Attention based DeepLOB for Prediction in Limit Order Books

Major Project Report

Submitted by

Pulyapudi Aditya (B201015EC)
Yalavarthi Roshan Sai (B201022EC)
D Saicharan (B200970EC)
Chevula Sai Anivesh (B201026EC)

Under the guidance of

Dr.Anup Aprem

In partial fulfillment for the award of the Degree of

BACHELOR OF TECHNOLOGY
IN
ELECTRONICS AND COMMUNICATION ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY, CALICUT
NIT CAMPUS P.O., CALICUT
KERALA, INDIA 673601.

NATIONAL INSTITUTE OF TECHNOLOGY, CALICUT

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

*This is to certify that the major project report entitled "**Attention based DeepLOB for Prediction in Limit Order Books**" is a bonafide record of the Project done by **Pulyapudi Aditya** (B201015EC), **Yalavarthi Roshan Sai** (B201022EC), **D Saicharan** (B2009708EC) and **Chevula Sai Anivesh** (B201026EC) under our supervision, in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Electronics and Communication Engineering** from **National Institute of Technology Calicut**, and this work has not been submitted elsewhere for the award of a degree.*

Dr. Anup Aprem
Assistant Professor Grade I,
NIT Calicut

Dr. Jaikumar M.G
HOD ECED,
NIT Calicut

Place: Calicut

Date: 02 - 05 - 2024

ACKNOWLEGEMENT

We would like to take this opportunity to express our deepest gratitude to everyone who helped us in completing the project. We have great pleasure in expressing my gratitude and obligations to Dr. Anup Aprem, Assistant Professor Grade I, Department of ECE, for his valuable guidance, and suggestions to make this work a success. We express our gratitude to Dr. Jaikumar M. G, Head of the Department, Department of ECE, for his wholehearted cooperation and encouragement. We also acknowledge our gratitude to other faculty members in the Department of Electronics and Communication Engineering, family, and friends for their wholehearted cooperation and encouragement.

Pulyapudi Aditya

Yalavarthi Roshan Sai

D Saicharan

Chevula Sai Anivesh

May 2024

National Institute of Technology Calicut

ABSTRACT

The financial landscape is complex, and accurately predicting market movements is crucial for smart decision-making. Particularly in the case of high-frequency transactions involving Limit Order Books (LOB). Traditional methods struggle with the intricacies of financial time series data, especially in the context of limit order book (LOB) information.

In response to these challenges, deep learning models can be used to address the unique demands of financial forecasting. DeepLOB [1] is one of these deep learning models. It is very good at finding complex patterns and temporal dependencies in financial time series data because it uses both convolutional neural networks (CNNs) and long short-term memory (LSTM) networks for short-term and long-term dependencies respectively, as well as the Inception module [1] to deal with non-linear and different timepatterns.

In this project, we introduce a novel deep learning model called DeepLOB with Attention (ADLOB), designed specifically for the challenges of financial forecasting with high-frequency data (LOB's). The novel addition to DeepLOB by us is a weighted attention mechanism [2] to enhance the model's capabilities, allowing it to focus on key elements and patterns of the input data and utilize those patterns to make a more accurate decision on the movement of the mid-price.

We further tried variations of deep learning methods and compared their results. To measure the capabilities of the deep learning models, we applied backtesting procedures to the models and observed the results of backtesting upon a capital investment.

CONTENTS

List of Figures **6**

List of Tables **7**

1 Introduction **10**

1.1 Limit Order Books (LOB) 10

1.2 Motivation 12

1.3 Problem Statement 12

2 Literature Survey **13**

2.1 Deep Learning in Financial market Prediction 14

2.2 DeepLOB for Limit Order Books 14

2.3 Attention Mechanism 15

3 Theory **17**

3.1 Background 17

3.1.1 Stock Market and Futures 17

3.1.2 Stock Market 18

3.1.3 Limit Order Books 19

3.2 Requisite Deep Learning Concepts 19

3.2.1 Convolutional Neural Network 19

3.2.2 Inception Module 22

3.2.3 Recurrent Neural Network 23

3.2.4 Attention Meachanism 26

4 Methdology and Implementation **27**

4.1 Data Set and Preprocessing 27

4.1.1 FI-2010 dataset . . . 27
4.1.2 Dataset Description and Labelling . . . 27
4.1.3 Preprocessing . . . 29
4.2 DeepLOB (Deep Learning for Limit Order Books) . . . 30
4.2.1 Convolutional Layer . . . 30
4.2.2 Inception Module . . . 31
4.2.3 LSTM (Long Short-Term Memory) . . . 32
4.3 ADLOB (Attention in DeepLOB for Limit Order Books) . . . 33
4.3.1 Attention Mechanism . . . 34
4.4 Variations to ADLOB . . . 35
4.5 Backtesting on Deep learning Models . . . 36
4.5.1 Backtesting on Capital Investment . . . 37

5 Results and Discussion **38**
5.1 Testing LOB Models . . . 38
5.1.1 Performance Metrics for Backtesting . . . 41
5.1.2 Results of Backtesting . . . 42
5.1.3 Backtesting on Capital Investment . . . 43

6 Conclusion **45**
BIBLIOGRAPHY . . . 45

List of Figures

1.1 Limit Order Book (LOB) 11

3.1 Diagram of CNN 22
3.2 An exemplary Architecture of Inception Module 23
3.3 Diagram of Recurrent Neural Network 24
3.4 Diagram of LSTM 25

4.1 Model Diagram of DeepLOB 30
4.2 Inception Module 32
4.3 Model Diagram of ADLOB 33
4.4 Weighted Attention Mechanism Diagram 35
4.5 Model Diagrams of Bi-ADLOB and Enc-Dec-ADLOB, respectively . . 36

5.1 Confusion Matrices of LOB Models from Top to Bottom and Left to Right DeepLOB , ADLOB , Bi-Directional ADLOB and Encoder Decoder ADLOB 39
5.2 UI for Backtesting on Capital Investmnet 43
5.3 Graphs of Current Assest value at two different time steps 44
5.4 Graphs of Current Assest value at two different time steps 44

List of Tables

2.1 Comparative Study Table - Literature Survey. 16

5.1 Accuracy Scores of LOB models on limited data 38
5.2 DeepLOB Classification Report 40
5.3 ADLOB Classification Report 40
5.4 Bi-ADLOB Classification Report 40
5.5 Encoder Decoder ADLOB Classification Report 40
5.6 Backtesting Results for DEEPLOB 42

List of Publications

Attention based DeepLOB for Prediction in Limit Order Books

(Undergoing)

List of Abbreviations

ML	Machine Learning
DL	Deep Learning
AI	Artificial Intelligence
LOB	Limit Order Books
CNN	Convolutional Neural Network
LSTM	Long Short-Term Memory
RNN	Recurrent Neural Network
ADLOB	Attention in DeepLOB
Bi-ADLOB	Bidirectional Attention in DeepLOB
Enc-Dec-ADLOB	Encoder-Decoder Attention in DeepLOB
NLP	Natural Language Processing
UI	User Interface

CHAPTER 1

Introduction

The project aims to enhance mid-price movement prediction in limit order book (LOB) data with attention-based LOB models.

By incorporating attention mechanisms, we aim to enhance prediction accuracy. By evaluating the performance of deep learning LOB models using historical LOB data. Comparing accuracy scores is part of the project's goal to find the best model variations for predicting mid-price movements.

Backtesting procedures will be conducted to assess the profitability and robustness of the developed models in simulated trading scenarios, with the ultimate goal of maximizing trading profits in high-frequency trading environments.

1.1 Limit Order Books (LOB)

Limit order books (LOBs) are essential components of financial markets, offering transparency by organizing buy and sell orders for a security according to price levels. These orders represent the collective intentions of market participants, delineating the range of prices at which they are willing to transact. LOBs play a crucial role in price discovery, providing real-time insights into supply and demand dynamics.

The buy side of an LOB lists bids from highest to lowest prices, reflecting buyers' willingness to purchase at various levels. **Conversely, the sell side displays asks from lowest to highest prices**, indicating sellers' desired prices for their holdings. Market depth, derived from the volume of orders at each price level, signifies the liquidity of the market, with deeper markets associated with higher liquidity and narrower bid-ask spreads.

Order execution within an LOB follows predefined rules, with market orders

matched against the best available limit orders based on price and time priority. This process ensures fair and efficient trade execution while facilitating price discovery. Additionally, LOBs enable market participants to gauge sentiment, identify trends, and anticipate price movements by monitoring changes in the order book.

In summary, limit order books are foundational structures in financial markets, providing transparency, liquidity, and efficiency. They serve as vital tools for price discovery, order execution, and market analysis, empowering participants to make informed trading decisions and navigate market complexities effectively. The figure below represents a Limit Order Book (LOB), with the left side indicating limit buy orders and the right side indicating sell orders relative to the mid-price.

