Supplementary material

Quantum Circuit Learning with Error Backpropagation Algorithm and Experimental Implementation

Masaya Watabe[1], Kodai Shiba[1,2], Chih-Chieh Chen[2], Masaru Sogabe[2], Katsuyoshi Sakamoto[1,3], Tomah Sogabe[1,2,3]*

[1] Engineering department, The University of Electro-Communications, Tokyo, Japan
[2] Grid, Inc. Tokyo, Japan
[3] i-PERC, The University of Electro-Communications, Tokyo, Japan

E-mail: sogabe@uec.ac.jp

S.A Dot product node

Backpropagation of dot product node.

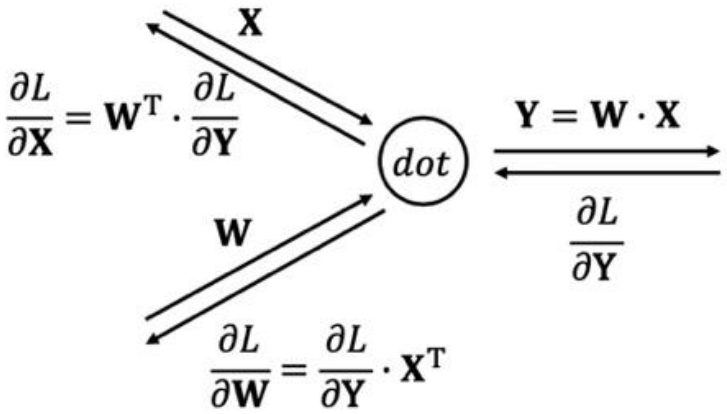

Figure 1 Graph of dot product node.

X is an input state. The size of **X** corresponds to N$(= 2^n)$ of states with n qubits. **W** is the network weights and corresponds to a gate in the quantum circuit. The size is N × N.

$$\mathbf{X} = (x_1, x_2, \dots, x_N)^{\mathrm{T}}$$

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & \cdots & w_{1,N} \\ \vdots & \ddots & \vdots \\ w_{N,1} & \cdots & w_{N,N} \end{bmatrix}$$

Output **Y** is written as $\mathbf{Y} = \mathbf{W} \cdot \mathbf{X}$. Here, when there is a gradient $\frac{\partial L}{\partial \mathbf{Y}}$ with respect to **Y** of the loss function L, the gradient of each L with respect to **X** and **W** is calculated as follows.

$$\frac{\partial L}{\partial \mathbf{X}} = \mathbf{W}^{\mathrm{T}} \cdot \frac{\partial L}{\partial \mathbf{Y}}$$

$$\frac{\partial L}{\partial \mathbf{W}} = \frac{\partial L}{\partial \mathbf{Y}} \cdot \mathbf{X}^{\mathrm{T}}$$

S.B Rotation gate $U(\theta)$ node

The weight **W** in the full-connected network is made to correspond to the rotation gate in the quantum circuit.

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} \\ w_{2,1} & w_{2,2} \end{bmatrix} = \begin{bmatrix} u_{1,1}(\theta) & u_{1,2}(\theta) \\ u_{2,1}(\theta) & u_{2,2}(\theta) \end{bmatrix} = \boldsymbol{U}(\theta)$$

Unlike conventional fully-connected networks, the elements of the unitary gate $\boldsymbol{U}(\theta)$ matrix are not independent of each other and have a common parameter θ. Therefore, the gradient of L with respect to θ is obtained by backpropagation using the calculation graph shown in Fig. 2.

