentially decaying factor γ, one for the numerator and one for the denominator in (13) across the iterations [26]. Note that the baseline can also be updated locally and separately for each local parameter.

Moreover, training the model using the local update rule does not require estimating the response functions, i.e., the expectation of the outcomes of the quantum neurons. For quantum hardware, one can only take a finite number of measurements, so one can never determine a circuit's expectation values exactly. This makes training the proposed architecture less computationally intensive than variational quantum circuits relying on parameters shift rules that require taking the difference of two circuit expectation values, with forward and backward shifts in angles [22], [27], [16].

IV. Experiments

In this section, we provide experimental results to elaborate on the performance of the proposed training scheme.

A. Data Set

We consider a prototypical image data set, namely Bars-and-Stripes (BAS) which is widely used in related studies [14]–[16], [28], [29]. Each sample of size $d \times d$ illustrates bars, stripes, or random patterns. The samples are transformed from $d \times d$ pixel images to binary strings x of length d^2. As in [14], [15], [28], [29], we consider $d = 4$. The samples containing bars or stripes are labeled as $z = 1$, whilst the random patterns are labeled as $z = -1$.

B. Schemes and Benchmarks

We compare the performance of the SVO-based strategy with the perception-like scheme proposed in [14] to train single quantum neurons. To adapt the scheme in [14] to the two-layer architecture studied in this paper, we train each neuron individually as in [14]. A final classification decision is then made by the classical neuron in the second layer via a majority rule, i.e., a label is selected if it is chosen by at least half of the neurons. Specifically, following [14], the weight vector w^q_m for QGLM m is updated as follows. If the respective neuron classifies a sample with a negative label as positive, one flips a fixed portion $\eta \in [0, 1]$ of the binary weights, which are selected at random among the positions at which the signs of the weights and the sample coincide. Conversely, if the respective neuron classifies a positive sample as negative, one flips the same portion of the weights, which are selected at random among the positions at which the signs of the weights and the sample differ.

C. Model Architecture and Hyperparameters

We consider the hybrid classical-quantum model in Fig. 1 with $M = 32$ hidden quantum neurons, and a single classical output neuron. We use $|\mathcal{D}| = 30$ randomly selected samples for training, and testing is carried out with the same number of independently generated samples. We use SGD with 4000 training iterations. For the sign-flips scheme, the fraction of flipped signs is set to $\eta = 0.625$ (which was optimized

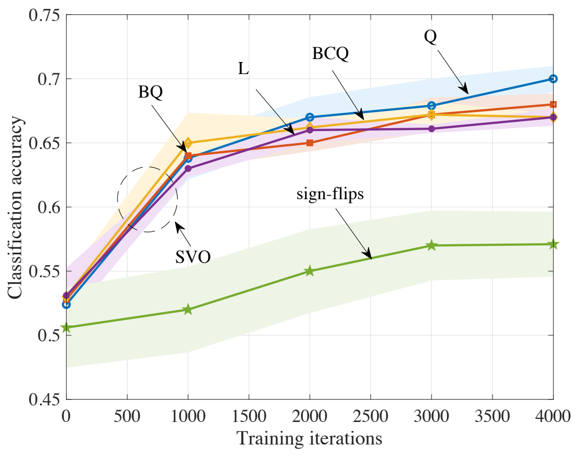

Fig. 2: Classification accuracy as a function of the training iteration for the benchmark sign-flips scheme [14] and the proposed SVO-based procedure for the BAS data set. The results are averaged over 5 independent trials.

numerically through exhaustive search). For SVO, the learning rate η is altered based on a cyclical schedule as in [30] as

$$\eta = \eta_{\text{base}} + (\eta_{\text{max}} - \eta_{\text{base}}) \\ \times \max(0, (1-t)) \times \left(1 + \sin\left(\frac{c\pi}{2}\right)\right), \quad (14)$$

where i denotes the current training iteration; $s = 1000$ denotes the step size; the base and maximum values for the learning rate are set to $\eta_{\text{base}} = 0.1$, and $\eta_{\text{max}} = 0.9$, respectively; and we have defined $c = \lceil (1 + i/(2s)) \rceil$ and $t = |i/s - 2c + 1|$. In addition, for SVO, the batch size is set to $|\mathcal{D}^b| = 16$, and the number of samples of the model parameters used to evaluate the expectations in (12) is set to 10. We adopt a standard sigmoid for the response function $g^c(\cdot)$ of the CGLM.

D. Results

We plot the classification accuracy for all schemes and benchmarks in Fig. 2 as a function of the training iterations. For SVO, we consider all response functions, while we illustrate only the quadratic (Q) response function for the sign-flips scheme, since it behaves similarly for all response functions. The sign-flips scheme [14] is seen to improve as training proceeds, however, the accuracy saturates after around $3,000$ iterations. The reason is that there is no single hyperplane separating the bars and the stripes from the random patterns, making the task difficult to solve for individual neurons in the ensemble. Conversely, the proposed SVO scheme is seen to achieve higher classification accuracy for all response functions. In particular, the QGLM using the Quadratic (Q) response function yields fastest convergence and achieves the best performance. Due to the additional bias terms resulting from the swap test routine, the QGLMs relying on the Biased quadratic (BQ) and Biased centered quadratic (BCQ) response functions are slower to learn, but ultimately converge after around $3,000$ training iterations.

