Distributed Quantum Machine Learning: A Survey and Outlook

Wai Ming Chan[1]

[1]Department
of Electrical and Computer Engineering, University of Kansas, Lawrence, KS 66045, USA

March 18, 2024

Abstract

1 Introduction

The quantum computing in machine learning and optimization has attraction great attention, particularly in the variational quantum algorithms (VQAs) which are implemented by parameterized quantum circuits (PQCs). VQAs has shown efficient implementations on noisy NISQ machines, as well as theoretical guarantees for quantum superiority. Nevertheless, modern VQAs are still facing challenges in terms of expensive or unaffordable quantum resources, such as the number of qubits, the depth of quantum circuits, and the number of quantum measurements. For example, in classical machine learning, the training data is usually in a large volume, and the current quantum circuit might not be able to handle the large-scale data in a single quantum chip. For example of solving classification problems with n labels in quantum neural networks (QNNs), the convential optimizer requires to query the single quantum circuit at least $O(n)$ times. This overhead prohibits the scalability of QNNs in large-scale classification problems.

A natural way to accelerate the training of VQAs is to parallelize the quantum circuits to fulfill the joint optimization. However, designing such a parallel quantum circuit is non-trivial and it is sharply inconsistent with classical distributed optimization or quantum distributed computing. There are three reasons. First, the gradient operated in VQAs is biased, induced by the system imperfection such as sample error and gate noise, whereas most classical distributed methods assume the unbiased gradients. Second, unlike the classical distributed optimization, VQAs are immune to the communication bottleneck, due to the small number of trainable parameters. Third, quantum distributed computation focuses on using multiple less powerful quantum circuits to simulate a standard quantum circuit. Therefore, the focus of this project is to investigate the distributed quantum machine learning and optimization, and to provide a survey and outlook on this topic.

2 Problem Statement

We consider the distributed quantum machine learning consists of a central server and multiple local clients $\{\mathcal{Q}_i\}_{i=1}^{Q}$ where each local client consists of a quantum circuit $\mathcal{Q}_i$. The central server is responsible for the global optimization of the quantum circuits, and the local clients are responsible for the local optimization of the quantum circuits. The goal of the distributed quantum machine learning is to minimize the global cost function $\mathcal{L}(\theta,\mathcal{D})$, where θ is the global parameter and $\mathcal{D}$ is the global dataset. The dataset $\mathcal{D}=\{\mathbf{x}_i,y_i\}_{i=1}^{N}$ is distributed among the local clients, and the cost function $\mathcal{L}_i(\theta,\mathcal{D})$, for classification task, is defined as

$$\mathcal{L}(\theta,\mathcal{D})=\frac{1}{2N}\sum_{i=1}^{N}(y_i-\mathcal{Q}(\mathbf{x}_i;\theta))^2+\lambda\|\theta\|^2, \tag{2.1}$$

where $\mathcal{Q}(\mathbf{x}_i;\theta)$ is the quantum circuit, and $\lambda\|\theta\|^2$ is the regularization term.

For original central QNN training, we want to minimize the cost function $\mathcal{L}(\theta,\mathcal{D})$ by solving the optimization problem

$$\min_{\theta}\mathcal{L}(\theta,\mathcal{D}), \tag{2.2}$$

by using the gradient-based optimization algorithms, such as the stochastic gradient descent (SGD).

For distributed QNN training, we want to minimize the cost function $\mathcal{L}(\theta,\mathcal{D})$ by combining the local model updates or gradients from the local clients, i.e.,

$$\operatorname*{argmin}_{\theta}\mathcal{L}(\theta,\mathcal{D})=f\Big(\{\mathcal{L}_i(\theta_i,\mathcal{D}_i)\}_{i=1}^{Q}\Big), \tag{2.3}$$

where $f(\cdot):\theta_1\times\cdots\times\theta_Q\mapsto\theta$ is the aggregation function, such as the average or the weighted average and the local model updates or gradients are computed by the local clients, i.e.,

$$\theta_i^{(t)}=\theta_i^{(t-1)}-\eta\nabla\mathcal{L}_i(\theta_i^{(t-1)},\mathcal{D}_i), \tag{2.4}$$

where η is the learning rate, and $\nabla\mathcal{L}_i(\theta_i^{(t-1)},\mathcal{D}_i)$ is the gradient of the local cost function $\mathcal{L}_i(\theta_i^{(t-1)},\mathcal{D}_i)$.

