Quantum Machine Learning Algorithm

July 11, 2021

Abstract

Fuelled by increasing computer power and algorithmic advances, machine learning techniques have become powerful tools for finding features in data. Quantum systems produce atypical patterns that classical systems are thought not to produce e iciently, so it is reasonable to postulate that quantum computers may outperform classical computers on machine learning tasks.

The quantum machine learning originated at 2009, while HHL(Harrow-Hassidim-Lloyd) algorithm was tendered for solving linear equation. It is based on QFT(Quantum Phase Estimation) and Phase Estimation Algorithm, and is extended to QPCA(Quantum Principle Component Analysis) and QSVM(Quantum Support Vector Machine).

Also, with the research of deep learning, neural network has become more and more essential. The breakthrough of Quantum neural network occurred about 2018, when the calculation of gradient based on quantum computers was proposed.

Our curriculum paper focus on traditional quantum machine learning and the basic gradient calculation scheme of quantum neural network.

Keywords: quantum machine learning, HHL algorithm, QPCA, QSVM, quantum neural network, quantum gradient, quantum circuit learning.

1 The HHL algorithm

1.1 Problem formulation

HHL algorithm is for solving the linear equation. The problem is: given a system $A\vec{x} = \vec{b}$, find vector $\vec{x}$ for matrix A and vector $\vec{b}$, assuming matrix A is Hamilton.

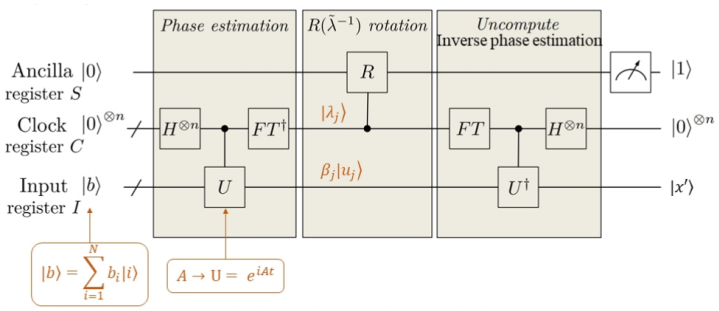

Figure 1: HHL algorithm

1.2 The algorithm modules

- **State preparation and gate construction**: we assume that we gain a operator $U = e^{-iAt}$, and we neglect the construction of U here and pose an example in QPCA to construct the gate. The state $|b\rangle = \sum_{i=1}^{N} b_i |i\rangle = \sum_{j=1}^{N} \beta_j |u_j\rangle$ is encoded from $\vec{b}$ using amplitude encoding. Note that $|b\rangle$ is a supposition state rather than eigenstate in phase estimation. [1]

- **Phase estimation module:**
  (1) In phase estimation: $|0\rangle |u\rangle - - > |\lambda\rangle |u\rangle$
  (2) In HHL: $|0\rangle |b\rangle = |0\rangle (\sum_{j=1}^{N} \beta_j |u_j\rangle) - - > \sum_{j=1}^{N} \beta_j |\lambda_j\rangle |u_j\rangle$
  Note that this phase estimation module construct an entangled state $\sum_{j=1}^{N} \beta_j |\lambda_j\rangle |u_j\rangle$.

- **Rotation module**: we use the Clock register in Figure 1 as the control qubits. Therefore, we get the output:

$$\sum_{j=1}^{N} (\sqrt{1 - C^2/{\lambda_j}^2}\,|0\rangle + C/\lambda_j\,|1\rangle)\beta_j\,|\lambda_j\rangle\,|u_j\rangle$$

- **Uncompute Module**: the QFT combined with $U^{\dagger} = e^{-iAt}$ turn the clock qubit to $|0\rangle^{\otimes n}$,

$$\sum_{j=1}^{N} (\sqrt{1 - C^2/{\lambda_j}^2}\,|0\rangle + C/\lambda_j\,|1\rangle)\beta_j\,|0\rangle\,|u_j\rangle$$

- **Measurement**: we perform the measurement for the ancillary qubit, since it entangles with Input register qubit and clock register, if the output is $|1\rangle$, then we get:

$$|x\rangle \sim \sum_{j}^{N} C\beta_j/\lambda_j |u_j\rangle$$

which is proportional to the output we need.

