Rapid Communications

Implementing perceptron models with qubits

R. C. Wiersema* and H. J. Kappen
Department of Biophysics, Donders Institute, Radboud University, Nijmegen, The Netherlands

(Received 17 May 2019; published 26 August 2019)

We propose a method for learning a quantum probabilistic model of a perceptron. By considering a cross entropy between two density matrices we can learn a model that takes noisy output labels into account while learning. Although some work has been done that aims to utilize the curious properties of quantum systems to build a quantum perceptron, these proposals rely on the *ad hoc* introduction of a classical cost function for the optimization procedure. We demonstrate the usage of a quantum probabilistic model by considering a quantum equivalent of the classical log-likelihood, which allows for both a quantum model and training procedure. We show that this allows us to better capture noisiness in data compared to a classical perceptron. By considering entangled qubits we can learn nonlinear separation boundaries, such as XOR.

DOI: 10.1103/PhysRevA.100.020301

I. INTRODUCTION

One of the goals of quantum machine learning is to integrate quantum physics with machine learning to develop novel algorithms for learning classical data, so-called quantum-inspired models [1–5]. Along with these developments another goal has been to come up with machine learning algorithms for quantum computers, either by designing specific algorithms for quantum computers [6,7], or by speeding up the underlying linear algebra routines [8]. Examples of the former include employing adiabatic quantum annealers to train a binary classifier [6] and using a quantum computer to calculate an classically intractable kernel function [7], whereas the latter includes support vector machines [9], support matrix machines [10], A-optimal projections [11], and principal component analysis (PCA) [12]. However, most of these proposals remain unfeasible due to the current limitations of modern quantum computers, which still lack long qubit (the quantum mechanical description of a single spin-$\frac{1}{2}$ particle) coherence times and high gate fidelity [13].

Inspired by the success of deep learning [14], there has been interest to develop quantum equivalents of neural networks that can be trained more efficiently or are more expressive than their classical counterparts [15–22]. Quantum-inspired proposals utilize quantum effects in different ways: employing a superposition of perceptrons [17], using qubit weights [18,20], or learning a unitary transformation between input and output [19]. Quantum computing work in this direction involves using an inverse quantum Fourier transform to obtain a nonlinear step function [21] or tracing out parts of a quantum circuit to create an autoencoder [22]. However, all these proposals introduce a classical cost function for learning, omitting the underlying probabilistic motivation for their model. The usage of quantum probabilistic cost functions is still relatively unexplored.

Constructing quantum probabilistic models from density matrices is a recent direction of quantum machine learning research [5,23], where one exploits quantum effects in both the model and training procedure by constructing a differentiable cost function in terms of density matrices. Density matrices are used in quantum mechanics to describe statistical ensembles of quantum states. They are represented by a positive semidefinite Hermitian matrix with trace 1. In this Rapid Communication, we will use density matrices to construct a model that minimizes a generalization of the classical likelihood function for learning, replacing the classical perceptron bit with a qubit. Others have attempted to generalize probability theory to density matrices [24]. However, the equivalent of conditional probabilities, conditional density matrices, do not preserve positive definiteness so states can be assigned a negative probability [25]. Our approach bypasses this difficulty because we construct a data density matrix from the probability amplitude of the empirical data distribution, which is always positive semidefinite.

The desired perceptron is a linear classifier that can be used for binary classification. It assigns a probability

$$p(y = 1|\mathbf{x}) = f(\mathbf{x} \cdot \mathbf{w}) \tag{1}$$

to class $y = 1$, based on input $\mathbf{x}$ and trainable weights $\mathbf{w}$ with $f(x)$ a nonlinear activation function. The activation function of the perceptron is often taken to be a sigmoid, since it produces an output between 0 and 1 and is equivalent to logistic regression. The perceptron is of particular interest in machine learning because it is the building block of multilayer neural networks, the driving force behind deep learning.

In Sec. II we will consider a qubit perceptron that uses a generalization of the classical likelihood function for learning. Some numerical results for toy data sets are discussed in Sec. III, where we show that our qubit model is better at assigning class probability for noisy data. In Sec. IV we will consider two entangled qubits as a perceptron that can learn nonlinear problems by assigning a nonlinear separation boundary.