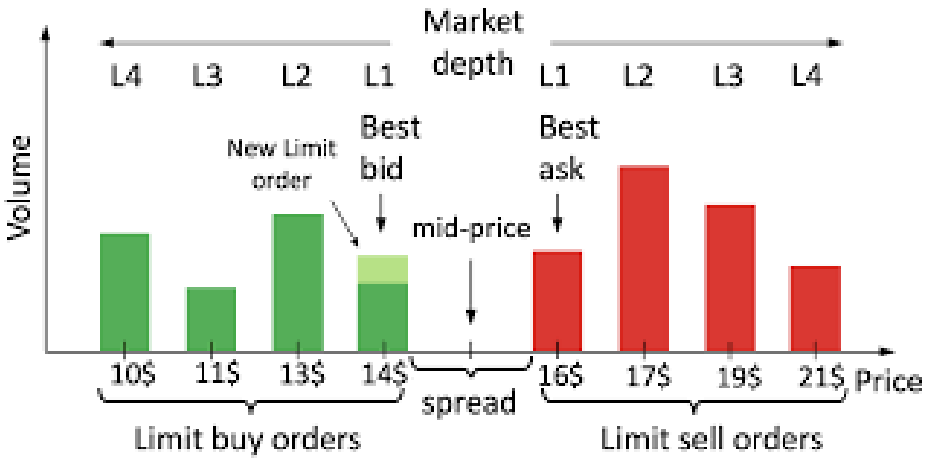

Figure 1.1: Limit Order Book (LOB)

1.2 Motivation

The motivation behind this project lies in addressing the need for advanced predictive tools in high-frequency trading.

Traditional methods often struggle to capture the complexities of limit order book (LOB) data, hindering accurate mid-price movement predictions. Leveraging deep learning models like ADLOB, specifically designed for LOB data, the project aims to enhance profitability and risk management in fast-paced markets.

Backtesting ensures the reliability of the models, paving the way for their practical implementation in trading strategies.

1.3 Problem Statement

Problem Statement : Prediction of the movement of the mid-price of a stock in limit order books using deep learning models.

- Utilize advanced deep learning models such as DeepLOB [1], ADLOB, etc. to predict price movements accurately in financial markets, especially in high-frequency transactions using Limit Order Book (LOB) data.
- Optimize profits by using predictions from deep learning models to execute strategic trades efficiently in the dynamic landscape of financial markets.

CHAPTER 2

Literature Survey

The study of predicting price movements in financial markets using deep learning techniques has gained significant attention in recent years due to its potential applications in algorithmic trading and financial decision-making [9,10,11]. Research on predicting price movements in financial markets has a rich history within the financial literature, spanning numerous studies [4]. While there is ongoing debate regarding market efficiency, consensus suggests that financial markets exhibit some degree of predictability.

Two primary approaches to forecasting financial time-series data have emerged:

1. Statistical parametric models [6] and
2. Data-driven machine learning methods [5].

Traditional statistical methods often assume a parametric process underlying the time-series data. The Traditional methods like Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) have been used [6]. The FI-2010 dataset [1] yields noticeably improved results when the Bag-of-Features model (BoF)—which is represented as a neural layer in the work of [6] is trained using backpropagation approach. But, these static approaches may not fully optimize the model's objective.

However, it is widely acknowledged that stock returns exhibit complex and nonlinear behavior. Machine learning techniques offer the advantage of capturing arbitrary nonlinear relationships without relying on prior knowledge of the data.

2.1 Deep Learning in Financial market Prediction

In recent years, there has been a surge of interest in predicting price movements using machine learning algorithms applied to limit order book (LOB) data . Nevertheless, only a small number of published publications [12 ,13] have used CNNs to analyse financial microstructure data, and the CNN architectures that are now in use are not very advanced and have not undergone sufficient research.The incorporation of feature extraction and representation as a component of the learnt model is arguably one of the main advances of contemporary deep learning.

The Long Short-Term Memory (LSTM) [16] was originally proposed to solve the vanishing gradients problem of recurrent neural networks, and has been largely used in applications such as language modelling and sequence to sequence learning. Unlike CNNs which are less widely applied in financial markets, the LSTM has been popular in recent years, [14,15] all utilising LSTMs to analyse financial data. Specifically, [17] tests a multi layer LSTM model using limit order data from 1000 stocks. Their findings demonstrate a consistent out-of-sample prediction accuracy over time, suggesting the possible advantages of deep learning techniques.

2.2 DeepLOB for Limit Order Books

One of the seminal works in this field involving CNNs and LSTMs combined is the DeepLOB model proposed by Zhang et al. [1].

This model introduced the combination of convolutional neural networks (CNNs) and long short-term memory (LSTM) networks to capture spatial and temporal dependencies in high-frequency LOB data. It introduces the first hybrid deep neural network to predict stock price movements using high frequency limit order data. Unlike traditional hand-crafted models, where features are carefully designed, it utilises a CNN and an InceptionModule to automate feature extraction and use LSTM units to capture time dependencies.

The DeepLOB model demonstrated promising results in predicting price movements from LOB data, paving the way for further research in this area.

2.3 Attention Mechanism

Attention mechanisms have also been widely studied in the context of sequence modeling and natural language processing tasks. The "Attention Is All You Need" paper by Vaswani et al. [2] introduced the Transformer architecture, which relies entirely on attention mechanisms for capturing dependencies between input and output sequences. This paper revolutionized the field of sequence modeling and inspired the incorporation of attention mechanisms into our deep learning model for financial forecasting.

In the context of sequence prediction, the decision transformer model [3] employs a causally masked transformer to capture the optimal action to take at different time steps in financial data. This work demonstrated the effectiveness of attention mechanisms in improving the accuracy of financial forecasting models. This work demonstrated the effectiveness of attention mechanisms in improving the accuracy of financial forecasting models.

The ***DeepLOB with Attention model proposed in the paper*** builds upon existing research by combining CNNs, LSTMs, and attention mechanisms to achieve improved performance in forecasting price movements from LOB data.

To the best of our knowledge, there is no existing research that combines Convolutional Neural Networks (CNNs) with Long Short-Term Memory (LSTM) networks, followed by an Attention Mechanism, to anticipate stock price movements. This study is the first comprehensive investigation to deploy an Attention-based Deep Learning model for predicting financial prices.

The literature survey reflects a diverse range of approaches and methodologies in the quest for an accurate prediction of the price of a stock. Ongoing efforts aim to enhance increase the prediction accuracy of the deep learning model and optimize decision-making processes using decision transformers to maximize the profit from the deep learning model's predicted price.

Table 2.1: Comparative Study Table - Literature Survey.

Approach	**Features**	**Limitations**
Statistical Parametric Models	- Assume a parametric process underlying the time-series data. - Utilizes traditional methods like Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA).	- May not fully optimize the model's objective. - Static approaches may not fully capture the complex and nonlinear behavior of stock returns
CNNs	- Capture short term dependencies in high-frequency LOB data based on the size of the filter in convolution layer. - Combined with LSTM for improved performance in predicting price movements.	- Limited research and advancement in CNN architectures for financial microstructure data.
LSTMs	- Solve the vanishing gradients problem of recurrent neural networks. - Popular for analyzing financial data due to their effectiveness.	- Training times tend to be considerably prolonged when dealing with extensive sequences in comparison to both Recurrent Neural Networks (RNNs) and Transformer architectures.
Attention Mechanisms	- Revolutionized sequence modeling by capturing dependencies between input and output sequences. - Effectively improve the accuracy of financial forecasting models.	- Not extensively explored in financial markets.
Deep Learning Model Combination (**DeepLOB**)	- Combining CNNs, and LSTMs achieves improved performance in forecasting price movements from LOB data. - First comprehensive investigation into combining CNNs and LSTMs with inception module for predicting financial prices.	- Lack of existing research (to the best of our Knowledge) on combining CNNs with LSTMs followed by an Attention Mechanism which are revolutionary in sequence modelling problems.

CHAPTER 3

Theory

3.1 Background

3.1.1 Stock Market and Futures

The stock market functions as a marketplace where investors engage in the buying and selling of ownership stakes in various publicly traded companies, represented by shares. These shares confer partial ownership of the respective companies to shareholders, thereby enabling them to potentially profit from the success and growth of those businesses through dividends and capital appreciation.

Conversely, the futures market operates as a platform where traders speculate on the **future prices of underlying assets** such as currencies, stock indices, or commodities, through the trading of futures contracts. These contracts entail agreements to buy or sell assets at predetermined prices on specified future dates, providing avenues for investors to hedge risks or capitalize on anticipated price movements.

While the stock market and futures market possess distinct characteristics and mechanisms, they often exhibit interplay due to shared influences from economic factors and global events. Consequently, understanding the dynamics and nuances of both markets is paramount for investors navigating the complexities of financial markets.

In this discourse, we shall delve deeper into the operational frameworks of the stock market and futures market, examining their respective parallels, divergences, and the critical variables shaping their pricing dynamics.

Buy:

In the stock market, **"buying"** denotes the acquisition of a security with the anticipation of its appreciation in value over time. This transaction is typically facilitated through brokerage firms or online trading platforms. Investors make buying decisions based on several factors, including the financial health of the company, industry trends, and recommendations from financial analysts.

Sell:

In the stock market, **"selling"** pertains to the action of divesting oneself of a security, typically with the anticipation of its declining value over time. Investors may opt to sell a stock for various reasons, including realizing gains, limiting losses, or aligning their portfolio with their investment goals.