2 Quantum principle component analysis

2.1 Problem formulation

Principal component analysis is a tool in data analysis for revealing the internal structure of the data by projecting the data on to the direction of maximal variance. The essential work is still eigenvalue decomposition.
Note that:
(1) The input is a mixed state ρ, which is different from HHL;
(2) The construction of $U = e^{-i\rho t}$ is non-trivial that needs to be paid attention to;

2.2 The construction of U

- **The construction of mixed state** ρ:
(1)We suppose that the density matrix corresponds to the covariance matrix of a set of data vectors $\vec{a_i} \in \mathbb{C}^d$. Define the covariance matrix $\rho = AA^\dagger$. In quantum mechanical form, $A = \sum_i |a_i| |a_i\rangle \langle e_i|$,where the $|a_i\rangle$ are normalized.[2]
(2) Assume we use the QRAM[3] (Quantum Random Access Memory) that takes $|i\rangle |0\rangle |0\rangle$ to $|i\rangle |a_i\rangle ||a_i|\rangle$, to construct the entangled state $\sum_i |a_i| |e_i\rangle |a_i\rangle$, **note that the mixed state stems from entangled state in higher dimension**.
(3) The density matrix for second register (the partial trace on second register) is proportional to ρ.
(4) Using the QRAM, the construction of state ρ is $O(logd)$, where d is the dimension of state vector.

- **The construction for U**:
Assume we have a density matrix σ now and want to perform the operator $U = e^{-i\rho t}$ on it, which formulates as $e^{-i\rho t}\sigma e^{i\rho t}$.(Note that the operator on density matrix is the form of $\sigma^{'} = U\sigma U^\dagger$).We first prove two equalities and then explain the function.

$$e^{-i\rho\Delta t}\sigma e^{i\rho\Delta t} = \sigma - i\Delta t[\rho, \sigma] + O(\Delta t^2) \tag{1}$$

$$tr_P(e^{-iS\Delta t}\rho \otimes \sigma e^{iS\Delta t}) = \sigma - i\Delta t[\rho, \sigma] + O(\Delta t^2) \tag{2}$$

Where S stands for Swap operator, tr_P stands for partial trace for second register.
Proof for Equation 2:

We use the Taylor expansion, and get:

$$e^{-i\rho\Delta t}\sigma e^{i\rho\Delta t} = (I + \rho\Delta t + ...)\sigma(I - \rho\Delta t + ...) = \sigma - i\Delta t[\rho, \sigma] + O(\Delta t^2)$$

Proof for Equation 2:
Since $S^2 = I$, we can show easily that,

$$e^{-iS\Delta t} = cos\Delta t I - isin\Delta t S$$

$$e^{iS\Delta t} = cos\Delta t I + isin\Delta t S$$

Therefore,

$$e^{-iS\Delta t}\rho \otimes \sigma e^{iS\Delta t} = (cos^2\Delta t)\rho \otimes \sigma + (sin^2\Delta t)\sigma \otimes \rho - isin\Delta t cos\Delta t(S\rho \otimes \sigma - \sigma \otimes \rho S)$$

Note that

$$S |a\rangle |b\rangle = |b\rangle |a\rangle$$

$$\langle a| \langle b| S = \langle b| \langle a|$$

For the sake of simplification, we use Einstein notation:

$$S\rho \otimes \sigma = (\rho_{ij} |i\rangle \langle j|)(\sigma_{kl} |k\rangle \langle l|) = (\rho_{ij} |k\rangle \langle j|)(\sigma_{kl} |i\rangle \langle l|)$$

$$Tr_P(S\rho \otimes \sigma) = \rho_{ij}\sigma_{jl} |i\rangle \langle l| = \rho\sigma$$

Similarly,

$$Tr_P(\sigma \otimes \rho S) = \sigma\rho$$

Finally, we get the Equation 1 using the approximation $sin\Delta t \sim \Delta t$ and $cos\Delta t \sim 1$.

