ISSN 1816-6075 (Print), 1818-0523 (Online)
Journal of System and Management Sciences
Vol. 11(2021) No.3, pp. 75-100
DOI:10.33168/JSMS.2021.0305

An Optimal Limit Order Book Prediction Analysis Based on Deep Learning and Pigeon-Inspired Optimizer

Mohammad Zainal[1], Ibrahim Gad [2], Hameed AlQaheri[1]

[1]College of Business Administration, Kuwait University, Sabah Al Salem University City,

[2]Kuwait City, Kuwait

zainal@cba.edu.kw

Abstract. Currently, In the system-driven electronic markets, stocks and futures contracts are traded based on the centralization of buy and sell orders in the boundary order book (LOB), as provides much more information about stocks than their prices, such as dynamics of the price and predictability of the next trading move. This paper proposes an optimal limit order book analysis for five stocks, including Amazon (AMZN), Apple (AAPL), Google (GOOG), Intel (INTC), and Microsoft (MSFT). It is based on deep learning and an enhanced pigeon-inspired (PIO) algorithm. The system reduces the dimentaionlty of LOB data sets by using a pigeon-inspired optimizer to determine the most significant features. The fitness function is used to evaluate the fitness value of each solution based on TPR and FPR and the feature count. The optimized LOB feature selection is evaluated using the Decision Tree (DT) classifier. A new deep neural network with high-frequency order series includes convolutional, dense layers and Inception units to predict future stock price movements (Submission, Cancellation, Deletion, Execution visible and hidden orders) in an extensive high-frequency LOB database that supports improving the operational performance of the trading process. The proposed model is evaluated using the LOB dataset, and the results show that the model performs better in predicting the different classes. The analysis of variance ANOVA supports the obtained results that test for the significant difference among the means of all items according to their event types.

Keywords: Limit order book, deep learning, pigeon optimization, prediction, ANOVA.

1. Introduction

Artificial intelligence (AI) in finance has been a popular topic in academia and the financial industries in the past few decades. Several studies have been published that yielded different models. Meanwhile, in machine learning (ML), deep learning (DL) has started to get a lot of attention recently, mainly due to its superiority over classical models. Today there are a lot of different applications of DL learning, and the wide interest continues. Funding is one specific area where DL models are gaining traction; however, the playing field is wide open, and there are still plenty of research opportunities (Ahmet et al., 2020; Joanna, 2020).

Some recent trends in market design are the provision of real-time Limit Order Book (LOB) information, the introduction of competing for orders-driven venues in traditional merchant markets, and new pure electronic order book systems. With the increasing availability of order book data, these trends are generating a renewed interest in the microstructure of system-driven net markets. The distinguishing feature of these trading platforms is their high degree of pre-trade transparency: the ability of market participants to monitor content limit order book (Tripathi and Dixit, 2020; Domowitz and Wang, 1994; Marco and Sasha,2008; Palguna and Pollak, 2016; Xiao et al., 2016).

Predicting stock price movements based on deep learning and high-frequency data has been studied extensively in recent years. Data flows' complex and chaotic behavior has given rise to nonlinear methods such as those we see in machine learning and deep learning. The high-frequency of the limit order books data analysis has captured the machine learning community (Ntakaris et al., 2019; Xue et al., 2021).

Ntakaris et al. (2019) discuss the problem of features design, developing a new set of handcrafted features, and conducting a comprehensive experimental evaluation of liquid and illiquid stocks. In addition, the authors present a wide range of econometric features that capture the statistical properties of the underlying securities for the average price prediction task. A new experimental online learning protocol that treats the above task as a multi-objective optimization problem and predicts is also discussed. Convolutional neural networks and long-term memory neural networks are adapted to with multi-objective optimization for predicting the mid-price movement,

Avraam et al. (2020) introduce short-term recurrent memory networks (LSTM) and convolutional neural networks (CNNs) to generate static features on the LOB. These generated features are tested in the task of forecasting average price movements in the limited order book. The introduced model combines the ability of a CNN to extract valuable features with the ability of LSTMs to analyze time series has been proposed and evaluated. The combined model outperforms individual LSTM and CNN models in the prediction prospects tested.

Nousi et al. (2019) proposed two Autoencoders and Bag-of-Features-based feature learning algorithms to predict future price movements using limit order book data. Autoencoders (AEs) are type neural networks that define their input data for themselves by multiple levels of nonlinear neurons, while Feature bag models (BoF) allow the extraction of fixed-length representations of samples consisting of multiple feature vectors, for example, trait vectors extracted from different locations of the image or from different time points of a time series. Three Machine Learning algorithms are tested and validated using combinations of these features on three prediction scenarios amd two different evaluation setups.

Nakayama et al. in (2019) apply a convolutional neural network and logistic regression models based features to predict the direction of short-term price movements. Their results show the highest expectation power from the order and cancel information.

Forecasting the movements of stock prices is one of the most challenging problems in financial markets analysis. Zhang et al. (2019) introduced a large-scale deep learning model to predict price movements from cash stocks' limit order book (LOB) data. Convolutional filters were used to capture the spatial structure of LOBs and capture longer temporal dependencies that depend on long-term memory modules. A sensitivity analysis was performed to understand the rationale behind the model predictions and reveal the components of the most relevant LOBs. Based on various features generated from the order book data observed simultaneously, several non-parametric forecasts of the average price in the limit order book have been proposed by Palguna and Pollak (2016), simultaneously and in the recent past.

To improve execution on cryptocurrency exchanges by learning strategies for placing optimum limit orders, Matthias (2021) presents an optimizing execution strategy execution through deep reinforcement learning that support both professional asset managers and private investors as the quality of implementation that affects portfolio performance at important economic we design a problem-specific training environment that offers purpose-built reward functionality, handcrafted market-state features, and virtual limit order exchanges.

This paper proposes an optimal limit order book analysis for five stocks, including Amazon (AMZN), Apple (AAPL), Google (GOOG), Intel (INTC), and Microsoft (MSFT). It is based on deep learning and an enhanced pigeon-inspired (PIO) algorithm. The system reduces the dimentaionlty of LOB data sets by using a pigeon-inspired optimizer to determine the most significant features. The fitness function is used to evaluate the fitness value of each solution based on TPR and FPR and the feature count. The optimized LOB feature selection is evaluated using the Decision Tree (DT) classifier. A new deep neural network includes convolutional, dense layers and Inception units to predict future stock price movements in an extensive high-frequency LOB database.

The remainder of the paper is organized as follows. Section (2) provides a brief explanation of the basics of the methods used in this paper. Section (3) discusses the characteristics of Limit Order Books Datasets. Section (4), the proposed optimal limit order books prediction phases are discussed along with the steps involved and the characteristic features for each phase. In Section (5), we provide the empirical results and ANOVA analysis. Section (6) concludes the paper.

2. Preliminaries

This section provides a brief explanation of the basics of the methods used in this paper, including limit order books, the Pigeon-inspired optimizer, and the Deep Learning model, along with some of the essential basic concepts. A more comprehensive review can be found in sources as (Tripathi and Dixit, 2020; Palguna and Pollak, 2016; Duan and Qiao, 2014; Yushan et al., 2019; Li et al., 2019).

