Expert Systems With Applications 213 (2023) 118899

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Deep LOB trading: Half a second please!

Jie Yin, Hoi Ying Wong *

Department of Statistics, The Chinese University of Hong Kong, Hong Kong

ARTICLE INFO

Keywords:
Limit order book
Trading system
Deep learning
Chinese A-share market
Imbalanced classification
Optimization

ABSTRACT

We introduce an expert deep-learning system for limit order book (LOB) trading for markets in which the stock tick frequency is longer than or close to 0.5 s, such as the Chinese A-share market. This half a second enables our system, which is trained with a deep-learning architecture, to integrate price prediction, trading signal generation, and optimization for capital allocation on trading signals altogether. It also leaves sufficient time to submit and execute orders before the next tick-report. Besides, we find that the number of signals generated from the system can be used to rank stocks for the preference of LOB trading. We test the system with simulation experiments and real data from the Chinese A-share market. The simulation demonstrates the characteristics of the trading system in different market sentiments, while the empirical study with real data confirms significant profits after factoring in transaction costs and risk requirements.

1. Introduction

High-frequency trading (HFT) is a popular form of algorithmic trading that leverages electronic trading tools and high-frequency financial data. A typical HFT algorithm is based on limit order book (LOB) data (Baldauf & Mollner, 2020; Brogaard et al., 2014; Kirilenko et al., 2017). The example of LOB data shown in Fig. 1 illustrates the bid and ask prices and their 5-level queues for a stock at two consecutive time points (ticks). In this study, we implement a LOB trading strategy to enter and exit the market by processing LOB data. For mature markets, such as the U.S. and Europe, the real-time LOB is event-based and updates at high speed of at least milliseconds and up to nanoseconds. The dataset from the Nasdaq Nordic stock market in Ntakaris et al. (2018) contains 100,000 events per stock per day, and the dataset from the London Stock Exchange in Zhang et al. (2019) contains 150,000. In contrast, exchanges in the Chinese A-share market publish the level II data, essentially 10-level LOB, every three seconds on average, with 4500–5000 daily ticks. This snapshot data provides us with the opportunity to leverage the longer tick-time interval and make profits using machine learning algorithms.

There are two main methods for modeling LOB. Statistical models (Abergel et al., 2020; Avellaneda & Stoikov, 2008; Cartea et al., 2014; Cont et al., 2010; Roşu, 2009) are proposed to describe the structure and dynamics of the LOB using various processes. However, the inherent noise and stochastic nature of financial markets severely limit the utilization of these models, as patterns of stock prices are too sophisticated and volatile to be captured. The development of computational resources and the availability of large amounts of historical data allow us to exploit novel machine learning techniques and develop data-driven models. Recently, the literature consists of important contributions (Kercheval & Zhang, 2015; Ntakaris et al., 2019; Passalis et al., 2018; Tran et al., 2018; Tsantekidis et al., 2017a, 2017b) to the short-term price movement forecasting of stocks based on LOB data. They mainly focus on the mid-price prediction but the actual transaction seldom occurs at the mid-price. The liquidity spread and transaction cost should be seriously considered in LOB trading. In this paper, we propose a LOB trading algorithm that exploits these predictions and achieves optimization with practical concerns within 0.5 s (more precisely, less than 0.03 s). Referring to classic trading strategies in the context of LOB data (Abudy & Wohl, 2018; Obizhaeva & Wang, 2013) and contemporary trading using deep learning (Pun et al., 2020; Tsang & Wong, 2020), we investigate the profits and risks associated with a novel deep LOB trading system.

We test our data-driven LOB trading system with two types of data: simulated data generated using the zero-intelligence agent-based model in Collver (2017) and real LOB data from the Chinese A-share market. The simulated data reflect different hypothetical market sentiments: uptrend, downtrend, and flat. We generate data for the three sentiments by manipulating the liquidity provider and market maker bid probability and then examine their effects on profitability. To replicate a market environment with unknown sentiments, we also simulate a dataset over a long-term period with mixed sentiments and test it using the rolling window method over data representing more than half a year. For the empirical data, we first implement our system on the benchmark dataset with 20 stocks provided by Huang et al. (2021). Analysis of both the simulation and benchmark datasets confirms that

* Corresponding author.
E-mail addresses: jyin@link.cuhk.edu.hk (J. Yin), hywong@cuhk.edu.hk (H.Y. Wong).

https://doi.org/10.1016/j.eswa.2022.118899
Received 16 December 2021; Received in revised form 21 July 2022; Accepted 21 September 2022
Available online 28 September 2022

Table 1
Summary statistics of the proprietary CS-100 dataset calculated from the training set for the first rolling window period. "SZ/SH" shows the number of stocks from the Shenzhen Stock Exchange and Shanghai Stock Exchange.