- **Complexity Advantage**:
(a)To retrieve ρ we use a QRAM with $O(logd)$ operations;
(b) Since the swap operator S is sparse it can be performed efficiently using Equation 2 to accomplish the operation of Equation 1. Repeated application of Equation 2 with n times allows one to construct $e^{-in\rho\Delta t}\sigma e^{in\rho\Delta t}$.
(c) To simulate $e^{-i\rho t}$ to accuracy ϵ, $n \sim O(t^2\epsilon^{-1})$ steps are needed. **Actually, the reason why we need different t is that we want to construct varied U^k in phase estimation Therefore, we use t as the regulation parameter.** Overall, the complexity is $O(nlogd)$ to construct $U = e^{-i\rho t}$.

- **Algorithm Process**: We input the mixed state ρ into the phase estimation structure using $U = e^{-i\rho t}$. For varying times t we apply this operator on ρ itself which results in the state

$$\sum_i r_i |\mathcal{X}_i\rangle \langle \mathcal{X}_i| \otimes |r_i\rangle \langle r_i| \tag{3}$$

where $|\mathcal{X}_i\rangle$ is the eigenstate and the $|r_i\rangle$ is eigenvalue state. Through quantum sampling, we can get the large eigenvalues.

3 Quantum support vector machine

3.1 Problem formulation

The SVM(Support Vector Machine) is a supervised machine learning tool, which is harnessed to classify data. Given a training dataset of $\mathcal{D} = \{(x^1, y^1), ..., (x^M, y^M)\}$ with $x_i \in \mathcal{R}^d$ and $y_i \in \{-1, 1\}$, the task for linear SVM is to find a hyperplane $w\vec{x} + \vec{b}$ to separate data.

3.2 Transformation of the problem

We neglect the math inference in classical part, and provide the equivalent problem below[4]:

$$\begin{aligned} argmax L(\vec{\alpha}) &= \sum_j y_j \alpha_j - 1/2 \sum_{j,k}^{M} \alpha_j K_{jk} \alpha_k \\ s.t. \sum_j \alpha_j &= 0, y_j \alpha_j > 0 \end{aligned} \tag{4}$$

where $K_{ij} = \vec{x_j} \cdot \vec{x_k}$. The formula can be transferred to the following:

$$\begin{pmatrix} 0 & \vec{1}^T \\ \vec{1} & \boldsymbol{K} + \gamma \mathbf{1} \end{pmatrix} \begin{pmatrix} b \\ \vec{\alpha} \end{pmatrix} = \begin{pmatrix} 0 \\ \vec{y} \end{pmatrix} \tag{5}$$

which becomes the problem of solving linear equation.

4 Quantum neural network classifier

4.1 Problem formulation

The task that the naive neural network model aims to solve is data classification, which belongs to the supervised learning. To formulate this problem, let $\mathcal{X}$ be a set of inputs and $\mathcal{Y}$ a set of outputs. Given a dataset of $\mathcal{D} = \{(x^1, y^1), ..., (x^M, y^M)\}$, we intends to predict $y \in \mathcal{Y}$ to match the y^i for any provided $x^i \in \mathcal{X}$. We assume that $\mathcal{X} = \mathbb{R}^N$ and $\mathcal{Y} = \{0, 1\}$ in the following passage.

