Computational Physics Projects - 2024

MnP Club, IIT Bombay

Quantum Machine Learning

Priyansh Singh

22b1856@iitb.ac.in

Contents

1 Introduction

– to be updated –

2 Week 1 - Introduction to Qiskit

2.1 The Circuit - Quantum Teleportation

– to be updated –

2.2 Implementation

– to be updated –

2.3 Results

– to be updated –

3 Week 2 - Introduction to VQAs

3.1 The Circuit - Variational Quantum Eigensolver

– to be updated –

3.2 Implementation

– to be updated –

3.3 Results

– to be updated –

4 Week 3 - Solving Differential equations

Here, using objects called "Differential quantum circuits", we explore how quantum circuits can represent derivatives and hence try to solve non-linear differential equations.
The differential equation is denoted as $F[d_x f(x), f(x), x] = 0$, with the boundary condition being $f(x_0) = u_0$.
The following article refers to the research paper titled "Solving nonlinear differential equations with differentiable quantum circuits" by Oleksandr Kyriienko, Annie E. Paine and Vincent E. Elfving.

4.1 A brief introduction to the methodologies used

A "Quantum Differential Equation Solver", as we may call it, would need to represent a "trial solution function" on a quantum circuit, along with a method to compute the required derivatives as per the DE.
The key aspect of the research involves a method called a "Quantum Feature Map" aided by the parameter-shift rule to represent functions and their derivatives on quantum hardware respectively.
The real function evaluation itself can be obtained as the expectation of a predefined Hermitian operator.

4.2 The avenues to achieve the proposed solution

4.2.1 Quantum Feature Maps

These are unitary circuits parameterized by a variable "x" and a non-linear function "$\varphi(x)$", represented as $\hat{U}_\varphi(x)$. These, when applied to the input state $|\emptyset\rangle$ map x to $\hat{U}_\varphi(x)|\emptyset\rangle$. This is called the "latent space mapping".
The quantum feature map circuit serves the purpose of closely representing highly non-linear functions and their derivatives. Listed below is one of the possible frameworks for the feature map circuit using sequential Qubit rotations, the "Product Feature Map", along with a specific class of it called the Chebyshev feature map.

- **Product Feature Map**: This methodology involves applying a non-linear transformation to x followed by a set of Qubit rotations. It is also allowed to have the circuit made up of multi-layered rotations.
  For an N Qubit system, the mathematical representations can be given as:
  Single layered approach:

$$\hat{U}_\varphi(x) = \bigotimes_{j=1}^{N} \hat{R}_{\alpha,j}(\varphi[x]), \tag{1}$$

  Multi-Layered approach:

$$\hat{U}_{\varphi}(x) = \prod_{\ell=1}^{L} \bigotimes_{j \in \mathbb{N}_\ell} \hat{R}^{(\ell)}_{\alpha,j}(\varphi_\ell[x]). \tag{2}$$

Where $\hat{R}_{\alpha,j}$ is the Pauli rotation operator acting on Qubit j, involving complex exponentiation of one of the three Pauli matrices, chosen by $\alpha \, \epsilon \, \{x, y, z\}$.

- **Derivative of the Product Feature Map**: Working out the derivative of $\hat{U}_{\varphi}(x)$:

$$\frac{d}{dx}\hat{U}_{\varphi(x)} = \frac{1}{2}\left(\frac{d}{dx}\varphi(x)\right)\left(\sum_{j'=1}^{N}\bigotimes_{j=1}^{N}\hat{R}_{y,j}(\varphi[x] + \frac{\pi}{2}\delta_{j,j'})\right) \tag{3}$$

The parameter-shift rule form becomes visible when the expectation with respect to an operator $\hat{C}$ is written out:

$$\frac{d}{dx}\langle\emptyset|\hat{U}_{\varphi}(x)^{\dagger}\hat{C}\hat{U}_{\varphi}(x)|\emptyset\rangle = \frac{1}{4}\left(\frac{d}{dx}\varphi(x)\right)\left(\langle\hat{C}\rangle^{+} - \langle\hat{C}\rangle^{-}\right) \tag{4}$$

where:

$$\langle\hat{C}\rangle^{\pm} = \sum_{j'=1}^{N}\bigotimes_{j=1}^{N}\langle\varnothing|\hat{R}^{\dagger}_{y,j}(\varphi[x] \pm \frac{\pi}{2}\delta_{j,j'})\hat{C}\hat{R}_{y,j}(\varphi[x] \pm \frac{\pi}{2}\delta_{j,j'})|\varnothing\rangle \tag{5}$$

- **Chebyshev Feature Map**: This is a sub-class of the product feature map, obtained by choosing $\varphi(x)$ as $2n\cos^{-1}x$ for some integer n.
The reason for this choice becomes evident as the true form of the rotation operator is revealed as being made up of two Chebyshev functions by the Euler's identity.

  A short note on Chebyshev functions and their utility here: Chebyshev functions are simple polynomial functions aided by trigonometric relations:

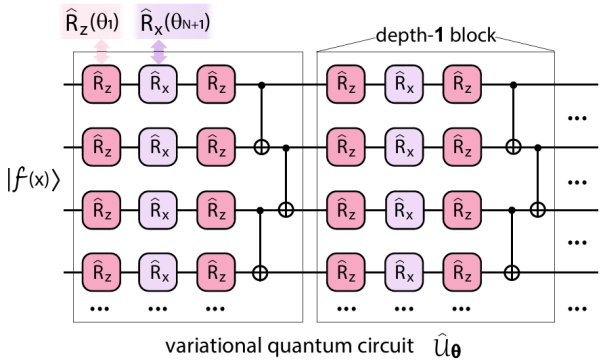
$$T_n(cos(\theta)) = cos(n\theta)$$
$$U_n(cos(\theta))sin(\theta) = sin((n+1)\theta)$$

  These are powerful in that when used as basis, two functions can be multiplied (called "**chaining**") to give rise to another two functions orthogonal to the initial two functions, and hence can provide a rich representation of functions.
  The chaining property enables to grow the basis set through state concatenation.