2.1. Limit Order Books (LOBs)

The limit order book records the pending limit orders kept by a security specialist operating at an exchange. A limit order is a type to buy or sell a security at a specified price or better (Palguna and Pollak, 2016). LOBs work with two types of orders, namely limit orders and market orders (Rajeshkanna and Arunesh, 2020). A limit order is a commitment made at a specified time to trade a particular volume of securities at a predetermined price limit. The market order is executed instantly at the best available price(s). Unexecuted limit orders are stored in the LOB until they are matched or canceled. In the LOB microstructure, traders provide liquidity through limit orders and consume liquidity through market orders. As described in Figure (1), the definition of the limit order is an order submitted at the time (t_i) and prize (p_i) and if the size (w_i) is less than zero; then the limit order is a commitment as a sell order; else, its commitment as a buy order up to $|w_i|$ units of the traded asset at a price not less than p_i for the sell order and not greater than p_i for buy. While the limit order book is defined as a set of all active orders in the market at the time t_i (Mason, 2013).

2.2. Pigeon-Inspired Optimizer (PIO)

Pigeons are the most famous and popular birds globally, and the Egyptians previously employed them to convey messages, a practice repeated in several military operations (Duan and Qiao, 2014). It is possible to rapidly locate a homing pigeon's home using three homing methods: the magnetic field, the sun, and landmarks. The map and compass operator models are provided in this optimizer based on the magnetic field and the sun, whereas the landmark operator model is presented based on landmarks.

Two factors are formed by following a set of criteria to idealize certain homing pigeon behaviors. These rules are as follows:

(1) Map and compass Pigeons can sense the Earth's magnetic field by using magnetism to form a map in their brains. To navigate, they use the altitude of the

sun as a compass to change their direction. The sun and magnetic particles become less important as they fly closer to their destination. The rules are established in this map and compass operator with the pigeon i's position X_i and velocity V_i. The positions and velocities in a D-dimension search space are modified with each iteration of the algorithm. The following equations can be used to find the new location X_i and velocity V_i of pigeon I at the t^{th} iteration, which is as follows:

$$V_i(t) = V_i(t-1).e^{-Rt} + \text{rand}.\left(-X_g - X_i(t-1)\right) \quad (1)$$

$$X_i(t) = X_i(t-1) + V_i(t) \quad (2)$$

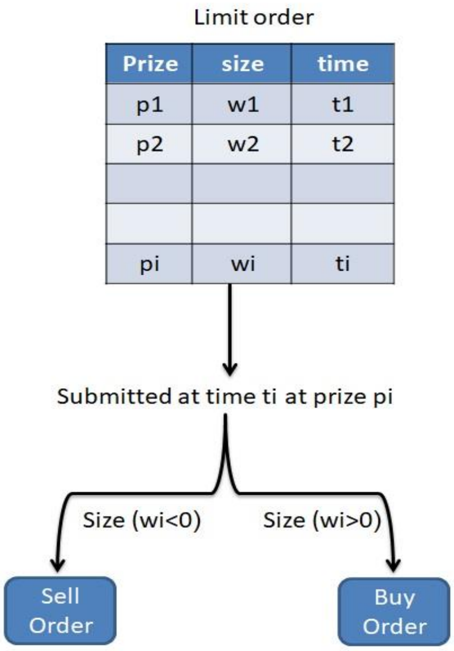

Fig. 1: The description of the limit order.

Where R is the map and compass factor, rand denotes a random integer, and X_g denotes the current global best position, which may be determined by comparing the locations of the pigeons. Comparing all possible flying positions, it becomes evident that flying in a right-centered pigeon's position is now the most optimum. Following Equation (3), each pigeon can modify its flying path by following this specific pigeon.

(2) Landmark operator: When pigeons fly close to their objective, they rely on nearby landmarks to guide them to their destination. They will likely fly directly to the destination if their route includes familiar sights. If they are distant from their goal and unfamiliar with the landmarks, they will likely follow pigeons already familiar with the locations.

When using the landmark operator, the number of pigeons is reduced by half in each generation by N_p. In contrast, the pigeons are far from their goal and are unfamiliar with the area landmarks. For simplicity, let Xc(t) be the center of several pigeon's locations at the tth iteration and assume that every pigeon is capable of flying directly to the destination. The following is the rule for updating the location of pigeon if at the tth iterations:

$$N_P(t) = \frac{N_P(t-1)}{2} \tag{3}$$

$$X_c(t) = \frac{\sum X_i(t).\,\mathrm{FITNESS}(X_i(t))}{N_P \sum \mathrm{FITNESS}(X_i(t))} \tag{4}$$

$$X_i(t) = X_i(t-1) + \mathrm{rand}.\,(X_c(t) - X_i(t-1)) \tag{5}$$

The fitness function evaluates each solution's fitness value based on True positive rate (TPR), False positive rate (FPR), and the number of features. The cost function is calculated based on the TPR, and FPR as shown in Equation 6 [4].

$$\mathrm{Fitness} = W_1 * \frac{SF}{TF} + W_2 * FPR + W_2 * \frac{1}{TPR} \tag{6}$$

Where w1+w2+w3=1, TF is the features total number, SF is the selected sub features by the PIO model.

The performance of the pigeon individual is measured by its fitness (6) value. The optimal position of the $N_c th$ iteration for each pigeon may be indicated with X_p, while the best position of the $N_c th$ iteration can be denoted with $X_p = Min\ (X_{i1}, X_{i2}, X_{i3}, \ldots .X_{iNc})$.

To each bird. After then, we compare the fitness of all of the pigeons and determine the new optimum solution. If $N_c = 1$ is reached, the map and compass operator is stopped and the next operator is operated instead.

The fitness values of all pigeons should be used to rank them. As predicted by Equation (6), half of the pigeons with poor fitness will trail behind the other half of the birds with higher fitness (6). According to Equation (4), we can then locate the center of all pigeons, and this center is the desired location to arrive at. When all pigeons alter their flight path following Equation 1, they will all arrive at their destination (5). After that, make a note of the optimal solution parameters and the optimal cost. If $N_c \geq 1$ is encountered, the landmark operator is terminated, and the results are printed out on the screen.

3. Limit Order Books Dataset

All of the experiments are carried out utilizing the LOBSTER dataset, which contains a wealth of information on the market activity of each stock traded on the NASDAQ exchange. Some important papers and journals in this area, such as Quantitative Finance, have highlighted LOBSTER as one of the data sources they work with. The LOB datasets are available for each stock exchange on the

NASDAQ. Limit order book data tool LOBSTER is an online limit order book data tool that provides high-quality limit order book data that is simple to use and provides an event-by-event explanation of every micro-scale market activity. LOBSTER offers various pricing levels, with the option to choose from up to 200 various price levels based on - if 'trades and quotations' are required, i.e., level 1, level 10, or level 20. In addition, LOBSTER offers comprehensive event details such as the following: Unique identifiers are assigned to each of the submissions, cancellations, and executions (both visible and hidden) that occur in the NASDAQ platform between 09:30 am – 04:00 pm on each trading day. The following information is provided for each limit order event that occurs within the specified price range: Price, size, and buy/sell indication are all shown together with a time stamp (Jonas et al., 2021; Ntakaris et al., 2019).

