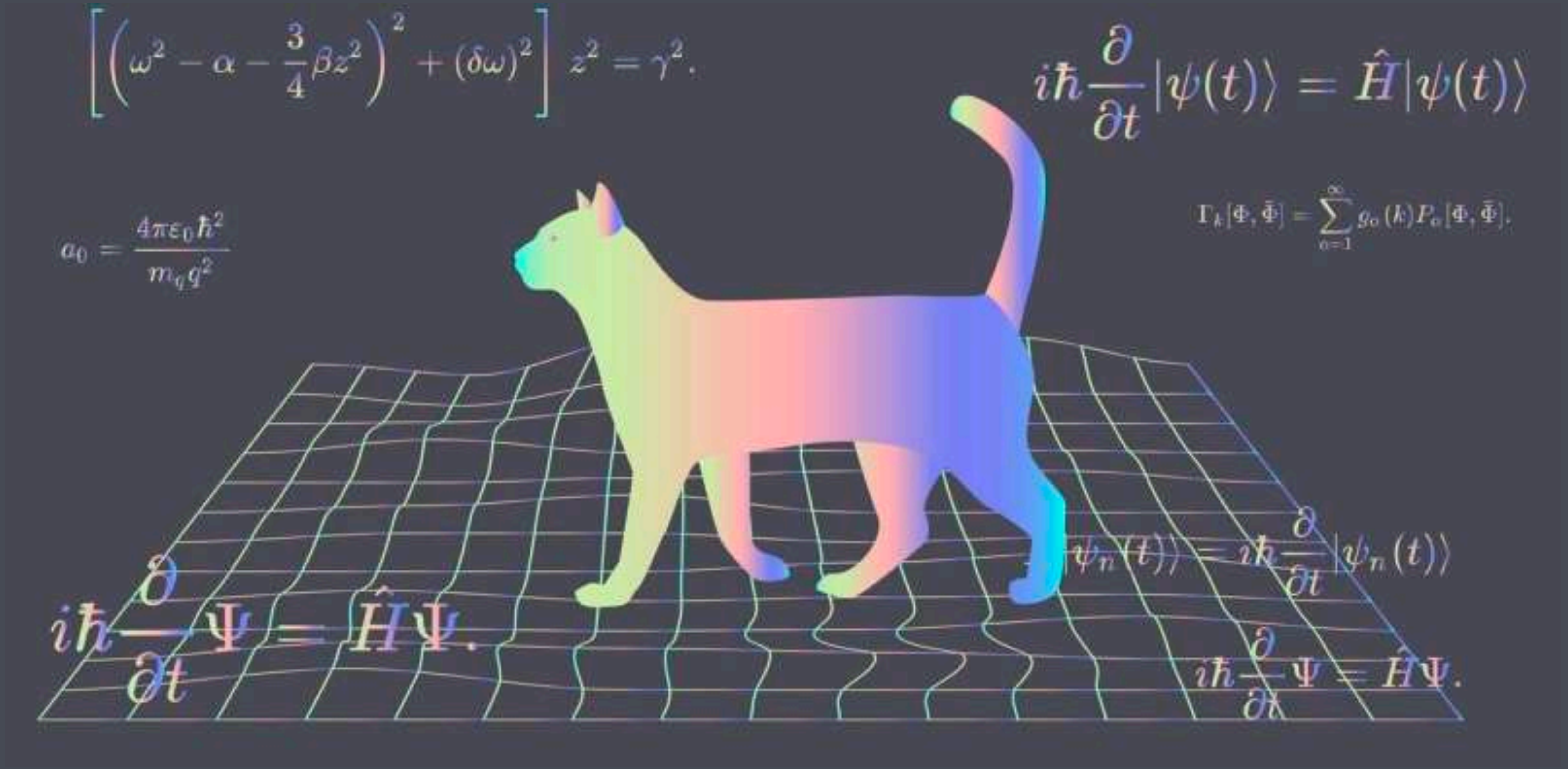
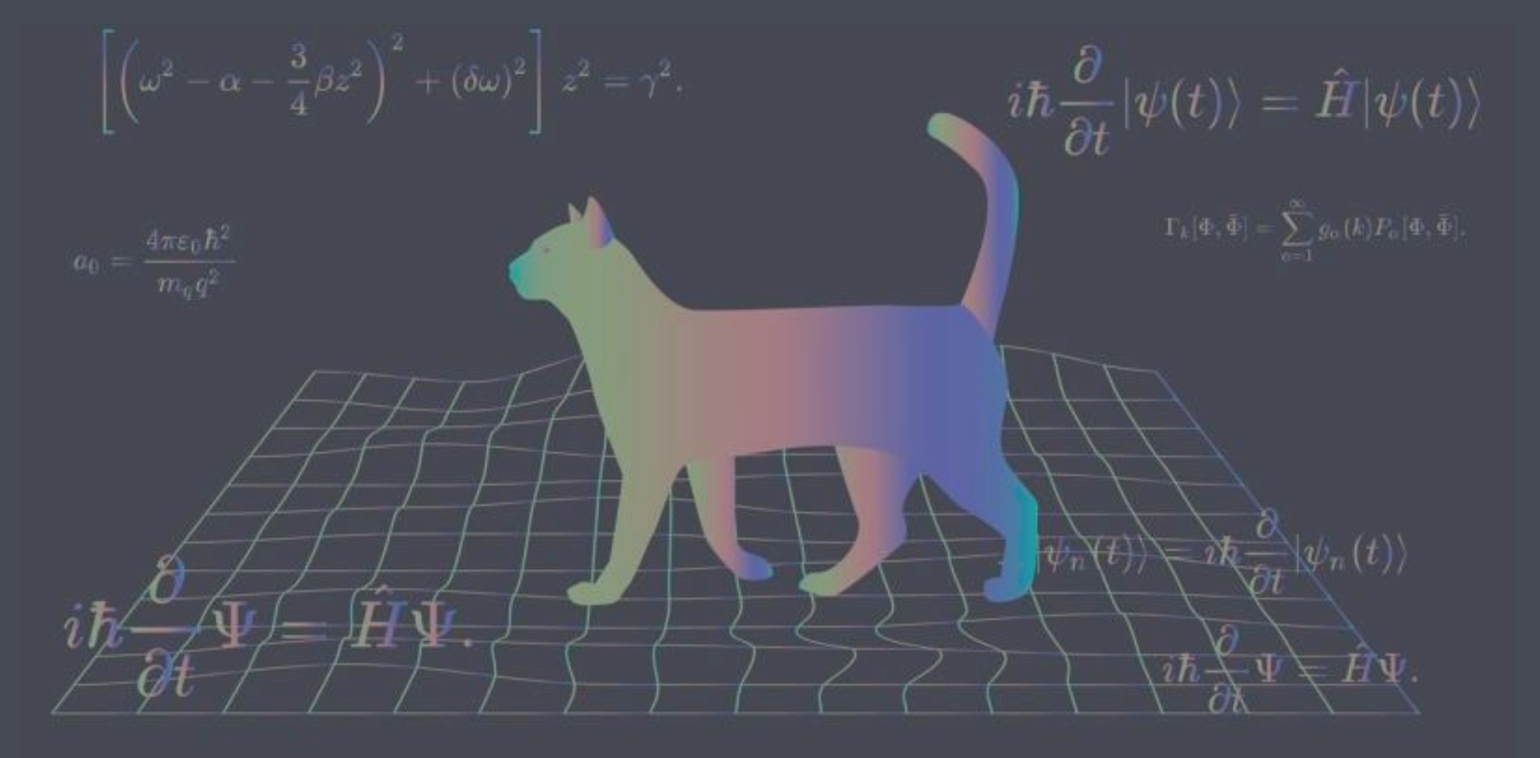


Cryptocurrency Forecasting with Quantum Machine Learning

CatsInHilbertSpace

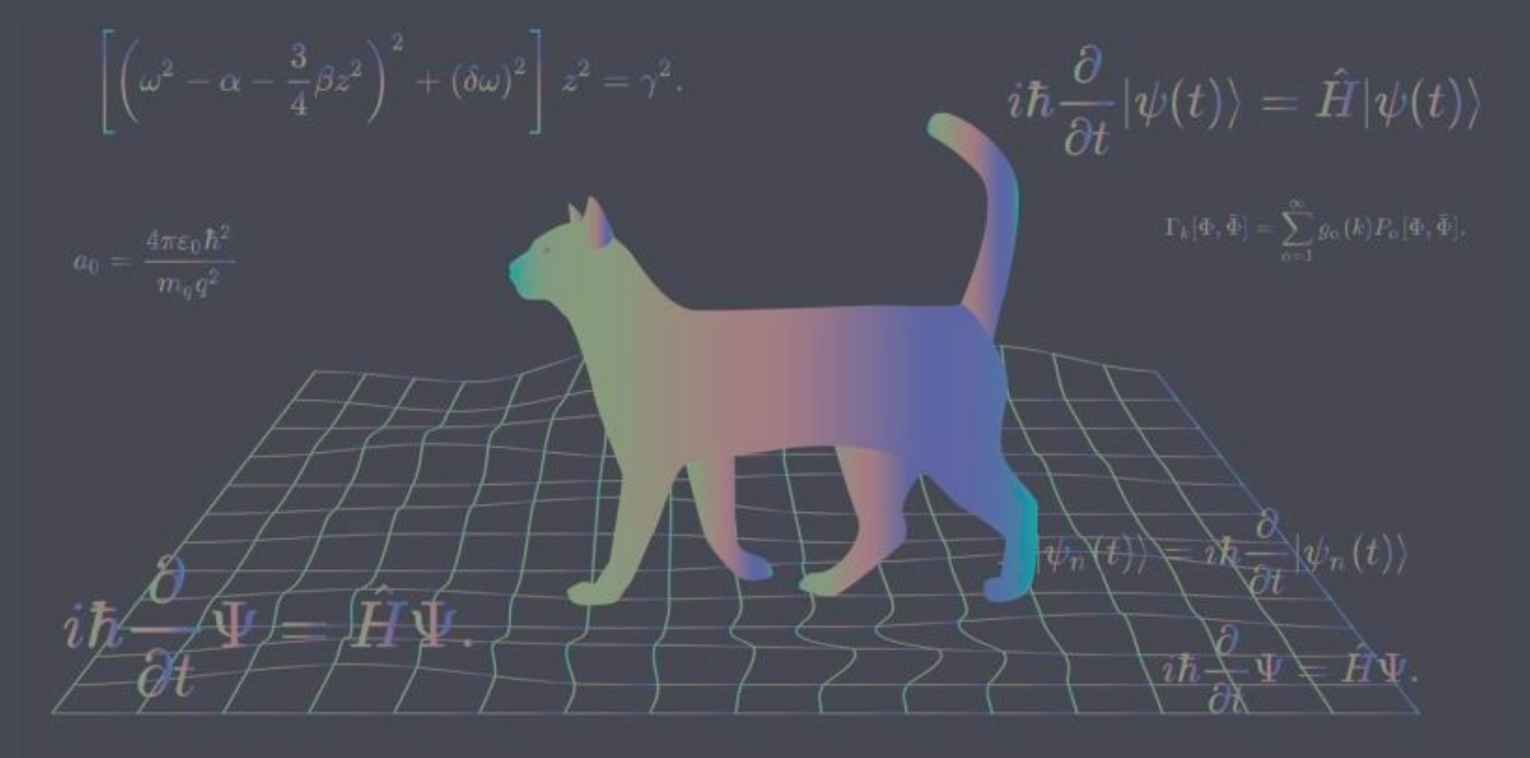
Project Overview



- The purpose of this work is to compare quantum and classical machine learning methods for time series forecasting of cryptocurrency