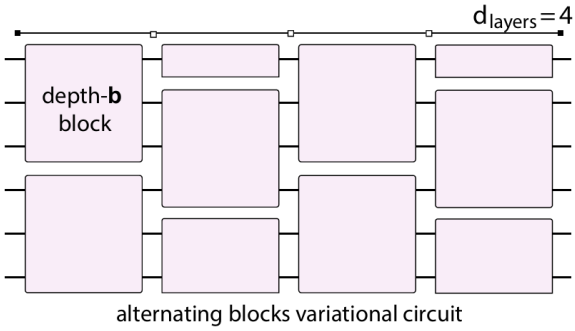
4.2.2 Variational Quantum Circuit - Ansatzes

The ansatz manipulates the latent space basis function and brings the derivatives and the function to the required form.
A difference from the approach used in the Variational Quantum Eigensolver is that here the gradient is computed by the quantum circuit itself and fed to a classical optimizer such as GD, Adam, SGD, etc. whereas the VQE approach just uses the quantum circuit as an oracle to compute the ansatz state followed by Hamiltonian expectation computation.

- **Hardware efficient ansatz**: Rotations followed by CNOT gates for entanglement.



- **Alternating blocks ansatz**:



4.2.3 Cost Operator

- **A predefined hermitian operator**: A hermitian operator $\hat{C}$ is used for which the expectation value, parameterized by a non-linear dependence on x and the variational angles can be used to evaluate the real value of the function.
  The function can be evaluated as:

$$f(x) = \langle f_{\varphi,\theta}(x)|\hat{C}|f_{\varphi,\theta}(x)\rangle \tag{6}$$

4.2.4 Loss Function

- **Quantifying "distance"**: The purpose of any loss function in optimization problems is to measure how far the solution is to the desired value in some vector space relevant to the problem case. Here, a "distance" metric is needed to evaluate how far the $F[d_x f(x), f(x), x]$ form is from 0, the desired value.
  A standard choice for the distance measure is the MSE, while MAE and KL Divergence are also beneficial for specific problems.

4.2.5 Boundary Handling

Unlike an analytical pen and paper approach to solving differential equations where we first obtain a general form of the solution and then adjust parameters of the general form to fit the boundary conditions, here the DQC takes a numerical approach, and hence the need of including the boundary condition somewhere in the loss arises.

- **Pinned handling**: Adds a boundary term to the overall loss function:

$$\mathcal{L}_\theta[d_x f, f, x] = \mathcal{L}_\theta^{(\text{diff})}[d_x f, f, x] + \mathcal{L}_\theta^{(\text{boundary})}[f, x] \tag{7}$$

- **Floating handling**: Boundary term is added in the function evaluation:

$$f(x) = f_b + \langle f_{\varphi,\theta}(x)|\hat{C}|f_{\varphi,\theta}(x)\rangle \tag{8}$$

  f_b is updated at each iteration:

$$f_b \leftarrow u_0 - \langle f_{\varphi,\theta}(x_0)|\hat{C}|f_{\varphi,\theta}(x_0)\rangle \tag{9}$$

- **Optimized handling**: This is the standard "offload tasks to the optimizer" approach, where we just expect the optimizer to come up with a "good" solution, here the function is computed from the mapping as:

$$f(x) = f_c + \langle f_{\varphi,\theta}(x)|\hat{C}|f_{\varphi,\theta}(x)\rangle, \tag{10}$$

  The f_c term adjusts the function to satisfy the boundary, and itself is a parameter for the optimizer to adjust.

4.3 Algorithm Workflow

With the components in hand, the workflow can now be described.

- The first step involves choosing the feature map circuit, the non linear function $\varphi(x)$, the VQC, the cost operator, the loss function, classical optimizer and regularization, if any.
- The parameters for the optimizer to update are the ansatz parameters θ, whereas the x variable used in the feature map circuit is the variable for function evaluation.
- A grid of points $\mathbb{X}$ to be used to evaluate the solution of the method at each iteration, are initialized.
- Also, the variational parameters θ are randomly initialized.
- The Derivative Quantum Circuits are constructed based on the differentials involved in the DE.
- The loss function is computed for the entire grid of points.
- The gradient of the loss function with respect to θ is computed for the optimizer.
- The algorithm is a quantum-classical loop, where a classical optimizer algorithm, say, Adam, calls the trial function and its derivatives as quantum oracles to compute the parameter update term from the loss gradient.
- When the loss converges, the algorithm returns $\theta_{optimal}$ for which:

$$f(x) = \langle \emptyset|\hat{U}_\varphi(x)\hat{U}_\theta|\hat{C}|\hat{U}_\theta\hat{U}_\varphi(x)|\emptyset\rangle$$

4.4 Implementation

– to be updated –

4.5 Results

– to be updated –

References

- Quantum Computation and Quantum Information by Michael A. Nielsen and Isaac L. Chuang
- Variational Quantum Algorithms: `https://arxiv.org/abs/2012.09265`
- Solving nonlinear differential equations with differentiable quantum circuits: `https://arxiv.org/abs/2011.10395`