Forecasting of FTSE 100 Index Price Movement Using Quantum Computing Approaches

Aakash Swami and Gajendra Malviya
Data & Decision Sciences,
TCS Research

I. Introduction

Quantum Computing is a new paradigm that promises a transformation in the way we approach problems in quant finance. It promises improvements to current approaches, brings speed-ups, and improves accuracies in AI/ML, optimization, and simulation problems. One of the fundamental building blocks in finance is to predict the price of an asset that has a dependency on several macro and micro factors, behaviors, and dynamics of the market. Several approaches have been proposed for price prediction problems using classical methods and it is expected that we can further enhance the predictions using quantum approaches. The objective is to effectively predict the price of the FTSE 100 index based on historical data using the quantum approach and benchmark this against the classical approach.
In this report, we have presented two quantum approaches for stock price prediction. First is quantum LSTM. In quantum LSTM we have used the concept of data reuploading and have shown its significance. Second, we propose a novel quantum neural network approach. The result signifies the advantage of the quantum approach in terms of fewer training parameters required and faster learning when compared to classical. The rest of the report is organized as follows: literature review in section II, describing data preparation in section III, then describing classical approach and classical approach result in section IV, quantum approach and comparison of classical and quantum approach in section V, making prediction in section VI, followed by the conclusion in section VII.

II. Literature review

Among literature that uses the classical approach, Moghaddam et al. [1] used a neural network for stock price prediction. Oukhouya et al. [2] compared different machine learning methods—SVR, XGBoost, LSTM, and multilayer perceptron (MLP)—for forecasting the Moroccan stock market. The results show that using the grid Search optimization algorithm, the SVR and MLP models outperform the other models and achieve high accuracy in forecasting daily prices. Xion et al. [3] predict stock prices using LSTM and ARIMA models and show that LSTM performs better than ARIMA.
Among different literature that used the quantum approach for stock price prediction, Dharwad [4] explored the potential applications of various quantum algorithms for stock price prediction by conducting a series of experimental simulations using both classical as well as quantum Hardware. Kaushik et al. [5] treated the problem as solving a system of linear equations and used a variational quantum linear solver (VQLS) for predicting stock price. Dimitrios et al. [6] used a quantum neural network for stock price prediction and compared that with LSTM. Chen et al. [7] proposed quantum LSTM. The paper replaces the neural part of LSTM with a quantum circuit specifically a variational quantum circuit and analyzed the result. Perez et al. [8] proposed the concept of data reuploading. The paper highlights that a single qubit with data reuploading can generalize any function similar to a neural network with a single layer. The paper also shows the analogy between single-qubit data reuploading and neural networks.
We found that the different quantum techniques available that can be beneficial to our work are variational or parameterized quantum circuits, data reuploading, and quantum LSTM. We plan to utilize the concept of data reuploading and quantum LSTM technique for our work.

III. Data Preperation

Data preparation allows us to explore, clean, combine, and format data for sampling and deploying ML models. The data preparation involves steps like generating side features, interpolation, and data normalization. We took FTSE data from 9th Sep 2011 till 1st March 2024 and performed data preparation as shown below.

- **Considering exogenerous feature**: We further used a few features that we think can influence the FTSE future price. These include:
  - GBP USD: Capture British Pound - US Dollar relation.
  - Gold price: Capture gold price change.
  - Crude futures: Capture change in crude futures price.
  - Libor rate: The London Inter-Bank Offered Rate is an interest rate average calculated from estimates submitted by the leading banks in London

Few of these features have high, low, open, and close prices. Instead of using these price values as features, we used the difference of high and low and the difference of open and close as features.

- **Generate side features - Technical indicators**: Many investors use technical indicators of a stock to decide whether to invest or not. Thus, we use various technical indicators as side features.
  - Simple moving average (SMA): The SMA calculates the moving average for a given window size. When price action tends to stay above the moving average, it signals that the price is in a general uptrend. If price action tends to stay below the moving average, then it indicates that it is in a downtrend.
  - Moving averages convergence divergences (MACD): The MACD turns two trend-following indicators, moving averages, into a momentum oscillator by subtracting the longer moving average(large step size) from the shorter one(small step size).
  - Bollinger Bands: The Bollinger bands indicator are volatility bands placed above and below a moving average. They are based on standard deviations of price movements
  - Exponential moving average (EMA): The EMA is a widely used technical chart indicator that tracks changes in the price of a financial instrument over a certain period. Unlike simple moving averages (SMA), EMA puts more emphasis on recent data points like the latest prices.
- **Interpolation**: The missing value is filled via interpolation.
- **Data normalization**: Normalization is one of the most frequently used data preparation techniques, which helps us to change the values of numeric columns in the dataset to use a common scale.