- In particular, this project analyzes the value of Ethereum, between 2015 and 2021. The project focuses on making near-term predictions of the open price given data from previous days

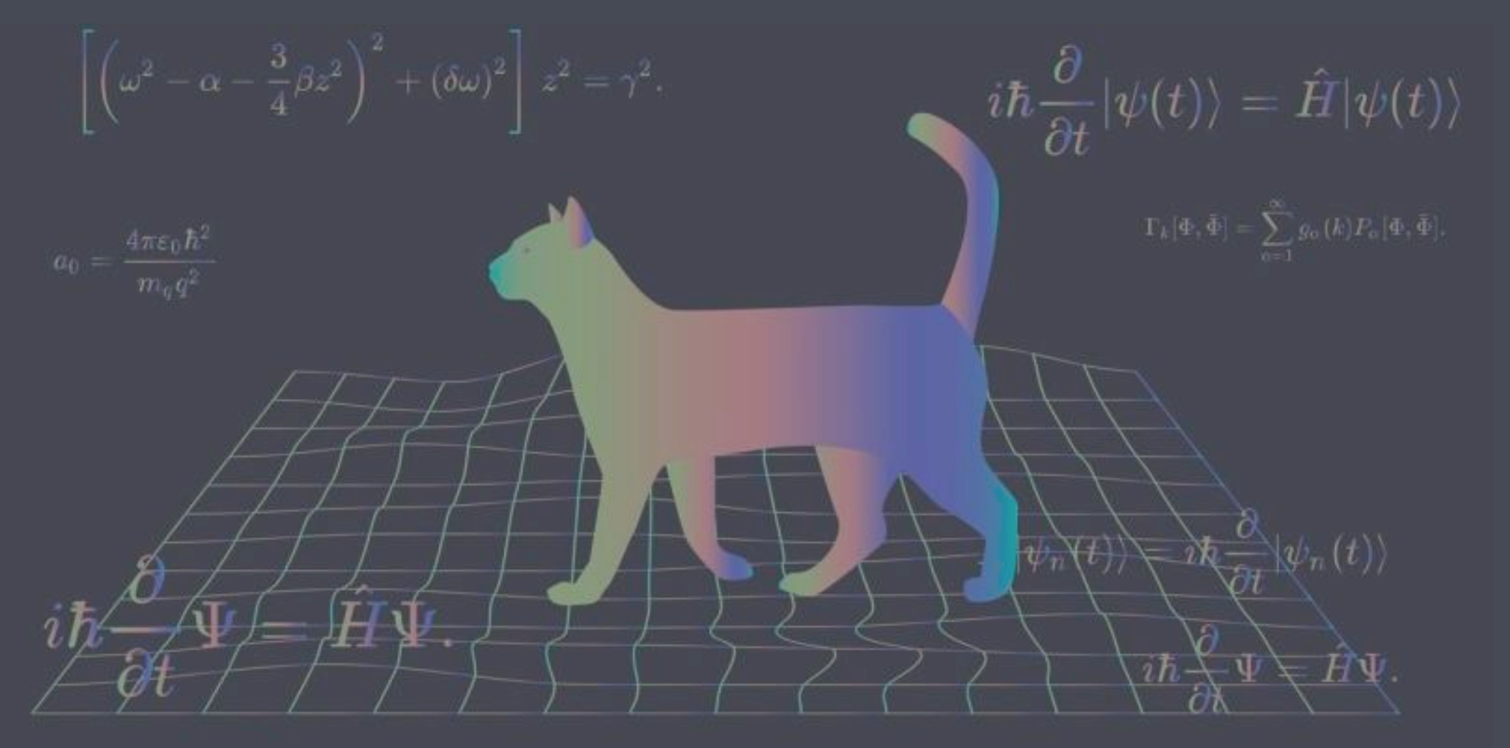
- The techniques include a classical CNN, Hybrid CNN with quantum layer, and custom quantum variational algorithm for sequences



Motivation

- Cryptocurrencies are notoriously volatile and challenging to predict
- However, there is a great financial incentive in predicting their value
- Ethereum is one of the most popular cryptocurrencies with a value that has grown drastically from $7 in February 2016 to around $2600 in February 2022
- This motivated forecasting the Ethereum Open price of Ethereuem given data such as Open, Close, and Volume of the previous (e.g. 5) days
- The goal was to compare quantum and classical forecasting models of Ethereum

Data



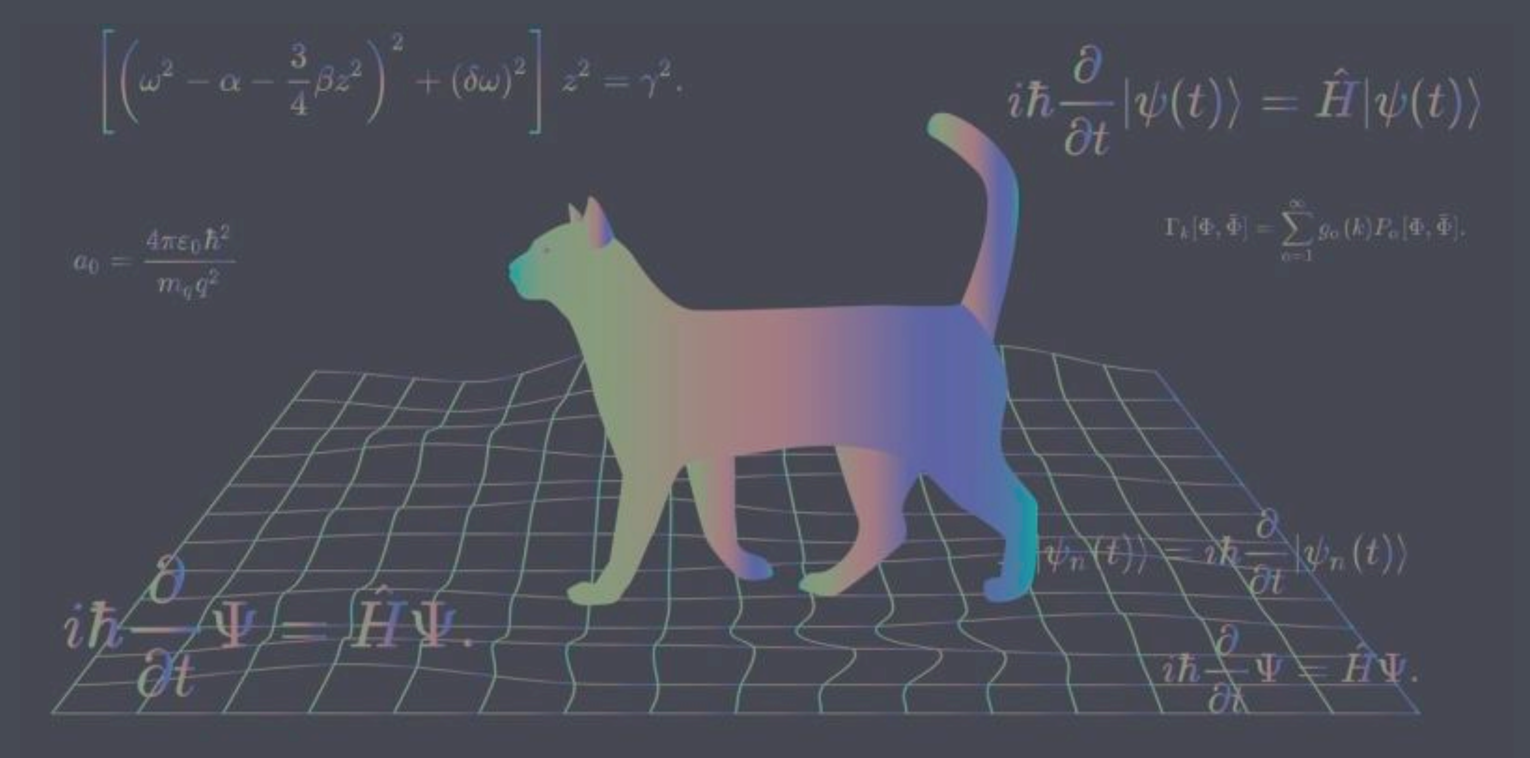
- Ethereum Data - Arpit Verma: https://www.kaggle.com/varpit94/ethereum-data

- This data consists of the Open, High, Low, Close, Adj Close, and Volume for each day between August 2015 to December 2021

Date	# Open	# High	# Low	# Close	# Adj Close	# Volume
Date	Price from the first transaction of a trading day	Maximum price in a trading day	Minimum price in a trading day	Price from the last transaction of a trading day	Closing price adjusted to reflect the value after accounting for any corporate actions	Number of units traded in a day
6Aug15 – 6Dec21	0.43 – 4.81k	0.48 – 4.89k	0.42 – 4.72k	0.43 – 4.81k	0.43 – 4.81k	102k – 84.5b
2015-08-07	2.831620	3.536610	2.521120	2.772120	2.772120	164329
2015-08-08	2.793760	2.798810	0.714725	0.753325	0.753325	674188
2015-08-09	0.706136	0.879810	0.629191	0.701897	0.701897	532170
2015-08-10	0.713989	0.729854	0.636546	0.708448	0.708448	405283
2015-08-11	0.708087	1.131410	0.663235	1.067860	1.067860	1463100
2015-08-12	1.058750	1.289940	0.883608	1.217440	1.217440	2150620
2015-08-13	1.222240	1.965070	1.171990	1.827670	1.827670	4068680
2015-08-14	1.810920	2.261880	1.754750	1.827870	1.827870	4637030
2015-08-15	1.802890	1.877240	1.570980	1.688900	1.688900	2554360
2015-08-16	1.684350	1.695240	1.089810	1.566030	1.566030	3550790
2015-08-17	1.581190	1.581190	1.185340	1.203610	1.203610	1942830
2015-08-18	1.215300	1.331160	1.087050	1.087050	1.087050	1485680