4.2 The workflow frame

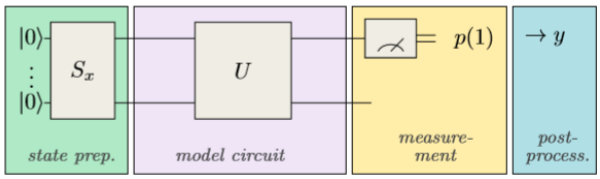

Figure 2: Four steps in quantum neural network classifier

The main idea of classifier design is to turn a generic quantum circuit consisting of single gates and 2-qubit gates into a model for classification.
(1) step1: Prepare the state $|\psi(x)\rangle$; we use amplitude encoding to map the x^i into n-qubit state $\psi(x)$;
(2) step2: Get the $|\psi^{'}(x)\rangle = U_{(\theta)}\,|\psi(x)\rangle$;
(3) step3: Measure the first qubit;
(4) step4: Repeat to get the estimated distribution of the probability[5];

The decision function is:

$$\begin{aligned} \pi(x;\theta,b) &= p(q_0 = 1, x, \theta) + b; \\ f(x;\theta,b) &= \begin{cases} 1 & \text{if } \pi(x;\theta,b) > 0.5 \\ 0 & \text{if } \pi(x;\theta,b) < 0.5 \end{cases} \end{aligned} \tag{6}$$

where $p(q_0 = 1, x, \theta) = \sum_{k=2^{n-1}-1}^{2^n} |(U(\theta)\phi(x))_k|^2$.

4.3 The gate circuit model

We formulate the single-qubit gate as $G(\alpha, \beta, \gamma)$:

$$G(\alpha,\beta,\gamma) = \begin{pmatrix} e^{i\beta}cos\alpha & e^{i\gamma}sin\alpha \\ -e^{-i\gamma}sin\alpha & e^{-i\beta}cos\alpha \end{pmatrix} \tag{7}$$

Since we only want to use single-qubit gate and two-qubit control gate, the common circuit based

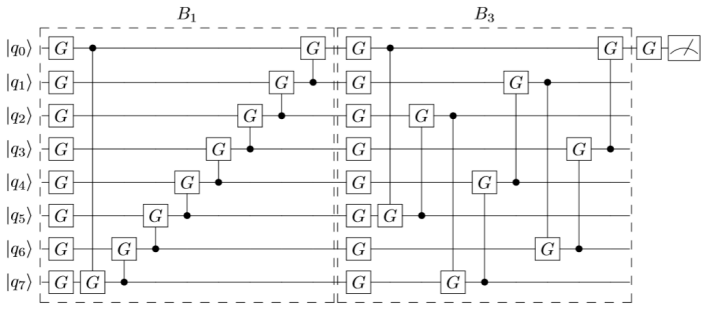

Figure 3: the title of figure

on the principle of cyclic code is in the Figure 3. 16 two-qubit control gates are utilized for constructing entanglement effect.

There are three kinds of gates in Figure 3:

- **Single qubit gate**:

$$U_l = \mathbb{I}_0 \otimes \mathbb{I}_1 \otimes \cdots \otimes G_k \otimes \cdots \otimes \mathbb{I}_{n-1}$$

- **Measurement qubit gate**:

$$U_{measure} = \sigma_z \otimes \mathbb{I}_1 \otimes \cdots \otimes \mathbb{I}_{n-1}$$

Note that we can reformulate the measurement :

$$p(q_0 = 1, x, \theta) = 1/2 + 1/2\mathbb{E}(\sigma_z) \tag{8}$$

where

$$\mathbb{E}(\sigma_z) = \langle\psi(x)| U^\dagger \sigma_z \otimes \mathbb{I}_1 \otimes \cdots \otimes \mathbb{I}_{n-1} U |\psi(x)\rangle \tag{9}$$

- **Controlled single-qubit control gate**
A common case of an imprimitive two-qubit gate is a singly-controlled single-qubit gate $C(G)$.

4.4 The gradient calculation scheme

We use Equation 8 to formulate the derivative.

$$\begin{aligned} \partial_\mu \pi(x;\mu) &= \partial_\mu p(q_0 = 1, x, \theta) \\ &= Re\{\langle \partial_\mu U(\theta)\psi(x)| \sigma_z \otimes \mathbb{I}_1 \otimes \cdots \otimes \mathbb{I}_{n-1} |U(\theta)\psi(x)\rangle\} \end{aligned} \tag{10}$$

assuming that $\partial_\mu U(\theta) = U_L \cdots \partial_\mu U_i \cdots U_1$, which means only U_i is related to μ.