The generated data includes around 29 features ['3M', 'BBB_20_2.0', 'BBL_20_2.0', 'BBM_20_2.0', 'BBP_20_2.0', 'BBU_20_2.0', 'Close_copy', 'Crude Futures_close', 'Crude Futures_volume', 'Crude_H-L', 'Crude_O-C', 'EMA_14', 'EMA_21', 'EMA_7', 'FTSE_H-L', 'FTSE_O-C', 'GBP USD ', 'GBP_USD_H-L', 'GBP_USD_O-C', 'Gold in USD close', 'Gold_H-L', 'Gold_O-C', 'MACD_12_26_9', 'MACDh_12_26_9', 'MACDs_12_26_9', 'SMA_14', 'SMA_21', 'SMA_7', 'Volume'] and the target is the close price.

IV. Classical Approach

A. Classical method

Classical methods are based on statistical models that capture the patterns and relationships in the historical data, and extrapolate them to the future. We use Long Short-Term Memory (LSTM), ARIMA, standard ANN, multiple linear regression(MLR), and XGBoost as classical models. The training data used for training LSTM is different from that used for training standard ANN, multiple linear regression, and XGBoost. Below are the details on training data preparation.
Training data preparation for LSTM: The LSTM are trained on the time series data. Preparation of the time series training data means that it uses a certain number of past observations to predict the future. We define 'sequence length' and 'window'. The 'sequence length' decides how many days the LSTM considers in advance and the 'window' decides how many days we want to predict in the future. If the 'sequence length' is n and the 'window' is w, then the LSTM considers the last n observations to predict the next w day price.
Training data preparation for standard ANN, MLR, and XGBoost: The training data for training standard ANN, MLR, and XGBoost involve using current features X mapped to future target Y. This involves shifting the target by given sequence length. Here we provide more detail on the LSTM model(shown in Figure 1).
LSTM: LSTM networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems. A few properties of LSTM are:

- LSTMs have three types of gates: input gates, forget gates, and output gates that control the flow of information.
- The data feeding into the LSTM gates are the input at the current time step and the hidden state of the previous time step.
- Three fully connected layers with sigmoid activation functions compute the values of the input, forget, and output gates. All values of the three gates are in the range of (0,1).
- The input node is typically computed with a tanh activation function. Intuitively, the input gate determines how much of the input node's value should be added to the current memory cell's internal state.
- The forget gate determines whether to keep the current value of the memory or flush it.
- The output gate determines whether the memory cell should influence the output at the current time step.

LSTM model equations are:

$$f_t = \sigma(W_f \cdot v_t + b_f) \tag{1}$$

$$i_t = \sigma(W_i \cdot v_t + b_i) \tag{2}$$

$$C_t = tanh(W_c \cdot v_t + b_c) \tag{3}$$

$$c_t = f_t * c_{t-1} + i_t * C_t \tag{4}$$

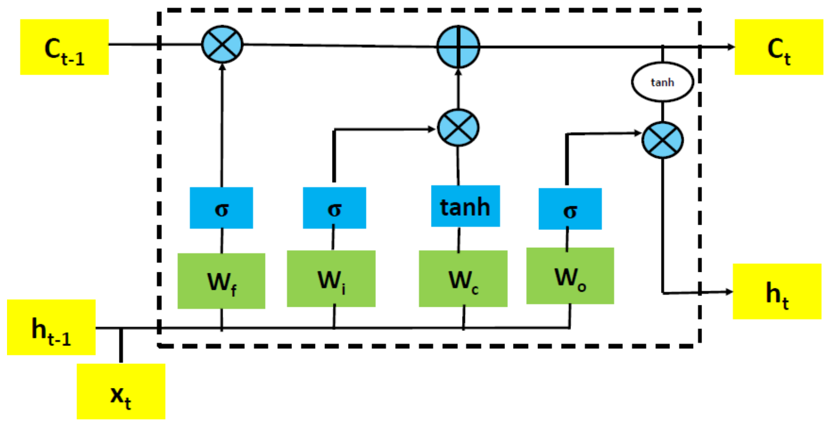


Fig. 1: LSTM

$$o_t = \sigma(W_o \cdot v_t + b_o) \tag{5}$$

$$h_t = o_t * tanh(c_t) \tag{6}$$

We observe that the test accuracy of standard ANN is not good(discussed later) and hence we propose a novel neural network model(Proposed-ANN). The training of this novel neural network model is done on the same data as used for training LSTM. **Proposed-ANN** The proposed-ANN is shown in Figure 2. The approach involves:

1) First passing the past n days' feature data through a dense layer. This generates the latent feature for each day.
2) Second applying a Flatten layer to the latent feature of the past n days and placing them in a sequence.
3) At last bringing the nonlinear interaction between these latent factors for the last n days using a dense layer to predict the future.

