Attention based DeepLOB for Prediction in Limit Order Books

Pulyapudi Aditya (B201015EC), Yalavarthi Roshan Sai(B201022EC),
D Saicharan (B200970EC), Chevula Sai Anivesh (B201026EC)
Guide : Dr. Anup Aprem
Group :- 49

***Abstract*—Predicting price movements from Limit Order Books (LOB) has historically been a challenging task due to the inherent complexity and high-dimensional nature of the data. In this project, with the base model as DeepLOB [1], our approach involves the incorporation of an attention layer at the end of the architecture.**

Contributions : Introducing Attention Mechanism [2] to the Model after LSTM in DeepLOB [1].

The resulting accuracy of DeepLOB and Attention to DeepLOB (ADLOB) are 66.42 percent and 74.41 percent respectively, an increase in accuracy by 9 percent.

***Index Terms*—1. Introduction 2. Methodology 3. Attention 4. Dataset 5. Conclusion**

I. INTRODUCTION

THE financial landscape is complex, and accurately predicting market movements is crucial for smart decision-making. Traditional methods struggle with the intricacies of financial time series data, especially in the context of limit order book (LOB) information.

This report introduces a deep learning model called DeepLOB with Attention, designed specifically for the challenges of financial forecasting. The model combines convolutional neural networks (CNNs) and long short-term memory (LSTM) networks to capture temporal dependencies and spatial patterns in financial time series. An attention mechanism is included to enhance the model's capabilities, allowing it to focus on key elements of the input data.

The goal of this project is to develop more robust and accurate forecasting models in finance.

II. DATASET

We utilized the FI-2010 dataset, employing normalized z-score values for a feature set encompassing ask price,bid price, ask volume, and bid volume with 10 orders on each side of the Limit Order Book . Our data analysis focused on standard trading hours, specifically excluding auction periods. Our dataset comprises both the feature set and corresponding labels. Rows 1 to 144 contain the feature data, where the initial 40 entries represent LOB points. Each LOB point encompasses 10 levels for 'ask price,' 'bid price,' 'ask volume,' and 'bid volume,' in addition to 104 manually-engineered features. Furthermore, there are five prediction horizons (k=10, 20, 30, 50, 100). we used the most recent 100 states of the LOB as input to our model and trained it on label corresponding to prediction horizon k=10. In terms of classification, labels are available from row 145 onwards, with '1' indicating an up-movement, '2' signifying a stationary condition, and '3' representing a down-movement [1].

Specifically, a single input is defined as $X = [x_1, x_2, \ldots, x_t, \ldots, x_{100}]^T \in R^{100\times 40}$, where $x_t = \left[p_a^{(i)}(t), v_a^{(i)}(t), p_b^{(i)}(t), v_b^{(i)}(t)\right]_{i=1}^{n=10}$, . Here, $p^{(i)}$ and $v^{(i)}$ denote the price and volume size at the i-th level of a limit order book.

From normalized limit order data , mid-price is used to find the price moment

$$p_t = \frac{p_a^1(t) + p_b^1(t)}{2} \quad (1)$$

$$m_+(t) = \frac{1}{k}\sum_{i=1}^{k} p_{t+i} \quad (2)$$

m+ denote the mean of the next k mid-prices and pt is the mid-price defined in Equation (1) and k is the prediction horizon.

$$l_t = \frac{m_+(t) - p_t}{p_t} \quad (3)$$

The labels are then decided based on a threshold
Labeling:

$l_t > \alpha : +1(\text{up})$
$l_t < -\alpha : -1(\text{down})$
else:$0(\text{stationary})$

III. METHODOLOGY

The core of the proposed DeepLOB with Attention model lies in its architecture, combining Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM) networks, and an Attention Mechanism .

Convolutional Layer :- Convolutional layers use a small kernel to systematically analyze input data, emphasizing the organization of limit order book information. Specifically , the Convolutional Layer processes the input sequence, which represents the 100 most recent updates of an order book, with each update having 40 features per timestamp.

The initial convolutional filter has a size of (1 × 2), indicating that it analyzes two adjacent features at a time. The stride of (1 × 2) is used to control how the filter moves across the input. This is crucial for consolidating information between price and volume at each order book level.Convolutional layers exhibit parameter sharing, a property that helps minimize parameter estimation and reduce the risk of overfitting. Stride plays a crucial role in this by ensuring different parameters for distinct dynamics.We note that very short time-dependencies are already captured in the convolutional layer which takes "space-time images" of the LOB as inputs[1].

Inception Module:- The Inception Module enhances the model's adaptability by combining different convolutions, similar to using various moving averages in technical analysis for observing time-series momentum. Unlike manually setting weights, Inception Modules let the model learn optimal weights during training[1].