For single qubit gate and the tow-qubit control gate, the gradient calculation is in the following:

- **Derivative for single qubit gate**
The derivatives for gate $G(\alpha, \beta, \gamma)$ in Equation 7 in respect to parameters α, β, γ is :

$$\begin{aligned} \partial_\alpha G &= G(\alpha + \pi/2, \beta, \gamma) \\ \partial_\beta G &= 1/2G(\alpha, \beta + \pi/2, 0) + 1/2G(\alpha, \beta + \pi/2, \pi) \\ \partial_\gamma G &= 1/2G(\alpha, 0, \gamma + \pi/2) + 1/2G(\alpha, \pi, \gamma + pi/2) \end{aligned} \tag{11}$$

One can see that the derivative calculation only ends in phase shift in parameters.

- **Derivative for controlled single qubit**
Differentiating a controlled single qubit gate is not that immediate, but fortunately we have

$$\partial_\mu G = 1/2(C(\partial_\mu G) - C(-\partial_\mu G)) \tag{12}$$

which means that the derivative of the controlled single qubit gate is half of the difference between a controlled derivative gate and the controlled negative version of that gate.

Therefore, we see the gradient is

$$\partial_\mu \pi(x;\mu) = \sum_j a_j Re\{\langle U(\theta_j')\psi(x)| \sigma_z |U(\theta)\psi(x)\rangle\} \tag{13}$$

where θ_j' is a modified vector of parameters corresponding to a term appearing in Equation 11, and a_j is the corresponding coefficient also stemming from the Equation 11.

4.5 Compute the gradient terms

Lemma 1. *Given two unitary quantum circuits A and B that act on a n qubit register to prepare the two quantum states* $|A\rangle$,$|B\rangle$*, we can get* $p(a=0) = 1/2 + 1/2Re\langle A|B\rangle$ *,where a is ancillary qubit register.*

Proof. We circuits A and B to construct $|A\rangle$ and $|B\rangle$ conditioned on the ancillary qubit,

$$\frac{1}{\sqrt{2}}(|0\rangle|A\rangle + |1\rangle|B\rangle)$$

Perform the H gate,

$$\frac{1}{2}(|0\rangle(|A\rangle + |B\rangle) + |1\rangle(|A\rangle + |B\rangle))$$

where $p(a=0) = 1/2 + 1/2Re\langle A|B\rangle$.

□

According to Lemma 1, we can get that

$$Re\langle A|B\rangle = 2p(a=0) - 1 \tag{14}$$

which shows that we can compute $Re\{\langle U(\theta_j')\psi(x)|\sigma_z|U(\theta)\psi(x)\rangle\}$ through a minor modification for the original circuit by adding an ancillary qubit to control the gate with phase shift in derivative process.

5 Quantum circuit learning

5.1 Problem formulation

However, the process proposed by the paper [5] is not that intuitive with the Equation 14. Another beautiful and more general model was also proposed around 2018, which has been treated as the foundation of quantum neural network since then.[6]

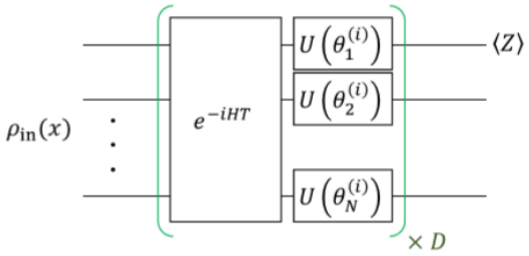

Figure 4: Quantum circuit learning structure: N denotes the number of qubits, while D denotes the repetition times of the $U(\vec{\theta})$ block.

The basic structure is like Figure 4. The $U(\vec{\theta})$ is needed to learn to minimize the loss function. Commonly, for task of classification, we use so-called one-hot encoding. For simplicity, we consider the case of two-category classification, using $y \in \{0, 1\}$ to label the categories and the number of the output qubit is one. We estimate the probability for each class by measuring the expectation values of the projection: $P(y = l) = Tr(\rho_{out} |l\rangle \langle l|)$, where $l = 0, 1$ and $\rho_{out} = Tr^{'}(|\Phi(\theta)\rangle\langle\Phi(\theta)|)$ with $Tr^{'}$ meaning that tracing out all other but the output qubit like the Z in Figure 4.