Hold:

In trading, **"holding"** refers to maintaining a position in a security without engaging in further buying or selling activities. This strategy is commonly associated with a long-term investment approach, where investors anticipate the gradual appreciation of a particular security over time.

3.1.2 Stock Market

The stock market plays a crucial role in the global economy by facilitating the transfer of capital from investors to companies. It's a complex system influenced by various factors like government regulations, company performance, investor sentiment, and economic conditions.

Stocks represent ownership in publicly traded companies, and their value depends on factors like profitability, revenue growth, competitive position, and overall market conditions. Investors buy and sell stocks based on their expectations for the company's future prospects, and supply and demand determine stock prices.

Volatility, or how much stock prices change over time, is a key characteristic of the stock market. It's influenced by factors like market sentiment, economic news, company earnings reports, and geopolitical events. While high volatility can create

opportunities for investors to profit from short-term price changes, it also increases market risk and unpredictability.

Futures Market

The futures market serves as a marketplace for trading futures contracts, which are arrangements to purchase or sell an underlying asset at a future price and date. Futures contracts can be used to speculate on an asset's future price or to protect against price swings.

Leverage, which enables investors to control a substantial portion of an asset with a relatively little amount of cash, is one of the main benefits of futures contracts.

Similarites and Differences

While there are similarities between the stock market and the futures market, they also have significant differences. One key contrast is that futures contracts are based on the price of an underlying item, whereas stocks represent ownership in a company.

Another important difference lies in the level of risk and volatility. While both markets can be volatile, futures trading is generally riskier and more volatile due to the use of leverage in futures contracts. This can lead to both substantial profits and significant losses.

Additionally, the factors influencing each market differ. The futures market is often driven by technical factors like price trends and trading volumes, whereas the stock market is typically influenced by fundamental factors such as company performance and economic conditions.

3.1.3 Limit Order Books

3.2 Requisite Deep Learning Concepts

3.2.1 Convolutional Neural Network

Convolutional Neural Networks (CNNs) represent a specialized form of deep neural networks tailored for efficiently processing grid-like data, particularly images. In recent years, CNNs have emerged as the cornerstone of computer vision, exhibiting

unparalleled performance across various tasks such as image classification, object detection, and segmentation.

The three primary types of layers within CNNs are:

1. **Convolutional Layer**:- The initial layer of a Convolutional Neural Network (CNN) is typically comprised of convolutional filters tasked with extracting features from input data. This convolutional layer serves as the cornerstone of CNN architecture, where the bulk of processing occurs. These filters traverse the entire data.

   The outcome of the series of dot products between the filter and input data yields a feature map, activation map, or convolved feature. Following each convolution operation, a Rectified Linear Unit (ReLU) transformation is applied, introducing non-linearity to the model. This activation function enhances the model's capacity for learning complex relationships and improves its representational power.

2. **Pooling Layer**:- The pooling layer, also known as downsampling, serves to reduce the dimensionality of the input data, thereby decreasing the number of parameters in the model. Similar to the convolutional layer, the pooling operation involves sweeping a filter across the entire input. However, unlike the convolutional layer, the pooling filter lacks weights.

   In pooling, the filter aggregates values within its receptive field to populate the output array. There are primarily two types of pooling:

   Max Pooling: As the filter traverses the input, it selects the pixel with the highest value to transmit to the output array. Max pooling is commonly preferred over average pooling due to its effectiveness.

   Average Pooling: The filter computes the average value within its receptive field as it moves across the input and assigns this average value to the output array.

   While the pooling layer does discard some information, it confers several benefits to the CNN. These include reducing complexity, enhancing efficiency, and mitigating the risk of overfitting by summarizing essential information from the input.

3. **Fully Connected Layer** ;- The Fully-Connected (FC) layer, as its name suggests, establishes direct connections between every node in its output layer and those in the preceding layer. Unlike convolutional and pooling layers, which operate on localized features, FC layers integrate information from the entire preceding layer.

   In essence, the FC layer performs the classification task based on the extracted features and patterns from earlier layers. Each node in the output layer of the FC layer corresponds to a specific class or category, and its activation represents the likelihood of the input belonging to that class.

   Typically, FC layers employ a softmax activation function to generate probability scores for each class, ensuring proper categorization with probabilities ranging from 0 to 1. Contrastingly, convolutional and pooling layers often utilize Rectified Linear Unit (ReLU) functions to introduce non-linearity into the model, enhancing its capacity to capture complex relationships within the data.

   In summary, the FC layer acts as the final decision-making component of the CNN, leveraging the features learned by earlier layers to classify inputs accurately.

Training a Convolutional Neural Network (CNN) entails refining the weights associated with its filters and fully connected layers to minimize the disparity between the predicted and actual outputs given a specific input. This optimization process predominantly relies on backpropagation, a gradient-based technique. Backpropagation computes the gradient of the loss function concerning the network's weights, facilitating their adjustment in a direction that minimizes the loss. The figure 3.2 represents a standard CNN model.

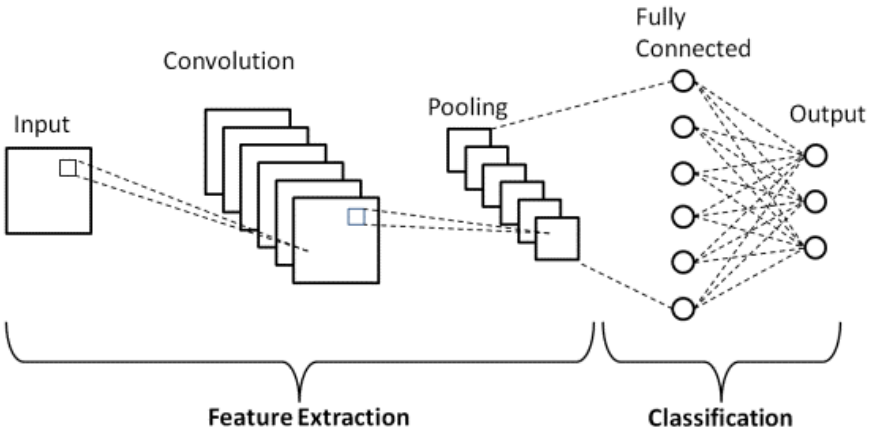

Figure 3.1: Diagram of CNN

3.2.2 Inception Module

The Inception module, a cornerstone of the Inception architecture, addresses the challenge of efficiently capturing features at different spatial scales while maintaining computational efficiency. By employing parallel convolutional branches with filters of various sizes, the module enables simultaneous feature extraction across multiple scales within a single layer. This approach enhances the network's ability to represent complex patterns and structures in the input data, leading to improved performance in tasks such as image classification and object detection.

Within the Inception module, convolutional filters of different sizes, including 1x1, 3x3, and 5x5, are applied in parallel to the input data. Additionally, pooling operations are employed to downsample the input, further enhancing feature representation across different resolutions. Dimensionality reduction techniques, such as 1x1 convolutions, are utilized to mitigate the computational cost associated with processing feature maps from parallel branches. The outputs from these branches are concatenated along the depth dimension, merging features captured at various scales into a single tensor for subsequent processing.

Overall, the Inception module offers significant advantages in feature representation, computational efficiency, and parameter reduction. By capturing features at multiple scales simultaneously and efficiently utilizing parallel operations, the module enhances the network's ability to extract informative and discriminative features from the input data. This architectural innovation has played a pivotal role in ad-

vancing the field of deep learning, leading to state-of-the-art performance in a wide range of computer vision tasks.

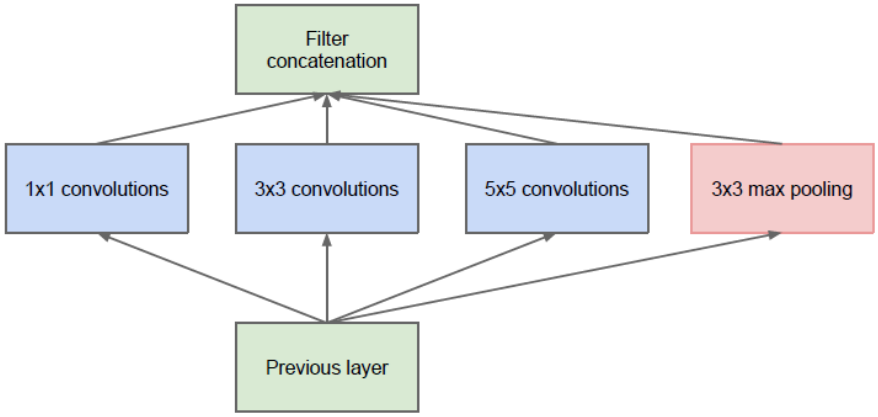

Figure 3.2: An exemplary Architecture of Inception Module

3.2.3 Recurrent Neural Network

Recurrent Neural Networks (RNNs) are a specialized family of neural networks designed to handle sequential data such as time series, audio, and text. They excel in capturing temporal relationships within data by maintaining a hidden state that carries information from previous inputs to the current time step. This ability to retain context and history makes RNNs highly suitable for tasks in natural language processing, speech recognition, and time series analysis.