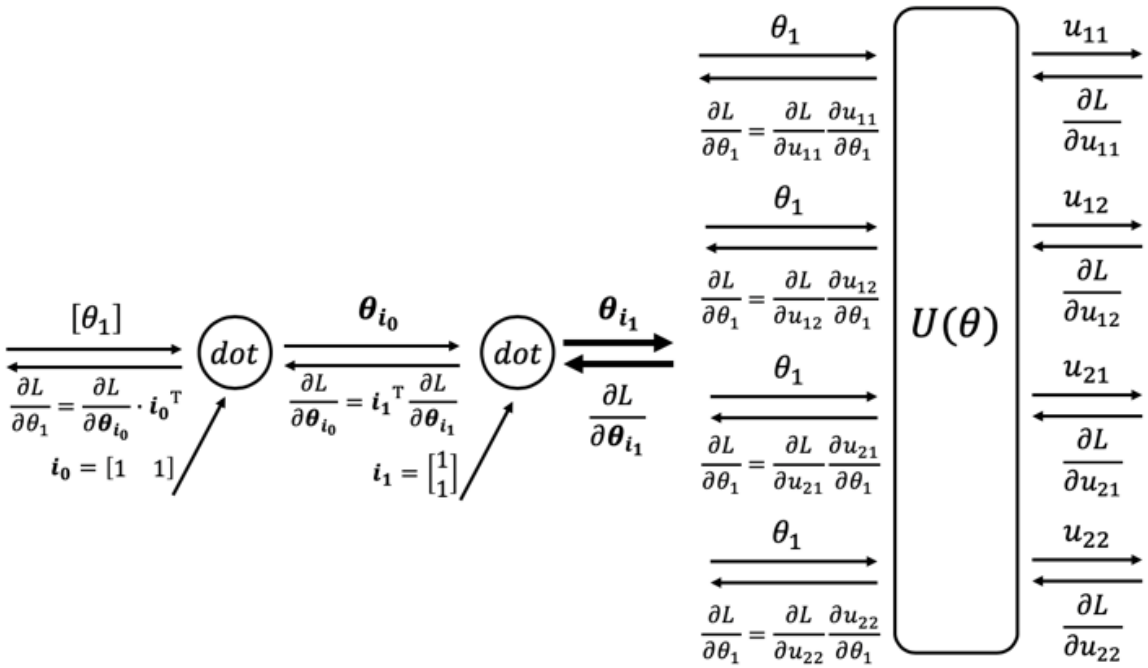

Figure 2 Graph of rotation gate $\boldsymbol{U}(\theta)$ node

For example, we chose R_Y gate,

$$\boldsymbol{U}(\theta) = \mathbf{Ry}(\theta) = \begin{bmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix},$$

$$\frac{\partial \mathbf{Ry}(\theta)}{\partial \theta} = \begin{bmatrix} \frac{\partial u_{1,1}}{\partial \theta} & \frac{\partial u_{1,2}}{\partial \theta} \\ \frac{\partial u_{2,1}}{\partial \theta} & \frac{\partial u_{2,2}}{\partial \theta} \end{bmatrix} = \begin{bmatrix} -\sin\frac{\theta}{2} & -\cos\frac{\theta}{2} \\ \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \end{bmatrix},$$

We define vector $\boldsymbol{i_0} = [1 \quad 1]$ and $\boldsymbol{i_1} = \boldsymbol{i_0}^{\mathrm{T}}$. The gradient is

$$\frac{\partial L}{\partial \boldsymbol{\theta}_{i_1}} = \begin{bmatrix} \frac{\partial L}{\partial \theta} & \frac{\partial L}{\partial \theta} \\ \frac{\partial L}{\partial \theta} & \frac{\partial L}{\partial \theta} \end{bmatrix} = \begin{bmatrix} \frac{\partial L}{\partial u_{1,1}}\frac{\partial u_{1,1}}{\partial \theta} & \frac{\partial L}{\partial u_{1,2}}\frac{\partial u_{1,2}}{\partial \theta} \\ \frac{\partial L}{\partial u_{2,1}}\frac{\partial u_{2,1}}{\partial \theta} & \frac{\partial L}{\partial u_{2,1}}\frac{\partial u_{2,1}}{\partial \theta} \end{bmatrix},$$

$$\frac{\partial L}{\partial \boldsymbol{\theta}_{i_0}} = \boldsymbol{i_1}^{\mathrm{T}} \cdot \frac{\partial L}{\partial \boldsymbol{\theta}_{i_1}},$$

$$\frac{\partial L}{\partial \theta} = \frac{\partial L}{\partial \boldsymbol{\theta}_{i_0}} \cdot \boldsymbol{i_0}^{\mathrm{T}}.$$