*roeland.wiersema@student.ru.nl

II. QUANTUM PERCEPTRON

Consider a classification problem where we have a data set consisting of input vectors $\mathbf{x} \in \mathbb{R}^d$ of length d with corresponding labels $y \in \{1, -1\}$. In supervised machine learning it is our goal is to find the parameters $\mathbf{w}$ for the function $p(y|\mathbf{x}; \mathbf{w})$ that assigns a high probability to the correct label y for each input $\mathbf{x}$. The classical negative log-likelihood is given by

$$\mathcal{L}_{cl} = -\sum_{\mathbf{x}} q(\mathbf{x}) \sum_{y} q(y|\mathbf{x}) \ln p(y|\mathbf{x}; \mathbf{w}). \tag{2}$$

Here, $q(\mathbf{x})$ is the empirical probability of observing $\mathbf{x}$, $q(y|\mathbf{x})$ is the empirical conditional probability of observing label y for data $\mathbf{x}$, and $p(y|\mathbf{x}, \mathbf{w})$ is the proposed model conditional probability distribution of the data. By performing a gradient descent we can find the optimal parameters for our model, which is equivalent to minimizing the cross entropy between distributions p and q.

To extend the classical likelihood in Eq. (2) to the realm of quantum mechanics we require a description of our model and the conditional probability $q(y|\mathbf{x})$ in terms of density matrices. The density matrix contains the classical uncertainty we have about a quantum state. If this matrix is rank one, we have what is known as a pure state in which case there is no classical uncertainty about what quantum state the system is in. If the density matrix has rank > 1, then we have a so-called mixed state [26]. For our model we will consider a parametrized mixed state, since this will allow us to capture the uncertainty in the data. To perform learning, we require a learning rule that preserves the Hermiticity, positive semidefiniteness and trace of the density matrix.

We consider the specific case where the data consist of N discrete vectors $\mathbf{x} \in \{1, -1\}^d$ with d bits and $y \in \{1, -1\}$ labels. We define the quantum log-likelihood as a cross entropy between a conditional data density matrix $\eta_{\mathbf{x}}$ and a model conditional density matrix $\rho_{\mathbf{x}}$, analogous to Eq. (2). For each $\mathbf{x}$ we construct a wave function based on the empirical conditional probabilities $q(y|\mathbf{x})$,

$$|\Psi\rangle = \sqrt{q(1|\mathbf{x})}|1\rangle + \sqrt{q(-1|\mathbf{x})}|-1\rangle, \tag{3}$$

where the states $|1\rangle, |-1\rangle$ are the eigenstates of the σ^z operator. The data density matrix is defined as $\eta_{\mathbf{x}} \equiv |\Psi\rangle\langle\Psi|$, with components

$$\eta_{\mathbf{x}}(y, y') = \sqrt{q(y|\mathbf{x})}\sqrt{q(y'|\mathbf{x})}. \tag{4}$$

Note that this is a pure density matrix. $q(y|\mathbf{x})$ is an empirical distribution over the label y for each $\mathbf{x}$, and is fully determined by its conditional expectation value of y given $\mathbf{x}$ written as $b(\mathbf{x})$,

$$q(y|\mathbf{x}) = \tfrac{1}{2}[1 + b(\mathbf{x})y], \tag{5}$$

with

$$b(\mathbf{x}) = \frac{1}{M}\left(\sum_{\mathbf{x}'} y' \mathbb{I}(\mathbf{x}' = \mathbf{x})\right)$$

and

$$M = \sum_{\mathbf{x}'} \mathbb{I}(\mathbf{x}' = \mathbf{x}).$$

Succinctly put, every time $\mathbf{x}$ appears in the data, we add its corresponding label y' to the sum. Dividing by M, the total number of times the sample $\mathbf{x}$ appears in the data we obtain the conditional expectation value $b(\mathbf{x})$. We define the empirical probability

$$q(\mathbf{x}) = \frac{M}{N}$$

for M occurrences of $\mathbf{x}$ and N the total number of samples.

Our model is a density matrix $\rho(\mathbf{x}, \mathbf{w}; y, y') \equiv \rho_{\mathbf{x}}$. We use the following proposal,

$$\rho_{\mathbf{x}} = \frac{1}{Z} e^{-\beta H}, \tag{6}$$

where $H = \sum_k h^k \sigma^k$, with $h^k \in \mathbb{R}$ and σ^k the Pauli matrices with $k = (x, y, z)$. This is a finite-temperature description of a qubit, where we will set $\beta = -1$ for now. Using that $\exp(a\,\hat{\mathbf{n}} \cdot \boldsymbol{\sigma}) = \cosh(a) + \sinh(a) \sum_k \sigma^k$ and writing $\sum_k h^k \sigma^k = h \sum_k \frac{h^k}{h} \sigma^k$ with $h = \sqrt{\sum_k (h^k)^2}$, we find

$$\rho_{\mathbf{x}} = \frac{1}{Z}\left(\cosh h + \sinh h \sum_k \frac{h^k \sigma^k}{h}\right). \tag{7}$$