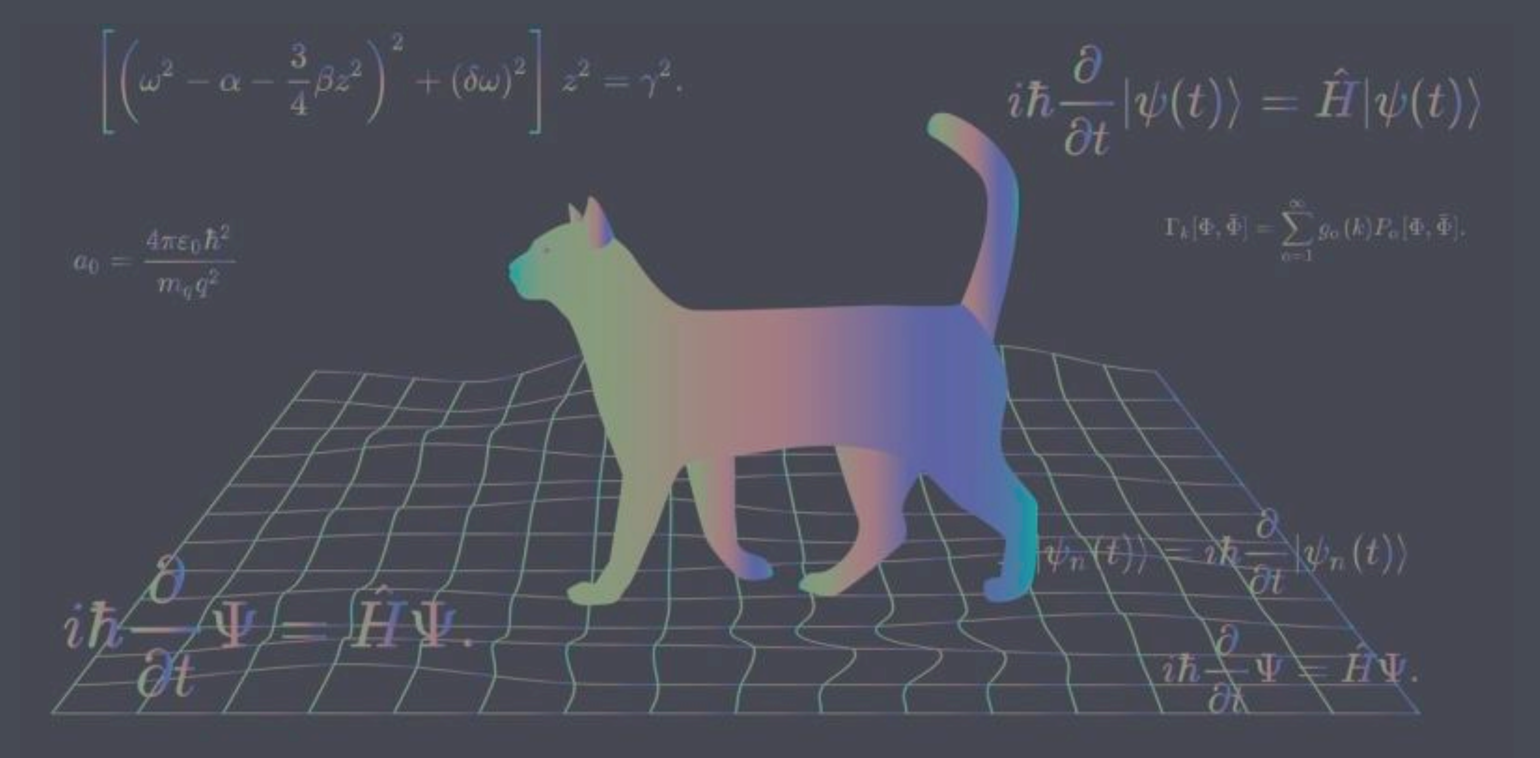
Date	Open	High	Low	Close	Adj Close	Volume
2015-08-07	2.83	3.54	2.52	2.77	2.77	164329
2015-08-08	2.79	2.8	0.71	0.75	0.75	674188
2015-08-09	0.71	0.88	0.63	0.7	0.7	532170
2015-08-10	0.71	0.73	0.64	0.71	0.71	405283
2015-08-11	0.71	1.13	0.66	1.07	1.07	1463100
2015-08-12	1.06	1.29	0.88	1.22	1.22	2150620
2015-08-13	1.22	1.97	1.17	1.83	1.83	4068680
2015-08-14	1.81	2.26	1.75	1.83	1.83	4637030
2015-08-15	1.8	1.88	1.57	1.69	1.69	2554360
2015-08-16	1.68	1.7	1.09	1.57	1.57	3550790
2015-08-17	1.58	1.58	1.19	1.2	1.2	1942830
2015-08-18	1.22	1.33	1.09	1.09	1.09	1485680
2015-08-19	1.17	1.32	1.17	1.26	1.26	1486240
2015-08-20	1.25	1.53	1.25	1.46	1.46	2843760
2015-08-21	1.48	1.56	1.35	1.4	1.4	2020970
2015-08-22	1.4	1.48	1.35	1.38	1.38	948310

Algorithms



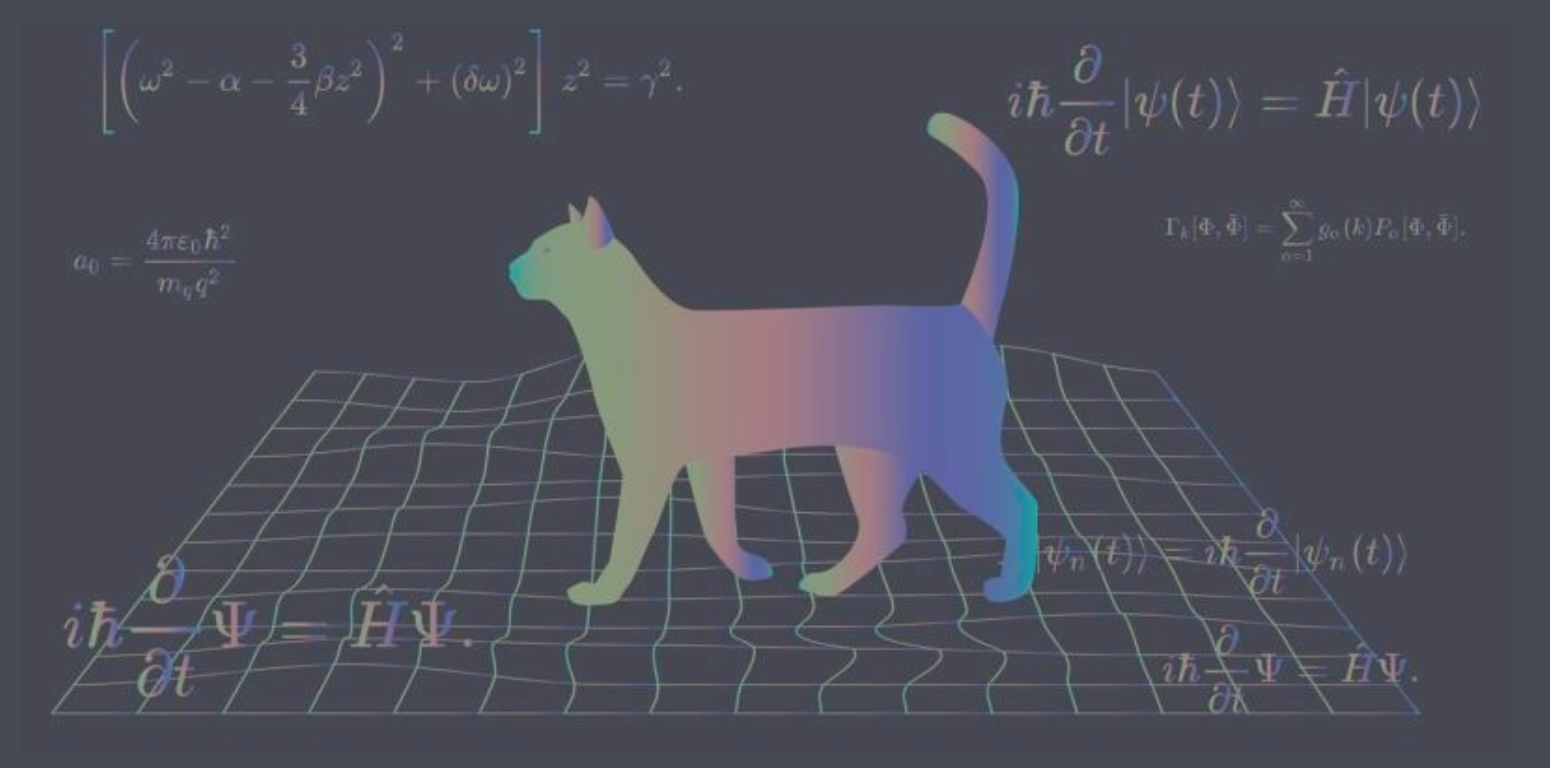
- **CNN**: Convolutional Neural Network
  - Primarily used in processing image data, but also proven useful and efficient in sequential data using 1D convolutional layers
  - Used two 1D convolutional layers followed by max pooling layers, and ending with two dense layers
- **Hybrid CNN**: Convolutional Neural Network with one-two quantum layers
  - Same structure as CNN but replaced one dense layer with one-two quantum layers
  - Quantum layer includes angle embedding of features and different circuit structures, such as PennyLane's defined strongly entangled layers
- **Variational Algorithm:** Custom variational algorithm
  - Circuit iterates over sequence, embedding each previous date's data with an angle embedding of the features into the six qubits and using similar circuit structures as in Hybrid CNN