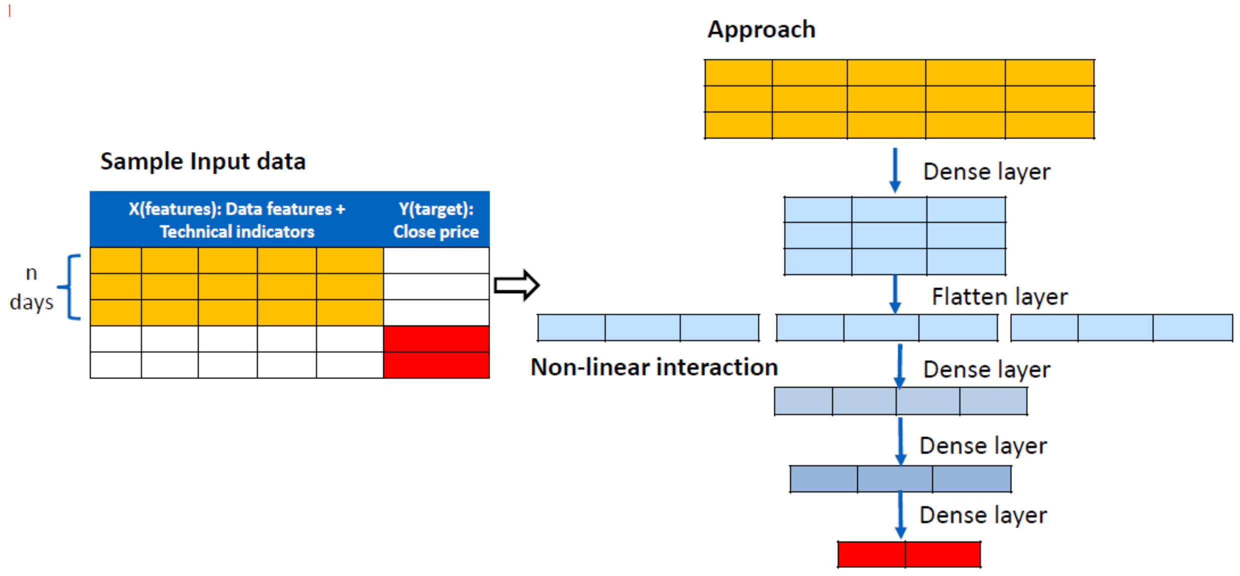


Fig. 2: Proposed-ANN model

Details on classical model architecture:

- LSTM: 2 layers ; Hidden unit: 16 ; Learning rate: 0.001
- Standard ANN: Archiecture $128- > 64- > 32- > 8- > 1$; Activation function: PRelu() ; Learning rate: 0.001
- ARIMA: p=0,q=1,d=0
- Proposed-ANN : $20- > Flattenlayer- > 96- > 32- > 10$

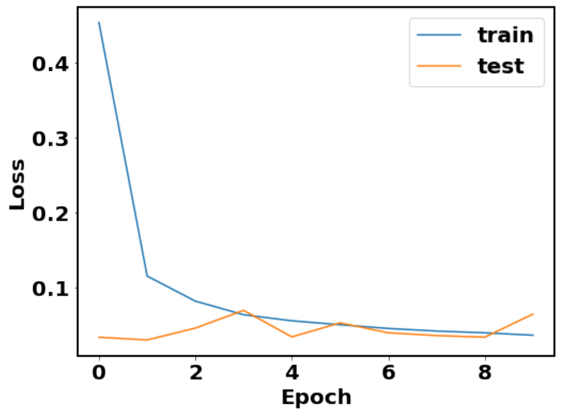
B. *Classical Results*

We used 96% of the data for training (3079 days) and reserved 4% for testing(129 days). Table I shows the MAE and RMSE comparison between different classical models. When using LSTM and proposed-ANN we used the past 60 days('sequence

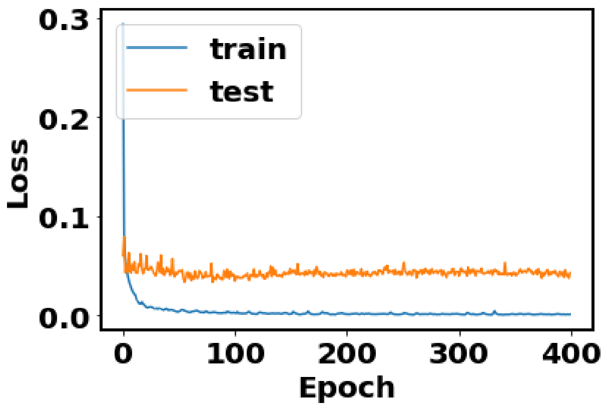
TABLE I: MAE and RMSE error

Model	Train MAE	Test MAE	Train RMSE	Test RMSE
LSTM	83.40	79.15	106.97	95.78
MLR	133.94	121.88	186.69	145.01
XGBoost	10.95	130.23	14.95	157.36
Standard ANN	27.64	116.37	37.38	141.78
Proposed ANN	15.23	101.10	18.20	119.53

length') to make predictions for the next 10 days('window'). When using standard ann, XGBoost, and MLR, we used sequence length as 10, meaning we shifted our target Y by 10 compared to feature X. The LSTM, standard ANN, XGBoost, and proposed ANN give good training accuracy. However, only LSTM and proposed ANN give good test accuracy. The ARIMA model also gave poor accuracy with the future prediction as a constant value (flat line). Thus, we plan to move forward with LSTM and the proposed-ANN. Figure 3 shows the loss curve for LSTM and proposed-ANN. The loss curve of LSTM is shown for 12 epochs and the loss curve of proposed ANN is shown for 400 epochs. For proposed-ann train and test error saturates to a particular value with epochs. Figure 4 shows the test prediction made from 19th Feb 2024 to 1st March 2024 using LSTM and proposed ANN. Both proposed ANN and LSTM performed well with LSTM giving better results.
Few observations :