	Number of stocks (SZ/SH)	Ratio of long/ short signals	Mean (std) of mid-prices
Group 1	25 (14/11)	0.26/0.25	70.76 (69.21)
Group 2	25 (13/12)	0.22/0.21	53.44 (95.03)
Group 3	25 (20/5)	0.18/0.17	47.47 (103.82)
Group 4	25 (21/4)	0.12/0.13	11.25 (12.36)

Table 2
A sketch of the basic features of stock SZ300869 on July 21, 2021.

Time	$p_b^{(1)}(t)$	$v_b^{(1)}(t)$	$p_b^{(2)}(t)$	$v_b^{(2)}(t)$	…	$p_a^{(1)}(t)$	$v_a^{(1)}(t)$	$p_a^{(2)}(t)$	$v_a^{(2)}(t)$	…
09:32:15	62.65	200	62.62	500		62.72	200	62.84	100	
09:32:18	62.66	100	62.65	200		62.72	200	62.84	100	
09:32:21	62.66	500	62.62	100		62.84	100	62.94	900	
09:32:24	62.66	700	62.62	100		62.84	100	62.93	600	
09:32:27	62.66	700	62.62	100		62.84	100	62.93	600	

the signal quality improves as the number of signals increases. As such, we extend the empirical study to find more profitable stocks in a proprietary dataset of approximately 4000 stocks. Based on the number of signals combined with the in-sample performance, we select the top 100 stocks and assemble the proprietary CS-100 dataset described in Table 1. Regardless of the number of samples or the type of stocks, it is sufficient to evaluate the performance of our system. Consistent with the simulation and benchmark dataset results, we find that the system yields significant profits from the selected stocks in our proprietary dataset when controlling for the risk level and accounting for transaction costs.

Data preprocessing is the first stage of our system. Data normalization for features and labeling for signals are required for classification. Instead of simply labeling the mid-price movement as in Kercheval and Zhang (2015) and Tsantekidis et al. (2017a), we consider the direct trading actions, including *long*, *short*, and *none*. This approach is inspired by the previous application of deep learning to trade signals (long or short) in the context of VIX futures (Avellaneda et al., 2021). The signals are determined by the approximate wealth changes during a fixed and limited holding period, during which we set stop-loss and take-profit points. These settings are heterogeneous for different stocks, and we provide a method to assign the values of these hyperparameters based on the historical average ratio of the best ask to the best bid price. Furthermore, the threshold of signals can be adjusted according to investors' risk aversion. This type of labeling closely reflects actual transactions and earnings. The main contribution of this paper is a new integral deep LOB trading system that embraces model training, prediction, and optimization. Inspired by the model architecture in Zhang et al. (2018, 2019), we adopt the deep convolutional neural network (DCNN) model, which has a structure of convolutional layers and includes an inception module and LSTM module. However, because of the characteristics of imbalanced classification, we replace the categorical cross-entropy loss (CE) function with the focal loss (FL) function. It is necessary to pay more attention on the minority cases and capture the patterns of these valuable *long* and *short* signals. Then, the model trained daily or weekly can predict trading actions and the probability of each choice at every tick. The next step is to trade the securities based on the information yielded by the predictions. Instead of investing the same proportion consistently, we devise an optimization scheme using the fractional Kelly growth criterion under risk control, which is further achieved by the risk measure, value at risk (VaR). Based on the estimates of historical VaR and returns for successful/failed actions, we provide a theoretical closed-form solution for the optimal investment proportion. Finally, we demonstrate the significance of this novel system in multiple experiments.

We apply our system to the datasets introduced above and evaluate its performance using various approaches. First, some regular measures of out-of-sample classification under distinct settings are calculated and presented. Because the actual returns differ from the approximate wealth changes defined for labeling purposes, it is inappropriate to compare the accuracy without trading these predicted signals. After trading, the performance is obviously discrepant in terms of average profit per signal and accumulated profit over certain time periods. Furthermore, a t-test is conducted to examine if the average profit per signal is significantly greater than the profit per signal yielded with different settings. In general, the results obtained from both the simulation and empirical data support the idea of using the FL function and the optimization scheme. These mechanisms greatly enhance returns and reduce risks. One explanation is that the FL function helps to better capture signal patterns and the optimization process utilizes more information beyond the predicted actions and incorporates the historical performance of the training set. The profits gained in the proprietary CS-100 dataset are satisfactory under our realistic conditions considering the bid–ask spreads and transaction fees.

