**Report of CNN-LSTM Model for stock market prediction:**

**1} Preparing the data**

We prepare the data to be fed into the CNN algorithm in the following manner :

* Using past 60-minute trading data to predict the next 1 minute closing price. This activity can be recursively repeated to predict say, the next 5 candles (1 minute)
* Taking 11 features listed below for each minute-data entry.

**Features:**

1. OP Open price
2. HI High price
3. LO Low price
4. CO Close price
5. Volume
6. %K Stochastic
7. EMA x-days exponential moving average
8. MOM Momentum measures change in stock price over last x days
9. MACD x days moving average convergence and divergence
10. RSI Relative strength index
11. ATR Average true range

**Scaling :**

1. The input matrix is normalized for Gaussian distribution, at μ = 0 and σ=1
2. The target features are scaled by target\_scaling\_factor, with default value 1000

**2} Objective**

Predicting the Target Variable: Close prices

Predicting the next *n* one-minute closing prices.Default, n=5

**3} ML Model - CNN + LSTM**

The details of the implementation are mentioned in the research paper here : [A\_CNN-LSTM-based\_model\_to\_forecast\_stock\_prices (1).pdf](https://drive.google.com/file/d/1hJYiERkCVQEg2axMBXHdRGWP366b6uni/view?usp=sharing)

The network :

1. 32 filters of 1D Convolution on input shape (60,11) :
2. MaxPooling 1D with stride =1 and appropriate padding
3. LSTM layer with 64 units
4. Final Fully connected-layer with 5 outputs

Total number of trainable parameters = 25,541

Parameter information

| **Parameter** | **Value** |
| --- | --- |
| Convolution Activation | tanh |
| LSTM Activation | tanh |
| Default past candles | 60 |
| Default prediction candles | 5 |
| Train size | 0.8 |
| Batch size | 64 |
| Learning rate | 0.001 |
| Optimizer | Adam |
| Max Epochs | 100 |
| Early Stopping | Patience = 4 |

The train ratio should be kept as large as possible (recommended >0.75) for the below two reasons :

1. Time series data is generally sensitive to recent events and hence can cover the fluctuations in near future.
2. The model can learn the periodic occurrences of volatility in the stock.

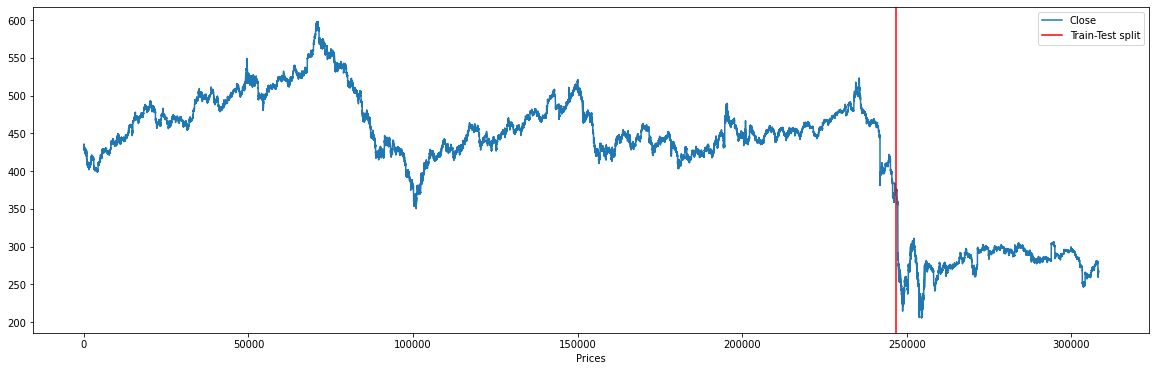
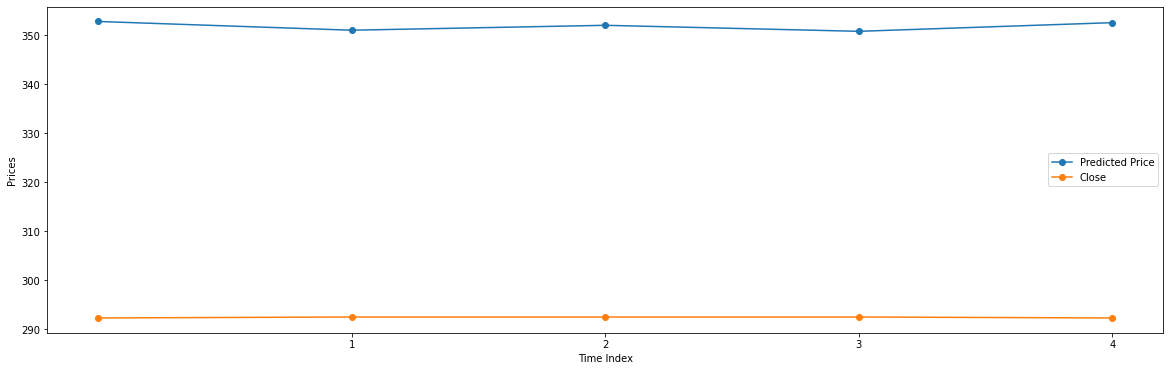
**4} Inference**

We use Mean Absolute Error (MAE) for the evaluation.

Mean Train error : 2.4, early stopping after 34 epochs

Mean Test error: 59.1

This is the difference in prices as predicted by the model and that of true prices



**Momentum accuracy**

We can calculate the momentum accuracy, if the model is able to predict the price change (increase/decrease correctly).

It is roughly 28 % for this model

**5} Conclusions**

Due to the extremely volatile nature of crude near the train test split, and the consideration the Machine Learning model is not able to predict the parameters properly.

Considerations

1. Whenever the market is volatile, it better to take action on the basis of technical indicators like EMA, RSI
2. ML model would perform better in case of a side-ways market as the predictions are much more stable.

**6} Future Works**

Temporal Fusion Transformer is a much more advanced time-series prediction model performed by the Google Research Team which can accurately predict adverse events.

[TFT github](https://github.com/google-research/google-research/tree/master/tft)

**References**

1. <https://machinelearningmastery.com/using-cnn-for-financial-time-series-prediction/>
2. <https://github.com/kritim13/cfa_neural_networks_in_action_2020>
3. <https://www.youtube.com/watch?v=HFu0Jes0x8I>

4. <https://machinelearningmastery.com/cnn-long-short-term-memory-networks/>