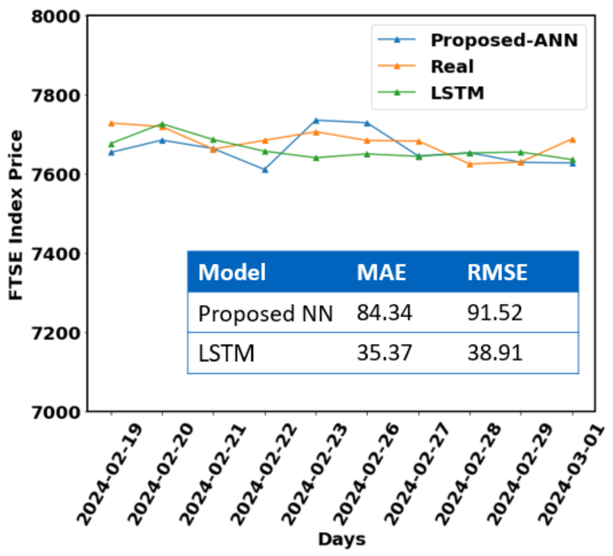


(a) LSTM



(b) Proposed-ANN

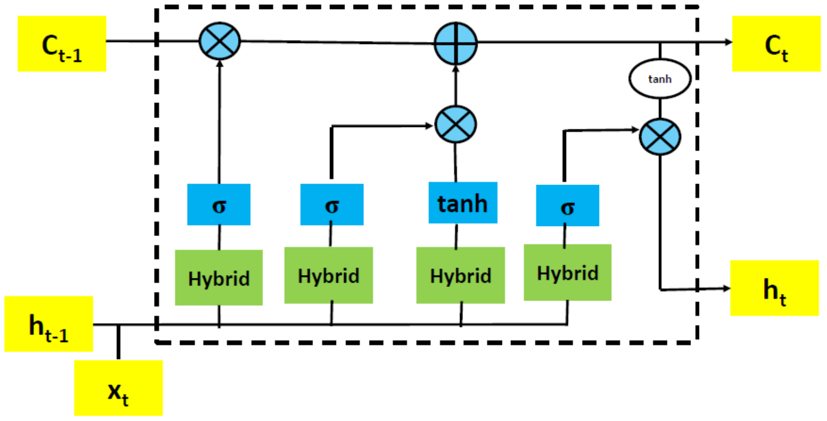
Fig. 3: MSE loss with epochs



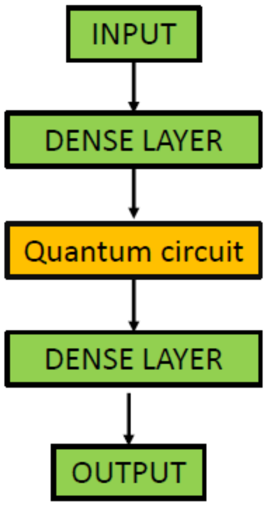
Model	MAE	RMSE
Proposed NN	84.34	91.52
LSTM	35.37	38.91

Fig. 4: Classical model comparison

1) The LSTM test error seems to be less at the start (first few epochs) after which it started oscillating. These oscillations are with large amplitude and thus the loss is not smooth. In turn, the loss of the proposed-ann model is more smooth than that of LSTM.



(a) Quantum LSTM



(b) Hybrid circuit

Fig. 5: Quantum LSTM

2) The test error with LSTM is less if we intentionally stop LSTM at a particular epoch where we observe low test error. However, this seems to be a hit-and-trial procedure. For example, in Figure 4 we intentionally stop LSTM at 9th epoch as it was giving less test error.

V. Quantum Approach

The proposed quantum approach involves creating the Quantum version of the classical approach. Instead of using the full data set we used data from Aug 2013 till 1st March 2024 for quantum analysis and compared with the classical on the similar model settings. We used Quantum LSTM and Quantum Neural network.

A. *Quantum LSTM and Results*

We use the hybrid quantum-classical model of LSTM, which we call Quantum LSTM. This involves extending the classical LSTM into the quantum realm by replacing the classical neural networks in the LSTM cells with a 'hybrid' circuit as shown in Figure 5. The hybrid circuit is the stack of classical dense layer and quantum circuit. Each of the quantum circuits includes:

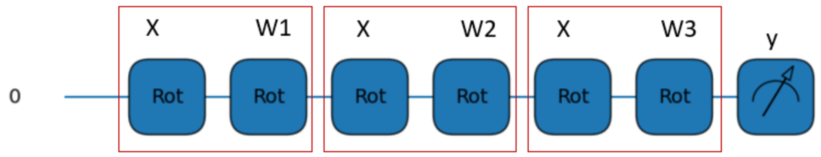


Fig. 6: Single qubit DRC

1) Angle Encoding: Angle encoding serves as a method for integrating classical information into quantum states, enabling quantum computers to effectively handle and manipulate classical data. We used single qubit arbitrary rotation [10] to encode classical data in quantum circuits.
2) Ansatz: We leverage the concept of data reuploading(DRC) [8] for this. DRC involves a single-qubit quantum circuit where data is re-introduced several times in a sequence using arbitrary unitary operations. DRC acts like a universal classifier much like a single hidden-layered Neural Network. An example quantum circuit of DRC is shown in Figure 6. The equation for a single layer is:

$$L(i) = U(X)U(W) \tag{7}$$

Here U is arbitrary single-qubit rotations. If we incorporate data and processing angles in a single step, then we can compactify the circuit and a layer will only need a single rotation to introduce data and tunable parameters.

$$L(i) = U(a_i + w_i \cdot X) \tag{8}$$

Here, each data point can be uploaded with some weight w_i. These weights will play a similar role as weights in artificial neural networks.
We plan to use the concept of multi-qubit DRC as shown in Figure 7. We divide out input data features X in a set of three (X1, X2, etc). Then we encode the set of three data dimensions into multiple qubits. By repeating this embedding of input data and adding successive uploading layers, a highly complex feature Hilbert space can be created in an attempt to improve the learning capacity of the algorithm. We use entanglement between layers to improve the accuracy [8]. Figure 8 shows the actual quantum circuit used in Quantum LSTM. To understand the effect of data reuploading we used both circuit 'DRC': Utilizing data reuploading (Figure 8(a)) and NO-DRC: Not using data reuploading(Figure 8(b)).

3) Measurement from all qubits is the expectation value of Pauli Z.

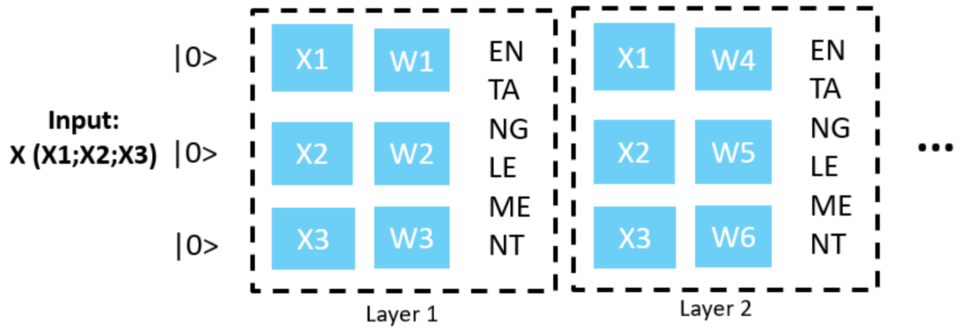
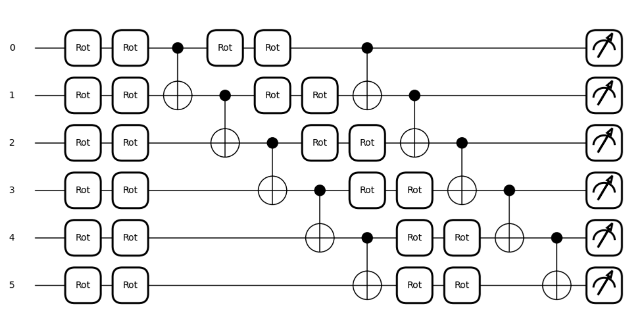
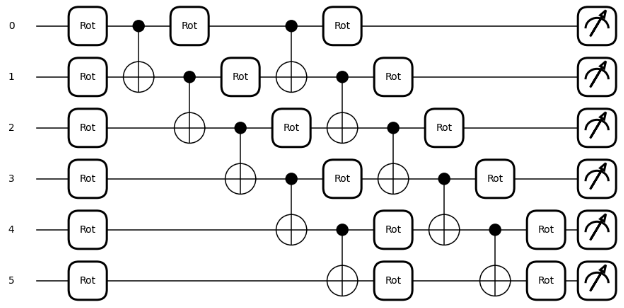


Fig. 7: Conceptual Multi-qubit DRC



(a) Multi-qubit DRC(2 layer)



(b) Multi-qubit NO-DRC(2 layer)

Fig. 8: Quantum circuit

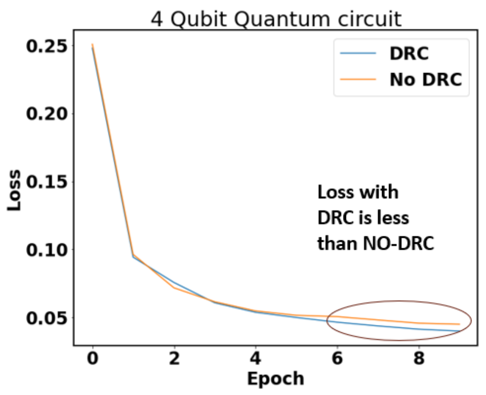
Quantum LSTM result

Here we discuss the result of the quantum approach and how it compares to the classical. The data reuploading concept helps in better learning. This is evident from Figure 9, where the training loss decreases more for DRC than NO-DRC after a certain epoch. The effect of data reuploading is analyzed for both 4-qubit and 6-qubit quantum circuits.
Figure 10 shows the training loss change for both classical and quantum LSTM. The training loss decreases more for quantum lstm at the first few epochs, showing the effectiveness of quantum LSTM at the initial epoch. Then the training loss remains mostly the same as that of classical LSTM. The test loss of classical and quantum is nearly the same and seems to oscillate.