Software & Services



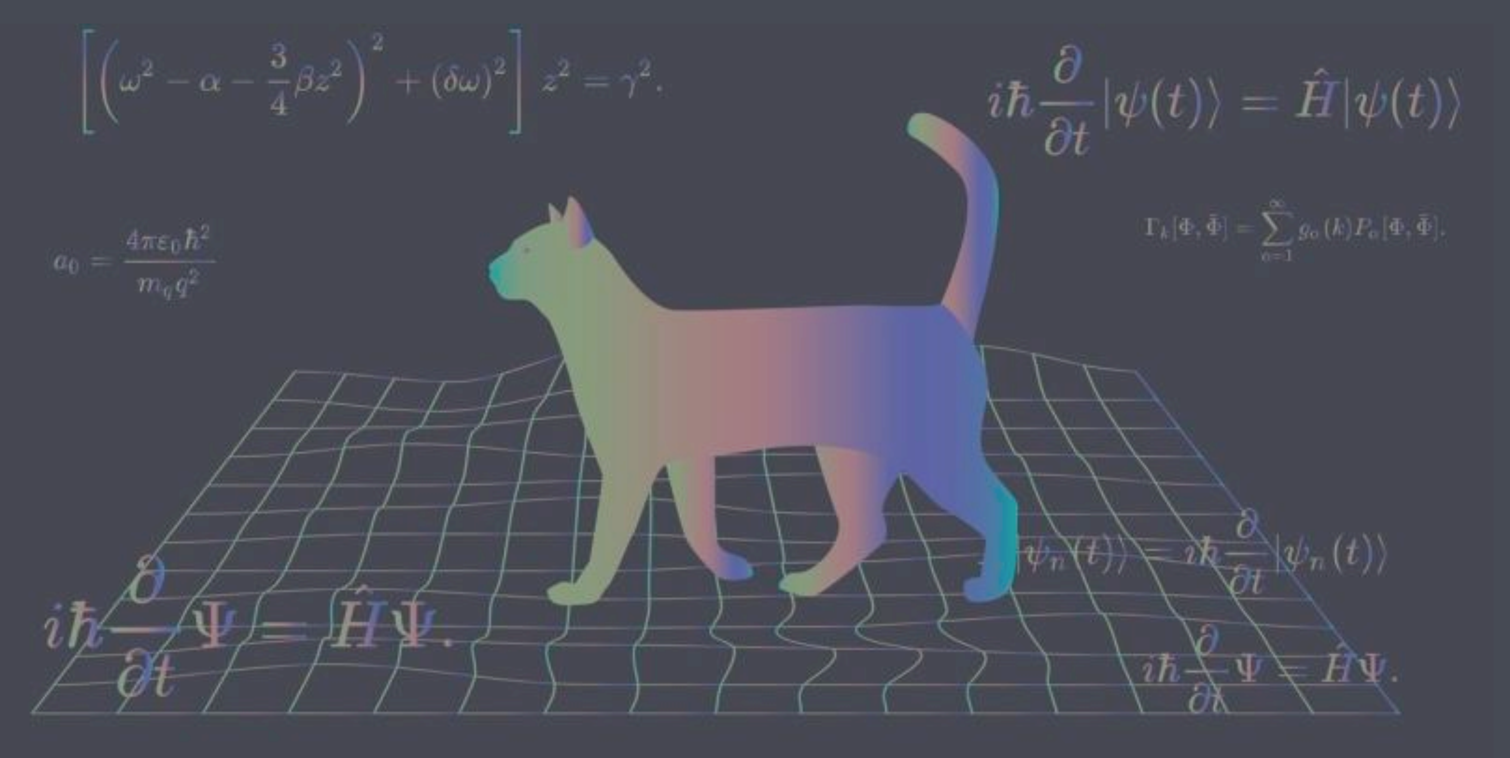
- **PennyLane**: for the Hybrid CNN and Variational Algorithm
- **Braket**: for running on simulators
- **Torch**: for CNN and Hybrid CNN
- **Scikit-Learn**: for preprocessing data
- **Matplotlib**: for plotting data
- **Pandas** and **Numpy**: for representing data

Procedure

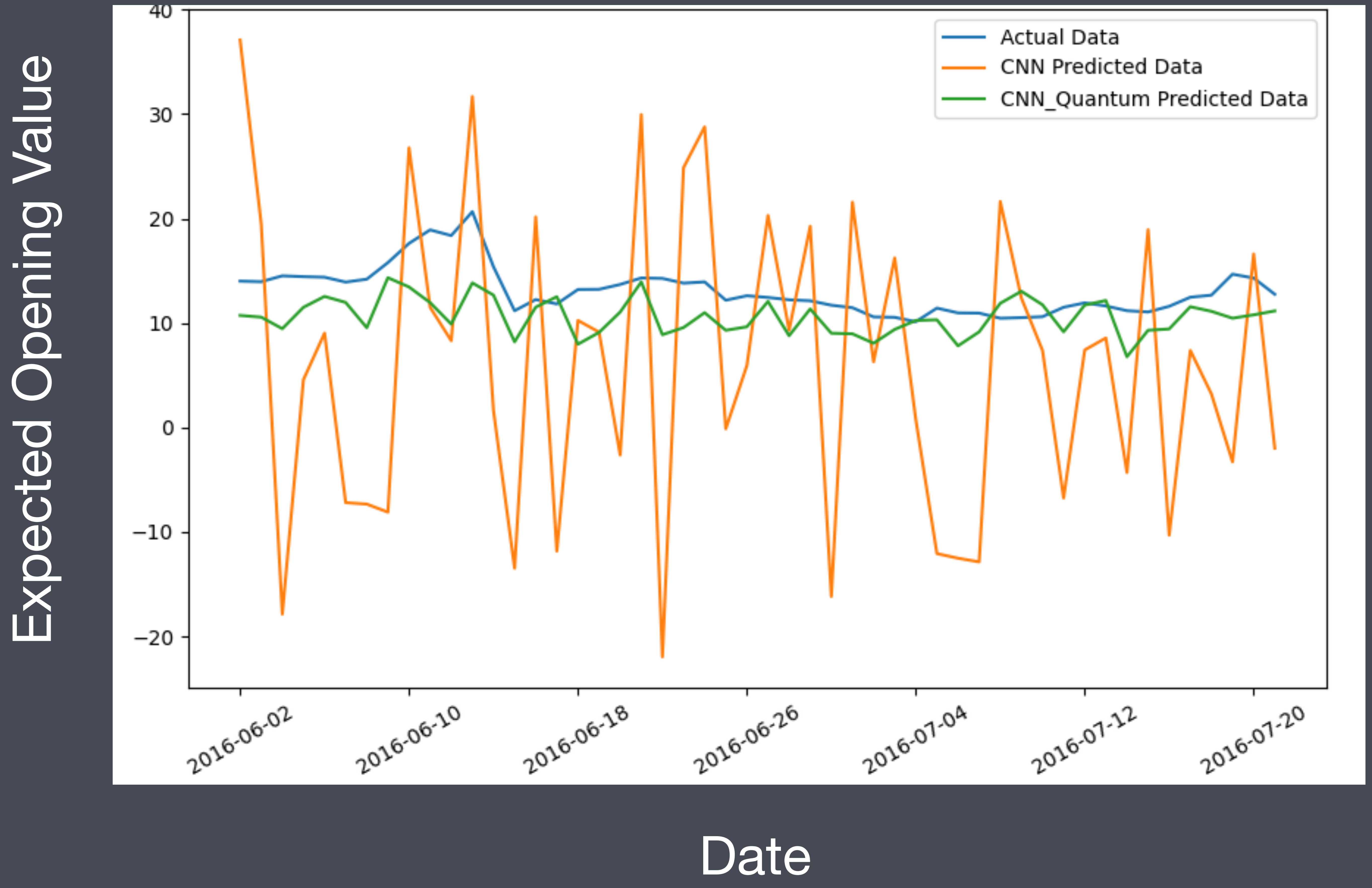


- **Read**: read data in as pandas dataframe
- **Preprocess**: basic preprocessing, including standardizing the features, scaling *y* between 0 and 1, and reshaping data.
- **Train**: optimize square loss comparing training batch processed predictions and true values
- **Test**: compare square loss of predictions and true values
- **Write**: write results to csv file, including information about parameters
- **Plot**: plot results in several formats, comparing different model predictions

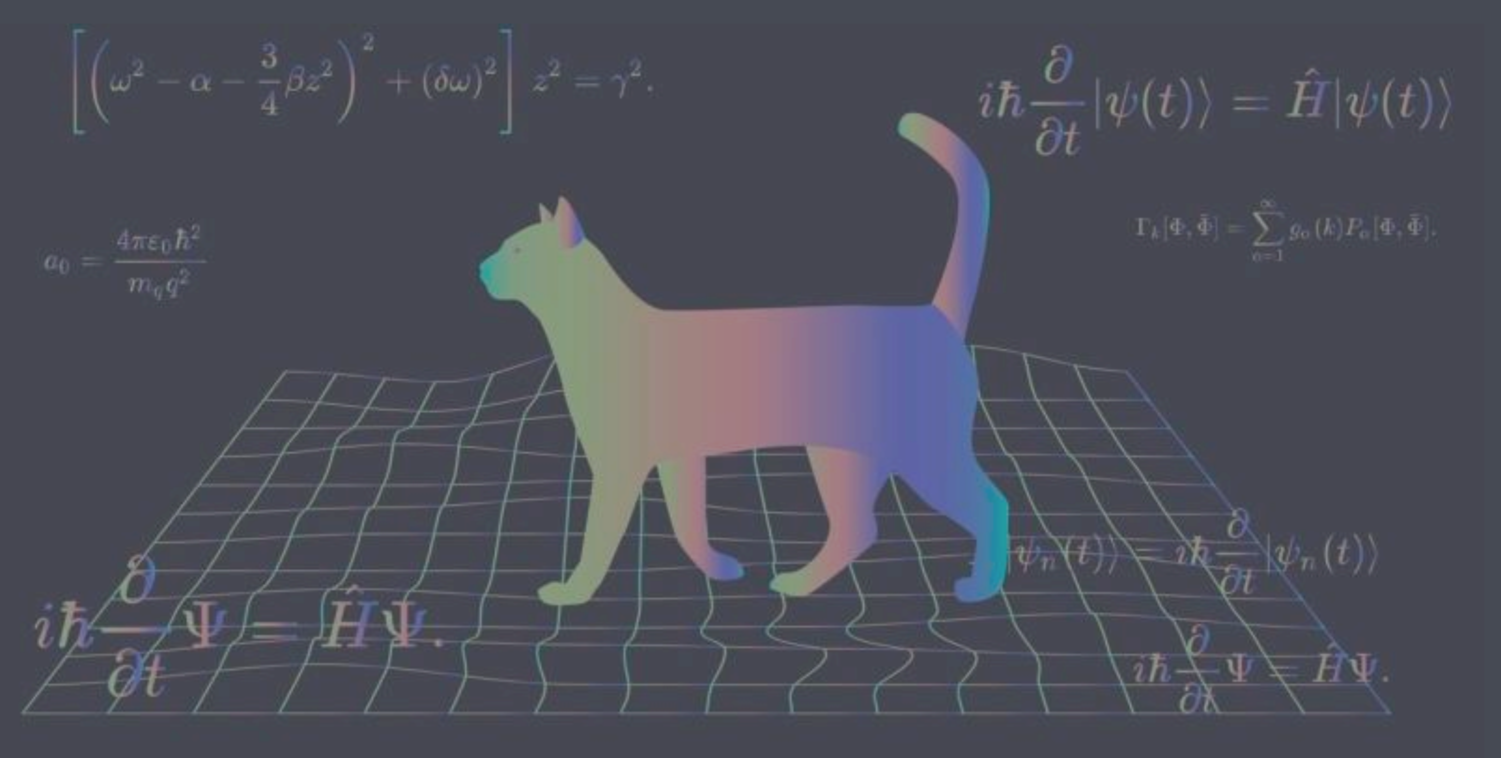
Plot 1:
Local Simulator CNN (+ Quantum)



