A Variational Algorithm for Quantum Neural Networks

Antonio Macaluso, Luca Clissa, Stefano Lodi, Claudio Sartori

Raphael Blankson

November 24, 2020

Novosibirsk State University

Table of contents

Intro

Intro

1. Quantum Variational algorithms are hybrid computations designed to tackle optimization problems.
2. This paper introduces a novel algorithm for quantum Single Layer Perceptron(qSLP).
3. The work shows the design of quantum circuit to perform linear combinations in superposition.
4. The proposed algorithm was tested on synthetic data using both simulators and real quantum device.

Definitions

- Quantum Supremacy
- Superposition
- Entanglement
- Quantum Forking
- Quantum Register
- Quantum Oracle

Quantum Variational Algorithm

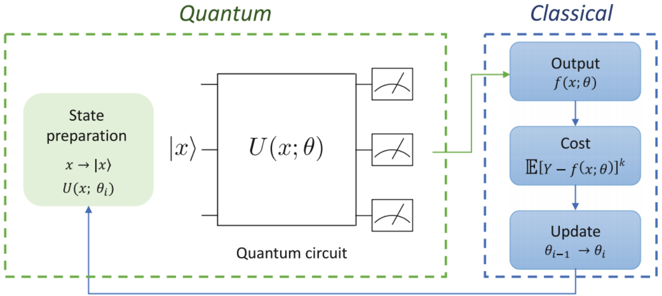

Fig. 1. Scheme of a hybrid quantum-classical algorithm for supervised learning. The quantum variational circuit is depicted in green, while the classical component is represented in blue. (Color figure online)

Neural Network as Universal Approximator

$$f(x_i) = \sigma_{\text{out}}\left(\sum_{j=1}^{H} \beta_j \sigma_{\text{hid}}\left(L(x_i;\theta_j)\right)\right).$$

Variational Algorithm For Single Layer Neural Network

Encode Data in Amplitude Encoding

$$|x\rangle = \sum_{k=1}^{2^n} x_k \, |k\rangle \longleftrightarrow x = \begin{pmatrix} x_1 \\ \vdots \\ x_{2^n} \end{pmatrix}.$$

Activation Function

- Restrictions of linear and unitary operations imposed by laws of quantum physics makes it difficult to implement a proper activation function.
- The most promising attempt uses a repeat-until-success approach to achieve non-linearity
- In this work, implementation of non-linear activation function was not considered

Gates as Linear Operators

- A variational circuit is composed of series of gates parameterized with $\theta_{l=1,\ldots,L}$ and is given by the product of matrices

$$U(\theta) = U_L \cdots U_l \cdots U_1,$$

- Gates can be expressed in terms of single-qubit gates, G_i as

$$U_l = \mathbb{1}_1 \otimes \cdots \otimes G_i \otimes \cdots \otimes \mathbb{1}_n,$$

- G_i is complex valued thus describes a more general operation than the classical SLP which only describes real operation.
- Parameterizing this gate with Pauli-Y rotation restricts the computation to the real domain.

Quantum Single Hidden Layer Network with Two Neurons

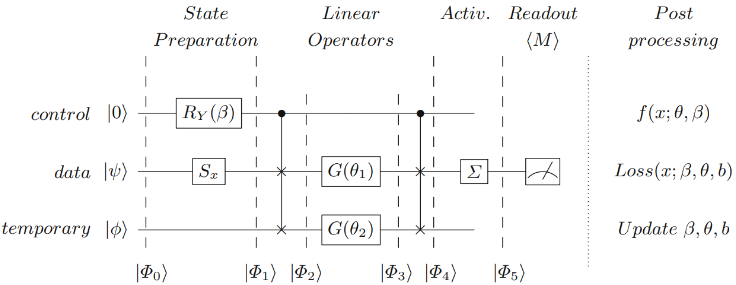

Fig. 2. Quantum circuit for training a qSLP.