LOBSTER provides the most recent information available. In a database, data from the 27th of June 2007 up to the day before yesterday. Weekends and public holidays are not trading days, and as a result, these days are omitted from all of the analyses conducted. LOBSTER creates a 'message' and an ‘orderbook’ file for each trading day when a particular ticker is in the process of doing business. The 'orderbook' file contains the modification of the limit order book up to the number of levels specified over time. The 'message' file includes indications for the kind of occurrence that is triggering an update of the limit order book in the desired price range, as well as the price range that was requested.

LOBSTER data is organized in two separate files: 1) The message file contains a record of each market order, limit order, and cancellation that happens. 2) The order book file contains information on the market state immediately following the occurrence of the related event (i.e., the total volume of buy or sell orders at each price). The message and order book files are given in CSV format, which means that they may be read easily by any statistical software.

The bid orders and ask orders categories are the two major types of orders included in the LOB dataset (Jonas et al., 2021). Table 1 shows the structure of the order book file.

Table 1: The Variable Explanation for the Message File

Time	**Seconds after midnight with decimal precision of at least milliseconds and up to nanoseconds depending on the period requested**
Order ID	Unique order reference number
Size	Number of shares
Price	Dollar price times 10000
Event Type	
1	Submission of a new limit order
2	Cancellation (partial deletion of a limit order)

3	Deletion (total deletion of a limit order)
4	Execution of a visible limit order
5	Execution of a hidden limit order

The limit order deletion (event type 3) in the second line of the 'message' file removes 100 shares from the ask side for 118600. The change in the 'orderbook' file from lines one to two corresponds to this removal of liquidity. The volume available at the best ask price of 118600 drops from 9484 to 9384 shares.

The LOB searching and retrieval mechanism are working, as illustrated in Fig. 2.

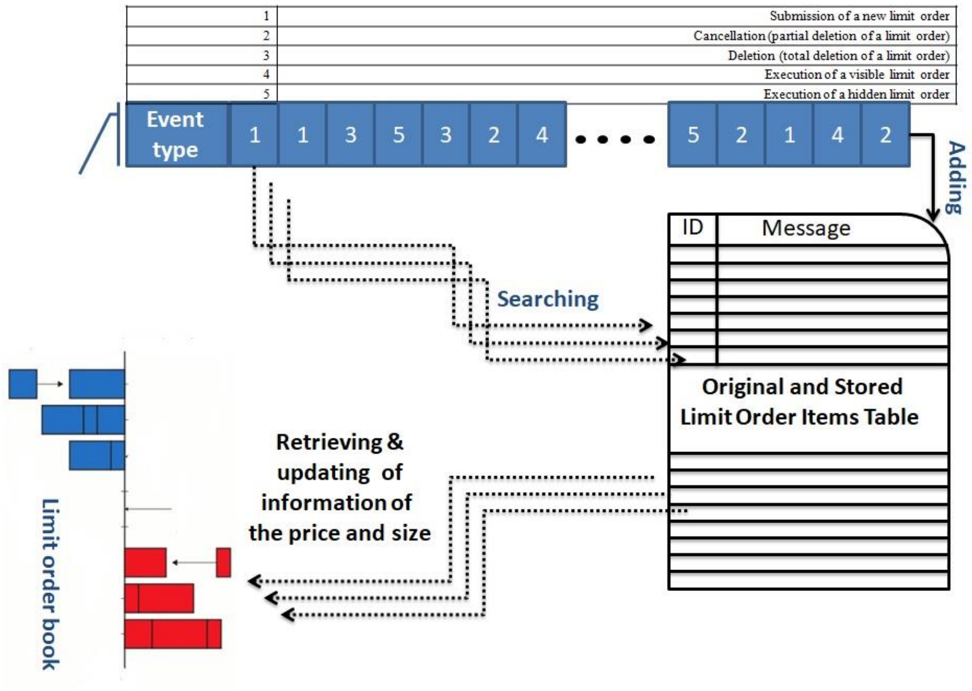
1 Submission of a new limit order
2 Cancellation (partial deletion of a limit order)
3 Deletion (total deletion of a limit order)
4 Execution of a visible limit order
5 Execution of a hidden limit order
Event type
1 1 3 5 3 2 4 • • • • 5 2 1 4 2
Adding
ID
Message
Searching
Original and Stored Limit Order Items Table
Retrieving & updating of information of the price and size
Limit order book

Fig. 2: The mechanism of the LOB working.

4. Research Methods

The proposed optimal limit order book activity comprises the following fundamental building phases: (1) Data preprocessing and labeling. In the first phase of the investigation, scaling processing is based on min-max scaler scales. It transforms each variable separately to fall within a specified range on the training set. (2) Pigeon-inspired optimization-based feature subset selection. Before feed, the data sets to the deep neural network, the sub-features are selected based on the Pigeon-inspired optimization algorithm. (3) The event type classification. Inception deep learning convolutional neural network architecture is used for classification five event types Submission of a new limit order (SNLO), Cancellation (partial deletion of a limit order) (PDLO), Deletion (total deletion of a limit order) (DLO), Execution of a visible limit order (EVLO), and Execution of a hidden limit order (EHLO). (4) Performance assessment of the system based on Sensitivity, Accuracy, False Positive Rate, and F-score. These phases are described in detail in the following section, along with the steps involved and the characteristic features for each phase (refer to Figure 3).

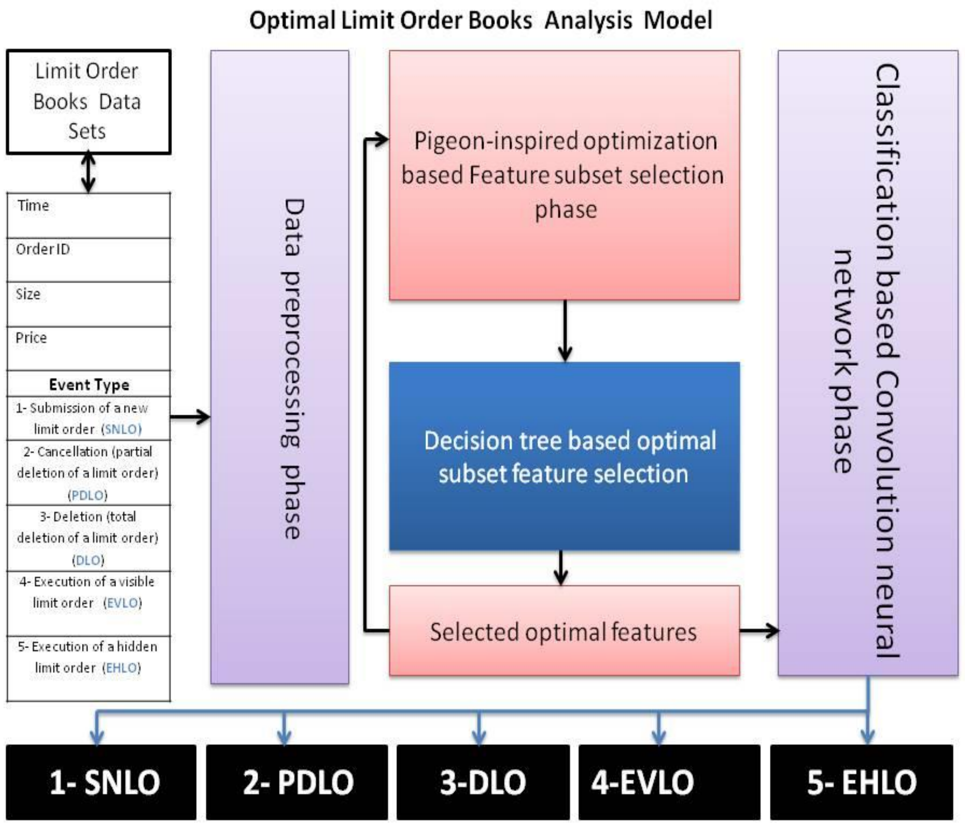
Optimal Limit Order Books Analysis Model
Limit Order Books Data Sets
Time
Order ID
Size
Price
Event Type
1- Submission of a new limit order (SNLO)
2- Cancellation (partial deletion of a limit order) (PDLO)
3- Deletion (total deletion of a limit order) (DLO)
4- Execution of a visible limit order (EVLO)
5- Execution of a hidden limit order (EHLO)
Data preprocessing phase
Pigeon-inspired optimization based Feature subset selection phase
Decision tree based optimal subset feature selection
Selected optimal features
Classification based Convolution neural network phase
1- SNLO
2- PDLO
3-DLO
4-EVLO
5- EHLO