- **Overall**
  - Size: Train: 100 samples, Test: 50
  - Iterations: 1000, Batch Size: 10
  - CNN Layers Out Channels: 128, 64
  - Lookback: 5
- **CNN**
  - Times: Train: 1.94s, Test: .002
  - Loss: Train: 9.45e-7, Test: 1.05e-5
- **CNN+Quantum**
  - Times: Train: 776.82s, Test: 2.44
  - Loss: Train: 1.03e-7, Test: **4.67e-7**

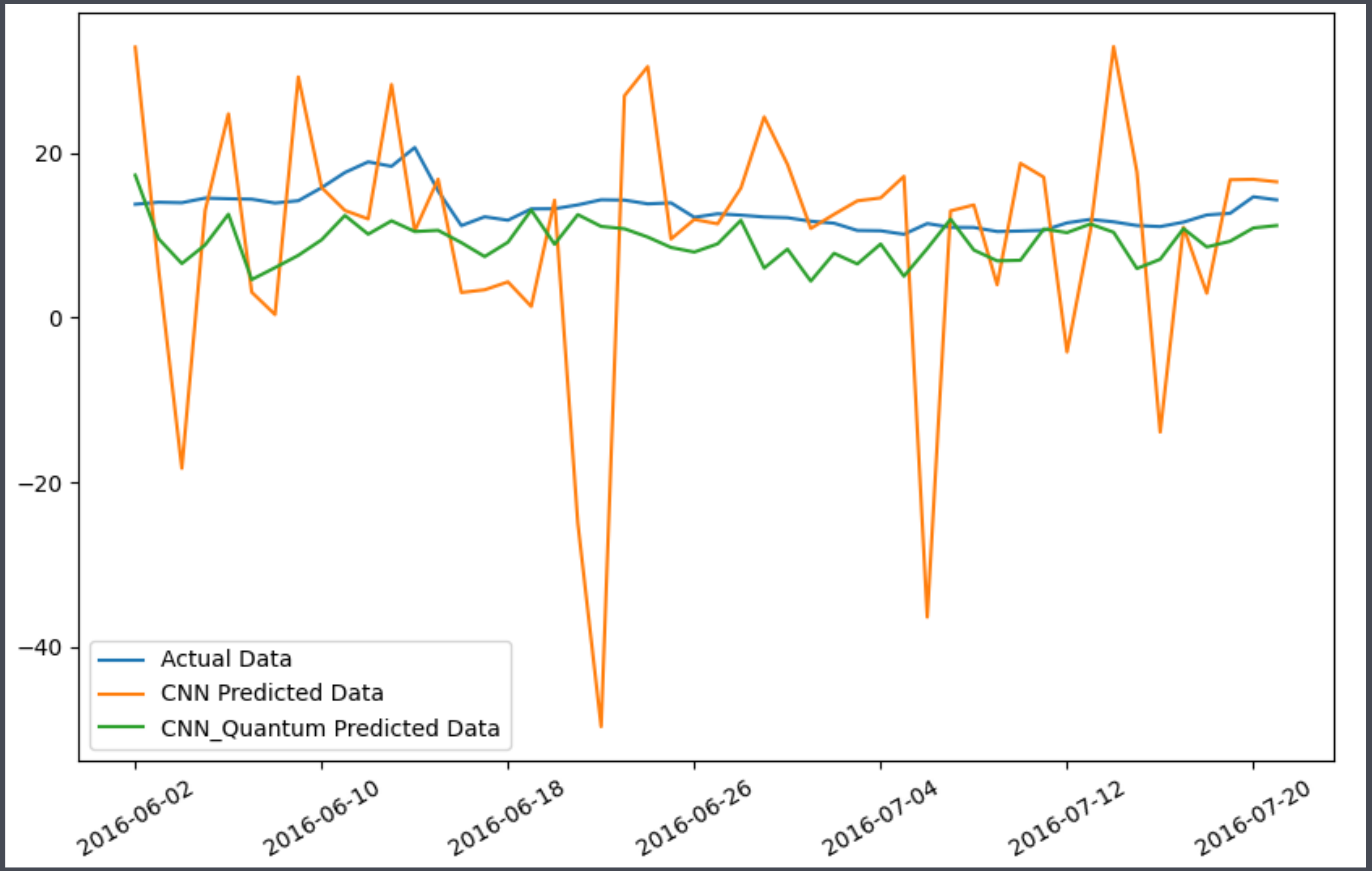


Plot 2: Local Simulator CNN (+ Quantum)



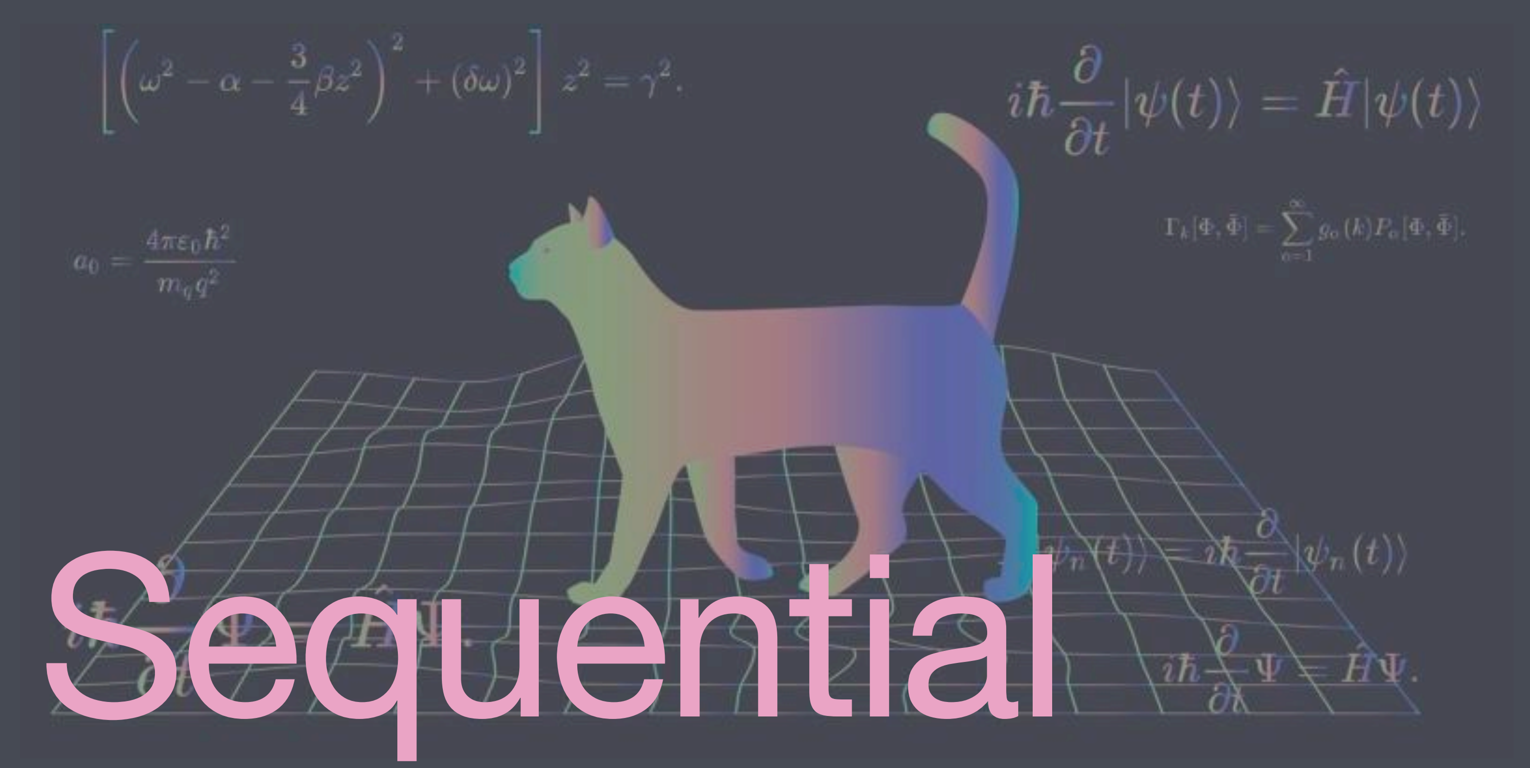
- **Overall**
  - Size: Train: 100 samples, Test: 50
  - Iterations: 1000, Batch Size: 5
  - CNN Layers Out Channels: 128, 64
  - Lookback: 4
- **CNN**
  - Times: Train: 4.69s, Test: .002
  - Loss: Train: 9.97e-07, Test: 1.123e-05
- **CNN+Quantum**
  - Times: Train: 154.53, Test: 0.86
  - Loss: Train: 1.30e-07, Test: **9.49e-07**

