Model Documentation

Our Approach: **FINAL RMS=2.8567**

In general, a set of fully connected layers are attached and used to classify the input data. However, all inputs to the fully connected layers are assumed independent of each other, while our features still contain time dependencies. In order to capture temporal relationship that exist in the extracted features, we replace the fully connected layers with LSTM units. The activation of a LSTM unit is fed back to itself and the memory of past activations is kept with a separate set of weights, so the temporal dynamics of our features can be modelled. We use 3 layers LSTM in our work, and find using more units can lead to overfitting.
In our model we predict the stock price 30 seconds ahead by actually finding the price movement happening after 30 secs. We labelled it into 3 classes.

1) +1 representing tick up
2) 0 representing no significant change
3) -1 representing tick down

The last output layer uses a softmax activation function to classify into 3 classes and hence the final output elements represent the probability of each price movement class (tick up, tick down or no change) at each time step.
For prediction of predMid, we use:

$$predMid = Mid\,price + class * tick\_size$$

Labelling

Because financial data is highly stochastic, if we simply compare pt and pt+k to decide the price movement, the resulting label set will be noisy. We adopt the idea of introducing a smoothed labelling method. First, m− denotes the mean of the previous k mid-prices and m+ denotes the mean of the next k mid-prices:

$$m_{-}(t) = \frac{1}{k}\sum_{i=0}^{k} p_{t-i}$$

$$m_{+}(t) = \frac{1}{k}\sum_{i=1}^{k} p_{t+i}$$

where pt is the mid-price defined in Equation (1) and k is the prediction horizon (k = 300 signifying 30 seconds on average). Then, we use m− and m+ to define the direction of price movement (lt) at time t by:

$$l_t = \begin{cases} -1, & \text{if } m_-(t) > m_+(t) \cdot (1+\alpha) \\ 1, & \text{if } m_-(t) < m_+(t) \cdot (1-\alpha) \\ 0, & \text{otherwise} \end{cases}$$

where the threshold α determines the smallest change in price that must occur for the mid-price to be considered upward (+1) or downward (−1). We choose the threshold α such that all the classes are balanced for each instrument during the training period. This can be understood as adjusting for the average volatility of a given instrument, i.e. if a stock is more volatile we require a larger upwards (downwards) price move to classify it as a +1 (−1).

Feature Description

Name	Formula (Definition)	Remarks
spread	ask - bid	Indicates the probability of a new order arrival and the unpredictability of mid price. A higher spread points to a higher probability for a new order arrival, but gives no information about the direction of mid price change.
arrival_rate	$\frac{change\ in\ bsize/asize}{\Delta t}$	Indicates how fast (or whether) mid price changes.
imbalance	$\frac{\left(\sum bsize - \sum asize\right)}{\left(\sum bsize + \sum asize\right)}$	Indicates the direction of mid price change. A positive imbalance indicates more demands and a consequent increase in mid price. We have accumulated bsize and size over 1s to calculate an integrated imbalance to account for a 'settling effect'
trade	0 - No Trade 1 - Trade	Indicates whether a trade happened. Trade will affect mid price in a way dictated by whether the trade was a buy or a sell
buy	1 - Trade price matched existing ask price 0 - Trade price matched existing bid price	Indicates whether the trade was a buy or a sell. A buy will reduce asize and indicate more demand pushing the mid price up. A sell will indicate more supply and will push mid price down
size	Size of trade	Indicates the magnitude of the effect a Trade will have on the mid price
mid	The mid price	Feeding mid price (and its history) gives an indication about the expected short term trend.

Model Description:

Optimizer used : Adam optimizer
Loss function used : Categorical cross entropy
Sequence length or history feeded : 35 timesteps

Layer 1:
Type: LSTM
No: of neurons = 30
Dropout rate = 0.2

Layer 2:
Type: LSTM
No: of neurons = 40

Layer 3:
Type:LSTM
No: of neurons = 40
Dropout rate = 0.2

Layer 4:
Type : Dense
No: of neurons = 3
Activation = Sigmoid

Getting started with running the model

We have some important files
1 - model.ipynb - Main pipeline to be executed
2 - core/model.py - contains class model which is our LSTM model
3 - core/data_processor.py - data_loader class for ease of loading big data
4 - config.json - contains all the important hyper parameters for model
5 - model.h5 - contains the current model weights we are using for replicability