Fig. 3: The architecture of the limit order book analysis.

4.1. Data Pre-processing Phase

There are five distinct values of target labels for the categorization assignment to choose from. The classifier is used to perform the assignment of continuous variables to discrete classes by calculating the result distribution of the training set and then applying it to the test set (Haase et al., 2021).

4.2. Pigeon-Inspired Optimization-Based Feature Subset Selection Phase

The PIO LOB feature selection is evaluated using the Decision Tree (DT) classifier Rajeshkanna and Arunesh (2020) from the scikit-learn project in Python since DT is more capable of dealing with feature interaction than other basic classifiers. The steps of the PIO LOB subset feature selection are illustrated as follows.

For choosing the best characteristics of LOB data, the complete implementation process for PIO is as follows. Start by initializing the parameters of the PIO algorithm, such as solution space dimension *D*, the population size N_p, map and compass factor *R*, and the number of iteration N_c. The next step is to assign a random velocity and position to each pigeon. Determine the fitness of the present pigeon and the best path for it to follow. We utilize the map and compass operator to update the velocity and path of each pigeon in the group.

Split LOB dataset into training and test sets.

- **Training phase**:
  - Initialize parameters
    - Space dimension (*D*)
    - Compas facto (*R*)
    - Number of iteration (N_c & N_p)
  - Initialize X_i for each pigeon randomly
  - Determine the fitness of the present pigeon
  - Evaluate pigeons $(X_{i1}, X_{i2}, X_{i3}, \dots . X_{iNc})$
  - Best pigeon $X_g \leftarrow$ minimum fitness value
- If $N_c \geq 1 \;\; \rightarrow \;\; N_c = N_c - 1$
  - Train the model with the selected feature using a Decision tree classifier
  - Test the model with the selected feature using a Decision tree classifier
  - Evaluate the model with the fitness function
  - Update the X_p and X_g
  - Update the pigeon velocity and its path
  - Return
- Elseif $N_p \geq 1 \;\; \rightarrow \;\; N_p = N_p/2$
  - Sort Pigeons by the fitness values
  - Computer $N_p\,(t) = \frac{N_p(t-1)}{2}$
  - Update pigeon position X_g
  - Return
  - Else
  - Return global solution X_g and selected sub-features
- **Test phase:** a: Use features selected
- Compute Error Rate for the Test set.

To train and verify the subset of features indicated by the proposed feature selection, a decision tree (DT) is utilized in the Pigeon swarm optimizer. After the obtained optimized LOB sub-feature is selected, it feeds to the deep learning for classification.

4.3. Classification based on Deep Learning model

The proposed deep network architecture consists of three primary blocks: convolutional layers, an Inception Module, and a Dense layer, as seen in Figure 4. The CNNs and Inception Modules are utilized to classify and automate feature extraction, particularly challenging in financial applications. Weights are vested during the inference process, and features learned from a huge training set are data-adaptive, eliminating the restrictions above. Afterward, a Dense layer is utilized to capture further temporal dependencies among the features that have been generated. The convolutional layer captured the concise time-dependencies that take the LOB as inputs. This work requires the history of LOB prices and sizes as inputs to the suggested algorithm, which is straightforward. Figure 4 describe the details of the classification phase.

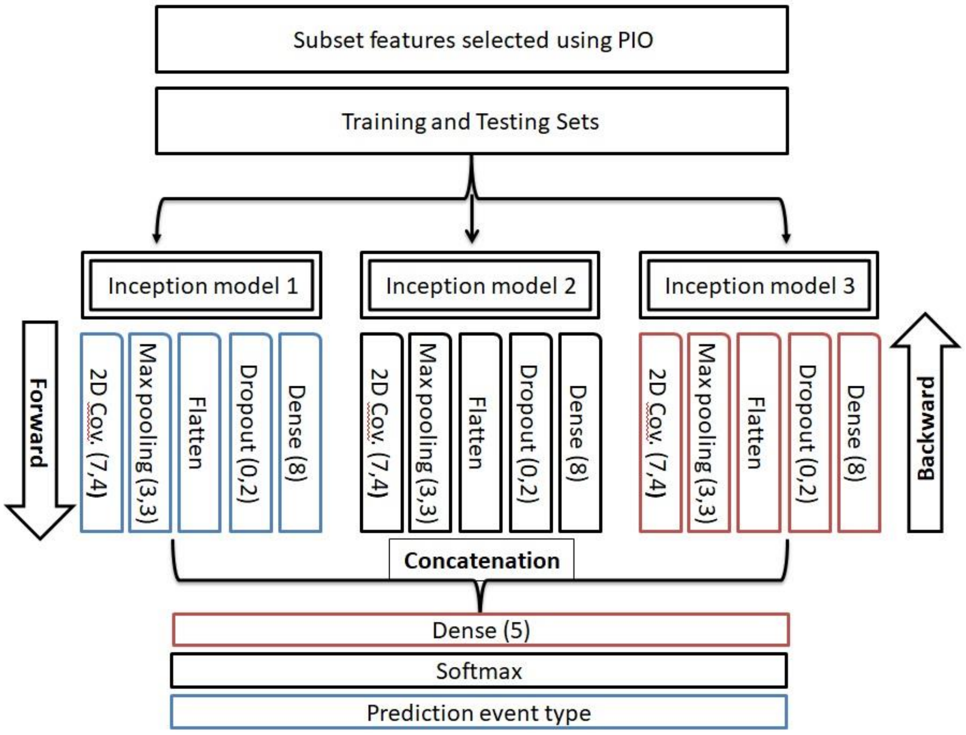

Fig. 4: Classification based on deep learning Inception models.

4.4. Performance metrics

The list of evaluation metrics that will be utilized to evaluate the proposed approach is defined in this section. The False Positive Rate (FPR) and True Positive Rate (TPR) metrics are used by many researchers to evaluate their feature and classification accuracy. The confusion matrix is used to calculate all the metrics that were chosen. It includes four primary parameters, namely: TP (True Positive), TN (True Negative), FP (False Positive), and FN (False Negative) (Hadeel et al., 2020). Table 2 describes the measure and its Equation.