V. Conclusion

In this paper, we studied a two-layer hybrid classical-quantum classifier with binary weights and activations in which a first layer of QGLMs is followed by a second classical combining layer. For the considered class of models, we proposed a SVO-based training scheme that results in local learning rules for joint weight optimization of the quantum and classical layers via stochastic gradient descent. The proposed method can be naturally extended to architectures with multiple layers of QGLMs (see [15]), although this is not elaborated on here due to the practical challenges associated with repeating amplitude encoding steps in between quantum layers.

References

[1] J. Preskill, "Quantum computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, 2018.
[2] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum machine learning," *Nature*, vol. 549, no. 7671, pp. 195–202, 2017.
[3] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, "Supervised learning with quantum-enhanced feature spaces," *Nature*, vol. 567, no. 7747, pp. 209–212, 2019.
[4] M. Schuld and F. Petruccione, *Machine Learning with Quantum Computers*. Springer, 2021.
[5] T. Hofmann, B. Schölkopf, and A. J. Smola, "Kernel methods in machine learning," *The annals of statistics*, vol. 36, no. 3, pp. 1171–1220, 2008.
[6] G. E. Hinton, "Boltzmann machine," *Scholarpedia*, vol. 2, no. 5, p. 1668, 2007.
[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.
[8] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.
[9] R. Mengoni and A. Di Pierro, "Kernel methods in quantum machine learning," *Quantum Machine Intelligence*, vol. 1, no. 3, pp. 65–71, 2019.
[10] M. H. Amin, E. Andriyash, J. Rolfe, B. Kulchytskyy, and R. Melko, "Quantum boltzmann machine," *Physical Review X*, vol. 8, no. 2, p. 021050, 2018.
[11] I. Cong, S. Choi, and M. D. Lukin, "Quantum convolutional neural networks," *Nature Physics*, vol. 15, no. 12, pp. 1273–1278, 2019.
[12] G. Verdon, T. McCourt, E. Luzhnica, V. Singh, S. Leichenauer, and J. Hidary, "Quantum graph neural networks," *arXiv preprint arXiv:1909.12264*, 2019.
[13] A. Mari, T. R. Bromley, J. Izaac, M. Schuld, and N. Killoran, "Transfer learning in hybrid classical-quantum neural networks," *Quantum*, vol. 4, p. 340, 2020.
[14] F. Tacchino, C. Macchiavello, D. Gerace, and D. Bajoni, "An artificial neuron implemented on an actual quantum processor," *npj Quantum Information*, vol. 5, no. 1, pp. 1–8, 2019.
[15] F. Tacchino, P. Barkoutsos, C. Macchiavello, I. Tavernelli, D. Gerace, and D. Bajoni, "Quantum implementation of an artificial feed-forward neural network," *Quantum Science and Technology*, vol. 5, no. 4, p. 044010, 2020.
[16] D. Arthur and P. Date, "A hybrid quantum-classical neural network architecture for binary classification," *arXiv preprint arXiv:2201.01820*, 2022.
[17] T. Bird, J. Kunze, and D. Barber, "Stochastic variational optimization," *arXiv preprint arXiv:1809.04855*, 2018.
[18] N. D. Mermin, *Quantum computer science: an introduction*. Cambridge University Press, 2007.
[19] O. Simeone, "A brief introduction to machine learning for engineers," *Foundations and Trends in Signal Processing*, vol. 12, no. 3-4, pp. 200–431, 2018.
[20] H. Kobayashi, K. Matsumoto, and T. Yamakami, "Quantum Merlin-Arthur proof systems: Are multiple Merlins more helpful to Arthur?" in *International Symposium on Algorithms and Computation*. Springer, 2003, pp. 189–198.
[21] Z. Zhao, J. K. Fitzsimons, and J. F. Fitzsimons, "Quantum-assisted gaussian process regression," *Physical Review A*, vol. 99, no. 5, p. 052331, 2019.
[22] M. Schuld, M. Fingerhuth, and F. Petruccione, "Implementing a distance-based classifier with a quantum interference circuit," *EPL (Europhysics Letters)*, vol. 119, no. 6, p. 60002, 2017.
[23] J. C. Spall, "A one-measurement form of simultaneous perturbation stochastic approximation," *Automatica*, vol. 33, no. 1, pp. 109–112, 1997.
[24] S. Mohamed, M. Rosca, M. Figurnov, and A. Mnih, "Monte carlo gradient estimation in machine learning." *J. Mach. Learn. Res.*, vol. 21, no. 132, pp. 1–62, 2020.
[25] D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, and J. Schmidhuber, "Natural evolution strategies," *Journal of Machine Learning Research*, vol. 15, no. 27, pp. 949–980, 2014.
[26] J. Peters and S. Schaal, "Reinforcement learning of motor skills with policy gradients," *Neural networks*, vol. 21, no. 4, pp. 682–697, 2008.
[27] J. Rivera-Dean, P. Huembeli, A. Acín, and J. Bowles, "Avoiding local minima in variational quantum algorithms with neural networks," *arXiv preprint arXiv:2104.02955*, 2021.
[28] M. Benedetti, D. Garcia-Pintos, O. Perdomo, V. Leyton-Ortega, Y. Nam, and A. Perdomo-Ortiz, "A generative modeling approach for benchmarking and training shallow quantum circuits," *npj Quantum Information*, vol. 5, no. 1, pp. 1–9, 2019.
[29] R. Y. Li, T. Albash, and D. A. Lidar, "Limitations of error corrected quantum annealing in improving the performance of boltzmann machines," *Quantum Science and Technology*, vol. 5, no. 4, p. 045010, 2020.
[30] L. N. Smith, "Cyclical learning rates for training neural networks," in *Proc. IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2017, pp. 464–472.