The distributed quantum machine learning problem is to design the aggregation function $f(\cdot)$ and the local model updates or gradients $\nabla\mathcal{L}_i(\theta_i^{(t-1)},\mathcal{D}_i)$, such that the global cost function $\mathcal{L}(\theta,\mathcal{D})$ is minimized.

3 Basic Review and References

We start with a distributed method of VQAs called proposed **Qu**antum **Di**stributed **O**ptimization (QUDIO) algorithm [1] which designs a distributed VQA to improve the runtime efficiency by distributing the optimization process across multiple local nodes. This work provides both theoretical proofs of how the distributed VQA accelerates from the centralized VQA, and also its implementation details. We will first review the algorithm and the implementation, and then discuss the theoretical proofs.

3.1 Theoretical Proofs of QUDIO

...

4 Algorithms, Circuits, Diagrams

Algorithm 1: The Pseudocode of QUDIO.

```
    Input: Initial parameters $\theta^{(0)} \in [0,2\pi)^{d_Q}]$, employed loss function $\mathcal{L}$, given dataset $\mathcal{D}$, and the
           hyper-parameters $\{Q,\nu,W,T\}$
 1  The central server partitions the given problem into $Q$ parts and allocates them to $Q$ local
      clients;
 2  for $t=0,\cdots,T-1$ do
 3      for Quantum processor $\mathcal{Q}_i$, $\forall i \in [Q]$ do
 4          $\theta_i^{(t,0)} = \theta^{(t)}$;
 5          for $w=0,\cdots,W-1$ do
 6              Compute the estimate gradient $g_i^{(t,w)}$;
 7              Update $\theta_i^{(t,w+1)} = \theta_i^{(t,w)} - \nu g_i^{(t,w)}$;
 8          end
 9      end
10  end
11  Synchronize $\theta^{(t+1)} = \frac{1}{Q}\sum_{i=1}^{Q} \theta_i^{(t,W)}$;
    Output: The optimal parameters $\theta^{(T)}$
```

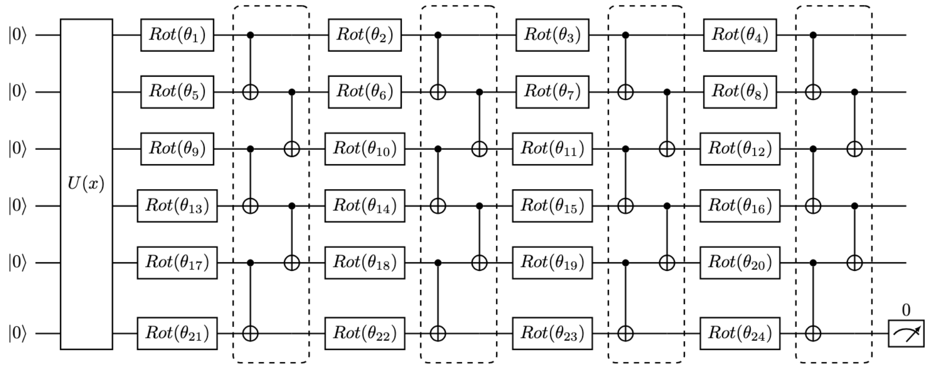

Figure 1: Circuit of the local node in QUDIO method [1]. The amplitude embedding method is represented by the block $U(x)$, and the quantum observable is set as $O = I_{2^5} \otimes |0\rangle\langle 0|$.

5 Simulation Results and Simulation Code

The source code of QUDIO can be found in the GitHub repository https://github.com/QQQYang/QUDIO. Because the original code seems contains some bugs and could not be executed in our local machine, we modify the code and keep the modified version into our project GitHub repository https://github.com/wai-ming-chan/project_distributedQNN.

Since we are not able to run the original code successfully, we cannot provide the simulation results in this midterm report. We will keep working on the code and provide the simulation results in the final report. Our plan is to create the circuits shown in Fig. 1 in pennylane (while the original code is written in their own framework), and then run the QUDIO algorithm as shown in Algorithm 1. We will also compare the results with the centralized VQA to show the efficiency of the distributed VQA.

6 Summary Outlook

In this midterm report, we have reviewed the distributed quantum machine learning and optimization, and provided a brief problem statement of this topic. We have also reviewed the QUDIO algorithm. We first reviewed their algorithm and implementation. We will next implement the QUDIO algorithm in pennylane/qiskit and compare the results with the centralized VQA. We will also study the theoretical proofs of the QUDIO algorithm and provide a summary in the final report.

References

[1] Y. Du, Y. Qian, and D. Tao, "Accelerating variational quantum algorithms with multiple quantum processors," *arXiv preprint arXiv:2106.12819*, 2021.