In our method, we use 1 × 1 convolutions(Bottle-Neck Layer) to break down the input into simpler forms. These forms go through transformations with filters (3 × 1 and 5 × 1), and the results are combined. In the Inception Module, we apply a max-pooling layer, with 1 × 1 convolutional layer featuring filters. This approach uses small neural networks to capture non-linear data properties, improving prediction accuracy[1].

LSTM:- LSTM units have a feedback mechanism, preserving past activations' memory. This helps model temporal dynamics in our features. Our implementation involves 64 LSTM units, requiring around 60,000 parameters—a significant 10x reduction if we use fully connected layers. The final output layer uses softmax activation, with output elements representing the probability of each price movement class at each time step[1].

IV. Attention

The attention layer acts like a spotlight, allowing the model to focus on specific parts of the input sequence when making predictions. Rather than treating all elements equally, the attention mechanism assigns varying levels of importance to different parts of the input sequence. This enables the model to pay more attention to relevant information, improving its ability to capture complex patterns and relationships in the data. Essentially, the attention layer helps the model "attend" to the most informative elements, contributing to more accurate predictions in our project [2].

Each query vector, (q) is matched against a database of keys to compute a score value. This matching operation is computed as the dot product of the specific query under consideration with each key vector,k_i .

$$e_{q,k} = q \cdot k_i \quad (4)$$

The scores are passed through a softmax operation to generate the weights:

$$aq, k = \text{softmax}(e_{q,k}) \quad (5)$$

The generalized attention is then computed by a weighted sum of the value vectors, V_{k_i} where each value vector is paired with a corresponding key ,allowing the model to determine the importance or weight of each element in the input sequence.:

$$\text{attention}(q, K, V) = \sum_{i=1}^{n} aq, k_i V_{k_i} \quad (6)$$

V. Simulation Results

The simulation was done in Google Colab platform with GPU backend on a shrunk down dataset for 50 epochs . And offline in JupyterNotebook on the original dataset for 200 epochs. The models (DeepLOB and ADLOB) were trained with the parameters and settings mentioned.

- Learning rate for Adam Optimizer: 0.0001
- Batch Size: 128

We experimented with 2 different models and had to tune the parameter, Look-back-timestep (T) , and the results of the model i.e.,Classification Report and Accuracy Score for DeppLOB and ADLOB as shown in TABLE.1,2 we observe increase in accuracy score from 77.01 to 79.05 (2 percent increase). Fig.1 depicts the Confusion Matrix without attention (DeepLOB) and with attention (ADLOB) respectively . As the dataset is small the increase in accuracy in not high (2 percent). Hence, we trained the model on the complete dataset offline in Jupyter Notebook (GPU: NVIDIA-RTX6000) for 200 epochs. TABLE.3,4 shows increase in accuracy from 66.42 percent to 74.41 percent (9 percent increase). Fig.2 depicts the Confusion Matrix without attention (DeepLOB) and with attention (ADLOB) respectively .

VI. Conclusion

In conclusion, the DeepLOB with Attention model presented is and promising approach for predicting financial market movements based on limit order book (LOB) data. The amalgamation of convolutional neural networks (CNNs), long short-term memory (LSTM) networks, and an Attention mechanism demonstrates the model's capability to capture both spatial and temporal dependencies inherent in financial time series.This lead to an increase in accuracy from 66 to 74 percent (9 percent increase)

While the model shows promise, it is essential to acknowledge its potential areas for improvement. Future research may focus on exploring additional data sources, refining model architecture like investigating more advanced attention mechanisms or adding Self Attention layer and converting it to Transformer based model (TransLOB) [3].

References

[1] Z. Zhang, S. Zohren and S. Roberts, "DeepLOB: Deep Convolutional Neural Networks for Limit Order Books," in IEEE Transactions on Signal Processing, vol. 67, no. 11, pp. 3001-3012, 1 June1, 2019, doi: 10.1109/TSP.2019.2907260.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, "Attention Is All You Need" in Advances in Neural Information Processing Systems 12 Jun 2017

[3] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, Igor Mordatch "Decision Transformer: Reinforcement Learning via Sequence Modeling "

VII. APPENDIX

Simulation Results with shrunken dataset for 50 epochs

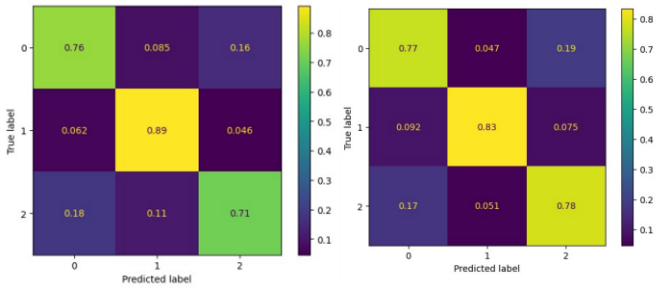

Fig-1:- Confusion Matrix for DeepLOB and ADLOB respectively

Label	Precision	recall	f1-score
0	0.7919	0.7599	0.7755
1	0.7793	0.8911	0.8315
2	0.7651	0.7119	0.7376
accuracy score			0.7701