We use the following loss function based on cross-entropy:

$$L(h(|\phi(x)\rangle, \vec{a})) = -\sum_{l} a_l \log g_l \tag{15}$$

where $\vec{a}$ represents the one-hot truly label of the data, h refers to the abstract function of the circuit and g_l refers to the measurement $Tr(|l\rangle \langle l| \rho_{out})$.

5.2 Another general and intuitive gradient calculation scheme

Firstly we assume that the $U(\vec{\theta})$ consists of a chain of unitary transformations so $U(\vec{\theta}) = \prod_{j=1}^{k} U_j(\theta_j)$ with $U_j(\theta) = exp(-i\theta_j \sum_j /2)$ and $\sum_j^2 = I$. Then we have the quantum gradient

$$\frac{\partial \langle L\rangle}{\partial \theta_j} = 1/2 \left\{ \left\langle L(\vec{\theta}) \right\rangle_{\theta_j + \pi/2} - \left\langle L(\vec{\theta}) \right\rangle_{\theta_j - \pi/2} \right\} \tag{16}$$

Proof
Denote $U_{j:k} = U_j ... U_k$, and consider any observable M, then we have

$$\left\langle M(\vec{\theta}) \right\rangle = Tr(M U_{k:1} \rho_{in} U_{k:1}^{\dagger})$$

Using the chain rule,

$$\frac{\partial \langle M\rangle}{\partial \theta_j} = -\frac{i}{2} Tr(M U_{k:j} \left[\sum\nolimits_j, U_{j-1:1} \rho_{in} U_{j-1:1}^{\dagger}\right] U_{k:j}^{\dagger})$$

Harnessing the equation,

$$\left[\sum\nolimits_j, \rho\right] = i \left[U_j(\pi/2) \rho U_j^{\dagger}(\pi/2) - U_j(-\pi/2) \rho U_j^{\dagger}(-\pi/2)\right]$$

Then, we get

$$\frac{\partial \langle M\rangle}{\partial \theta_j} = \frac{1}{2} Tr(M U_{k:j+1} U_j(\pi/2) \rho_j U_j^{\dagger}(\pi/2) U_{k:j+1}^{\dagger}) - \frac{1}{2} Tr(M U_{k:j+1} U_j(\pi/2) \rho_j U_j^{\dagger}(-\pi/2) U_{k:j+1}^{\dagger})$$

where $\rho_j = U_{j:1} \rho_{in} U_{j:1}^{\dagger}$.
Finally, we obtain,

$$\frac{\partial \langle M\rangle}{\partial \theta_j} = 1/2 \left\{ \left\langle M(\vec{\theta}) \right\rangle_{\theta_j + \pi/2} - \left\langle M(\vec{\theta}) \right\rangle_{\theta_j - \pi/2} \right\}$$

It shows that just by inserting $\pi/2$ rotation generated by $\sum_j$ and measuring the respective expectation values,we can evaluate the exact gradient of M.

5.3 Quantum advantage

Suppose we want to learn the output of quantum circuit learning that is allowed to use unlimited resources in the learning process, via classical neural networks. Then it has to learn the relation between inputs and outputs of a quantum circuit, which, in general, includes universal quantum cellular automata . This certainly could not be achieved using a polynomial-sized classical computational resource to the number of qubits used for quantum circuit learning. This implies that quantum circuit learning has the potential power to represent more complex functions than the classical counterpart.[6]

5.4 Simulation outcome

The paper [6] demonstrates the performance of the Quantum circuit learning framework on several prototypical machine learning tasks by numerically simulating a quantum circuit in the form of Figure 4 with $N = 6$ and $D = 6$.

The simulation results are in the following two pictures.