Expected Opening Value



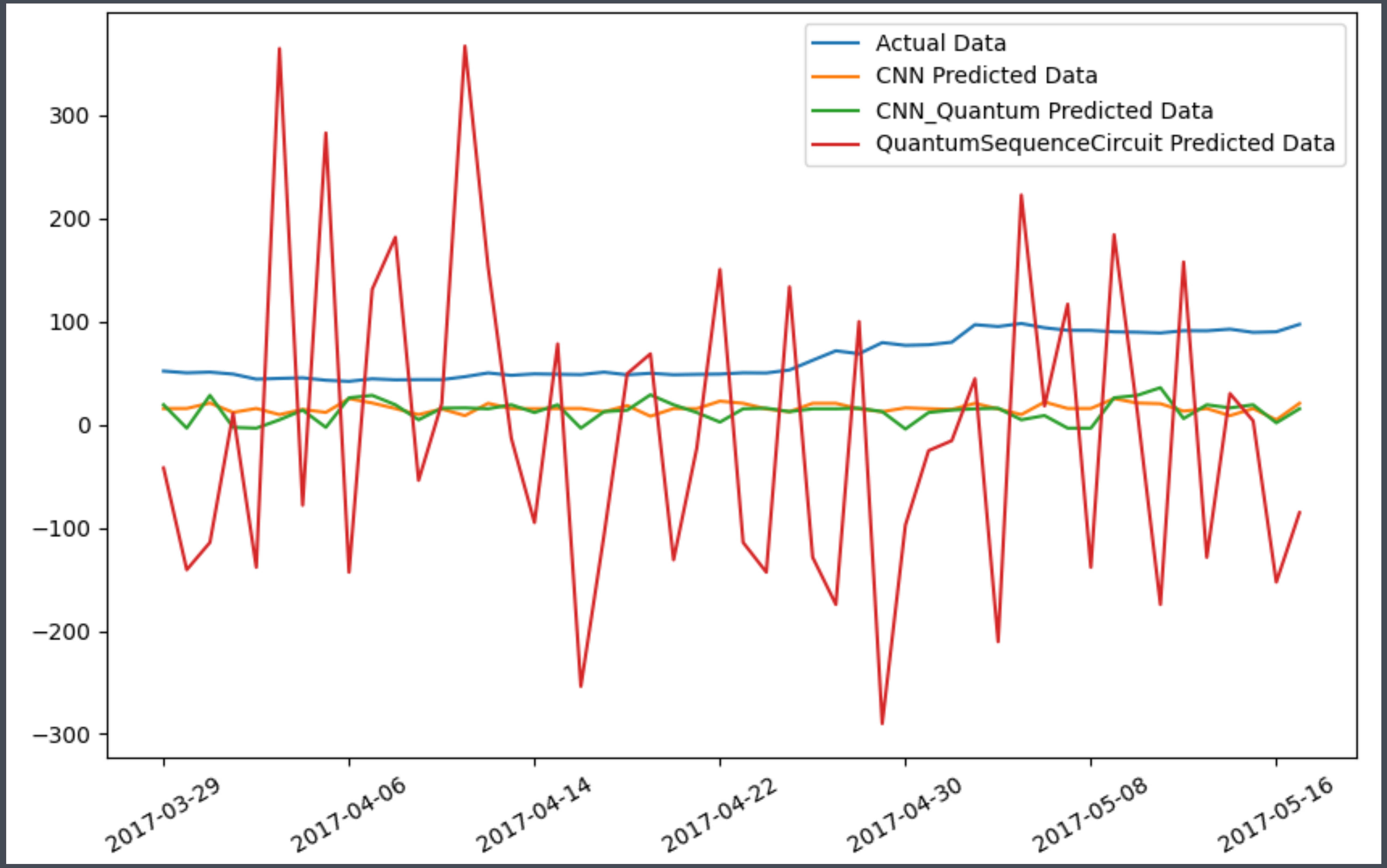
Date

Plot 3: Local Simulator CNN (+ Quantum) & Custom Variational Sequential



- **Overall**
  - Size: Train: 100 samples, Test: 50
  - Iterations: 1000 (100 for Variational), Batch Size: 10
  - CNN Layers Out Channels: 64, 32
  - Lookback: 5
- **CNN**
  - Times: Train: 3.74s, Test: .002
  - Loss: Train: 4.55e-6, Test: **0.00012**
- **CNN+Quantum**
  - Times: Train: 152.61s, Test: .833
  - Loss: Train: 1.85e-5, Test: 0.000138
- **Custom Variational Sequential**
  - Times: Train: 395.17s, Test: 0.35
  - Loss: Train: 0.005, Test: 0.002