Solving $\mathrm{Tr}\{\rho_{\mathbf{x}}\} = 1$ gives $Z = 2 \cosh h$. Then,

$$\begin{aligned} \rho_{\mathbf{x}} &= \frac{1}{2} I + \frac{1}{2} \tanh h \sum_k \frac{h^k \sigma^k}{h} \\ &= \frac{1}{2} I + \frac{1}{2} \sum_k m^k \sigma^k, \end{aligned} \tag{8}$$

where I is a 2×2 identity matrix and $m^k = \frac{h^k}{h} \tanh h$. Equation (8) gives us the general description of a qubit, which we have now described in terms of a density matrix. This definition spans the space of 2×2 Hermitian matrices, for all $h^k \in \mathbb{R}$. From the definition of m^k it is clear that $m^k \in (-1, 1)$. This means that $\rho_{\mathbf{x}}$ is positive semidefinite because the eigenvalues of $\rho_{\mathbf{x}}$ are

$$\lambda_{\pm} = \frac{1}{2}\left(1 \pm \sqrt{\sum_k (m^k)^2}\right) \geqslant 0. \tag{9}$$

From the eigenvalues we also see that $\rho_{\mathbf{x}}$ describes a mixed state, since it is only rank one if $\sum_k (m^k)^2 = 1$.

We now parametrize the field $h^k \to h^k(\mathbf{x})$ by setting $h^k(\mathbf{x}) = \mathbf{w}^k \cdot \mathbf{x}$ with $\mathbf{w}^k \in \mathbb{R}^d$, so that the qubit state is dependent on classical input data. We can absorb the inverse temperature $-\beta$ in the field $-\beta h^k \to h^k$ by rescaling the weights $\mathbf{w}^k$. Note that for each Pauli matrix k, we have one set of weights $\mathbf{w}^k$. To clean up the notation we omit the argument of h^k from now on. We now generalize Eq. (2) with our data and model density matrices $\eta_{\mathbf{x}}$ and $\rho_{\mathbf{x}}$ to obtain the negative quantum log-likelihood,

$$\mathcal{L}_q = -\sum_{\mathbf{x}} q(\mathbf{x}) \mathrm{Tr}\{\eta_{\mathbf{x}} \ln(\rho_{\mathbf{x}})\}. \tag{10}$$

This is the quantum mechanical equivalent of the classical log-likelihood which minimizes the "distance" between the density matrix representations of the data and the model. This

expression also appears in the quantum relative entropy, and for $\eta_{\mathbf{x}} > 0$ the quantum log-likelihood is convex in $\rho_{\mathbf{x}}$ [27]. Next, we rewrite this with our parametrized $\rho_{\mathbf{x}}$,

$$\begin{aligned}\mathcal{L}_q &= -\sum_{\mathbf{x}} q(\mathbf{x}) \mathrm{Tr}\{\eta_{\mathbf{x}} \ln(\rho_{\mathbf{x}})\} \\ &= -\sum_{\mathbf{x}} q(\mathbf{x}) \sum_{y,y'} \langle y'|\sqrt{q(y|\mathbf{x})}\sqrt{q(y'|\mathbf{x})} \ln(\rho_{\mathbf{x}})|y\rangle, \end{aligned} \tag{11}$$

with $\{|y\rangle\}$ a set of orthonormal vectors in the σ^z basis,

$$\begin{aligned}&-\sum_{\mathbf{x}} q(\mathbf{x}) \sum_{y,y'} \sqrt{q(y|\mathbf{x})}\sqrt{q(y'|\mathbf{x})} \\ &\quad\times \langle y'|\left(\sum_k h^k \sigma^k - \ln(2\cosh h)\right)|y\rangle. \end{aligned} \tag{12}$$

Calculating the statistics for the Pauli matrices gives

$$\sum_{y,y'} \langle y'| \sum_k h^k \sigma^k |y\rangle = \sum_{y,y'} \sum_k \langle y'|h^k \sigma^k |y\rangle, \tag{13}$$

which gives three delta functions that we can plug into Eq. (12) together with our definition of $q(y|\mathbf{x})$ from Eq. (5),

$$\begin{aligned}&\sum_{y,y'} \sqrt{q(y|\mathbf{x})}\sqrt{q(y'|\mathbf{x})}(h^x \delta_{y',-y} + iyh^y \delta_{y',-y} + yh^z \delta_{y',y}) \\ &\quad = h^x \sqrt{1-b(\mathbf{x})^2} + h^z b(\mathbf{x}). \end{aligned} \tag{14}$$