TABLE I
RESULTS FOR T=10 ON SHRUNKEN DATA WITHOUT ATTENTION (DEEPLOB)

Label	Precision	recall	f1-score
0	0.7874	0.7675	0.7773
1	0.8653	0.8332	0.8490
2	0.7383	0.7822	0.7596
accuracy score			0.7905

TABLE II
RESULTS FOR T=10 ON SHRUNKEN DATA WITH ATTENTION (ADLOB)

Final Simulation Results with complete dataset in JupyterNotebook (NVIDIA-RTX6000) for 200 epochs

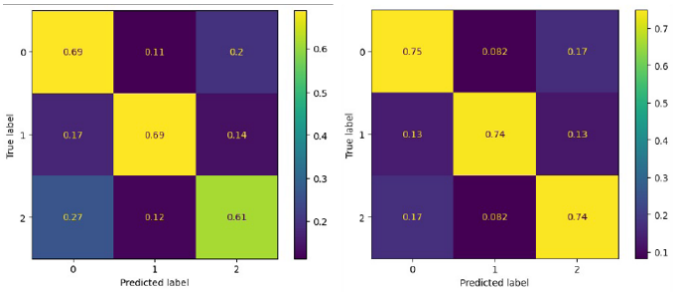

Fig-2:- Confusion Matrix for DeepLOB and ADLOB respectively

Label	Precision	recall	f1-score
0	0.6234	0.6884	0.6543
1	0.7549	0.6862	0.7189
2	0.6222	0.6134	0.6178
accuracy score			0.6642

TABLE III
FINAL RESULTS FOR T=100 WITHOUT ATTENTION (DEEPLOB)

Label	Precision	recall	f1-score
0	0.7198	0.7478	0.7335
1	0.8253	0.7401	0.7804
2	0.6951	0.7447	0.7191
accuracy score			0.7441

TABLE IV
FINAL RESULTS FOR T=100 WITH ATTENTION (ADLOB)

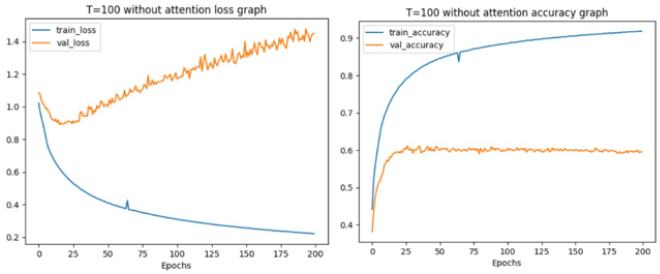

Fig-3:- Accuracy and Loss Graphs for DeepLOB

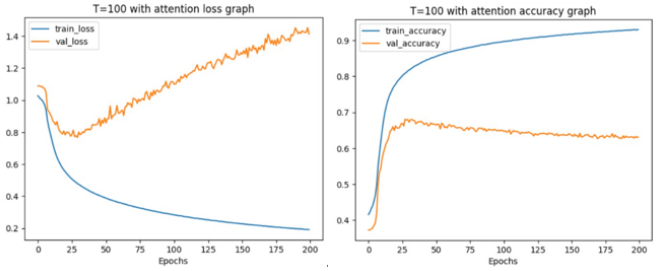

Fig-4:- Accuracy and Loss Graphs for ADLOB

Comparison of validation accuracy of DeepLOB and Attention in DeepLOB (ADLOB)

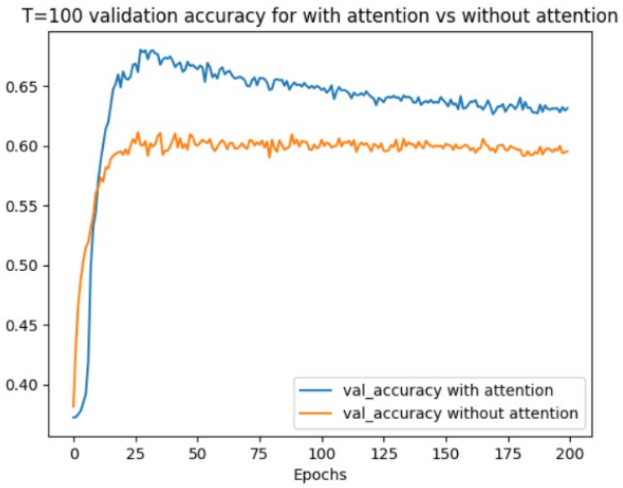

Fig-5