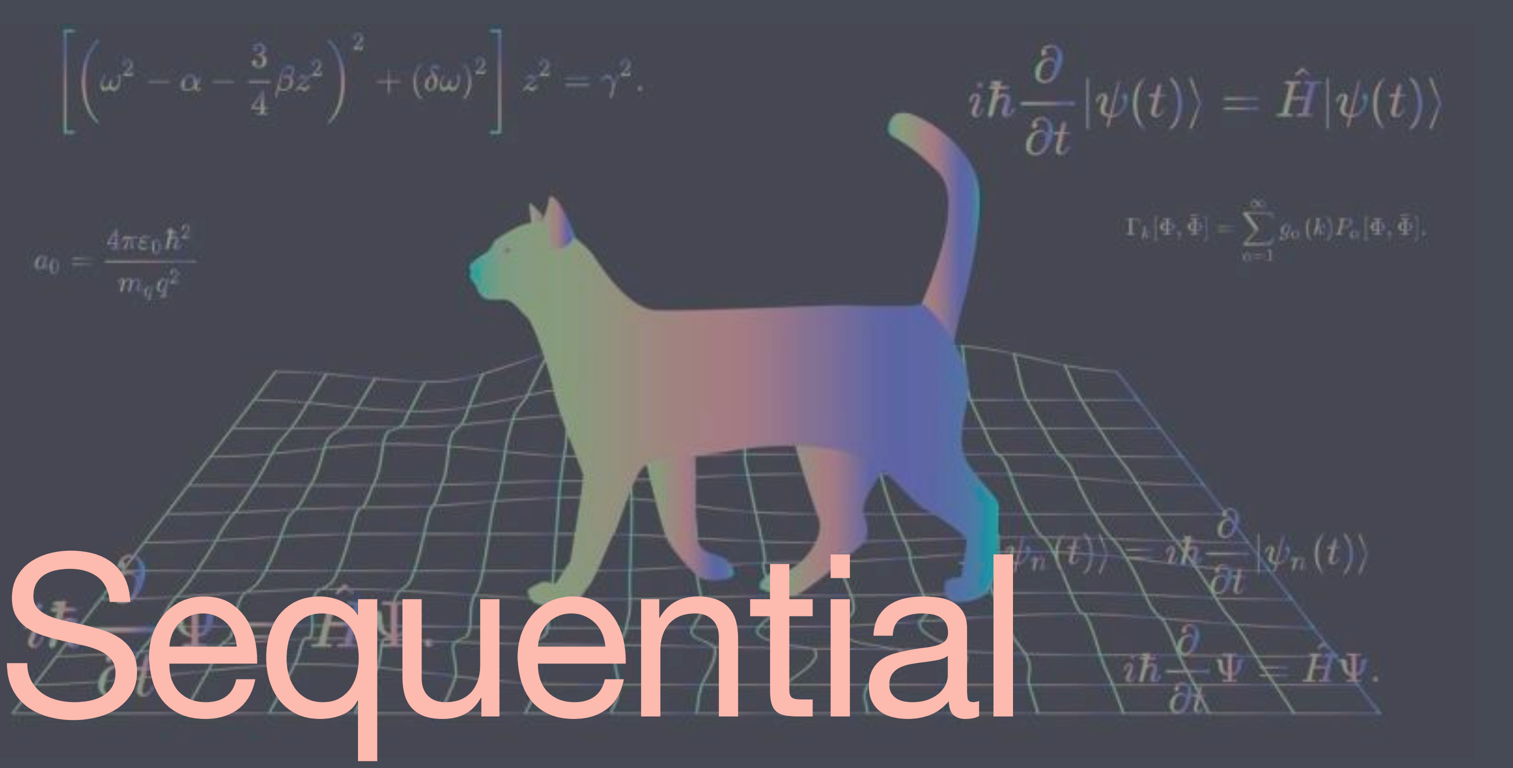
Expected Opening Value



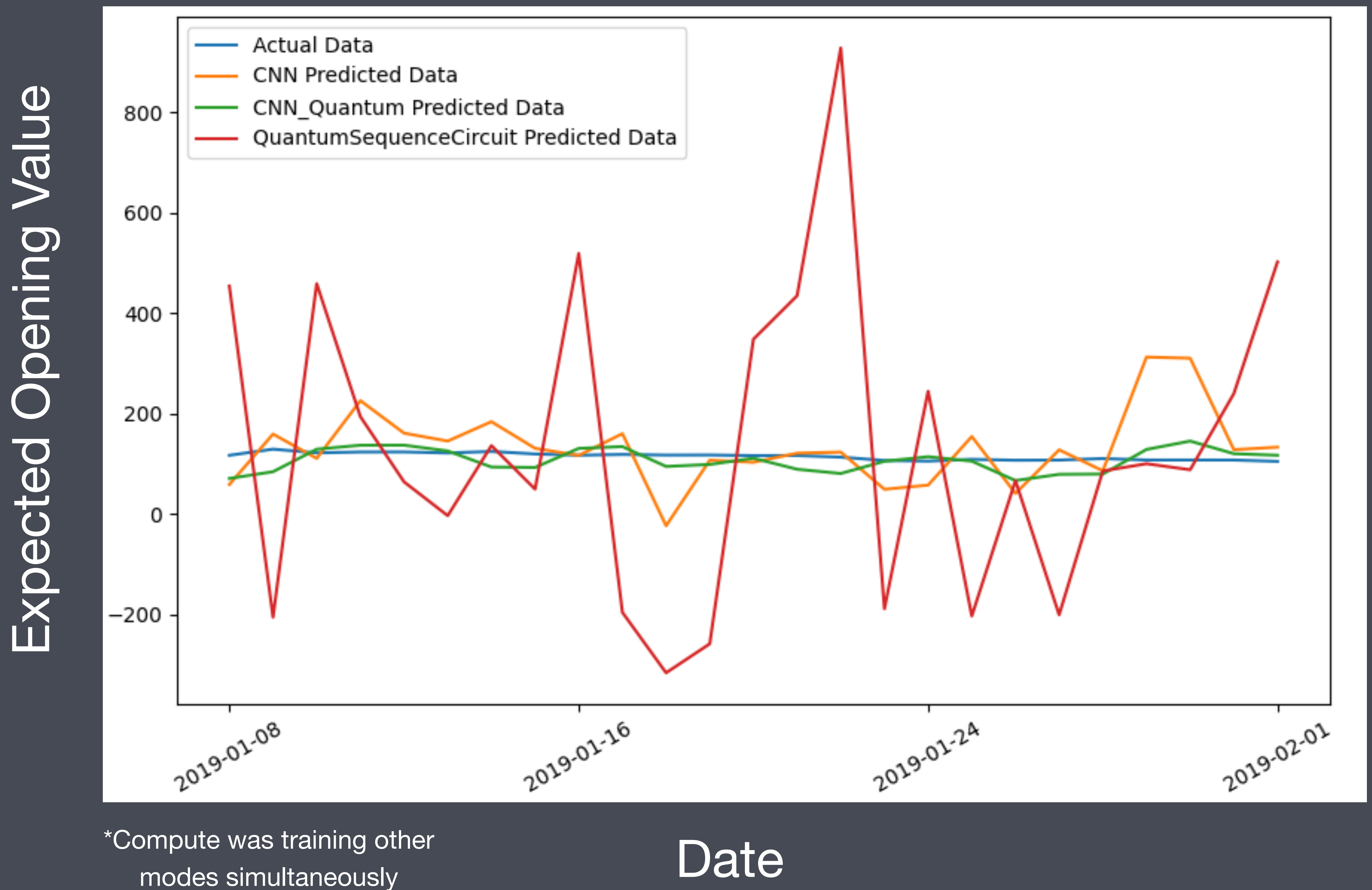
*Note the later start date

Date

Plot 4: Local Simulator CNN (+ Quantum) & Custom Variational Sequential

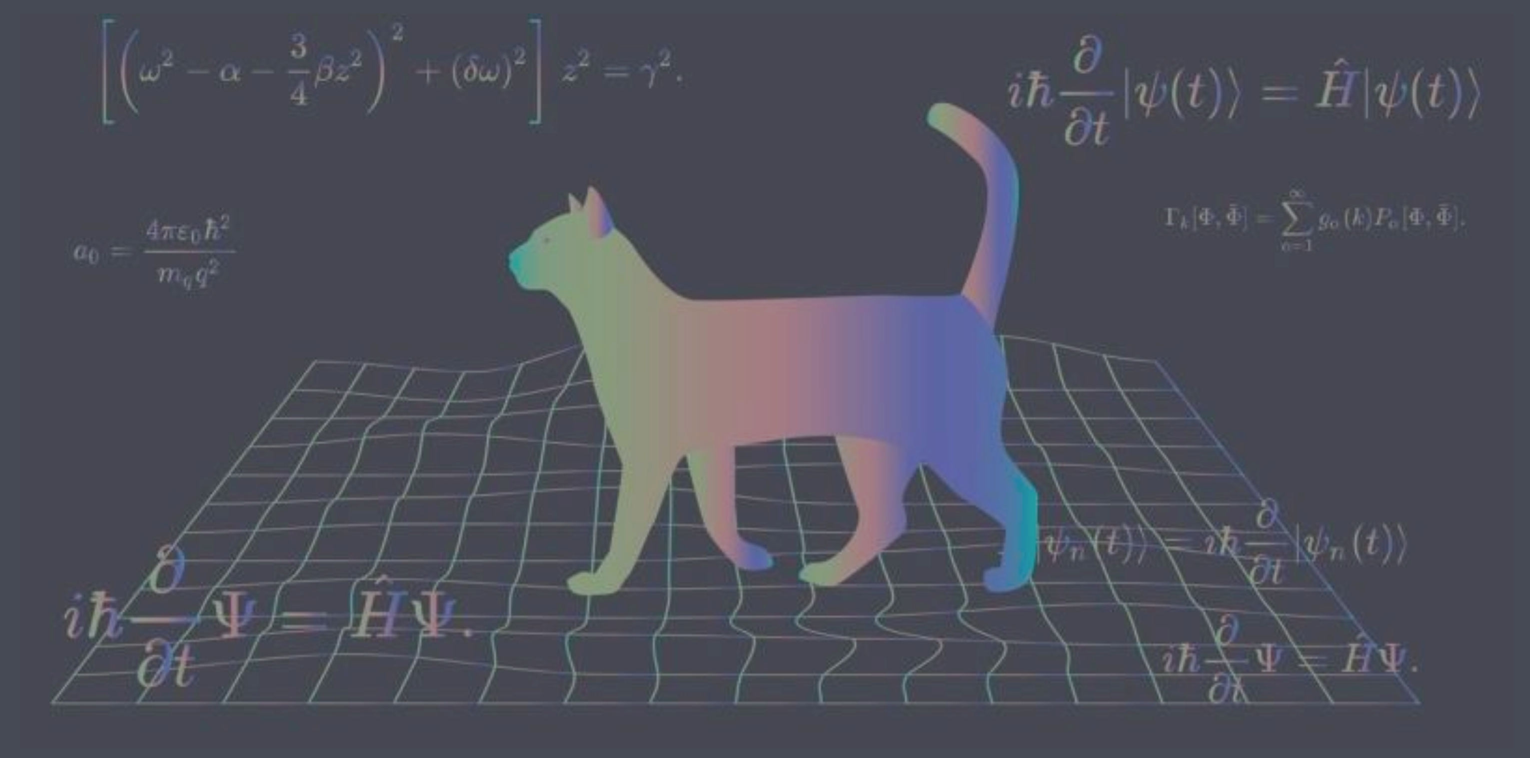


- **Overall**
  - Size: Train: 50 samples, Test: 25
  - Iterations: 1000 (100 for Variational), Batch Size: 5
  - CNN Layers Out Channels: 64, 32
  - Lookback: 4
- **CNN**
  - Times: Train: 4.15s, Test: .002
  - Loss: Train: 1.06e-6, Test: 0.00024
- **CNN+Quantum**
  - Times: Train:157.44s, Test: .423
  - Loss: Train: 1.28e-6, Test: **2.60e-5**
- **Custom Variational Sequential**
  - Times: Train: 2714s*, Test: .0.438
  - Loss: Train: 0.003, Test: 0.004



*Compute was training other modes simultaneously

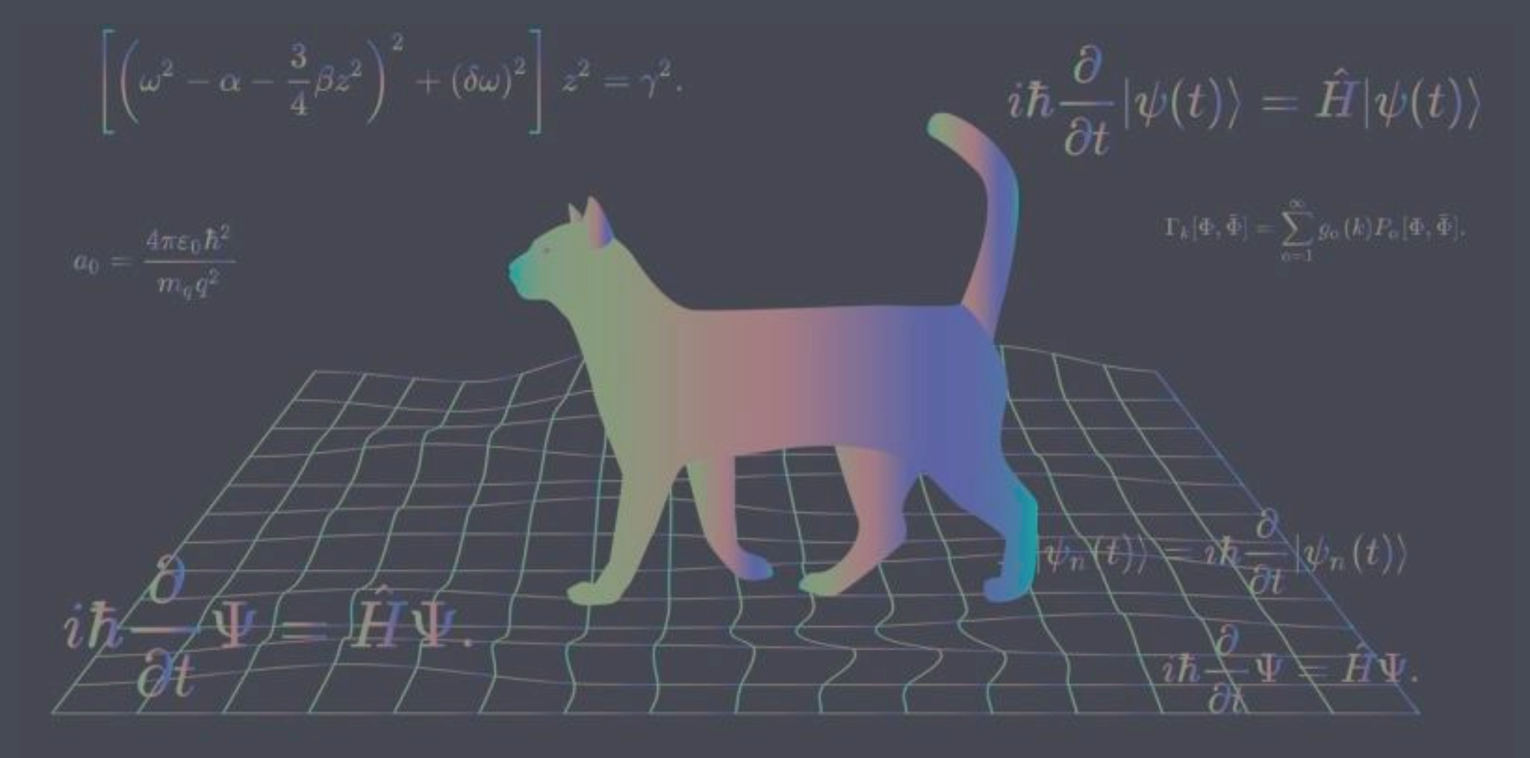
Overall Trends



- Overall, non-quantified trends that were observed from the different parameter choices and train and test sizes. The following rankings were observed:

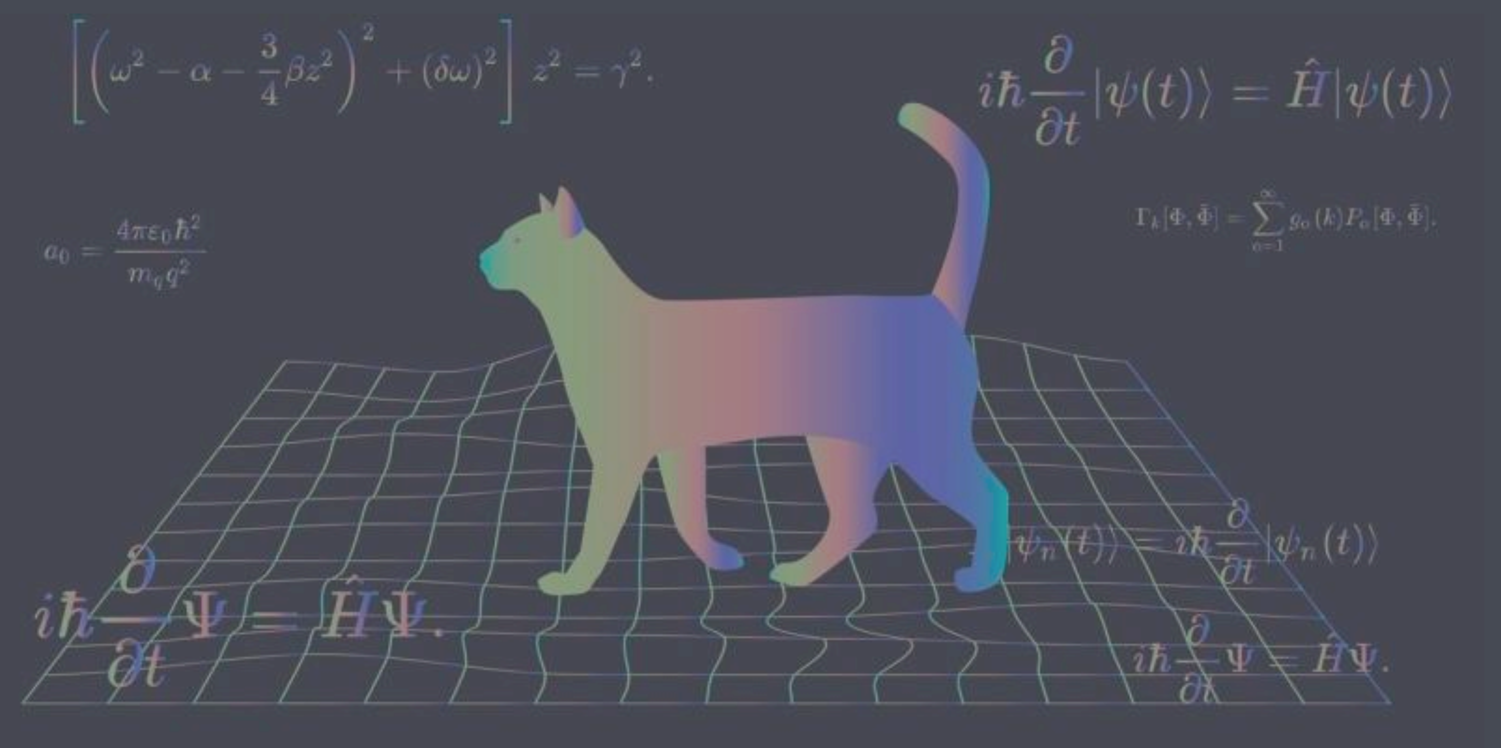
- ***Best performance*** (testing square loss)
  1. **Hybrid CNN + Quantum**
  2. CNN
  3. Custom Variational Sequential