The h^x term quantifies how often a sample occurs with a flipped output label and is the distinguishing factor from the classical perceptron. The source of this term is the σ^x matrix in the likelihood which flips the state $|y\rangle$ and scales h^x with the off-diagonal elements of $\eta_{\mathbf{x}}$. As a final likelihood we get

$$\mathcal{L}_q = -\sum_{\mathbf{x}} q(\mathbf{x})[h^x \sqrt{1-b(\mathbf{x})^2} + h^z b(\mathbf{x}) - \ln(2\cosh h)]. \tag{15}$$

In order to perform learning we have to find update rules that minimize the function in Eq. (15). To find the minimum we perform a gradient descent to update the parameters $\mathbf{w}^k$. Derive with respect to $\mathbf{w}^k$,

$$\begin{aligned}\frac{\partial \mathcal{L}_q}{\partial \mathbf{w}^x} &= -\sum_{\mathbf{x}} q(\mathbf{x})\left(\sqrt{1-b(\mathbf{x})^2} - \frac{h^x}{h}\tanh h\right)\mathbf{x}, \\ \frac{\partial \mathcal{L}_q}{\partial \mathbf{w}^y} &= \sum_{\mathbf{x}} q(\mathbf{x})\left(\frac{h^y}{h}\tanh h\right)\mathbf{x}, \\ \frac{\partial \mathcal{L}_q}{\partial \mathbf{w}^z} &= -\sum_{\mathbf{x}} q(\mathbf{x})\left(b(\mathbf{x}) - \frac{h^z}{h}\tanh h\right)\mathbf{x}. \end{aligned} \tag{16}$$

Update the weights at iteration t with

$$\mathbf{w}^k(t+1) = \mathbf{w}^k(t) - \epsilon\left(\frac{\partial \mathcal{L}}{\partial \mathbf{w}^k(t)}\right). \tag{17}$$

These are the learning rules for the quantum perceptron, with learning parameter ϵ for each gradient. Since the gradient step of $\mathbf{w}^y$ is proportional to $\mathbf{w}^y$, the fixed-point solution is $\mathbf{w}^y \to \mathbf{0}$ in the limit of many iterations. In the case that there exists a function $f(\mathbf{x}) = y$ (no noise in the data) for all data points, the statistics $b(\mathbf{x})$ become either 1 or -1, which gives a fixed-point solution $\mathbf{w}^x \to \mathbf{0}$. The h^z field then corresponds to the single field of a classical perceptron and the quantum perceptron approaches the classical case. However, in the case where there are samples which have both 1 and -1 labels, the weight $\mathbf{w}^x$ becomes finite and the solution of the quantum perceptron will diverge from the classical perceptron. This change in behavior is reflected in the probability boundaries, which differ from the classical case (see Supplemental Material A [28]).

We have yet to address how we actually retrieve the class label y from the model. Once trained, we can construct a state $\rho_{\mathbf{x}}$ of the qubit based on some input $\mathbf{x}$. The output labels $y \in \{-1, 1\}$ correspond to the states $|-1\rangle$, $|1\rangle$ by construction. An obvious measure of probability is the expectation value $\langle\sigma^z\rangle_{\rho_{\mathbf{x}}}$, which gives $p(y|\mathbf{x};\mathbf{w}) = \frac{1}{2}(1 + y\langle\sigma^z\rangle_{\rho_{\mathbf{x}}})$. For a finite-temperature system we have for the expectation value of some observable $\hat{A}$,

$$\langle\hat{A}\rangle = \mathrm{Tr}\{\hat{A}\rho\}. \tag{18}$$

From our definition in Eq. (8) we see that

$$\langle\sigma^z\rangle_{\rho_{\mathbf{x}}} = \mathrm{Tr}\left\{\sigma^z \frac{1}{2}\left(1 + \sum_k m^k \sigma^k\right)\right\} = \delta_{kz} m^k = m^z, \tag{19}$$

where we used that $\mathrm{Tr}\{\sigma^i\} = 0$ and $\mathrm{Tr}\{\sigma^i \sigma^j\} = 2\delta_{ij}$. The class probability is then constructed as

$$p(y|\mathbf{x};\mathbf{w}) = \tfrac{1}{2}(1 + ym^z). \tag{20}$$

III. RESULTS

In this section we apply the quantum perceptron to some toy data sets and compare with the classical perceptron with a sigmoid activation function, i.e., logistic regression. For both the classical and quantum perceptron we look at the mean squared error (MSE) to evaluate the performance of both methods,

$$\mathrm{MSE} = \frac{1}{N}\sum_i^N [y_i - p(y_i|\mathbf{x_i};\mathbf{w})]^2. \tag{21}$$