5 steps of Alogrithm

- (Step 1)
  - State Preparation includes encoding data *x*, in the amplitude of $|\psi\rangle$
  - apply parameterized Y-rotation $R_y(\beta)$ to the control qubit

$$
\begin{aligned}
|\Phi_1\rangle &= \big(R_y(\beta) \otimes S_x \otimes \mathbb{1}\big) |\Phi_0\rangle = \big(R_y(\beta) \otimes S_x \otimes \mathbb{1}\big) |0\rangle |0\rangle |\phi\rangle \\
&= (\beta_1 |0\rangle + \beta_2 |1\rangle) \otimes |x\rangle \otimes |\phi\rangle = \beta_1 |0\rangle |x\rangle |\phi\rangle + \beta_2 |1\rangle |x\rangle |\phi\rangle),
\end{aligned} \tag{7}
$$

where S_x indicates the routine that encodes the data, $|\beta_1|^2 + |\beta_2|^2 = 1$ and $\beta_1, \beta_2 \in \mathbb{R}$.

step 2 - quantum Forking

- Apply first controlled-swap gate to swap $|x\rangle$ with $|phi\rangle$ if control qubit is equal to $|1\rangle$

$$|\Phi_2\rangle = \frac{1}{\sqrt{E}}\Big(\beta_1 |0\rangle |x\rangle |\phi\rangle + \beta_2 |1\rangle |\phi\rangle |x\rangle \Big)$$

where E is a normalisation constant.

- 2 linear operations parameterized by (θ_1, θ_2) act on $|\psi\rangle$ and $|\phi\rangle$

$$\begin{aligned}
|\Phi_3\rangle &= \Big(\mathbb{1} \otimes G(\theta_1) \otimes G(\theta_2)\Big) |\Phi_2\rangle \\
&= \frac{1}{\sqrt{E}}\Big(\beta_1 |0\rangle G(\theta_1) |x\rangle |\phi\rangle + \beta_2 |1\rangle |\phi\rangle G(\theta_2) |x\rangle \Big) \\
&= \frac{1}{\sqrt{E}}\Big(\beta_1 |0\rangle |L(x;\theta_1)\rangle |\phi\rangle + \beta_2 |1\rangle |\phi\rangle |L(x;\theta_2)\rangle \Big).
\end{aligned}$$

Step 2 - quantum forking

- Apply the second controlled-swap gate to swap $|L(x;\theta_2)\rangle$ with $|\phi\rangle$ if control qubit is equal to $|1\rangle$

$$|\Phi_4\rangle = \frac{1}{\sqrt{E}}\Big(\beta_1 |0\rangle |L(x;\theta_1)\rangle |\phi\rangle + \beta_2 |1\rangle |L(x;\theta_2)\rangle |\phi\rangle\Big).$$

- The two linear operations are stored in $|\psi\rangle$ and then entangled with one state of control qubit.

Step 3 - Activation Function

$$
\begin{aligned}
|\Phi_5\rangle &= \Big(\mathbb{1} \otimes \Sigma \otimes \mathbb{1}\Big) |\Phi_4\rangle \\
&= \frac{1}{\sqrt{E}}\Big(\beta_1 |0\rangle\, \Sigma\, |L(x;\theta_1)\rangle\, |\phi\rangle + \beta_2 |1\rangle\, \Sigma\, |L(x;\theta_2)\rangle\, |\phi\rangle\Big) \\
&= \frac{1}{\sqrt{E}}\Big(\beta_1 |0\rangle \Big|\sigma_{hid}\big[L(x;\theta_1)\big]\Big\rangle |\phi\rangle + \beta_2 |1\rangle \Big|\sigma_{hid}\big[L(x;\theta_2)\big]\Big\rangle |\phi\rangle\Big).
\end{aligned}
$$

- The two linear operations $L(.)$ are put through the same activation function σ_{hid} represented by the gate $\sum$
- The results are then encoded in the quantum register $|\psi\rangle$

Step 4 - Measurement

- The measurement of $|\psi\rangle$ can be expressed as the expected value of the Pauli-Z operator acting on the quantum state $|x\rangle$

$$\langle M\rangle = \langle\Phi_0| U^\dagger(\beta,\theta)(\mathbb{1}\otimes\sigma_z\otimes\mathbb{1})U(\beta,\theta)|\Phi_0\rangle = \pi(x;\beta,\theta),$$

- $U(\beta,\theta)$ represents the qSLP circuit.
- The entire circuit must be run multiple times to get an estimate for $\pi(.)$