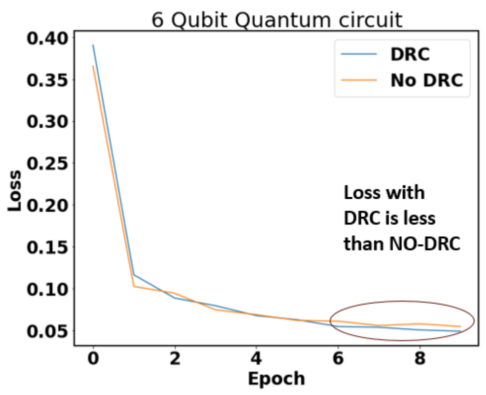
Table II shows the performance of Quantum LSTM and Classical LSTM in terms of error metric and number of taining parameters used in the model. The Quantum LSTM gave nearly the same train error but less test error when trained for the same number of epochs. Also, the number of parameters required for training quantum circuits is apprx. 5 times less than that of Quantum LSTM.

TABLE II: Comparison between classical and quantum

Models	**Performance**				Parameters
	Train MAE	Train RMSE	Test MAE	Test RMSE	
LSTM	74.93	97.74	109.78	130.50	5387
Quantum LSTM	73.81	97.08	76.87	98.09	1143

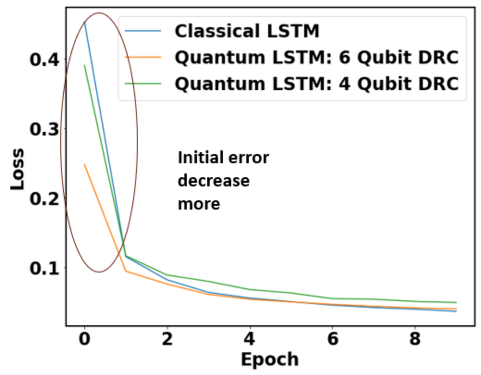


(a) 4 qubit quantum circuit

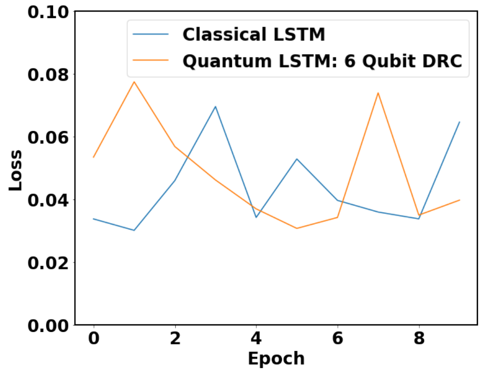


(b) 6 qubit quantum circuit

Fig. 9: Effect of Data reuploading : Training loss vs epochs



(a) Train loss



(b) Test loss

Fig. 10: Comparison between classical and quantum

B. *Proposed Quantum-ANN and Results*

The proposed quantum neural network is the extension of the proposed neural network using quantum layers. The Quantum Neural network as shown in Figure 11 involves:

1) First passing the n days feature data through the Quantum layer(1). This generates the latent feature for each day.
2) Second applying a Flatten layer to the latent feature of the past n days and placing them in a sequence.
3) At last bringing the nonlinear interaction between these latent factors for the last n days through a stack of dense and quantum layer(2) to predict the future

Details of Quantum layer(1) as shown in Figure 12(a) is:

1) Angle Encoding: We used single qubit arbitrary rotation to encode data in quantum computers.
2) Ansatz : Multi-qubit DRC
3) Measurement from all qubits is the expectation value of Pauli Z.

Details of Quantum layer(2) as shown in Figure 12(b) is:

1) Angle Encoding: We used single qubit arbitrary rotation to encode data in quantum computers.
2) Ansatz : The circuit consists of layers of single qubit operations U∈SU(2) and CX [11] entanglements.
3) Measurement from all qubits is the expectation value of Pauli Z.

The training of Quantum LSTM and Quantum neural network involves minimizing the loss function

$$MSE = \frac{1}{n}\sum(y-\hat{y})^2 \tag{9}$$

Here y is the actual value and $\hat{y}$ is the prediction.

Proposed quantum-ann result

Figure 13 shows the training loss change for both classical and quantum proposed ANN. The training loss with a Quantum neural network decreases more than that of the proposed ANN. The number of parameters in the classical proposed neural is 5636 and that in the quantum proposed neural network is 1355, thus almost 5 times less.