At their core, RNNs consist of interconnected neurons arranged in a directed cycle. Each neuron processes input data, produces an output, and feeds that output—along with the hidden state from the previous time step back into itself for the next time step. This feedback loop enables the RNN to retain memory of past inputs, allowing it to influence the current output based on this stored information.

The hidden state, a vector containing information about previous inputs, is the key component of an RNN. At each time step, the current input is combined with the previous hidden state, and a non-linear activation function is applied to update the hidden state. Subsequently, another non-linear function is applied to the updated hidden state to generate the RNN's output. However, training RNNs can be challenging due to issues such as vanishing and exploding gradients, where gradients become extremely small or large as they propagate through the network cycle. This

can lead to difficulties in learning consistent patterns over time, posing challenges in effectively training RNNs.

For each timestep t, the activation $a^{<t>}$ and the output $y^{<t>}$ are expressed as follows:

$$a^{<t>} = g_1\left(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a\right) \text{ and } y^{<t>} = g_2\left(W_{ya}a^{<t>} + b_y\right)$$

where $W_{ax}, W_{aa}, W_{ya}, b_a, b_y$ are coefficients that are shared temporally and g_1, g_2 activation functions.

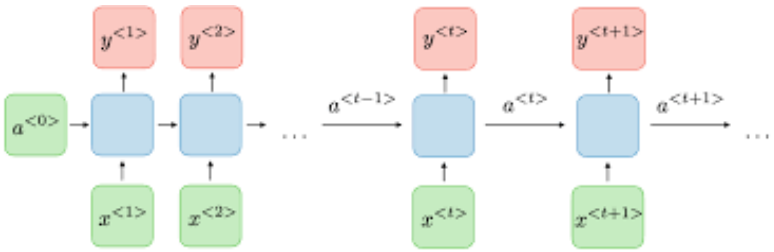

Figure 3.3: Diagram of Recurrent Neural Network

Various RNN architecture variations, notably Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) networks, have been introduced to mitigate the challenges posed by the vanishing and exploding gradient problem.

LSTM

The main goal of LSTM is to solve the vanishing gradient issue that frequently arises in conventional RNNs. This problem emerges when the network has trouble learning long-term relationships since the gradients in the backpropagation method grow incredibly tiny as they are transmitted over time.

Utilising memory cells to selectively remember or forget information at each time step is the fundamental tenet of LSTM. This objective is accomplished through a number of parts that make up the LSTM architecture. The input gate, forget gate, output gate, and memory cell are some of these parts. The LSTM equations for a single timestep for the below figure 3.4 are as follows:

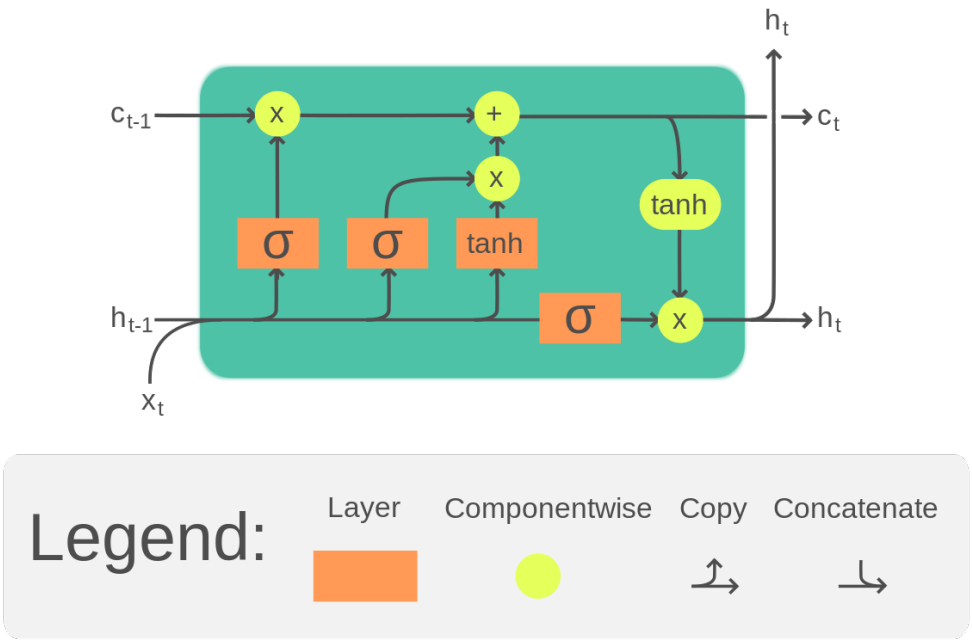

Figure 3.4: Diagram of LSTM

The input gate multiplies each input that the LSTM receives at a certain time step. How much of the fresh input should be let into the memory cell is decided by the input gate. How much of the prior memory cell state should be kept is decided by the forget gate. How much of the current memory cell state should be sent to the network's next layer is decided by the output gate.

Long-term dependencies are remembered by the memory cell itself. Based on the inputs, the previous memory cell state, and the gates, it is updated at each time step. Using a combination of addition and multiplication operations, the memory cell is updated.

Many different tasks, including speech recognition, language modelling, machine translation, and video analysis, have shown the effectiveness of the LSTM architecture. The popularity of LSTM in natural language processing (NLP) jobs, where it is used to represent sequential data like phrases or paragraphs, is due to its capacity to manage long-term dependencies.

3.2.4 Attention Meachanism

Attention mechanisms in neural networks draw inspiration from human cognitive processes, allowing models to prioritize relevant input data while disregarding irrelevant information. This dynamic allocation of importance significantly boosts the performance of neural network architectures, especially in tasks like natural language processing (NLP), image captioning, and machine translation, which involve sequential or structured data.

These mechanisms dynamically assign weights to different input elements, enabling the model to selectively focus on specific parts of the input sequence. Unlike traditional architectures, attention mechanisms adaptively weigh the importance of each element based on its relevance to the task at hand. They typically involve three components: the query (current context), key, and value (elements of the input sequence). The model computes compatibility scores between the query and keys, determining the relevance of corresponding values. These scores are transformed into attention weights using a softmax function, ensuring they represent a probability distribution over input elements. The weighted sum of values, based on these attention weights, produces the attended representation for further processing.

A major advantage of attention mechanisms is their ability to capture long-range dependencies in input data, even with lengthy sequences. By allowing dynamic focus on relevant parts, attention mechanisms enhance model interpretability and performance, yielding state-of-the-art results.

CHAPTER 4

Methdology and Implementation

4.1 Data Set and Preprocessing

4.1.1 FI-2010 dataset

FI-2010, a public benchmark financial market dataset collected in 2010 from Helsinki Stock Exchanges, commonly used for researching algorithmic trading strategies, market prediction models, and risk management techniques.

It includes high-frequency trading data such as order book information, trade prices, volumes, and timestamps, allowing detailed analysis of market dynamics and price movements.

This LOB dataset employs 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 midpoint of the Limit Order Book.* Our data analysis focused on standard trading hours, specifically excluding auction periods.

4.1.2 Dataset Description and Labelling

Our dataset comprises both the feature set and corresponding labels. The data is sourced from the Helsinki stock exchange and is structured as a table with 149 rows.

The *first 144 rows contain feature data*, with the initial 40 entries representing limit order book (LOB) points encompassing 10 levels for 'ask price,' 'bid price,' 'ask volume,' and 'bid volume,' along with 104 manually-engineered features.

The *last five rows contain labels* for the movement of the mid-price according to five prediction horizons ($k = 10, 20, 30, 50$, and 100).

In terms of classification of movement of mid-price, 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 $x_t = [p_a^{(i)}(t), v_a^{(i)}(t), p_b^{(i)}(t), v_b^{(i)}(t)]_{i=1}^{n=10}$. Here, $p^{(i)}$ and $v^{(i)}$ denote the price and volume size at the i-th level of a limit order book.

To calculate the movement of the mid-price, i.e., creating a label, the following steps are followed:

- The mid-price p_t is calculated as the average of the highest bid price and the lowest ask price:

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

  Here, p_t is the mid-price and it is the average of the highest Bid Price and the lowest Ask Price.

- The mean of the next k mid-prices, denoted as $m_+(t)$, is computed as:

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

  m_+ denotes the mean of the next k mid-prices, p_t is the mid-price defined in Equation (1), and k is the prediction horizon. This is used to predict the movement of mid-price, i.e., the calculation of the label.

- The movement of the mid-price, l_t, is calculated using m_+ and p_t:

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

- The labels are then determined based on comparison of the value of movement of the mid-price, l_t and a threshold, α:

$$\begin{cases} l_t > \alpha : +1 & \text{(up - label 0)} \\ l_t < -\alpha : -1 & \text{(down - label 2)} \\ \text{else: } 0 & \text{(stationary - label 1)} \end{cases} \quad (4.1)$$

The FI-2010 dataset [1] adopts the method in Equation (3) and we directly used their labels for fair comparison to other methods.

4.1.3 Preprocessing

Our deep learning models are trained using the previous 100 timestamps of the Limit Order Book (LOB), containing 40 features each (ask and bid price, ask and bid volume of 10 price levels), as input to predict the label (0, 1, or 2) corresponding to the prediction horizon $k = 10$.