5. Experimental Results and ANOVA Analysis

Experiments are performed only using the order book files. The training dataset consists of Intel Corporation's (INTC) LOB data from 04-02-2019 to 31-05-2019, corresponding to a total of 82 files. In contrast, the test dataset consists of Intel Corporation's LOB data from 03-06-2019 to 28-06-2019, obtained from 20 other files. All experiments are performed on snapshots of LOB with a depth of five representing the number of limit orders levels separated by hash size for each side of the order book (i.e., each row in the order book files corresponds to a vector of length 20). The LOB has two primary output data files; one includes the LOB data and the associated depths up to the fifth-best ask and bid, and the other provides the corresponding order events.

Table 2: The Evaluation Measures

Measure	**Equation**
Sensitivity (True Positive Rate (TPR), Recall)	$\mathrm{TPR} = \frac{TP}{TP + FN}$
Accuracy	$\mathrm{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$
False Positive Rate (FPR or False Alarms):	$\mathrm{FPR} = \frac{FP}{TN + FP}$
F-score (F-measure):	$\mathrm{F-score} = \frac{2 * TP}{2 * TP + FN + FP}$

Table 3 shows the different tickers in NASDAQ, such as Amazon (AMZN), Apple (AAPL), Google (GOOG), Intel (INTC), and Microsoft (MSFT). Also, Table 2 shows the distribution of the class labels for each five-level LOB dataset.

Table 3: The Distribution of the Class Labels for Each Ticker in the LOB Dataset

Label	**Apple (AAPL)**	**Google (GOOG)**	**Amazon (AMZN)**	**Intel (INTC)**	**Microsoft (MSFT)**	**Total**
1	143821	54907	77381	283287	293275	852671
2	2324	16	450	7863	5181	15834
3	120451	46072	66685	257397	263929	754534
4	23658	7764	8974	28923	29798	99117
5	11332	3913	2444	3559	3616	24864

Based on the combined LOB dataset of all tickers, Table 4 illustrates the distribution of each class label for the training and testing phases. The class labels are ordered in descending order from highest to lowest based on the total number of samples. As seen in this Table, the majority class is 1 due to the overall sample count of 852671, the training set size of 681485, and the testing set of 171186 samples. In contrast, the minority class is 2 due to the overall sample count of 15834, the training set of 12693, and the testing set of 3141 samples.

Several experiments on the selected LOB dataset were conducted to obtain comparative results using the proposed model. Google Colaboratory is a free online cloud-based Jupyter notebook platform that allows us to train the machine learning and deep learning models on computers provided for free by Google. The system contains CPUs, GPUs, and TPUs units, and it is accessible from anywhere in the world. Google Colab is an online browser-based platform that allows researchers to build deep learning models using the Keras library, which can be utilized inside the Python programming language. Furthermore, the experiments were carried out on a Google Colab's Virtual Machine, which had the following specifications outlined in detail: According to Google, the system includes two CPUs, 12 Gigabytes of RAM, and a hard drive with a capacity of 70 Gigabytes.

Table 4: The distribution of the class labels of the merged LOB dataset

Label	**Total**	**Train**	**Test**
1	852671	681485	171186
3	754534	604110	150424
4	99117	79332	19785
5	24864	19996	4868
2	15834	12693	3141

In this part, the PIO feature selection methodology is evaluated on the LOB dataset. The feature selection method is evaluated using the DT classifier from the scikit-learn project in Python since DT is more capable of dealing with feature interaction than other basic classifiers. To ensure that the algorithms are fair, the data preparation processes are applied to the LOB dataset. The PIO algorithm's experimental results are calculated based on TPR, FPR, F-score, and accuracy. As a result, the experimental results may differ from those described in similar studies.

The PIO approach can be made in a variety of manners by configuring the parameters of the PIO algorithm in different ways. Several experiments were carried out to examine the feasibility and efficacy of our suggested PIO method, and further comparative experimental findings are also included. The following parameters of the PIO algorithm were initialized: compass factor R = 0.09, Np = 64, and number of iterations Nc= 10.

Table 5 shows the TPR, FPR, and the selected set of attributes findings of the DT model using the AAPL LOB test set with Min-Max normalization. The third iteration had the greatest TPR, but it suffers from high FPR. The suggested PIO method achieves roughly the same outcomes for Xg, so the model trained using PIO features is more stable. Generally, the experimental results also show that the PIO algorithm is much better in stability and superiority.

Similarly, Table 6 presents the results of the PIO optimizer based on the set of features generated during the training of the DT classifier with the merged LOB dataset. Table 6 shows the findings of TPR, FPR, and the selected set of attributes for the best pigeon Xpg and the global best pigeon Xg using the merged LOB test set with Min-Max normalization. The last iteration had the greatest TPR, but it suffers from high FPR. The suggested PIO method achieves roughly the same outcomes for Xg, so the model trained using PIO features is more stable.

Table 5: The results of the PIO optimizer using AAPL LOB data

	Xg: GB = global best			**Xpg: PG**		
Iteration	TPR	FPR	Attributes	TPR	FPR	Attributes
0	0.625	0.991	[9, 12, 18]	0.596	0.962	[3, 7, 8, 11, 12, 14, 18]
1	0.625	0.991	[9, 12, 18]	0.593	0.951	[0, 3, 11, 12, 13, 14, 18]
2	0.625	0.991	[9, 12, 18]	0.615	0.987	[8, 10, 12, 14, 18]
3	0.625	0.991	[9, 12, 18]	0.606	0.985	[6, 7, 8, 12, 18]
4	0.625	0.991	[9, 12, 18]	0.603	0.989	[6, 8, 14, 15, 16]
5	0.625	0.991	[9, 12, 18]	0.605	0.972	[4, 8, 11, 14, 17]
6	0.625	0.991	[9, 12, 18]	0.607	0.985	[2, 6, 15, 16, 17]
7	0.625	0.991	[9, 12, 18]	0.604	0.974	[0, 1, 8, 11]
8	0.625	0.991	[9, 12, 18]	0.599	0.970	[2, 8, 11, 14, 17]
9	0.625	0.991	[9, 12, 18]	0.615	0.985	[9, 13, 14, 17]

Table 6: The results of The PIO optimizer using the merged LOB data

	Xg: GB = global best			**Xpg: PG**		
Iteration	TPR	FPR	Attributes	TPR	FPR	Attributes
0	0.620	0.948	[3, 7, 8, 11, 15]	0.634	0.997	[4, 8, 9, 14, 16, 18]
1	0.665	0.996	[6, 8, 12, 16, 17, 18]	0.665	0.996	[6, 8, 12, 16, 17, 18]
2	0.665	0.996	[6, 8, 12, 16, 17, 18]	0.629	0.995	[4, 6, 8, 11, 18]
3	0.869	1.000	[0, 2, 6, 14]	0.869	1.000	[0, 2, 6, 14]
4	0.869	1.000	[0, 2, 6, 14]	0.623	0.990	[7, 10, 13, 18]
5	0.869	1.000	[0, 2, 6, 14]	0.665	0.997	[6, 8, 10, 14, 17, 18]
6	0.869	1.000	[0, 2, 6, 14]	0.617	0.962	[3, 4, 6, 7, 16, 18]
7	0.876	1.000	[0, 2, 6]	0.876	1.000	[0, 2, 6]
8	0.876	1.000	[0, 2, 6]	0.864	0.998	[0, 2, 6, 8, 12, 14, 16, 18]
9	0.876	1.000	[0, 2, 6]	0.870	0.999	[4, 8, 10, 14]

As shown in Table 7, the results of the PIO optimizer based on Min-Max normalization were obtained by utilizing the set of features generated during the training of the DT classifier with the training AAPL LOB dataset. The results present an average of 10 operations of the PIO algorithm, and we can see at the 5th iteration TPR has the highest. In contrast, the eighth iteration achieved the highest FPR. The F-score measure has a better indication than other measures since it harmoniously presents the precision and recall results. Also, Table 7 illustrates the results of the F1-score for all iterations. The third iteration has the highest value of the F1-score when compared to the previous iterations. An additional factor that influences the quality of a solution for a feature selection technique is the number of features that have been selected. Development and testing of the model are influenced by the number of attributes included.

