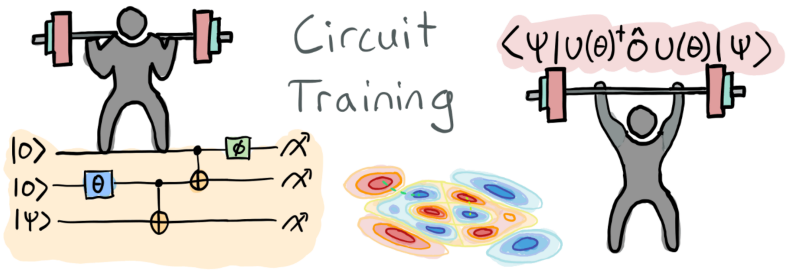

Circuit Training: Variational Quantum Classifier [500 points]

Version: 1

NOTE: Coding templates are provided for all challenge problems at this link. You are strongly encouraged to base your submission off the provided templates.

Overview: Circuit Training challenges

In this set of challenges, you'll explore methods and applications for training variational quantum circuits. Variational circuits play a role in quantum machine learning akin to that of neural networks in classical machine learning. They typically have a layered structure, and a set of tunable parameters that are learned through training with a classical optimization algorithm. Data is input to the circuit by embedding it into the state of a quantum device, and measurement results inform the classical optimizer how to adjust the parameters.

The circuit structure, optimization method, and how data is encoded in the circuit varies heavily across algorithms, and there are often problem-specific considerations that affect how they are trained. In this set of challenges, you'll explore how to construct and optimize a diverse set of circuits from the ground up, and become acquainted with some of today's popular quantum machine learning and optimization algorithms.

Problem statement [500 points]

Classification is a common classical machine learning task, and is also something that can be achieved with a quantum computer using variational circuits. In this problem, we've provided you with a set of data points that can be classified into three different categories (arbitrarily labeled `-1`, `0`, and `1`). Your job is to

design a *variational quantum classifier* that can classify unknown test data from the same distribution with an accuracy of more than 95%.

The data is the only thing we will provide for this problem—the number of qubits, circuit structure, and optimization methods are left to you.

Input

The file 1.in consists of 3 parts:

1. A set of training data points with dimensions `(250, 3)`
2. The categorical labels for the training points with dimensions `(250, )`
3. A set of testing data points with dimensions `(50, 3)`

The data has all been concatenated into a single string. We have provided helper functions that will parse the data into NumPy arrays.

The file 1.ans contains the categories of the testing points. To judge your solution, we will use the same training data points as you used to train your model, but we will use a different set of testing data points.

Output

The output of your program should be a string like the one in the file 1.ans, containing the predicted labels for the testing data separated by commas. The number of predicted labels should match the number of testing data points in the corresponding input file. The string will be compared to the string of true data labels for that testing set, and an accuracy will be computed by the grader.

Acceptance Criteria

In order for your submission to be judged as "correct":

- The outputs generated by your solution when run with a given `.in` file must match those in the corresponding `.ans` file to within the `Accuracy` specified below.
- Your solution must take no longer than the `Time limit` specified below to produce its outputs.

You can test your solution by passing the `#.in` input data to your program as stdin and comparing the output to the corresponding `#.ans` file:

```
python3 circuit_training_500_template.py < 1.in
```

WARNING: Don't modify any of the code in the template outside of the `# QHACK #` markers, as this code is needed to test your solution. Do not add any print statements to your solution, as this will cause your submission to fail.

Specs
Accuracy: $>$ **95%**
Time limit: **60 s**

Version History

Version 1: Initial document.