We utilized 203,800 time steps of 40 features as the training set, 50,950 as the validation set, and 139,587 as the testing set [1].

The most recent 100 states of the LOB's 40 features are considered to predict the movement of the mid-price. All LOB models were trained on labels corresponding to the prediction horizon $k = 10$.

Therefore, the input tensor for the models is of shape (N, 100, 40, 1), and the model predicts the movement of the mid-price or Label for every timestep, resulting in a tensor of shape (N, 3) one hot encoded.

- **Training Data:**
  - Features: (203701, 100, 40, 1)
  - Labels: (203701, 3)
- **Validation Data:**
  - Features: (50851, 100, 40, 1)
  - Labels: (50851, 3)
- **Test Data:**
  - Features: (139488, 100, 40, 1)
  - Labels: (139488, 3)

4.2 DeepLOB (Deep Learning for Limit Order Books)

The architecture of DeepLOB [1] is made up of convolutional neural networks (CNNs), an Inception module, and a LSTM network as shown in Fig. 4.1, to capture time dependencies and predict patterns in the data.

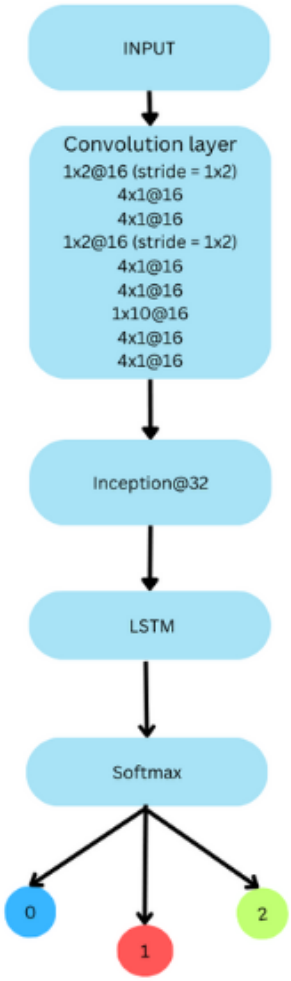

Figure 4.1: Model Diagram of DeepLOB

4.2.1 Convolutional Layer

- **First Layer:** The first layer summarizes information between price and volume $p(i), v(i)$ at each order book level. Hence, the filter has a size of (1×2) for the price and volume pair, indicating that it analyzes two adjacent features at a time. The stride of (1×2) is used to ensure that the volume and price pair have the same parameters.

- **Second CNN Layer:** The second CNN layer has a filter of size (4×1) and focuses on the same level, extracting features over four time steps, layer by layer.

- **Integration :** Finally, we integrate all the information using a large filter of size (1×10), resulting in the extraction of 40 features over 100 timestamps into one feature. As we have 16 filters, we will have 16 different dynamically extracted features.

It's important to note that very short-term dependencies are already captured in the convolutional layer, which takes (100, 40)-shaped tensors of LOB data as inputs [1].

4.2.2 Inception Module

By combining different convolutions, the Inception Module enhances the flexibility of the model. This concept is akin to the use of various moving averages in technical analysis to observe time-series momentum. The Inception Module enables the utilization of multiple filter sizes, removing restrictions imposed by a single filter size based solely on the data. Fig 4.2 shows the inception module used in DeepLOB and other LOB models.

- **Filter Sizes for Time Scales:** From Fig. 2, it's evident that the Inception module employs (3×1) and (5×1) filters, allowing the capture of information over multiple time scales. Without the Inception module, such versatility would not be achievable.

- **Bottle-Neck Layer:** The (1×1) convolution, known as the Bottle-Neck Layer, breaks down the input into simpler forms. These forms undergo transformations with filters of sizes (3×1) and (5×1), and the results are concatenated.

- **Max-Pooling Layer and Convolution:** In the Inception Module, a max-pooling layer is applied along with a (1×1) convolutional layer featuring filters. This approach *captures non-linear data properties,* thereby improving prediction accuracy [1].

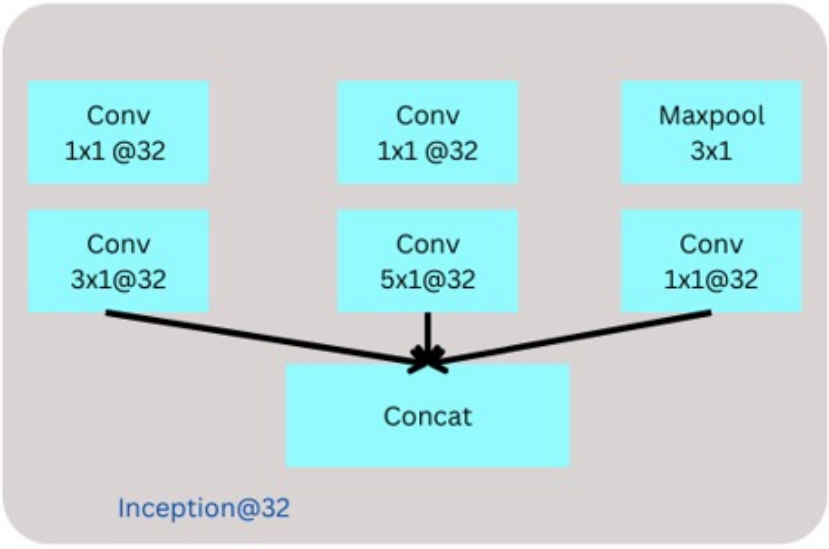

Figure 4.2: Inception Module

4.2.3 LSTM (Long Short-Term Memory)

:- LSTM units feature a feedback mechanism that preserves the memories of past activations. This capability aids in modeling the temporal dynamics present in our features.

- **Implementation Details:** Our implementation involves 64 LSTM units, which collectively require around 60,000 parameters. This represents a significant 10x reduction compared to using fully connected layers. The primary motivation for utilizing LSTM is to capture long-term temporal dependencies among the features, a task that CNNs may struggle with due to their limited filter size and lookback time step (T = 100).

- **Output Layer:** The ***final output layer for all LOB models consists of a softmax activation layer.*** The output of this layer represents the probability of each price movement class at each time step [1].

4.3 ADLOB (Attention in DeepLOB for Limit Order Books)

DeepLOB, which was the first literature to use CNN with LSTM on high-frequency LOB data to predict the movement of mid-price. ***By introducing a Weighted Attention mechanism after the LSTM layer, naming the model as Attention to DeepLOB (ADLOB).*** The resulting model architecture is shown in Figure 4.3.

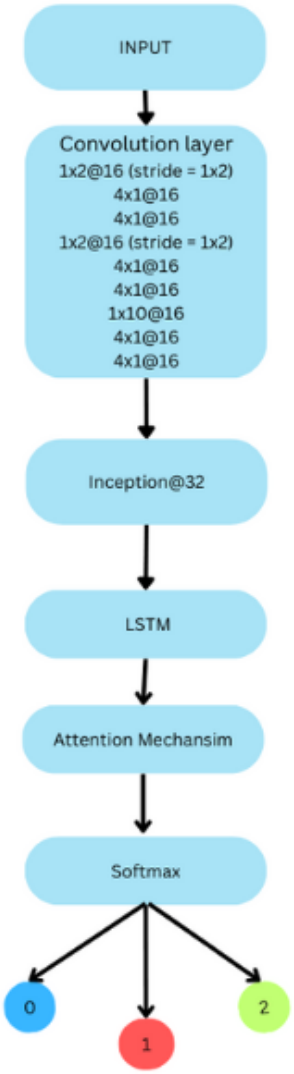

Figure 4.3: Model Diagram of ADLOB

4.3.1 Attention Mechanism

The attention layer acts like a spotlight, enabling the model to focus on specific parts of the input sequence when making predictions. Essentially, the attention layer helps the model "attend" to the most informative elements and patterns in the data, contributing to more accurate predictions in our project [2].

In the attention mechanism, there are query, key, and value vectors. Since this is a variation of the self-attention mechanism, the query and value will be the same, i.e., the input and the key will be trainable weight vectors.

Each query vector q is matched against a database of keys k_i to compute a score $e_{q,k}$. 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 $e_{q,k}$ are passed through a softmax operation to generate the attention weights $a_{q,k}$:

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

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.

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

Instead of treating all elements equally, the attention mechanism assigns varying levels of importance to different parts of the input sequence. This empowers the model to pay more attention to relevant information, thereby improving its ability to capture complex patterns and relationships in the data.