The following training and testing steps are performed for the Deep Learning (DL) model: a total of 1024 samples are created in each training batch, with each sample consisting of 10 consecutive LOB states. Class labels are converted to their one-hot representation. The selected optimizer is Adam. Its Keras implementation is chosen, and default values for its hyperparameters are kept (lr= 0.01). The categorical cross-entropy loss function is chosen due to its suitability for multi-class classification tasks. Manual hyperparameter investigation reveals that 100 training epochs are the best number of training epochs when considering the limitations of computing resources.

The dimensionality of the input data tends to have a large effect on the effectiveness of the classification model. At the same time, the low-dimensional representations produced by the PIO obtain competitive results in classification accuracy. The proposed CNN model is trained to predict the class label using the selected features from the different LOB datasets and the classification metrics used to assess and compare out-of-sample model performances. The model was evaluated using a variety of LOB data to determine the efficiency of the MLP designs. Table 9 summarizes the experimental results obtained from the different LOB datasets utilizing the deep learning model.

The proposed model was trained using five stocks: Amazon (AMZN), Apple (AAPL), Google (GOOG), Intel (INTC), and Microsoft (MSFT). Also, Table 8 shows the results of the DL for the combined LOB data, with accuracy, recall, and F-score values of 0.77, 0.43, and 0.436, respectively. Table 9 shows that the DL achieves the best accuracy, recall, and F-score outcomes on AAPL LOB data, with accuracy, recall, and F-score values of 0.749, 1, and 0.645, respectively. Furthermore, the classifier outperforms all other representations in the dataset regarding GOOG LOB data, which has the highest accuracy.

Table 9: The Results of the Deep Learning Model

LOB dataset	Accuracy	Precision	Recall	F1	MSE
AAPL	0.749	0.476	1	0.645	0.2096
GOOG	0.802	0.488	0.50	0.50	0.1976

Merged all LOB files	0.77	0.43	0.436	0.436	0.225

Table 7: The Results of the PIO Algorithm using AAPL LOB Data for Each Iteration

	Accuracy			**F1**			**TPR**			**FPR**		
	Max	Mean	Std	Max	Mean	Std	Max	Mean	Std	Max	Mean	Std
0	0.47834	0.46358	0.00740	0.30913	0.29269	0.00798	0.60637	0.57309	0.01290	0.98933	0.96123	0.01374
1	0.47824	0.46335	0.00761	0.30848	0.29317	0.00757	0.59285	0.57016	0.01139	0.99147	0.96095	0.01345
2	0.48053	0.46326	0.00834	0.30987	0.29266	0.00673	0.60533	0.57205	0.01004	0.99147	0.96168	0.01425
3	0.47725	0.46271	0.00667	0.30416	0.29249	0.00627	0.59442	0.57311	0.01133	0.98933	0.96212	0.01371
4	0.47851	0.46410	0.00784	0.30907	0.29208	0.00891	0.60979	0.57454	0.01287	0.99360	0.96188	0.01390
5	0.47342	0.46170	0.00705	0.30588	0.28965	0.00815	0.60567	0.57646	0.01314	0.99147	0.96615	0.01416
6	0.47453	0.46097	0.00754	0.30389	0.28878	0.00893	0.60752	0.57674	0.01284	0.99360	0.96578	0.01556
7	0.47460	0.46202	0.00754	0.30584	0.28960	0.00815	0.60372	0.57827	0.01366	0.99360	0.96828	0.01388
8	0.47833	0.46279	0.00838	0.30540	0.29113	0.00873	0.59829	0.57589	0.01295	0.98933	0.96455	0.01381
9	0.47524	0.46159	0.00743	0.30323	0.29011	0.00678	0.59735	0.57706	0.01156	0.98933	0.96418	0.01394
10	0.47407	0.46294	0.00706	0.30568	0.29177	0.00893	0.60853	0.57253	0.01276	0.99360	0.96449	0.01449

Similarly, Table 8 summarizes the outcomes of the PIO optimizer for each iteration using the merged LOB data. The results present an average of 10 runs for the PIO algorithm. The results show that the seventh iteration has the highest TPR value and achieved the highest FPR. The accuracy and F1 score have the highest values at the last iteration compared to the previous iterations.

Table 8: The Results of the PIO Algorithm using the Merged LOB Data for Each Iteration

	Accuracy			**F1**			**TPR**			**FPR**		
	max	mean	std	max	mean	std	max	mean	std	max	mean	std
0	0.53486	0.51046	0.01040	0.37858	0.35458	0.01530	0.63874	0.60244	0.00930	0.99709	0.94912	0.01678
1	0.53424	0.50981	0.01053	0.37906	0.35338	0.01761	0.66500	0.60380	0.01123	0.99645	0.94988	0.01857

2	0.53454	0.51193	0.01323	0.38012	0.35414	0.01915	0.62918	0.60178	0.00939	0.99451	0.94987	0.02114
3	0.52840	0.50689	0.01054	0.37448	0.34630	0.02603	0.86886	0.61122	0.03489	1.00000	0.95638	0.02066
4	0.53590	0.51154	0.01250	0.37904	0.35322	0.01907	0.62803	0.60420	0.00922	0.99258	0.95133	0.02146
5	0.53035	0.50785	0.01048	0.37493	0.34840	0.01978	0.66508	0.60673	0.01278	0.99709	0.95557	0.01954
6	0.53634	0.50796	0.00991	0.37896	0.35193	0.01369	0.61972	0.60238	0.00824	0.99000	0.95254	0.01675
7	0.53476	0.50608	0.01330	0.37891	0.34120	0.03114	0.87578	0.61315	0.03665	1.00000	0.95995	0.02336
8	0.53654	0.50915	0.01210	0.37962	0.34945	0.02525	0.86372	0.60853	0.03407	0.99806	0.95324	0.02105
9	0.53012	0.50931	0.01119	0.37583	0.35110	0.02493	0.86963	0.60823	0.03464	0.99935	0.95029	0.01905
10	0.53797	0.51154	0.01222	0.38078	0.35440	0.02065	0.66409	0.60273	0.01382	0.99670	0.94790	0.01990

Fig. 5 illustrates the accuracy of the deep learning model during training and testing after using the subset of features selected by the PIO. The figure illustrates how the number of features used in the training and testing stages affects the model's accuracy. Furthermore, as the number of epochs increases, the accuracy values in the training phase become increasingly equal. On the other hand, the accuracy of the testing step does not follow a smooth curve. As illustrated in Figure 5, the accuracy of the training step converges linearly over the last 60 iterations and continues to improve the quality of the solution. In contrast, accuracy in the testing stage converges much more slowly than accuracy in the training phase, and progress in the quality of the solution changes entirely throughout the number of iterations. Based on the data obtained, the model was more efficient during the training phase than during the testing phase.

