EEEN40690 Quantum Computing

Final Project
Quantum Support Vector Machines

Instructions

- This is the final project. This accounts for 30% of the marks for this module.
- In your report, please provide answers to the questions. Explain clearly how the answers are obtained and what are their meaning or interpretation. Include relevant intermediate steps of the solution and explain your approach.
- Make sure that the report is readable, and the graphs (if any) are presented according to scientific/engineering standards.
- Some of the questions are open-ended and include a research component. Please formulate clearly your hypothesis and explain what will prove (or disprove) your hypothesis. Make sure that you provide sufficient evidence (analytical results, numerical results, modelling and simulations, evidence from the literature) to support your answer to open-ended or research problems.
- The report must be submitted online through UCD Brightspace:

  My Brightspace → EEEN40690 → Assessment → Assignments → Final
- Late submissions will be accepted but a penalty will apply. In the case of late submissions, this module applies the standard UCD policy.
- Plagiarism and copying are offences under the terms of the Student Code, and you should be aware of the possible consequences.

Aim

The final project provides exposure to

- Support Vector Machine Classification
- Classical Kernels including Linear and RBF
- Quantum Kernels

Notes

You will use the following datasets in this project. These datasets do not have a column with a binary partition so you will need to define this partition and use this for training. Some examples of partitions are given below with the datasets.

- Pizza Dataset
  e.g: Brand 'A' or not Brand 'A', or some other binary partition.
- Wine Class Classification
  e.g: Target 0 or not Target 0, or some other binary partition
- Truffle Classification

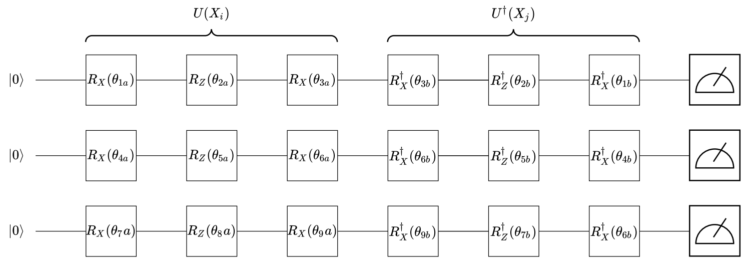
You will need to replace the column you choose with your binary partition, for example in the pizza dataset you would replace brand with 1 for Brand 'A' and 0 for not Brand 'A'. The truffle dataset has already been prepared as a binary dataset.

You may need to use a quantum circuit simulation library such as that provided by QuTip given the large number of qubits in this assignment.

Problems

1. We will first perform a classification using classical kernels. In the last homework you were introduced to creating a support vector classifier using both linear and RBF kernels. You should now repeat this for the datasets given above. You should clearly explain what binary partition you have used, how you have split your data for training and testing, and the accuracy of the classification in each case. This is somewhat open ended so be sure to explain your reasoning for any choices you have made.

2. To build a quantum kernel to perform QSVM we will need to first encode our input data in a quantum circuit. Following the encoding method used in *Peters et al. 2021*, we need to map all data to be encoded in the range $[-\frac{\pi}{2}, \frac{\pi}{2}]$ as we encode the data as parameters of rotation gates. The number of qubits you use is up to you, but be aware of the computational cost of increasing the number of qubits in your kernel. One example of an encoding is to map our data like so:



$U(X_i)$
$U^\dagger(X_j)$
$|0\rangle$
$R_X(\theta_{1a})$
$R_Z(\theta_{2a})$
$R_X(\theta_{3a})$
$R_X^\dagger(\theta_{3b})$
$R_Z^\dagger(\theta_{2b})$
$R_X^\dagger(\theta_{1b})$
$|0\rangle$
$R_X(\theta_{4a})$
$R_Z(\theta_{5a})$
$R_X(\theta_{6a})$
$R_X^\dagger(\theta_{6b})$
$R_Z^\dagger(\theta_{5b})$
$R_X^\dagger(\theta_{4b})$
$|0\rangle$
$R_X(\theta_7 a)$
$R_Z(\theta_8 a)$
$R_X(\theta_9 a)$
$R_X^\dagger(\theta_{9b})$
$R_Z^\dagger(\theta_{7b})$
$R_X^\dagger(\theta_{6b})$

Figure 1: An example of encoding classical information with 9 parameters into a quantum circuit.