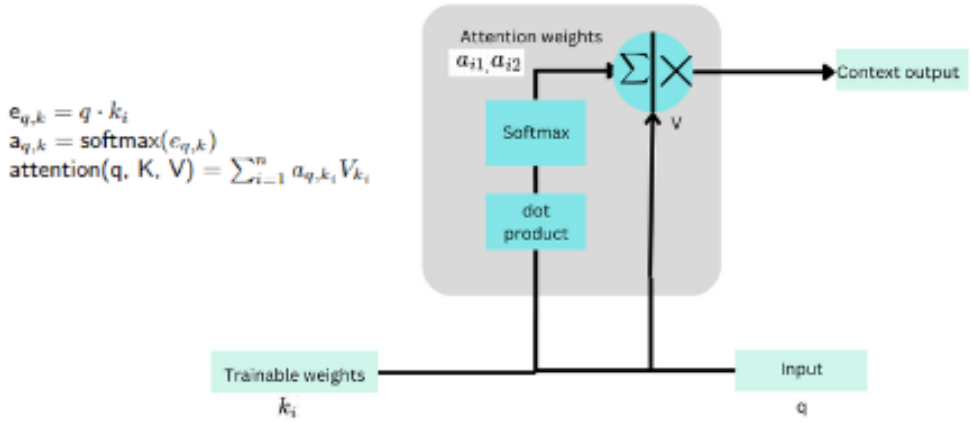

Figure 4.4: Weighted Attention Mechanism Diagram

4.4 Variations to ADLOB

We started experimenting with the variation of DeepLOB and attention mechanisms and simulated a total of 6 models on limited data sets.

The final 4 LOB models that we shoed promising results are:

- **DeepLOB**: This is the base model that was developed by Zhang et al [1].
- **ADLOB**: Attention on DeepLOB adds a Weighted Attention Layer after the LSTM layer to amplify the patterns in time series for accurate predictions
- **Bi-Directional ADLOB**: This slightly modifies ADLOB by changing the LSTM layer of ADLOB to a Bi-Directional LSTM. Thus, look for patterns in both directions instead of just one.
- **Encoder Decoder ADLOB**: Encoder Decoder ADLOB takes inspiration from *Bahdanau encoder-decoder attention mechanism* [8], which employs a bidirectional RNN as an encoder and an RNN decoder, with an attention mechanism in between.

Fig. 4.5 shows the remaining two model's architectures.

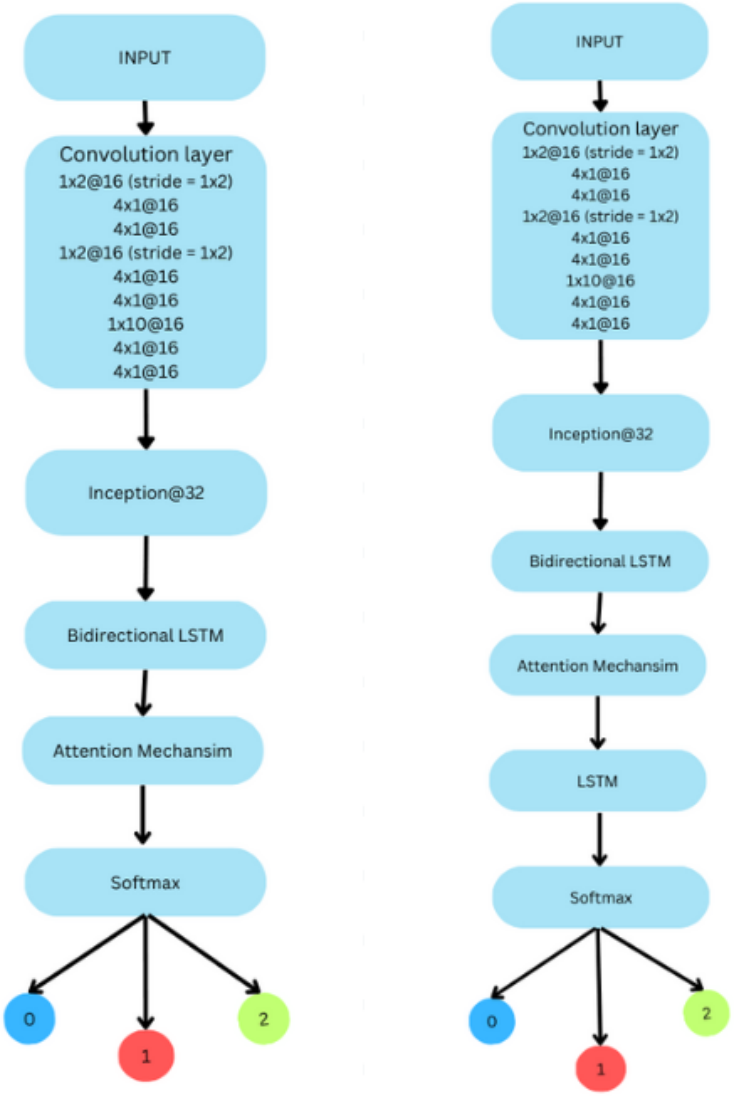

Figure 4.5: Model Diagrams of Bi-ADLOB and Enc-Dec-ADLOB, respectively

4.5 Backtesting on Deep learning Models

In order to evaluate the performance of deep learning models, we are conducting back-testing using historical financial data.

Backtesting involves assessing a trading strategy's performance using historical data to simulate its past performance. The objective is to evaluate the model's effectiveness in real-world scenarios and identify areas for improvement.

The backtesting procedure starts by defining the trading strategy based on predictions from deep learning models. This strategy typically involves making buy, sell, or hold decisions based on certain criteria derived from the model's output.We

used a simple backtesting strategy based on the model's predictions to calculate the performance metrics.
The strategy used is as follows:

Trade Execution:

A trade is executed based on the model's prediction for the next timestep.

- If the model predicts a 0 (up movement), we assume a long position, i.e., we **buy stocks**.
- If the model predicts a 1 (stationary movement), no trade is executed, i.e., we **hold the stocks** for future trades.
- If the model predicts a 2 (down movement), we assume a short position, i.e., we will **sell the stocks.**

When evaluating a trading strategy, the following general benchmarks are often used to assess its performance: Cumulative Return, Maximum Drawdown, Sharpe Ratio, Profit Factor, and Win Rate.

4.5.1 Backtesting on Capital Investment

The Final Model with the highest accuracy will be given a capital investment and the following strategy is applied to it.

1. If the model predicts a downward movement:
    - If the agent already holds shares, it sells all of them.
    - If the agent is not invested in shares, it maintains its current position and holds.
2. If the model predicts an upward or neutral movement:
    - If the agent has no shares, it uses all available funds to purchase shares.
    - If the agent already holds shares, it maintains its current position and holds.

CHAPTER 5

Results and Discussion

5.1 Testing LOB Models

We experimented with many variations of DeepLOB and attention mechanisms, including using bidirectional LSTM and transitioning from a weighted attention mechanism to a multiheaded attention mechanism. However, ***four different models (DeepLOB, ADLOB, Bi-ADLOB, and Enc-Dec-ADLOB) consistently exhibited high accuracy***.

We tuned hyperparameters such as look-back-timestep (T), learning rate, batch size, etc., until maximum accuracy was achieved. To accomplish this, we simulated the models on a limited dataset (trained upon 50,000 timesteps) and calculated their accuracies. Table 5.1 illustrates the accuracy scores of the four models.

Model	Accuracy Score
DEEP LOB	0.782
AD LOB	0.786
Bi-directoinal AD LOB	0.79
Encoder-Decoder	0.758

Table 5.1: Accuracy Scores of LOB models on limited data

We trained the models on the complete dataset offline in Jupyter Notebook (GPU: NVIDIA-RTX6000) for 200 epochs. The models were trained with the parameters and settings mentioned.

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

Fig. 5.1 depicts the Confusion Matrix of all four models respectively. From this, we observe that ADLOB is more accurate than other LOB models in predicting the correct movement of the mid-price.

