Neural Auto-designers for Enhanced Quantum Kernels - AMAL Project

See notes taken on the original paper here:
This note consists of a summary of the paper, outlining the main points of interests and the interesting analysis/implementations.

Todo >

- [ ] Write preliminaries
  - [ ] QC
  - [ ] QML
  - [ ] QKer
- [ ] Comments and analysis of how QuKerNet works
  - [ ] Search space
  - [ ] Train predictor
  - [ ] Top-k kernels search
  - [ ] Kernels fine tuning
- [ ] Compulsory implementations
  - [ ] Basic QuKerNet on tailored MNIST
    - [ ] mRMR vs Random feature selection
    - [ ] Each step enhance prediction
  - [ ] CNN as a neural predictor
- [ ] Optional implementations
  - [ ] KTA as a good substitute
  - [ ] Other datasets (CC/etc...)
  - [ ] GNN/exotic NN as predictor

Motivations of the study

- Pursuit of the global optima in kernel methods
- The space formed by quantum kernels encompasses functions that are computationally hard to evaluate classically
- Automatic design and enhance quantum kernels by optimizing the architecture of the kernels and the parameters of the different architectures

- Resource friendliness

Obstacles

- Qkers associated with inappropriate quantum feature maps
- Deep circuit depth, a large number of qubits, too many noisy gates -> vanishing similarity + degraded generalization ability
- Computational costs of quantum kernels (optimizing/train accuracy costs a lot of measurements etc...)
- Test accuracy of quantum kernels is low -> need to change target
- High-dimensional data on NISQ machines?

Quantum Preliminaries

- Quantum computing
- Quantum machine learning
- Quantum kernels

How QuKerNet works

- Set up search space
- Train the predictor neural network
- Search for the top-k kernels
- Fine-tune kernels with parameters optimization

Results of the paper

- Fig. 2(b) explanation?
- Better to use mRMR than random feature selection
- KTA is a good substitute for train accuracy
- The best kernels are found by Qukernet (vs HEAK, TEK, RBFK)
- Each step of Qukernet enhance performance

What we could bring/implement

- Basic qukernet on tailored MNIST
- Use it to compare mRMR and random feature selection
- verify that KTA is a good substitute
- Check that each step enhance performance
- Use CNN/GNN or another pertinent network for the neural predictor and compare

- Maybe change the way of embedding the QC and use another exotic NN