The half-second required by the system is put to good use in practice. For a single tick, the computation time required for the main procedures (from the beginning of data processing to the end of optimization) is recorded in Table 8. In addition to the algorithmic calculations, we reserve time for some mechanical order-related activities, such as order submission and execution in exchanges. This entire process is easily completed within 0.5 s. The Chinese A-share market can satisfy this tick-time condition with its update frequency of 3 s. Our empirical study shows that our deep LOB trading system is effective in the context of the Chinese market, which will encourage its use by other traders.

The rest of this paper is organized as follows. In Section 2, we introduce some basic concepts and describe the input LOB datasets. Then, the data preprocessing procedures are explained. Section 3 proposes our algo-trading system in detail. We evaluate the performance of the system in Section 4. Section 5 concludes the paper. The technical details are given in Appendix.

2. Data description and preparation

2.1. Basic concepts of LOB and market order

In order-driven markets, stocks are traded via matching buy and sell orders. When placing these orders, a trader has multiple choices, mainly divided into market orders and limit orders. A market order is an instruction to buy or sell a security immediately at the current market price without fixing the price. In contrast, a limit order is an instruction to buy or sell only at a price specified by the trader. Choosing different order types is a trade-off between execution price and waiting costs, as explained in Guilbaud and Pham (2013) and Roşu (2009).

The limit order book (LOB) provides a granular view of market data, listing all quotes at each price level. It contains information on both ask and bid sides. Furthermore, each side has a selection of limit orders placed at different prices, forming n levels. The first-level price of the ask side is termed the best ask price, and the difference between the best ask price and the best bid price is termed the spread. Fig. 1 shows a simplified LOB at time t and $t+1$ with the basic concepts. The changes of this LOB in Fig. 1 could be due to limit order cancellation or market order placement.

The market order is the most common and straightforward type of transaction that occurs in financial markets. When we expect the transaction to be executed as soon as possible, we issue a market sell order or market buy order. Most stock buyers and sellers choose one of these options. However, selling at the current best bid price or buying at the current best ask price means that traders give up the bid–ask spread. Although traders are not able to secure a satisfactory purchase price, they find it acceptable and have confidence about the future price

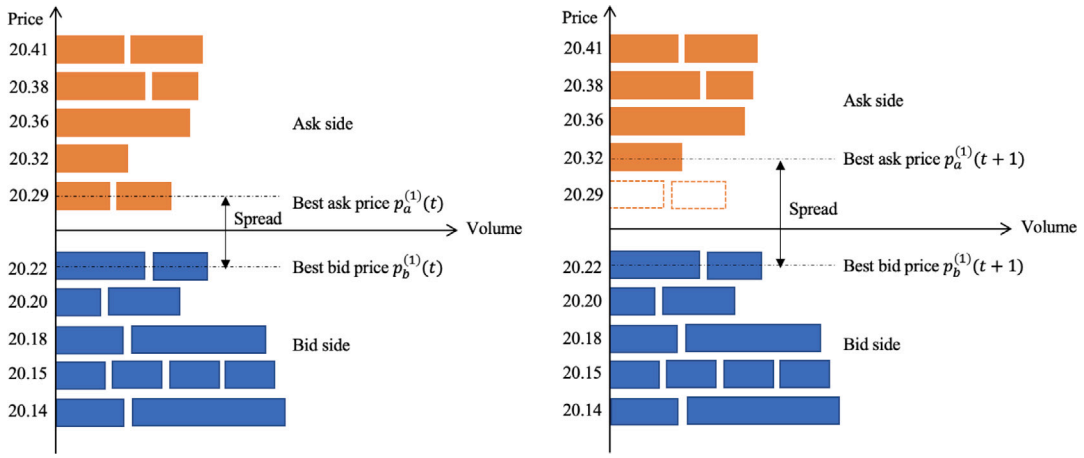

Fig. 1. A limit order book at time t and $t+1$. The first level of the ask side is changed because of the cancellation of limit orders or the arrival of market orders.