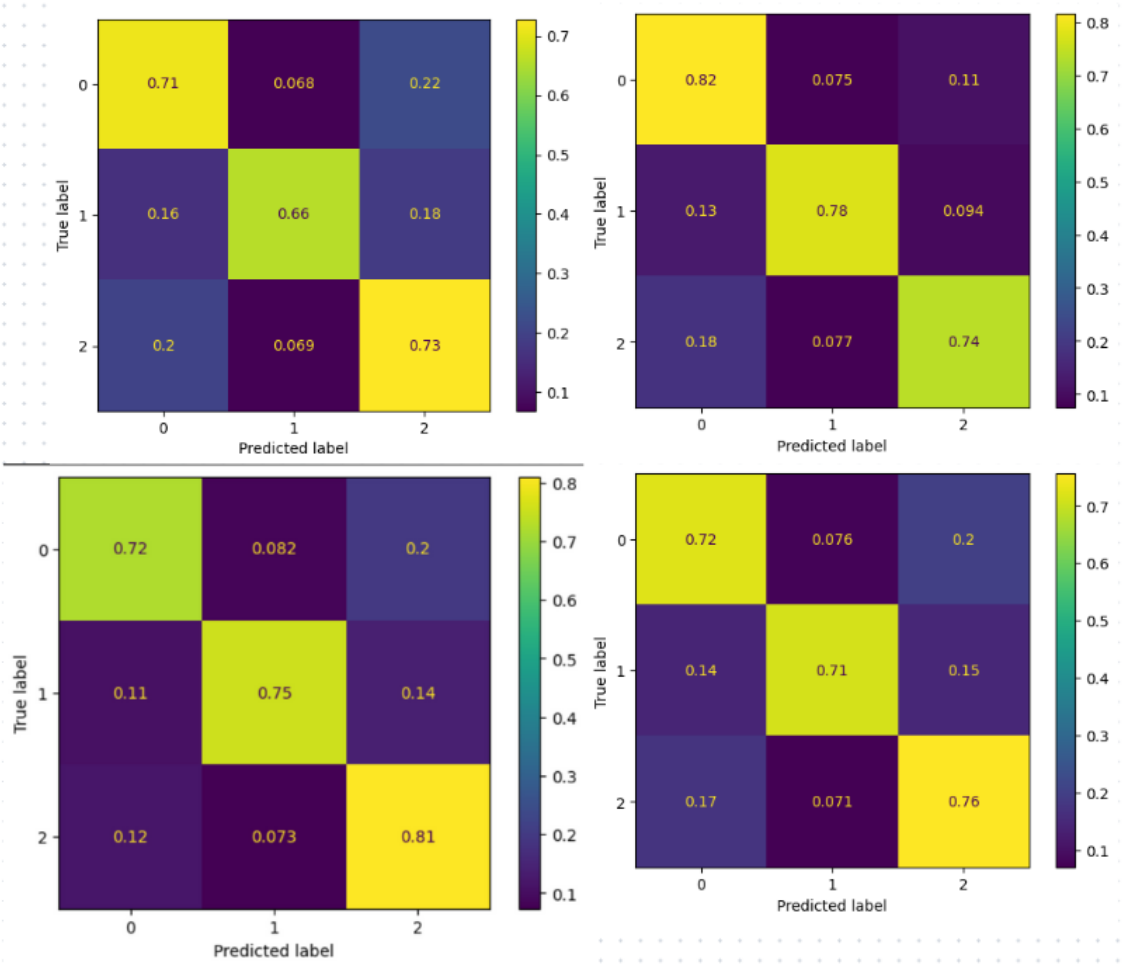

Figure 5.1: Confusion Matrices of LOB Models from Top to Bottom and Left to Right DeepLOB , ADLOB , Bi-Directional ADLOB and Encoder Decoder ADLOB

From these confusion matrices of the models, we observe the following:

- ADLOB exhibits the highest overall accuracy among all LOB models.
- ADLOB demonstrates the highest accuracy in predicting labels 0 and 1 compared to other LOB models.
- However, when it comes to predicting label 2 (i.e., down movement of mid-price), Bi-ADLOB and Enc-Dec-ADLOB are more accurate.

Label	Precision	recall	f1-score
0	0.6744	0.7130	0.6932
1	0.8340	0.6569	0.7349
2	0.6219	0.7285	0.6710
accuracy score			0.6985

Table 5.2: DeepLOB Classification Report

Label	Precision	recall	f1-score
0	0.7342	0.8157	0.7728
1	0.8435	0.7776	0.8092
2	0.7676	0.7401	0.7536
accuracy score			0.7789

Table 5.3: ADLOB Classification Report

Label	Precision	recall	f1-score
0	0.7717	0.7212	0.7456
1	0.8353	0.7541	0.7926
2	0.6862	0.8093	0.7427
accuracy score			0.7600

Table 5.4: Bi-ADLOB Classification Report

Label	Precision	recall	f1-score
0	0.7066	0.7224	0.7144
1	0.8354	0.7116	0.7686
2	0.6635	0.7557	0.7066
accuracy score			0.729

Table 5.5: Encoder Decoder ADLOB Classification Report

5.1.1 Performance Metrics for Backtesting

- **Cumulative Return:** Cumulative Return measures the total return of a trading strategy over a specified period.
  - **Benchmark for a good model:** A higher cumulative return indicates better performance. There is no specific benchmark, but a positive cumulative return is generally desired.
- **Maximum Drawdown:** Maximum Drawdown measures the largest single drop from peak to bottom in the value of a portfolio. It is calculated using the formula:

$$\text{Maximum Drawdown} = \max\left(\frac{P_{\text{peak}} - P_{\text{trough}}}{P_{\text{peak}}}\right)$$

  where P_{peak} is the highest portfolio value before a drawdown and P_{trough} is the lowest portfolio value during the drawdown.
  - **Benchmark for a good model:** A smaller maximum drawdown indicates a more stable performance. Generally, a maximum drawdown less than 20% is considered good.
- **Sharpe Ratio:** Sharpe Ratio measures the risk-adjusted performance of a trading strategy. It is calculated using the formula:

$$\text{Sharpe Ratio} = \frac{R_{\text{p}} - R_{\text{f}}}{\sigma_{\text{p}}}$$

  where R_{p} is the portfolio return, R_{f} is the risk-free rate, and σ_{p} is the standard deviation of the portfolio return.
  - **Benchmark for a good model:** A higher Sharpe Ratio indicates better risk-adjusted returns. Generally, a Sharpe Ratio greater than 1 is considered good.
- **Profit Factor:** Profit Factor is the ratio of gross profit to gross loss. It is calculated using the formula:

$$\text{Profit Factor} = \frac{\text{Total Gross Profit}}{\text{Total Gross Loss}}$$

- **Benchmark for a good model:** A higher profit factor indicates better profitability. Generally, a profit factor greater than 1 is considered good.

- **Win Rate:** Win Rate measures the percentage of profitable trades. It is calculated using the formula:

$$\text{Win Rate (\%)} = \frac{\text{Number of Profitable Trades}}{\text{Total Number of Trades}} \times 100$$

  - **Benchmark for a good model:** A higher win rate indicates a higher percentage of profitable trades. Generally, a win rate above 50% is considered good.

5.1.2 Results of Backtesting

Metric	**DeepLOB**	**ADLOB**	**Bi-ADLOB**	**Enc-Dec-ADLOB**
Cumulative Return	917.0000	3867.0000	3432.0000	2391.0000
Maximum Drawdown	91.0000	26.0000	34.0000	94.0000
Sharpe Ratio	0.2115	0.9025	0.8005	0.5564
Profit Factor	1.0935	1.4862	1.4188	1.2712
Win Rate (%)	51.2206	57.7605	56.9856	54.6614

Table 5.6: Backtesting Results for DEEPLOB

- The above table 5.6 represents the comparison of the backtesting metrics of 4 LOB models with a simple backtesting strategy, as discussed in Section 4.5.

- The ADLOB model performed better compared to other models in all of the metrics.

- Bi-ADLOB backtesting results are near Benchmark for the metrics.

- Both DeepLOB and Enc-Dec-ADLOB are not up to standard for backtesting using a simple strategy.

5.1.3 Backtesting on Capital Investment

Fig 5.2 shows the User Interface that we created for Backtesting

We developed a user interface (UI) that allows users to input the amount of capital they wish to invest. This capital is then utilized for backtesting strategy, as discussed in subsection 4.5.1.

The model employed for predictions is ADLOB, chosen based on its highest accuracy scores as observed from Tables 5.2 to 5.5. Furthermore, from Table 5.6, it is evident that ADLOB performs the best for simple backtesting strategies.

Figure 5.2 illustrates the user interface designed for conducting backtesting.

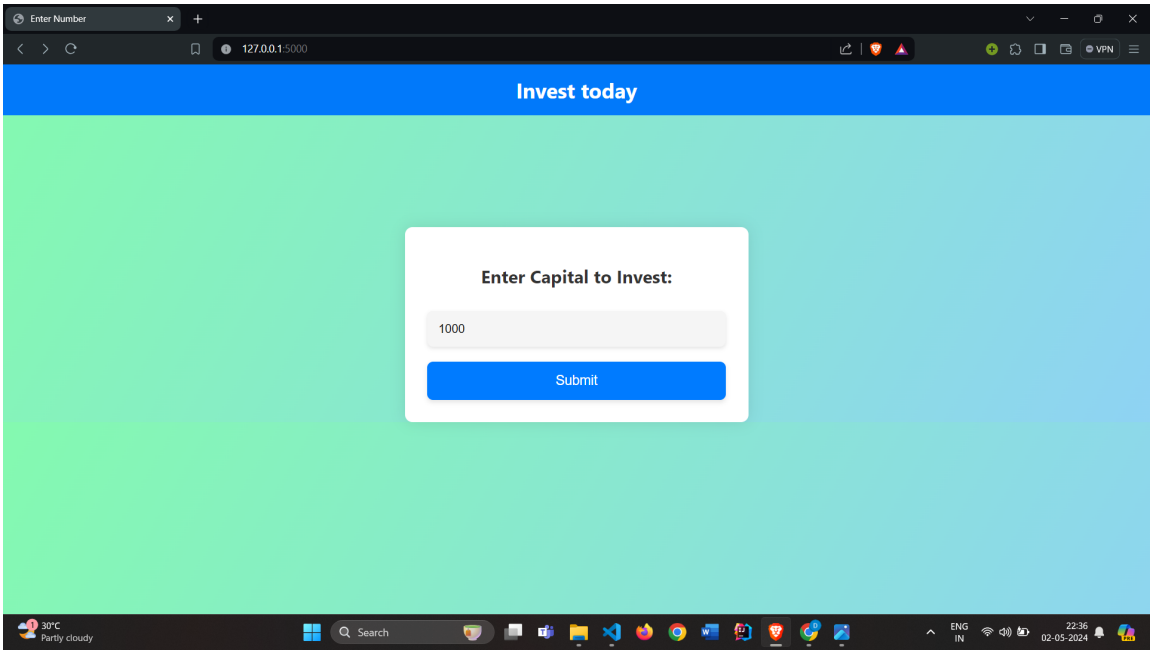

Figure 5.2: UI for Backtesting on Capital Investmnet

Figures 5.3 and 5.4 display the current asset value at different time instances, and the ***final return for a capital investment of 100 rupees utilizing predictions made by the ADLOB model is 3252 rupees. Hence, a profit of 2252 rupees.***