We always reach the global minimum through batch gradient descent because the cost functions are convex for both models. Due to the flatness of the likelihood function near the global minimum, convergence can be slow. Setting the threshold for convergence at $\Delta\mathcal{L} < 10^{-7}$ and the learning parameter at $\epsilon = 0.01$ ensures that we obtain fast convergence without sacrificing model accuracy for the problems discussed in this Rapid Communication.

A. Two-dimensional binary problem

In order to demonstrate the difference between the classical and quantum perceptron we consider a two-dimensional binary classification problem. If the problem is linearly separable, the classical perceptron converges to a solution where the two classes are perfectly separated. In the case where some samples are "mislabeled" the quantum perceptron should behave differently, because we account for noise in the learning rule.

Consider the data $\mathbf{x} = \{(1, 1), (1, -1), (-1, 1), (-1, -1)\}$ with labels $y = \{-1, -1, 1, -1\}$, respectively. This problem

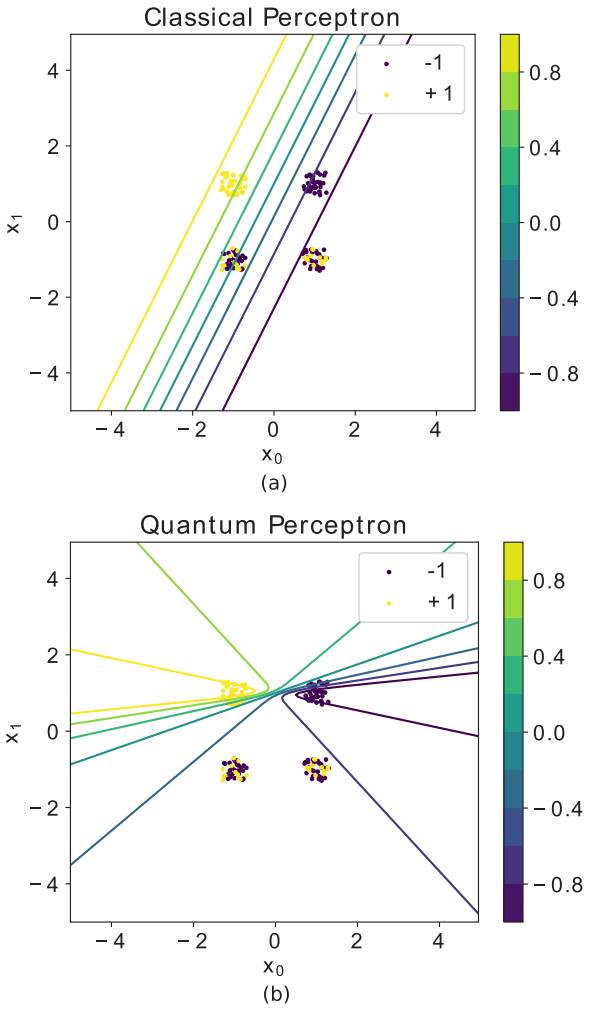

FIG. 1. Separation boundaries in the input space for a two-dimensional problem with $\mathbf{x} = (x_0, x_1)$. The contour lines indicate the expectation value $\mathbb{E}[y|\mathbf{x};\mathbf{w}] \in (-1, 1)$. The 0.0 line indicates the separation boundary where $p(y = 1|\mathbf{x};\mathbf{w}) = p(y = -1|\mathbf{x};\mathbf{w}) = \frac{1}{2}$. Random jitter is added to the plot to clarify which samples are noisy. (a) Classical perceptron. The classical perceptron assigns linear boundaries through the input space, where the distance between the boundaries is scaled with the sigmoid. (b) Quantum perceptron. The quantum perceptron assigns curved boundaries through the input space. Samples with mislabelings get assigned a lower expectation value which results in a lower MSE of MSE(quantum) ≈ 0.106 for the quantum perceptron vs MSE(classical) ≈ 0.154 for the classical perceptron. Note that if we threshold the quantum perceptron boundary at $p(y = 1|\mathbf{x};\boldsymbol{\theta}) = 0.5$, we get a linear boundary that would assign similar classes as in (a), even though the boundary is tilted with respect to the classical boundary. However, the quantum perceptron assigns high probabilities to classes about which it is certain [$\mathbf{x} \in \{(-1, 1), (1, 1)\}$] and lower probabilities to classes about which it is uncertain [$\mathbf{x} \in \{(-1, -1), (1, -1)\}$]. The classical perceptron does this significantly worse, which is reflected in the difference in MSE.