From above,

$$\frac{\partial L}{\partial \theta} = \frac{\partial L}{\partial \boldsymbol{\theta}_{i_0}} \cdot \boldsymbol{i_0}^{\mathrm{T}} = \left(\boldsymbol{i_1}^{\mathrm{T}} \cdot \frac{\partial L}{\partial \boldsymbol{\theta}_{i_1}}\right) \cdot \boldsymbol{i_0}^{\mathrm{T}}$$

$$\frac{\partial L}{\partial \theta} = \frac{\partial L}{\partial u_{1,1}}\frac{\partial u_{1,1}}{\partial \theta} + \frac{\partial L}{\partial u_{1,2}}\frac{\partial u_{1,2}}{\partial \theta} + \frac{\partial L}{\partial u_{2,1}}\frac{\partial u_{2,1}}{\partial \theta} + \frac{\partial L}{\partial u_{2,2}}\frac{\partial u_{2,2}}{\partial \theta}$$

S.C Observation Probability node

Output state $|\psi_{out}\rangle$ is

$$|\psi_{out}\rangle = c_0|0\rangle + c_1|1\rangle + \cdots + c_{N-1}|N-1\rangle,$$

where c_j is the probability amplitude and satisfies $|c_0|^2 + |c_1|^2 + \cdots + |c_{N-i}|^2 = 1$.

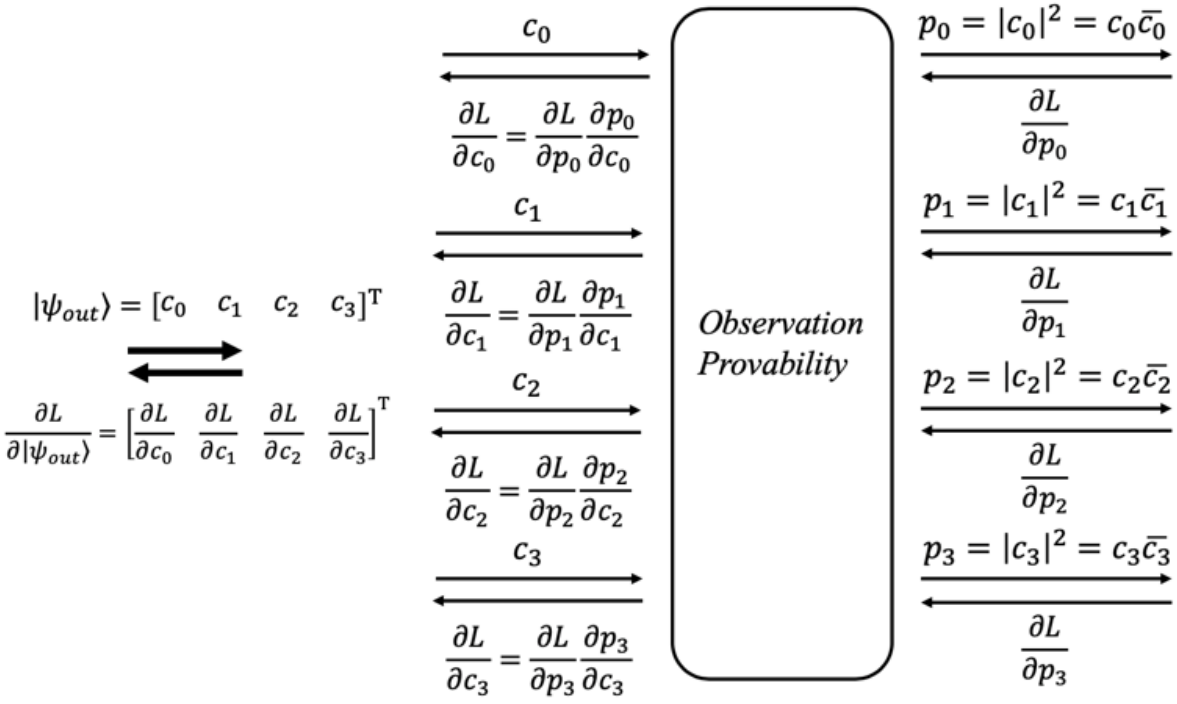

Figure 3 Graph of the probability amplitude node

The gradient of p_i with respect to a probability amplitude c_j can be calculated as follows. $\bar{c_i}$ is the complex conjugate of c_j.

$$\frac{\partial p_j}{\partial c_j} = \frac{\partial}{\partial c_j}\left(c_j \overline{c_j}\right) = \overline{c_j}$$

S.D Quantum circuit learning results with classification problems. The training data are added with noise.