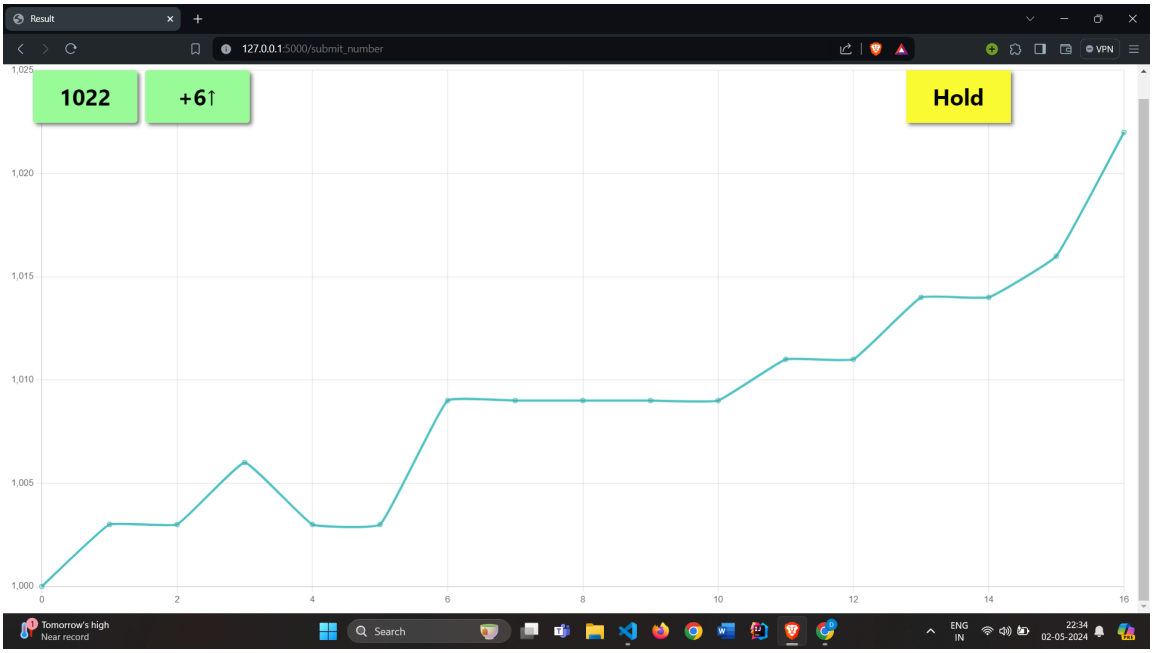

Figure 5.3: Graphs of Current Assest value at two different time steps

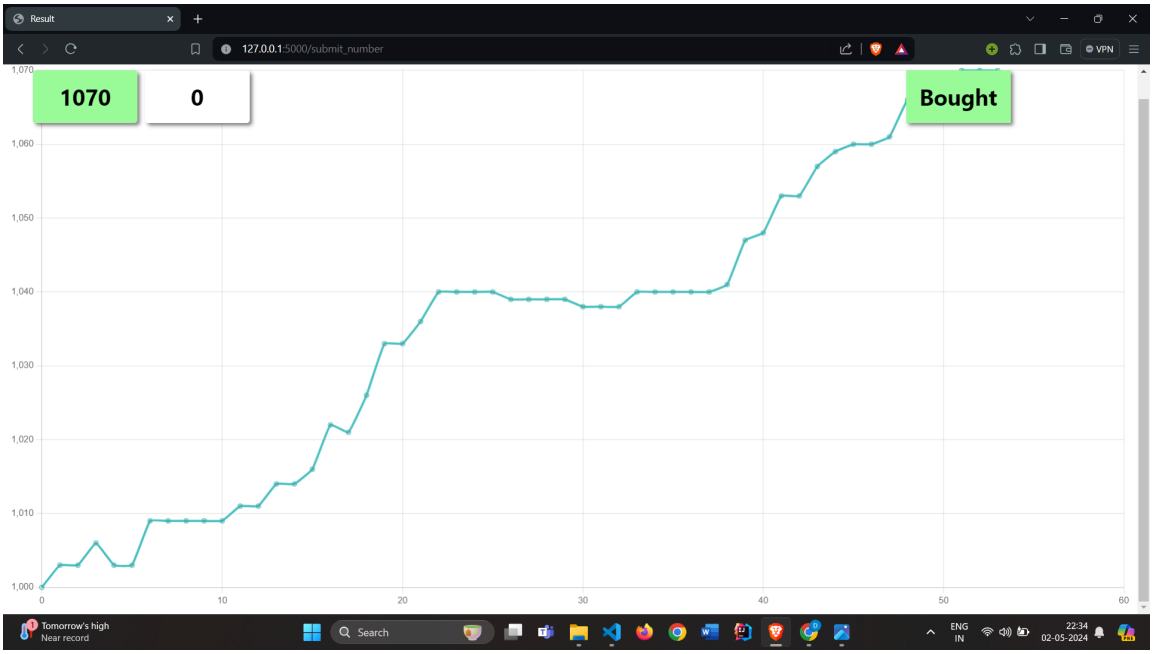

Figure 5.4: Graphs of Current Assest value at two different time steps

CHAPTER 6

Conclusion

In conclusion, the DeepLOB with Attention mechanism presentes is a promising approach for predicting financial market movements based on limit order book (LOB) data. Combining convolutional neural networks (CNNs), long short-term memory (LSTM) networks, and an attention mechanism shows that the model can handle the spatial and temporal dependencies that come up in financial time series.

This led to an increase in accuracy from 69 for DeepLOB to 77 percent for ADLOB **(8 percent increase in accuracy)**

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

BIBLIOGRAPHY

[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 "

[4] P. Bacchetta, E. Mertens, and E. VanWincoop, "Predictability in financial markets: What do survey expectations tell us?" J. Int. Money Finance,vol. 28, no. 3, pp. 406–426, 2009.

[5] J. Agrawal, V. Chourasia, and A. Mittra, "State-of-the-art in stock prediction techniques," Int. J. Adv. Res. Elect., Electron. Instrum. Eng., vol. 2, no. 4, pp. 1360–1366, 2013.

[6] N. Passalis, A. Tefas, J. Kanniainen, M. Gabbouj, and A. Iosifidis, "Temporal bag-of-features learning for predicting mid price movements using high frequency limit order book data," IEEE Trans. Emerg. Topics Comput.Intell., to be published.

[7] J. Sirignano and R. Cont, "Universal features of price formation in financial markets: perspectives from deep learning," arXiv preprint arXiv:1803.06917.

[8] Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio "Neural Machine Translation by Jointly Learning to Align and Translate" arXiv preprint arXiv:1409.0473

[9] D. T. Tran, M. Magris, J. Kanniainen, M. Gabbouj, and A. Iosifidis, "Tensor representation in high-frequency financial data for price change prediction," in Computational Intelligence (SSCI), 2017 IEEE Symposium Series on. IEEE, 2017, pp. 1–7.

[10] N. Passalis, A. Tefas, J. Kanniainen, M. Gabbouj, and A. Iosifidis, "Temporal bag-of-features learning for predicting mid price movements using high frequency limit order book data," IEEE Transactions on Emerging Topics in Computational Intelligence, 2018.

[11] A. Tsantekidis, N. Passalis, A. Tefas, J. Kanniainen, M. Gabbouj, and A. Iosifidis, "Forecasting stock prices from the limit order book using convolutional neural networks," in Business Informatics (CBI), 2017 IEEE 19th Conference on, vol. 1. IEEE, 2017, pp. 7–12.

[12] J.-F. Chen, W.-L. Chen, C.-P. Huang, S.-H. Huang, and A.-P. Chen, "Financial time-series data analysis using deep convolutional neural networks," in Cloud Computing and Big Data (CCBD), 2016 7th International Conference on. IEEE, 2016, pp. 87–92.

[13] J. Doering, M. Fairbank, and S. Markose, "Convolutional neural networks applied to high-frequency market microstructure forecasting," in Computer Science and Electronic Engineering (CEEC), 2017. IEEE, 2017, pp. 31–36.

[14] T. Fischer and C. Krauss, "Deep learning with long shortterm memory networks for financial market predictions," European Journal of Operational Research, vol. 270, no. 2, pp. 654–669, 2018.

[15] D. M. Nelson, A. C. Pereira, and R. A. de Oliveira, "Stock market's price movement prediction with LSTM neural networks," in Neural Networks (IJCNN), 2017 International Joint Conference on. IEEE, 2017, pp. 1419–1426.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural computation, vol. 9, no. 8, pp. 1735– 1780, 1997.

[17] J. Sirignano and R. Cont, "Universal features of price formation in financial markets: perspectives from deep learning," arXiv preprint arXiv:1803.06917, 2018.