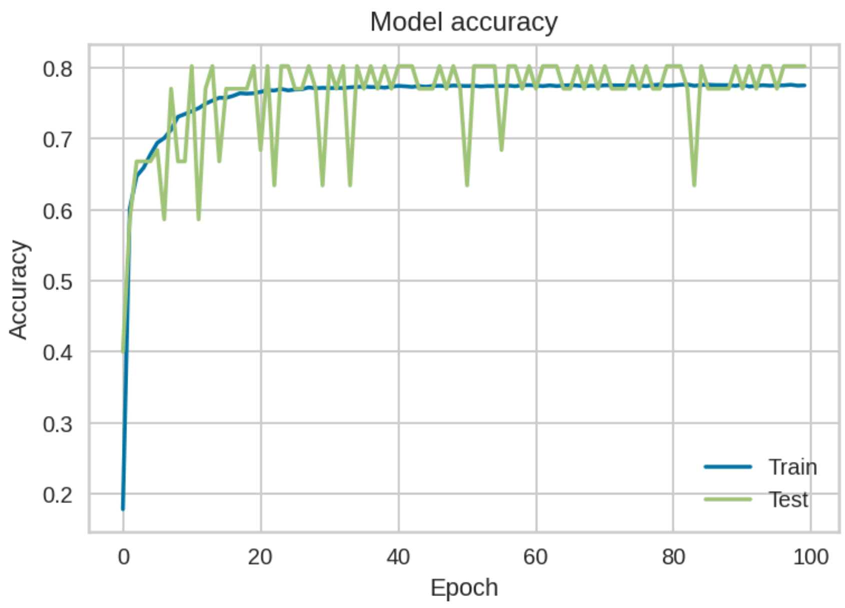

Fig. 5: The accuracy plot for the deep learning model using Google LOB data.

Table 10 illustrates the performance of the proposed approach in comparison to the most recent state-of-the-art models. The mean accuracy, recall, precision, and F1 score are calculated over all folds to determine overall performance. Because the LOBSTER/FI-2010 dataset is not well balanced, we recommend concentrating on F1 score performance to make fair comparisons. To compare our model to all of the current models, we used Single-Layer-Feedforward Network (SLFN), Attention-augmented-Bilinear-Network with one hidden layer (B(TABL)) and two hidden layers (C(TABL)), and DeepLOB.

Table 10 shows the results of the comparison with the most recent state-of-the-art models. The proposed classifier produces significantly better performance than the state-of-the-art techniques in terms of accuracy. This may be explained by the fact that PIO optimizers can capture the relationships between different features.

Table 10: The comparison with the State-Of-Art models

Model	**Dataset**	**Accuracy**	**Precision**	**Recall**	**F1**
The proposed method	AAPL LOBSTER	74.9	47.6	1	64.5
The proposed method	GOOG LOBSTER	80.2	48.8	50	50
The proposed method	All stocks LOBSTER	77	43	43.6	43.6
SLFN [4]	FI-2010	-	43	42	42
B(TABL) [5]	FI-2010	69.31	69.31	69.41	68.86
C(TABL) [5]	FI-2010	74.07	73.51	73.80	73.52
DeepLOB [2]	London Stock Exchange (LSE)	63.93	63.43	63.93	63.49

Also, the above results are supported by the analysis of variance ANOVA to test for the significant difference among the means of all items according to their event types. As shown in Table (11), the results clearly showed significant differences among the means of all event types for all the items except for Direction, Bid_Price_1, Bize_2, Bid_Size_3, Ask_Size_5, and Bid_Size_5 at significance level 0.05. While, The results show that there are significant differences among the means of all event types for all the items except for Bid_Price_1, Bize_2, Bid_Size_3, and Bid_Size_5 at significance level 0.1.

Table 11: LOB ANOVA Analysis

		Sum of Squares	**df**	**Mean Square**	**F**	**Sig.**
Order_ID	Between Groups	4.407E17	4	1.102E17	5230.897	.000
	Within Groups	3.449E17	16378	2.106E13		
	Total	7.856E17	16382			
Size	Between Groups	290135.638	4	72533.909	11.088	.000
	Within Groups	1.071E8	16378	6541.806		
	Total	1.074E8	16382			
Price	Between Groups	6.663E8	4	1.666E8	3.184	.013
	Within Groups	8.569E11	16378	5.232E7		
	Total	8.575E11	16382			

Direction	Between Groups	8.073	4	2.018	2.062	.083
	Within Groups	16030.049	16378	.979		
	Total	16038.122	16382			
Ask_Price_1	Between Groups	1.202E9	4	3.004E8	5.833	.000
	Within Groups	8.434E11	16378	5.150E7		
	Total	8.446E11	16382			
Ask_Size_1	Between Groups	1592947.110	4	398236.778	12.577	.000
	Within Groups	5.186E8	16378	31664.239		
	Total	5.202E8	16382			
Bid_Price_1	Between Groups	4.190E8	4	1.048E8	1.943	.100
	Within Groups	8.830E11	16378	5.391E7		
	Total	8.834E11	16382			
Bid_Size_1	Between Groups	4466647.891	4	1116661.973	11.673	.000
	Within Groups	1.567E9	16378	95660.329		
	Total	1.571E9	16382			
Ask_Price_2	Between Groups	1.224E9	4	3.061E8	6.007	.000
	Within Groups	8.345E11	16378	5.095E7		
	Total	8.357E11	16382			
Ask_Size_2	Between Groups	776733.372	4	194183.343	3.739	.005
	Within Groups	8.506E8	16378	51933.551		
	Total	8.513E8	16382			
Bid_Price_2	Between Groups	6.672E8	4	1.668E8	3.071	.015
	Within Groups	8.897E11	16378	5.432E7		
	Total	8.903E11	16382			
Bid_Size_2	Between Groups	138361.349	4	34590.337	1.484	.204

	Within Groups	3.817E8	16378	23305.389		
	Total	3.818E8	16382			
Ask_Price_3	Between Groups	1.127E9	4	2.817E8	5.587	.000
	Within Groups	8.258E11	16378	5.042E7		
	Total	8.269E11	16382			
Ask_Size_3	Between Groups	740358.247	4	185089.562	4.178	.002
	Within Groups	7.256E8	16378	44304.009		
	Total	7.264E8	16382			
Bid_Price_3	Between Groups	9.113E8	4	2.278E8	4.254	.002
	Within Groups	8.772E11	16378	5.356E7		
	Total	8.782E11	16382			
Bid_Size_3	Between Groups	53031.828	4	13257.957	.646	.630
	Within Groups	3.362E8	16378	20525.674		
	Total	3.362E8	16382			
Ask_Price_4	Between Groups	1.007E9	4	2.518E8	5.003	.000
	Within Groups	8.242E11	16378	5.032E7		
	Total	8.252E11	16382			
Ask_Size_4	Between Groups	826028.846	4	206507.212	3.619	.006
	Within Groups	9.345E8	16378	57061.078		
	Total	9.354E8	16382			
Time_sec	Between Groups	3461975.525	4	865493.881	16.092	.000
	Within Groups	8.809E8	16378	53784.254		
	Total	8.843E8	16382			
Bid_Price_4	Between Groups	9.694E8	4	2.424E8	4.751	.001
	Within Groups	8.354E11	16378	5.101E7		