Step 5 - Post Processing

The post processing is done classically and is task-dependent. For classification models, we need four steps

- adding a learnable bias term *b* to produce a continuous output
- applying a threshold operation

$$f(x_i; \beta, \theta, b) = \begin{cases} 1 & \text{if } \pi(x_i; \beta, \theta) + b > 0.5 \\ 0 & \text{else} \end{cases},$$

- computing the loss

$$SSE = Loss(\Theta; D) = \sum_{i=1}^{N} \left[y_i - f\left(x_i; \Theta\right)\right]^2,$$

- updating the parameters - **Nesterov accelerated optimizer**

Experiments

Experiments

- The circuit was implemented using Pennylane
- In addition, test was also done on QASM simulator and a real device
- Non-activation function was not used thus the model is just a linear classifier
- Generated linearly separable data for classification
  - 500 observations (250 per class) from 2 independent bivariate Gaussian distribution
  - 75% of data for training and 25% for testing

Results

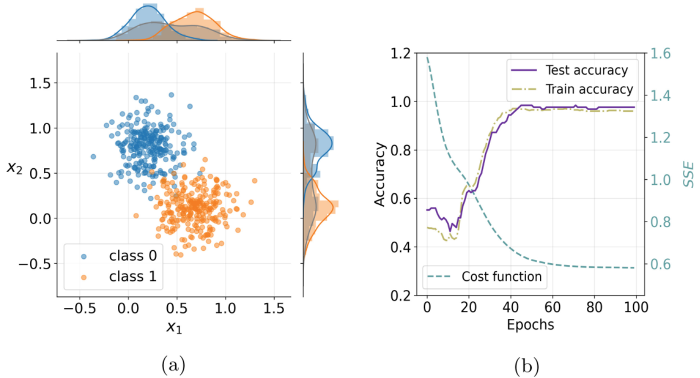

Fig. 3. The plot on the left illustrates the distributions of generated data in the two classes $(0, 1)$. The plot on the right shows the trends over training epochs of the cost function and the accuracy.

Results

Table 1: Test accuracy of multiple implementation

Pennylane	QASM	IBM (Vigo)	My Pennylane
94%	90%	64%	96%

- Experiments showed proposed architecture works well for linearly separable data as performance decreased largely when the problem level of complexity cannot be solved by a linear classifier

Generalization to H Hidden Neurons

Steps

- Turn control qubit into a non-uniform superposition parameterized by 2^d - D vector β by an oracle **B**

$$|\Phi_1\rangle = (\mathbb{1} \otimes B \otimes \mathbb{1})\,|x\rangle_{\text{data}}\,|0\rangle_{\text{control}}\,|0\rangle_{\text{output}} \\ \rightarrow \frac{1}{\sqrt{E}}\Big(\,|x\rangle \otimes \sum_j \beta_j\,|j\rangle \otimes |0\rangle\,\Big).$$

- Generate superposition of linear operation with parameters entangled with control register by the oracle which performs

$$|\Phi_2\rangle = \Lambda\,|\Phi_1\rangle \rightarrow \frac{1}{\sqrt{E}}\Big(\,|x\rangle \sum_j \beta_j\,|j\rangle\,|L(x;\theta_j)\rangle\,\Big).$$

- Apply the activation function $\sum$ to the third register

$$|\Phi_3\rangle = (\mathbb{1} \otimes \mathbb{1} \otimes \Sigma)\,|\Phi_2\rangle \rightarrow \frac{1}{\sqrt{E}}\Big(\,|x\rangle \sum_j \beta_j\,|j\rangle\,|\sigma[L(x;\theta_j)]\rangle\,\Big).$$

Steps

- As a result, the algorithm can be accessed by a single-qubit measurement
- The advantage is the number of hidden neurons H scales exponentially with the no. of states of the control register, 2^d

Conclusion

Conclusion

- Authors proposed implementation of a quantum version of SLP.
- The main idea is to use single state preparation routine and apply linear combination of superposition, each entangled with the control qubit
- Model trained with this algorithm can potentially approximate any desired function as long as enough hidden neurons and a non-linear activation are available
- The general case would be very beneficial for more hands-on experimentation
- We are still far from proving that Machine learning can benefit from Quantum Computing in practice

Thanks

Questions?