is trivial since it is linearly separable and all algorithms converge to the same solution ($\mathbf{w}^{x,y} = \mathbf{0}$ and $\mathbf{w}^{z} \approx \mathbf{w}_{\text{cl}}$). However, if we flip some of the output labels to simulate mislabeled samples or errors in the data, we suspect that the quantum perceptron will perform better. We make 40 copies of the four data points in the binary feature space and for $\mathbf{x} \in \{(1, -1), (-1, -1)\}$ we flip 30% of the outputs from -1 to 1. The probability boundaries of the perceptrons differ significantly, as can be seen in Fig. 1, which leads to a better assignment of the probability of the correct states.

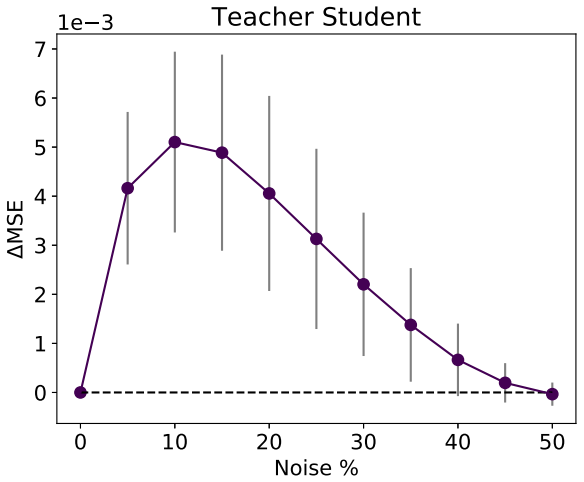

FIG. 2. $\Delta\text{MSE} = \text{MSE(classical)} - \text{MSE(quantum)}$ vs the percentage of labels flipped in the training data. Error bars indicate the standard deviation over 100 different $\mathbf{w}_{\text{teacher}}$ initializations. If the amount of noise is 0%, the classical and quantum perceptron will converge to the same solution. If the amount of noise is 50%, then both models cannot learn anything. Between these two points lies an area where the quantum perceptron outperforms the classical perceptron.

B. Binary teacher-student problem

A more complex, higher-dimensional problem is the teacher-student problem. The input data $\mathbf{x} \in \mathbb{R}^d$ consist of 600 random binary vectors of length $d = 8$, where $\mathbf{x} \in \{-1, 1\}^d$. We take a random weight vector $\mathbf{w}_{\text{teacher}} \sim \mathcal{N}(0, 1)$ and determine labels $y = \text{sgn}(\mathbf{x} \cdot \mathbf{w}_{\text{teacher}})$. We then create five duplicates of each input vector to ensure that there are multiple copies of each sample in the data set. Next, we flip some percentage of labels for 80% of this data set (the training set). This is done by generating a random permutation of the indices of the samples and flipping the label for the first x% of them. After training both the classical and quantum perceptron we predict the labels for the remaining 20% of the data (the test set) and calculate the difference in MSE between the two models. The percentage of flipped labels was incrementally increased by 5% from 0% to 50%. At each step in this schedule we learn 100 different $\mathbf{x}$ and $\mathbf{w}_{\text{teacher}}$ to gather statistics for the mean and variance of ΔMSE. The 100 generated problems are equal across the different percentages. This setup allows us to assert whether the algorithms can still find the original separation of the data even if noise is introduced. The performance of the quantum perceptron and classical perceptron is compared in Fig. 2.

We have shown that the quantum probabilistic description is better than a classical perceptron in capturing uncertainty in toy data sets. At the cost of introducing an additional

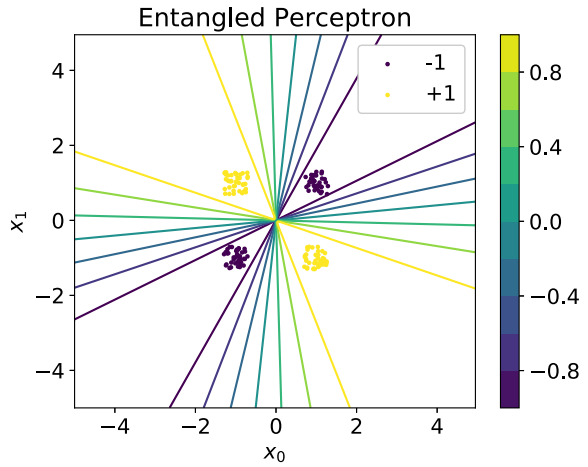

FIG. 3. The XOR problem. Perfect classification of this nonlinear data set requires four classical perceptrons in a two-layer configuration or a kernel transformation $(x_0, x_1) \to (x_0, x_1, \sqrt{x_0^2 + x_1^2})$. We show that the problem can be learned perfectly with two qubits.

parameter $\mathbf{w}^x$, the model is more expressive, which allows for a better characterization of the data.