	Total	8.364E11	16382			
Bid_Size_4	Between Groups	328927.778	4	82231.945	2.755	.026
	Within Groups	4.889E8	16378	29849.590		
	Total	4.892E8	16382			
Ask_Price_5	Between Groups	8.352E8	4	2.088E8	4.188	.002
	Within Groups	8.166E11	16378	4.986E7		
	Total	8.174E11	16382			
Ask_Size_5	Between Groups	444183.722	4	111045.931	2.157	.071
	Within Groups	8.433E8	16378	51489.080		
	Total	8.437E8	16382			
Bid_Price_5	Between Groups	1.052E9	4	2.629E8	5.498	.000
	Within Groups	7.833E11	16378	4.782E7		
	Total	7.843E11	16382			
Bid_Size_5	Between Groups	183467.155	4	45866.789	.975	.420
	Within Groups	7.704E8	16378	47038.797		
	Total	7.706E8	16382			

6. Conclusive Remarks

A new deep neural network with a high-frequency order chain includes convolutional and dense layers and inception units to predict future stock price movements (visual orders, cancels, deletes, visible and hidden execution) in a large-scale high-frequency LOB database that supports optimization for the operational performance of algorithmic trading. This paper proposes an optimal limit order book analysis for five stocks, including Amazon (AMZN), Apple (AAPL), Google (GOOG), Intel (INTC), and Microsoft (MSFT). It is based on deep learning and an enhanced pigeon-inspired (PIO) algorithm. The system reduces the dimentaionlty of LOB data sets by using a pigeon-inspired optimizer to determine the most significant features. Optimized LOB feature selection is evaluated using a Decision Tree (DT) classifier. Also, the results are supported by the analysis of variance ANOVA to test for the significant difference among the means of all items according to their event types which consequently proves that the experimental results are reasonably excellent.

In future extensions of this study, researchers may investigate more comprehensive deep learning models, which would be based on the feature extraction achieved by advanced optimization methods. In subsequent continuations of this work, we would like to investigate more detailed trading strategies using Reinforcement Learning, which is based on the feature extraction performed by PIO. In addition, researchers can apply this presented algorithm to solve other complicated optimization problems. Although the proposed model has many advantages, it also has several disadvantages. The most significant of them is that its performance is similar to that of the classical deep learning model, and the running time is a little longer. It requires tuning of the parameters of the deep learning layers.

References

Ahmet, M. O., Mehmet, U. G., Omer, B. S. (2020). Deep learning for financial applications: A survey. *Applied Soft Computing*, 93, 106384.

Avraam, T., Nikolaos, P., Anastasios, T., Juho, K., Moncef, G., and Alexandros, I. (2020). Using deep learning for price prediction by exploiting stationary limit order book features. *Applied Soft Computing*, 93, 106401。

Domowitz, I., and Wang, J. (1994). Auctions as algorithms: Computerized trade execution and price discovery. *Journal of Economic Dynamics and Control*, 18, 29-60.

Duan, H., and Qiao, P. (2014). Pigeon-inspired optimization: a new swarm intelligence optimizer for air robot path planning. *International Journal of Intelligent Computing and Cybernetics*, 7(1), 24-37.

Hadeel, A., Ahmad, S., Khair, E. S. (2020). A feature selection algorithm for intrusion detection system based on Pigeon Inspired Optimizer. *Expert Systems with Applications*, 148, 113249.

Joanna, W. (2020). Application of machine learning models and artificial intelligence to analyze annual financial statements to identify companies with unfair corporate culture. *Procedia Computer Science*, 176, 3037-3046.

Jonas, H., Huang, R., Nikolaus, H., Tomas, P., and Gus-tav, H. (2021). Limit order book data lobster dataset. https://lobsterdata.com/info/WhatIsLOBSTER.php.

Li, G., Xiao, M., and Guo, Y. (2019). Application of deep learning in stock market valuation index forecasting. *2019 IEEE 10th International Conference on Software Engineering and Service Science*, 551-554.

Marco, A., and Sasha, F. S. (2008). High frequency trading in a limit order book. *Quantitative Finance*, 8(3), 217-224.

Mason, A., Porter, S. W., Mark, M., Daniel, J. F., and Sam, D. H. (2013). Limit order books. Quantitative Finance, 13(11), 1709-1742.

Matthias, S. (2021). Deep reinforcement learning for the optimal placement of cryptocurrency limit orders. *European Journal of Operational Research*, https://doi.org/10.1016/j.ejor.2021.04.050.

Nakayama, A., Izumi, K., Sakaji, H., Matsushima, H., Shimada, T., and Yamada, K. (2019). Short-term stock price prediction by analysis of order pattern images. *2019 IEEE Conference on Computational Intelligence for Financial Engineering & Economics*, 1-5.

Nelli, F. (2015). Machine learning with scikit-learn. *Python Data Analytics*, 237-264.

Nousi P. (2019). Machine learning for forecasting mid-price movements using limit order book data. *IEEE Access*, 7, 64722-64736.

Nousi, P., Tsantekidis, A., Passalis, N., Ntakaris, A., Kanniainen, J., Tefas, A., Gabbouj, M., and Iosifidis, A. (2019). Machine learning for forecasting mid-price movements using limit order book data. *IEEE Access*, 7, 64722-64736.

Ntakaris, A., Mirone, G., Kanniainen, J., Gabbouj, M., and Iosifidis, A. (2019). Feature Engineering for mid-price prediction with deep learning. *IEEE Access*, 7, 82390-82412.

Palguna. D., and Pollak, I. (2016). Mid-price prediction in a limit order book. *IEEE Journal of Selected Topics in Signal Processing*, 10(6), 1083-1092.

Rajeshkanna, A., andArunesh, K. (2020). ID3 decision tree classification: an algorithmic perspective based on error rate. *2020 International Conference on Electronics and Sustainable Communication Systems*, 787-790.

Tran, D. T., Iosifidis, A., Kanniainen, J., and Gabbouj, M. (2018). Temporal attention-augmented bilinear network for financial time-series data analysis. *IEEE transactions on neural networks and learning systems*.

Tripathi, A., and Dixit, A. (2020). Limit order books: a systematic review of literature. Qualitative Research in Financial Markets, 12(4), 505-541.

Wray, A., Meades, M., and Cliff, D. (2020). Automated creation of a high-performing algorithmic trader via deep learning on level-2 limit order book data. 2020 IEEE Symposium Series on Computational Intelligence, 1067-1074.

Xiao, T., Huu, N. D., and Petko, S. K. (2019). Information content of the limit order book for crude oil futures price volatility. Energy Economics, 81, 584-597.

Xue, R., Ye, X., and Cao, X. (2021). Optimization of stock trading with additional information by Limit Order Book. *Automatica*, 127, 109507.

Yushan, Z., Han, H., Zhifeng, H., andZhou, H. (2019). Runtime analysis of pigeon-inspired optimizer based on average gain model. *2019 IEEE Congress on Evolutionary Computation*, 1165-1169.

Zhang, Z., Zohren, S., and Roberts, S. (2019). DeepLOB: deep convolutional neural networks for limit order books. *IEEE Transactions on Signal Processing*, 67(11), 3001-3012.