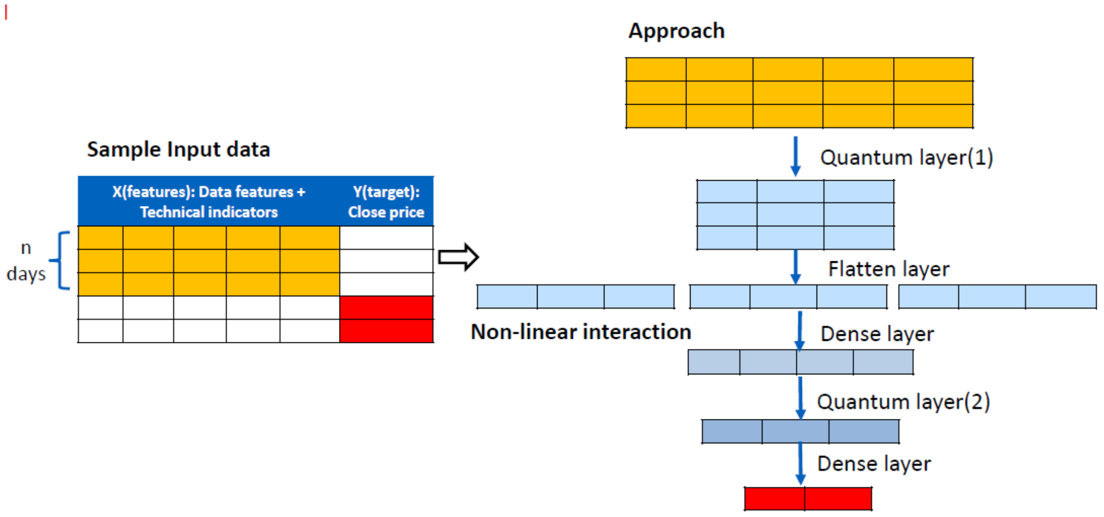
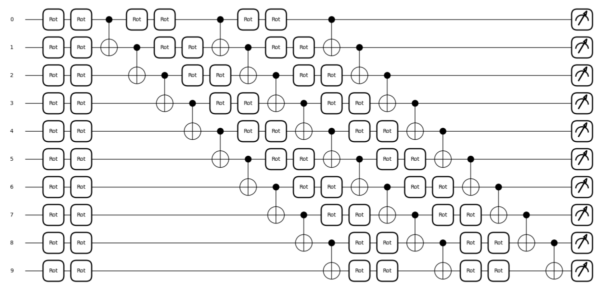
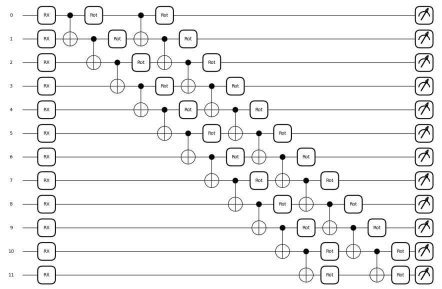


Fig. 11: Proposed quantum neural network

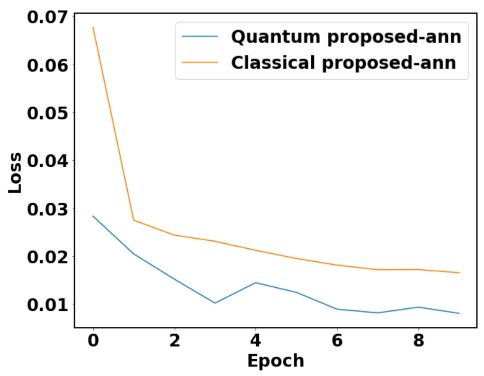


(a) Quantum layer(1)

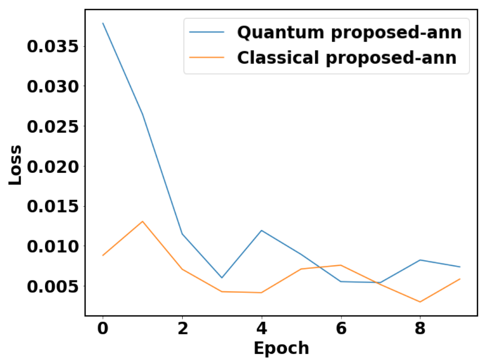


(b) Quantum layer(2)

Fig. 12: Quantum circuit used in proposed quantum neural network



(a) Train loss



(b) Test loss

Fig. 13: Comparison between proposed classical and quantum neural network

VI. Prediction

Here we present the final result on the prediction from 9th to 15th March. We present quantum results for only Quantum LSTM using a 6 qubit DRC circuit.