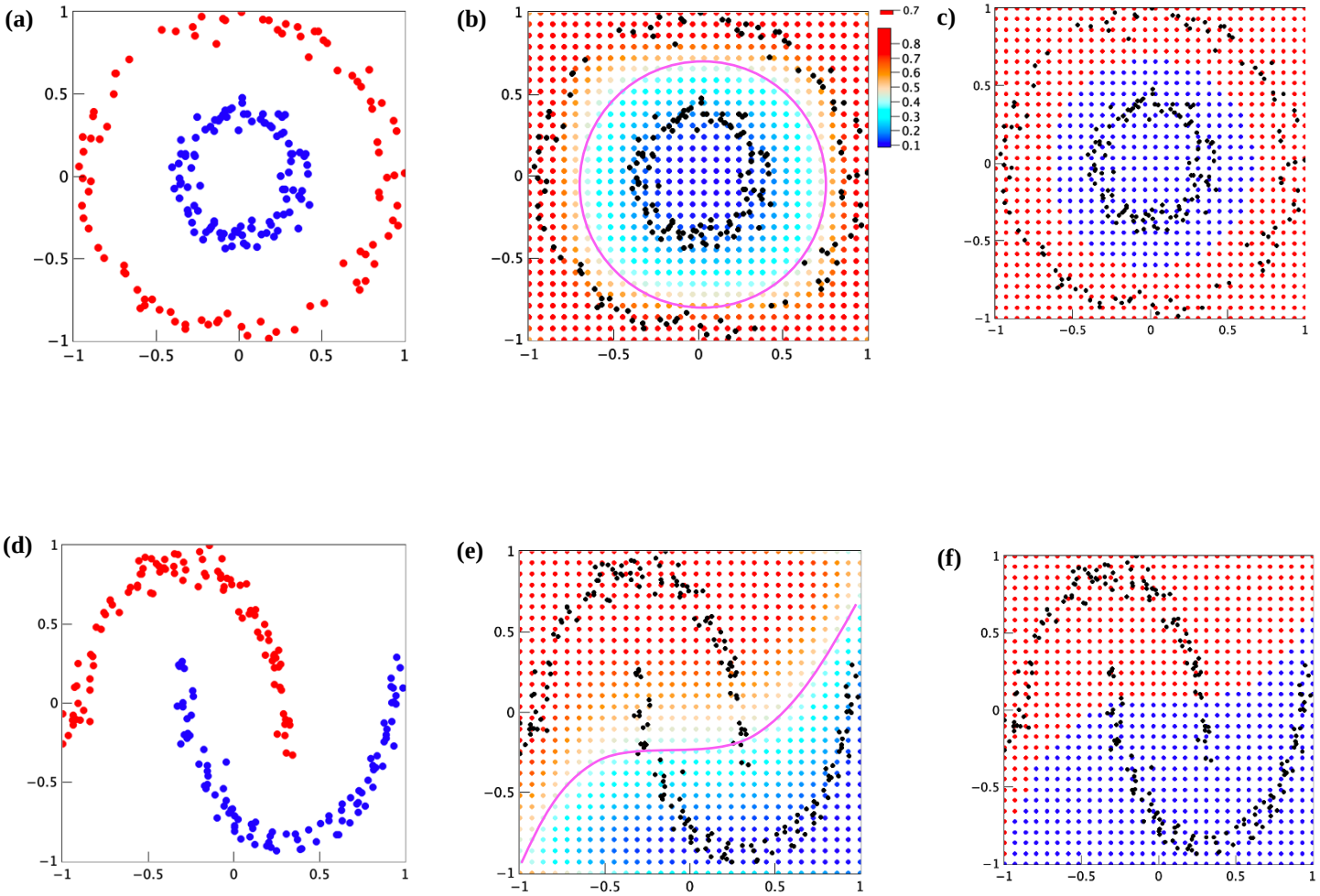

Figure 4 Quantum circuit learning results using error backpropagation for nonlinear binary classification problem with 4 qubit and 7 layer depth: (a) Training data set for make_circles, red for label '0' and blue for label '1' ; (b) Test results using the learnt parameter using the 200 make_circles dataset, pink line corresponding to the median boundary of the continuous probability; (c) scikit-learn-SVM classification results using the learnt support vectors; (d) Training data set for make_moons, red for label '0' and blue for label '1' ;; (e) Test results using the learnt parameter under the 200 make_moon dataset, pink line corresponding to the median boundary of the continuous probability; (f) scikit-learn-SVM classification results using the learnt support vectors.