IV. ENTANGLED PERCEPTRON

In this section we demonstrate the use of entanglement for learning. This can be achieved by extending the previous ideas to a multiqubit system. Consider the Hilbert space $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B$, with $i, j = 0, 1$. Let $\{|\phi_i\rangle\}$ be an orthonormal basis for the 2×2 Hilbert spaces $\mathcal{H}_A$ and $\mathcal{H}_B$. We can write down an arbitrary state in $\mathcal{H}$ as

$$|\phi\rangle = \frac{1}{\sqrt{N}} \sum_{i,j} h^{ij} |\phi_i\rangle \otimes |\phi_j\rangle, \tag{22}$$

where $h^{ij} \in \mathbb{C}$. We must normalize $|\phi\rangle$ accordingly to ensure that $\langle\phi|\phi\rangle = 1$, with $\langle\phi|\phi\rangle = \sum_{ij} h^{ij*} h^{ij} \equiv N$. This state can be described with a density matrix that is rank one because we are dealing with a pure state. Since $\rho \neq \rho_A \otimes \rho_B$ in general the state can be entangled. If we now look at the reduced density matrix ρ_B by tracing out qubit A we end up with a mixed state,

$$\rho_B = \frac{1}{N} \sum_{i,j,j'} h^{ij*} h^{ij'} |\phi_j\rangle\langle\phi_{j'}|. \tag{23}$$

If we take $h^{ij} = \mathbf{w}^{ij} \cdot \mathbf{x}$ with $\mathbf{w}^{ij} \in \mathbb{C}^d$, then we have constructed a quantum state parametrized by our inputs. With the data density matrix we used in Eq. (4) we can again minimize the quantum log-likelihood in Eq. (10) by replacing $\rho_{\mathbf{x}}$ with ρ_B. We can now learn nonlinear problems as can be seen in Fig. 3. An explanation of the quadrics and the shape of the boundaries as well as additional examples can be found in Supplemental Material B [28].

V. CONCLUSION

We extended the classical likelihood to a quantum log-likelihood and constructed a quantum perceptron from density matrices. The resulting algorithm is more resistant to noisy data when learning and takes this noisiness into account when predicting. This is due to the fact that there is a cost for flipped output labels in the quantum log-likelihood. For toy data sets we observed that the quantum perceptron is better at assigning probability to noisy samples, which resulted in improved performance. When we considered the extension to two entangled qubits, we could also learn nonlinear separation boundaries.

In this Rapid Communication we have only considered binary classification, but the quantum perceptron can easily be extended to multiclass regression for $C > 2$ classes by considering the SU(C) generators instead of the Pauli matrices. These generators span the space of $C \times C$ traceless Hermitian matrices. We are then working with qudits, which generalize the properties of qubits to d-level quantum systems.

A caveat of the quantum perceptron is that in order to outperform a classical perceptron, we require multiple copies of a sample $\mathbf{x}$ with conflicting labels y to be present in the data, otherwise $b(\mathbf{x}) = \pm 1$ for all data points and the algorithm simply reduces to the classical perceptron. Most real-world data sets, however, contain either a large number of features or continuous variables so that copies of samples $\mathbf{x}$ with conflicting labels are rare. This limits the increase in performance to edge cases where there is discrete input data with a small number of features. Nonetheless, we have shown that at the cost of introducing a single parameter $\mathbf{w}^x$, the density matrix construction is a more general model than the classical perceptron.

To conclude, we have shown that it is possible to learn a quantum model using a quantum cost function and that this can lead to improved performance for toy data sets. We believe that this modeling paradigm could be a fruitful direction for developing algorithms for noisy intermediate scale quantum computers, since the quantum probabilistic approach is still relatively unexplored in the current literature.

The code with the TENSORFLOW model of the quantum perceptron and ways to reproduce the figures in this Rapid Communication can be found on GitHub [29].