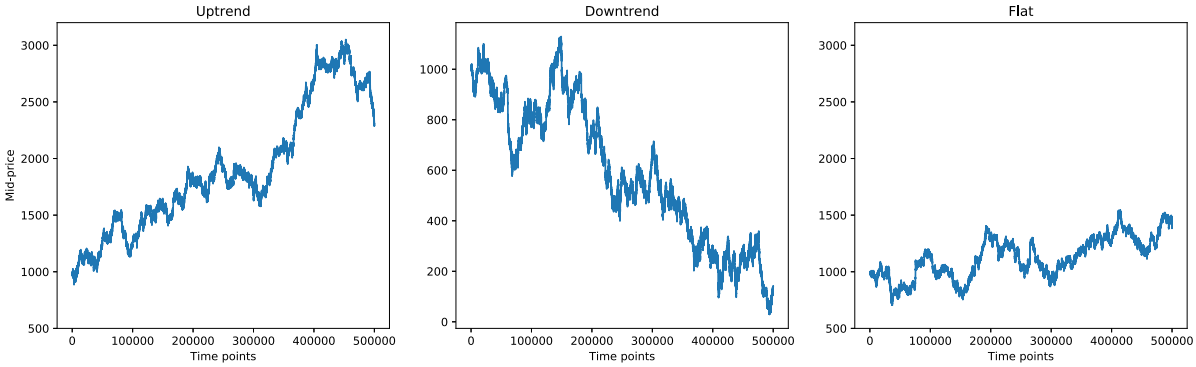

Fig. 2. Mid-prices of the simulated stocks for three hypothetical market sentiments.

trend. In HFT, market orders allow traders to enter or exit the market quickly, which is not the case in passive trading with limit orders.

HFT involves short holding periods and fast loss cutting. To reduce the risk of holding assets and to execute orders immediately, we sacrifice the bid–ask spread to take liquidity. Many studies do not use market orders because of difficulties in obtaining profits. For example, Zhang et al. (2019) perform trades on mid-prices. The current paper more closely reflects the real-world trading scenario, as we attempt to ensure the successful matching of these orders. Thus, the calculated order execution time is consistent with real-life scenarios, which helps to reflect the actual profits achievable on real exchanges.

2.2. Input LOB data

2.2.1. Simulation datasets

We simulate the LOB datasets using a zero-intelligence agent-based model. Fig. C.13 illustrates the framework of this simulation model, with the main objects being the exchange, traders, and orders. More details can be found in Collver (2017), and the code for this artificial market simulation can be downloaded from Github.[1] As the real LOB data may be unavailable to academic researchers, the simulated data can be used to validate and reproduce results from our trading system. The simulation datasets and the sample codes are uploaded to Github.[2]

The model parameters for one simulation dataset are given in Table C.11, and most of the parameters are the same as stated in Collver (2017, Table 1). To create different financial sentiments, we utilize the liquidity provider and market maker bid probability $q_{provide}$. Specifically, the constant $q_{provide} = 0.5$ corresponds to *flat*. When we randomly deviate $q_{provide}$ from 0.5 at certain time points, we get *uptrend* and *downtrend* to represent volatile markets. The mid-prices of the simulated stocks for three sentiments over the whole period are shown in Fig. 2.

We attempt to simulate market situations closer to the reality although no simulation can replace real data. There is another data presented in Fig. 3, as a mixture of different market sentiments. To demonstrate the consistency of our system and to make the results more convincing, we test it repeatedly using the rolling window method over a longer time period compared to data on single sentiment. If we match the data frequency with the actual LOB provider in the Chinese A-share market (approximately 4800 ticks per day), our simulated data cover a period of one year. By setting the size of the rolling window, we test approximately 21 periods of 6 days each. As such, the out-of-sample testing period covers half a year of data.

2.2.2. Empirical datasets from the Chinese A-share market

The simulation data help to demonstrate the profitability of the system under artificial markets and show the possible relationship between profits and the number of signals. Although the implementation is well designed and performs aptly, we acknowledge that the simulation data are limited in their ability to reveal the hidden characteristics of real markets. More experiments are required using historical datasets.

We empirically analyze LOB from two major Chinese A-share markets: the Shanghai Stock Exchange (SHSE) and Shenzhen Stock Exchange (SZSE). Both are pure order-driven trading mechanisms without designated market makers. Table 2 provides a sketch of the LOB data for the stock (code: 300869) in the SZSE. Such high-frequency tick-by-tick data are recorded based on the market quotes available to all

[1] https://github.com/JackBenny39/pyziabm
[2] https://github.com/jieyinjane/Deep-LOB-Trading