We map a vector $\boldsymbol{X}_i = (x_{1i}, x_{2i}, \cdots, x_{ni})$ to a sequence of angles $(\theta_{1i}, \theta_{2i}, \cdots, \theta_{ni})$ bound between $[-\frac{\pi}{2}, \frac{\pi}{2}]$. You should first normalise your vector and then map to the angles. We have not specified a mapping so you are welcome to suggest and compare a few. We use these angles in the rotation gates above to encode our information and denote the transform $U(\boldsymbol{X}_i)$. We then map another vector in the dataset to the transform $U(\boldsymbol{X}_j)$ and apply the adjoint, $U^\dagger(\boldsymbol{X}_j)$. Then measurement is performed, resulting in the following sequence:

$$U(\boldsymbol{X}_i)\,|000\rangle \tag{1}$$

$$U^\dagger(\boldsymbol{X}_j)U(\boldsymbol{X}_i)\,|000\rangle \tag{2}$$

Then we measure the expectation value:

$$\langle 000|\,U^\dagger(\boldsymbol{X}_j)U(\boldsymbol{X}_i)\,|000\rangle \tag{3}$$

Our quantum kernel is then defined as:

$$K(\boldsymbol{X}_i, \boldsymbol{X}_j) = |\,\langle 000|\,U^\dagger(\boldsymbol{X}_j)U(\boldsymbol{X}_i)\,|000\rangle\,|^2 \tag{4}$$

You should investigate the following problems for both datasets:

- You should investigate different encodings used in mapping classical information to quantum information. Comment on why different encodings are used in literature. Find some examples and comment on what are the advantages and disadvantages of the encoding schemes you find in literature.
- Perform encoding of your data in a circuit similar to Fig.1. Explain your encoding. The number of qubits need not be 3.
- You should define an SVM using a custom kernel. This custom kernel should be the quantum kernel you define for your encoding scheme. This can be done using Scikit Learn.
- Perform classification using the quantum kernel and compare your results with those found for the classical case. You should report the accuracy of your classifier and the confidence interval.

3. We now consider variations on the quantum kernel. You should comment on the results you find in each case for all datasets. Notice that no entanglement is possible in Fig.1. You should consider 3 variations on the quantum kernel encoding:
   - Introduce entanglers and comment how the accuracy of classification varies.

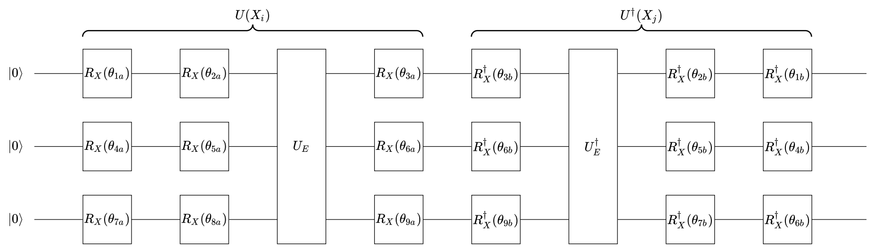


Figure 2

The entanglement unitaries U_E are CNOT gates:

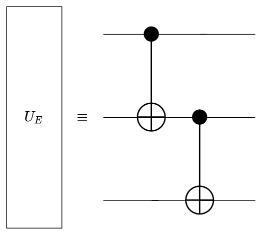


Figure 3

Is the location of the entangling gates important? You can try placing the CNOT gates anywhere in your encoding circuits, and even placing more between rotations.

- Encode your information in a wide circuit, in that you increase the number of qubits with very low depth of gates, something like the following:

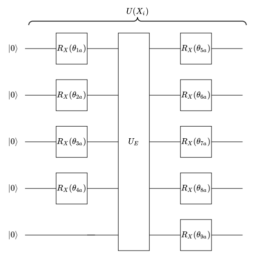


Figure 4

- Encode your information in a very deep circuit, with max 2 qubits, something like the following:

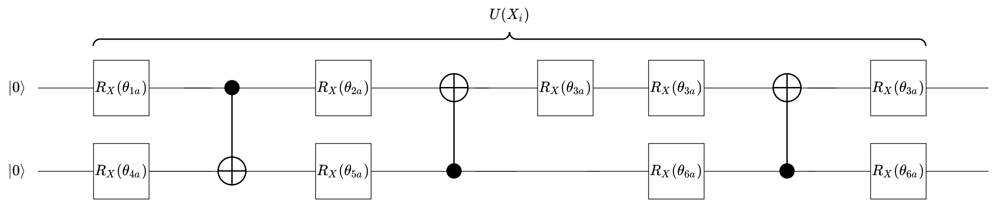


Figure 5

References

Peters, E., Caldeira, J., Ho, A. et al. Machine learning of high dimensional data on a noisy quantum processor. npj Quantum Inf 7, 161 (2021). `https://doi.org/10.1038/s41534-021-00498-9`