ACKNOWLEDGMENT

We thank J. Mentink and the people involved with the "Bits and Brains" project for inspiring discussions. This research was funded in part by ONR Grant No. N00014-17-1-256.

[1] S. Yang, M. Wang, and L. Jiao, A quantum particle swarm optimization, in *Proceedings of the 2004 Congress on Evolutionary Computation* (IEEE, Piscataway, NJ, 2004), Vol. 1, pp. 320–324.

[2] E. Stoudenmire and D. J. Schwab, Supervised learning with tensor networks, in *Advances in Neural Information Processing Systems* (Curran Associates, Red Hook, NY, 2016), Vol. 29, pp. 4799–4807.

[3] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, Nature (London) **549**, 195 (2017).

[4] Z. Xie and I. Sato, A quantum-inspired ensemble method and quantum-inspired forest regressors, in *Proceedings of the Ninth Asian Conference on Machine Learning* (PMLR, 2017), Vol. 77, pp. 81–96.

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

[6] H. Neven, V. S. Denchev, G. Rose, and W. G. Macready, Training a binary classifier with the quantum adiabatic algorithm, arXiv:0811.0416.

[7] M. Schuld and N. Killoran, Quantum Machine Learning in Feature Hilbert Spaces, Phys. Rev. Lett. **122**, 040504 (2019).

[8] A. W. Harrow, A. Hassidim, and S. Lloyd, Quantum Algorithm for Linear Systems of Equations, Phys. Rev. Lett. **103**, 150502 (2009).

[9] P. Rebentrost, M. Mohseni, and S. Lloyd, Quantum Support Vector Machine for Big Data Classification, Phys. Rev. Lett. **113**, 130503 (2014).

[10] B. Duan, J. Yuan, Y. Liu, and D. Li, Quantum algorithm for support matrix machines, Phys. Rev. A **96**, 032301 (2017).

[11] B. Duan, J. Yuan, J. Xu, and D. Li, Quantum algorithm and quantum circuit for A-optimal projection: Dimensionality reduction, Phys. Rev. A **99**, 032311 (2019).

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

[13] J. Preskill, Quantum computing in the NISQ era and beyond, Quantum **2**, 79 (2018).

[14] Y. LeCun, Y. Bengio, and G. Hinton, Deep learning, Nature (London) **521**, 436 (2015).

[15] M. Schuld, I. Sinayskiy, and F. Petruccione, The quest for a quantum neural network, Quantum Inf. Process. **13**, 2567 (2014).

[16] S. K. Jeswal and S. Chakraverty, Recent developments and applications in quantum neural network: A review, Arch. Computat. Methods Eng. (2018).

[17] J. Zhou, Q. Gan, A. Krzyżak, and C. Y. Suen, Recognition of handwritten numerals by quantum neural network with fuzzy features, Int. J. Doc. Anal. Recogn. **2**, 30 (1999).

[18] N. Kouda, N. Matsui, H. Nishimura, and F. Peper, Qubit neural network and its learning efficiency, Neural Comput. Appl. **14**, 114 (2005).

[19] R. Zhou and Q. Ding, Quantum M-P neural network, Int. J. Theor. Phys. **46**, 3209 (2007).

[20] S. Fuhua, Quantum-inspired neural network with quantum weights and real weights, Open J. Appl. Sci. **5**, 609 (2015).

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

[22] K. H. Wan, O. Dahlsten, H. Kristjánsson, R. Gardner, and M. S. Kim, Quantum generalisation of feedforward neural networks, npj Quantum Inf. **3**, 36 (2017).

[23] H. J. Kappen, Learning quantum models from quantum or classical data, arXiv:1803.11278.

[24] M. K. Warmuth and D. Kuzmin, A Bayesian probability calculus for density matrices, Mach. Learn. **78**, 63 (2009).

[25] N. J. Cerf and C. Adami, Quantum extension of conditional probability, Phys. Rev. A **60**, 893 (1999).

[26] M. A. Nielsen and I. L. Chuang, in *Quantum Computation and Quantum Information*, 10th ed. (Cambridge University Press, Cambridge, UK, 2011). pp. 98–111.

[27] E. Carlen, in *Trace Inequalities and Quantum Entropy: An Introductory Course* (American Mathematical Society, Providence, RI, 2010), Vol. 529, pp. 73–140.

[28] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevA.100.020301 for a mathematical analysis of the possible quantum perceptron separation boundaries, and additional examples of toy data sets.

[29] R. C. Wiersema, Code: Implementing perceptron models with qubits, https://github.com/therooler/qperceptron (2019).