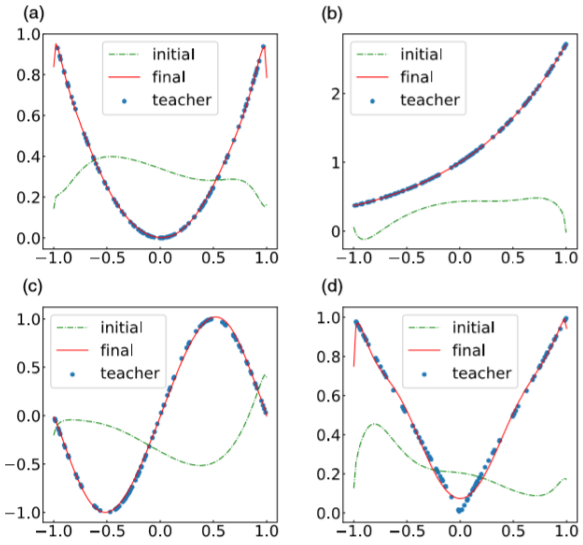

Figure 5: Demonstration of quantum circuit learning performance to represent functions. (a) x^2; (b) e^x; (c) $\sin x$; (d) $|x|$.

The numerical simulation code has been implemented in the repository `https://github.com/nkmjm/QCLL`, which is one of the best versions.

6 The thoughts of this curriculum paper

Thanks to Prof.Jing, I have the chance to get in touch with the basics of phase estimation and Grover algorithm and form a general concept for quantum computation algorithm. Since I am a

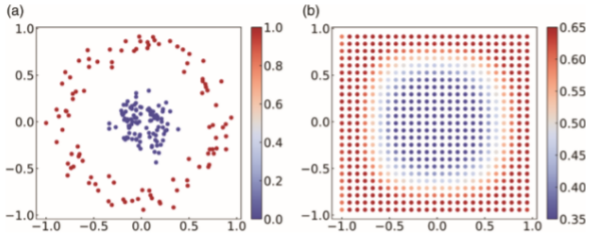

Figure 6: Demonstration of a simple non-linear classification task. (a) represents training data, and (b) shows the classification boundary.

student majoring in information science, quantum machine learning, combining machine learning and quantum computation, really intrigues me. After accomplish the reading of computation part of *Quantum Computation and Quantum Information*[7] and about six papers related to quantum machine learning, I try to write a survey as a record for this experience.

Actually, there are two things difficult in learning of quantum machine learning in my mind. Firstly, the process of creative algorithm construction really needs the familiarity of both machine learning and quantum computation like obtaining the quantum gradient through difference of the output expectation values in a modified quantum circuit. Another aspect is the analysis of error, such as the step counting with the requirement of accuracy ϵ in QPCA. As a beginner, I could almost figure out the process of the algorithm, but the analysis of error in quantum circuit is really challenging for me.

In sum, this is a tough area for study and for physical realization. But, it is truly interesting when you understand those amazing designs.

References

[1] A. W. Harrow, A. Hassidim, and S. Lloyd, "Quantum algorithm for linear systems of equations," *Physical review letters*, vol. 103, no. 15, p. 150502, 2009.

[2] S. Lloyd, M. Mohseni, and P. Rebentrost, "Quantum principal component analysis," *Nature Physics*, vol. 10, no. 9, pp. 631–633, 2014.

[3] M. Blencowe, "Quantum ram," *Nature*, vol. 468, no. 7320, pp. 44–45, 2010.

[4] P. Rebentrost, M. Mohseni, and S. Lloyd, "Quantum support vector machine for big data classification," *Physical review letters*, vol. 113, no. 13, p. 130503, 2014.

[5] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe, "Circuit-centric quantum classifiers," *Physical Review A*, vol. 101, no. 3, p. 032308, 2020.

[6] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, "Quantum circuit learning," *Physical Review A*, vol. 98, no. 3, p. 032309, 2018.

[7] M. A. Nielsen and I. Chuang, "Quantum computation and quantum information," 2002.