- ***Fastest*** (test and train times)
  1. **CNN**
  2. Hybrid CNN + Quantum
  3. Custom Variational Sequential



Discussion

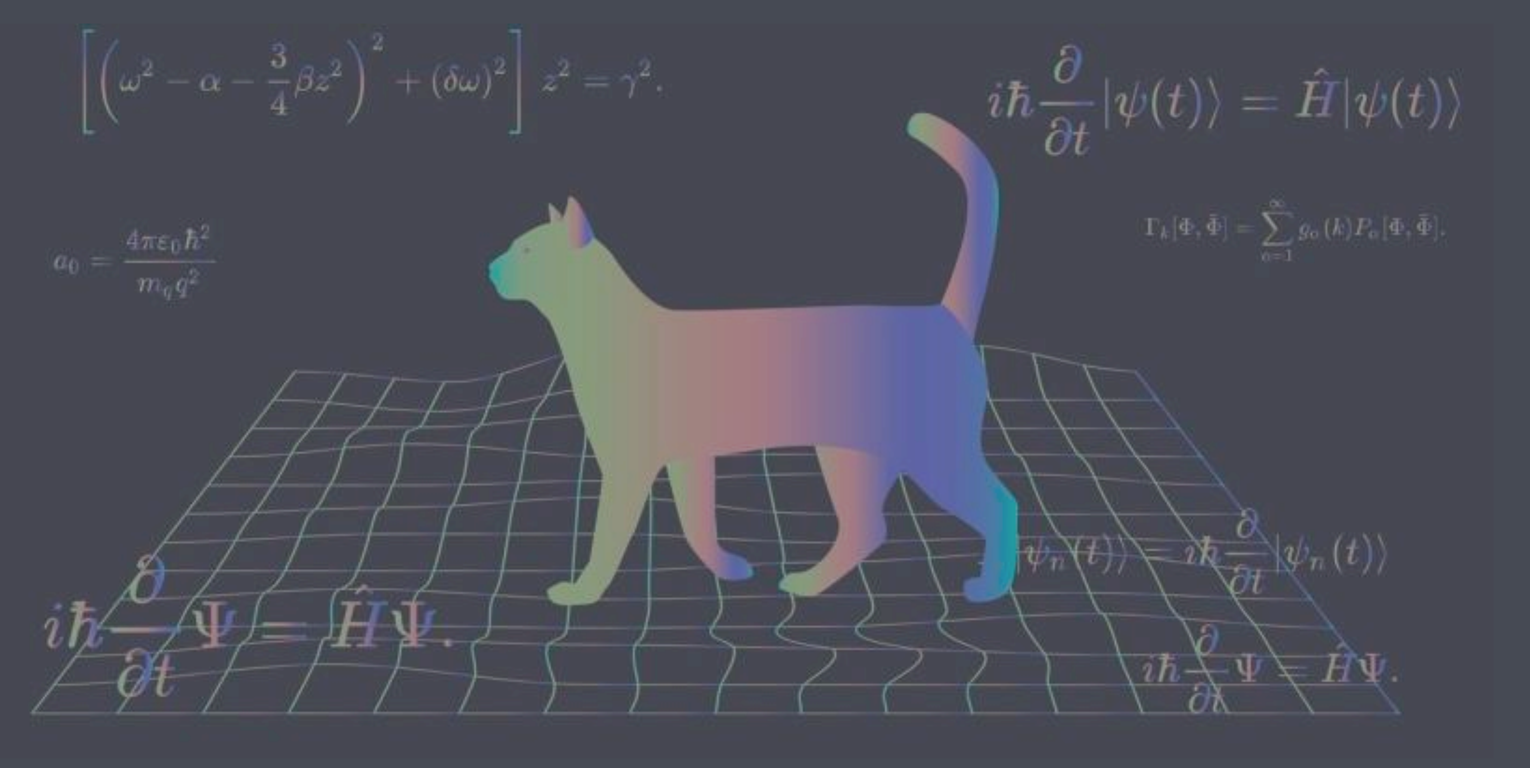
- There is a potential performance benefit to incorporating quantum layers
- The hybrid algorithm with quantum and classical layers outperforms the customized sequential model, and in many cases, the classical CNN as well
- At later dates, the value of the cryptocurrency was more difficult to predict than earlier dates
- Due to time constraints, although the models were run on Braket QPU and Simulators, the runs did not terminate in time to plot their results



Conclusion

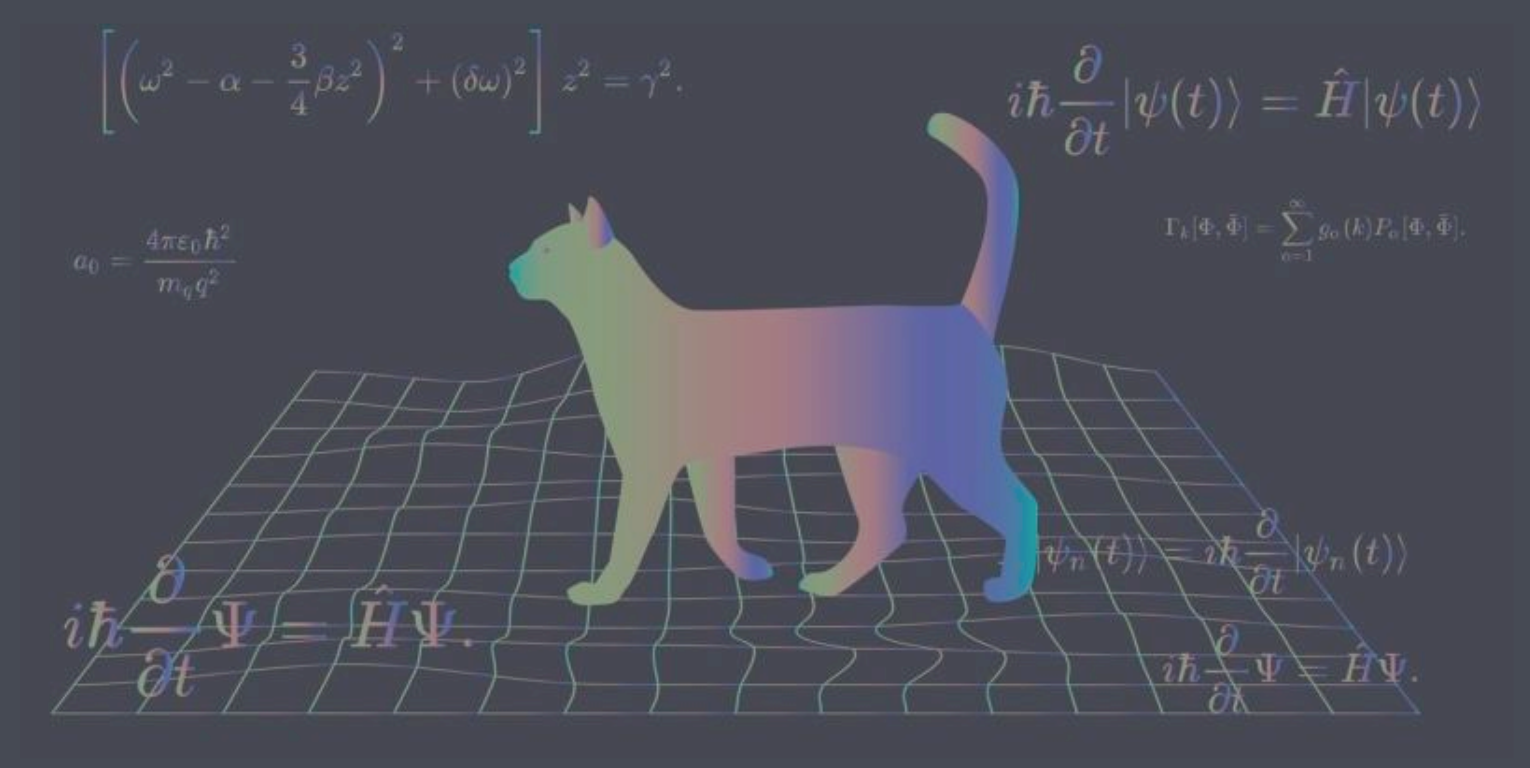
- Forecasting cryptocurrency prices is an important and challenging problem
- This project focused on smaller training and testing sizes, using recent data to make predictions about the near future
- Two quantum and one classical algorithm were used to forecast cryptocurrency prices
- The customized sequential quantum model was much slower to run and did not perform as well as the other two algorithms
- The hybrid quantum and CNN algorithm did perform well, at many points better than the purely classical CNN, indicating a performance benefit to incorporating quantum layers

Future Work



- Implement more classical and quantum algorithms, particularly recurrent algorithms such as LSTM and QRNN
- Continue to explore different parameter choices and layer structures to improve models
- Run on Braket QPUs, and use more shots
- Use different cryptocurrencies
- Predict different features e.g. Close price

Sources



- https://www.statista.com/statistics/806453/price-of-ethereum
- https://arxiv.org/abs/2006.14619
- https://controlandlearning.wordpress.com/2020/07/26/pytorch-basics-1d-convolution/
- https://cnvrg.io/pytorch-lstm/
- https://towardsdatascience.com/pytorch-lstms-for-time-series-data-cd16190929d7
- https://www.kdnuggets.com/2020/07/pytorch-lstm-text-generation-tutorial.html
- https://medium.com/swlh/stock-price-prediction-with-pytorch-37f52ae84632
- Cat image credit: Shutterstock