TABLE III: Prediction from 9th to 15th March

Model	**8th Mar**	**11th Mar**	**12th Mar**	**13th Mar**	**14th Mar**	**15th Mar**
Proposed ANN (Classical)	7715.72	7713.12	7666.08	7766.65	7769.66	7676.48
LSTM (Classical)	7613.73	7622.44	7597.53	7610.93	7623.29	7636.1094
Quantum LSTM	7662.46	7662.88	7645.06	7640.86	7660.05	7624.82

VII. Conclusion

In this report, we presented two approaches 1. Quantum LSTM utilizing the concept of data reuploading and 2 Quantum neural network. We observe that the quantum approach offers the advantage of low qubit count, low overall depth of circuits, learns more information in the first few epochs, comparable and even better results to classical in terms of training and test error, and requires fewer learning parameters compared to classical. The test error being less shows how the quantum approach can capture the market trends. With these approaches, any improvement in the classical method related to data preparation or others can also be made in quantum. The proposed neural network model also showed appreciable results and can be used to make predictions for a large number of days.

References

[1] Moghaddam, Amin Hedayati, Moein Hedayati Moghaddam, and Morteza Esfandyari. ”Stock market index prediction using artificial neural network.” Journal of Economics, Finance and Administrative Science 21, no. 41 (2016): 89-93.

[2] Oukhouya, Hassan, and Khalid El Himdi. ”Comparing machine learning methods—svr, xgboost, lstm, and mlp—for forecasting the moroccan stock market.” In Computer Sciences & Mathematics Forum, vol. 7, no. 1, p. 39. MDPI, 2023.

[3] Xiao, Ruochen, Yingying Feng, Lei Yan, and Yihan Ma. ”Predict stock prices with ARIMA and LSTM.” arXiv preprint arXiv:2209.02407 (2022).

[4] Dharwad, I. I. I. T. ”The Potential of Quantum Techniques for Stock Price Prediction.”

[5] Kaushik, Payal, Sayantan Pramanik, M. Girish Chandra, and C. V. Sridhar. ”One-Step Time Series Forecasting Using Variational Quantum Circuits.” arXiv preprint arXiv:2207.07982 (2022).

[6] Emmanoulopoulos, Dimitrios, and Sofija Dimoska. ”Quantum machine learning in finance: Time series forecasting.” arXiv preprint arXiv:2202.00599 (2022).

[7] Chen, Samuel Yen-Chi, Shinjae Yoo, and Yao-Lung L. Fang. ”Quantum long short-term memory.” In ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 8622-8626. IEEE, 2022.

[8] Pérez-Salinas, Adrián, Alba Cervera-Lierta, Elies Gil-Fuster, and José I. Latorre. ”Data re-uploading for a universal quantum classifier.” Quantum 4 (2020): 226.

[9] de Sá, Cláudio Rebelo. ”Variance-based feature importance in neural networks.” In International Conference on Discovery Science, pp. 306-315. Cham: Springer International Publishing, 2019.

[10] https://docs.pennylane.ai/en/stable/code/api/pennylane.Rot.html

[11] https://docs.quantum.ibm.com/api/qiskit/qiskit.circuit.library.EfficientSU2

Appendix A
Feature Importance

We calculated feature importance to assess the importance of each feature. [9] is used to measure the feature importance and the result is shown in Figure 1. The underlying principle [9] used to calculate feature importance assumes that the more important a feature is, the more the weights connected to the respective input neuron will change during the training of the model. To capture this behavior, a running variance of every weight connected to the input layer is measured during training. Even though we came up with feature importance we preferred to use all the features. Moving on we will use X to denote the features and Y to denote the target (FTSE Close price). The contents...

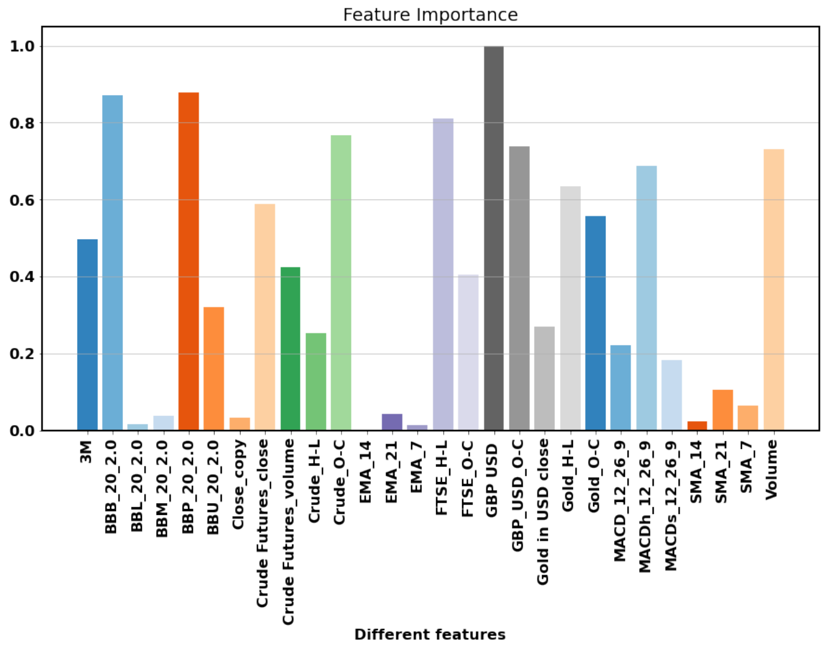


Fig. 14: Feature Importance