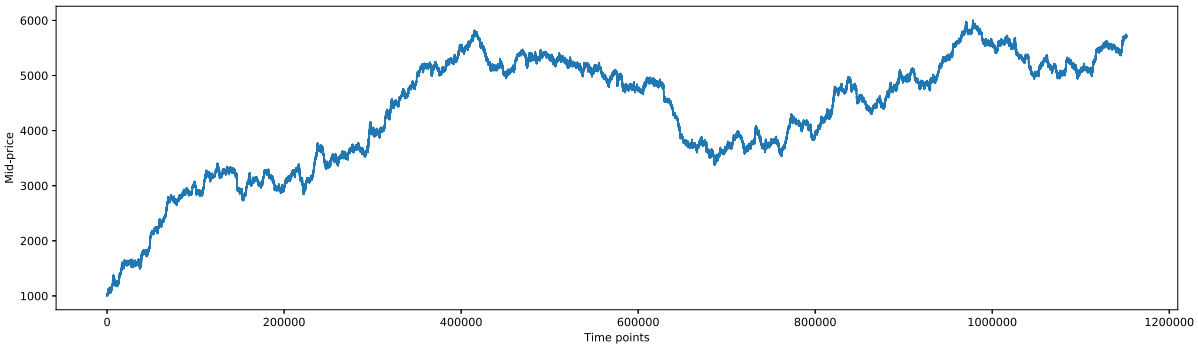

Fig. 3. Mid-prices of the simulated stock for a long-term period of approximately one year.

traders. Both exchanges update the LOB approximately every 3 s. On average, there are 4500–5000 ticks per stock per day. The time interval between two ticks is much longer than the interval of order books offered by other mature stock markets in which events are timestamped with decimal precision of at least milliseconds and up to nanoseconds. This tick-time interval provides traders with an opportunity to capture unreasonable orders and make profits.

Because of the trading rules of the Chinese A-share market, we restrict the daily trading interval to the continuous trading periods between 09:30:00–11:30:00 and 13:00:00–14:57:00 so that only normal trading activities occur and no auctions take place. Our analysis focuses on the data recorded during this period but excludes data after the daily price limit has been hit (10% of the previous day's closing price). This exclusion is due to the regulation of the Chinese A-share market.[3] When the price of any regular stock rises or falls by 10% within a day, trades are prohibited and no valid data can be collected. In addition, we combine different stocks into groups and train them together because Sirignano and Cont (2019) suggest that the universal features in limit order data can be extracted by deep learning techniques. Group training is also beneficial in terms of time and computational requirements. The specific details of the two datasets are described below.

- **CS-20 dataset**: A benchmark dataset provided by Huang et al. (2021). The authors provide LOB data for a small set of 20 stocks from the Chinese stock market that cover four consecutive months. The data are open access and publicly available on Github.[4] However, the time interval between two consecutive events is processed into 1 second. To match the actual snapshot flow of the SHSE and SZSE markets, we extract 10-level LOB data every 3 s.
- **Proprietary CS-100 dataset**: LOB data covering 100 stocks selected by ranking the number of signals combined with the in-sample performance from the entire Chinese stock pool covering around 4000 stocks. This proprietary dataset is provided by the Fintech company TradeMaster.[5] It contains 10-level real-time LOB covering five consecutive months. The 100 stocks are further separated into four groups according to their rankings; some summary statistics are displayed in Table 1.

Admittedly, the stocks given in the CS-20 dataset are not the most suitable for our deep LOB trading system. Some of them lack liquidity and exhibit large bid–ask spreads. Furthermore, the volatility of some stocks is too low because there are few orders or investors involved. Therefore, it is important to choose stocks to improve performance. Based on the results from the simulation and CS-20 data, we calculate the relationship between profits and the number of signals. According to this indicator and in-sample performance, we produce a proprietary dataset by selecting from approximately 4000 stocks in the Chinese A-share market. Interestingly, in Table 1, the mean of the mid-prices of the first group is the highest and that of the fourth group is the lowest, probably because when the price is very low, the ratio of spread to price is too high to be covered by the profit. This also explains why the number of signals is relatively small. Subsequent experiments confirm that our ranking and selection procedure is reasonable.

2.3. Normalization and labeling

The basic features derived from the LOB are prices and volumes at each level from both sides. Let $\boldsymbol{x}_t = \left[p_a^{(i)}(t), v_a^{(i)}(t), p_b^{(i)}(t), v_b^{(i)}(t)\right]_{i=1}^{n} \in \mathbb{R}^{4n}$, where $p_a^{(i)}(t)$ and $v_a^{(i)}(t)$ denote the ask price and ask volume, respectively, at level i at time t. Similarly, $p_b^{(i)}(t)$ and $v_b^{(i)}(t)$ denote the same values on the bid side. Table 2 shows some of the features (from the top two levels) of a sample stock SZ300869 from the Chinese A-share market. Further, n depends on the available levels, and usually equals 5 or 10. In all of the experiments, we set $n = 10$ to make full use of the LOB.

The input data (features) at time t are a combination of the most recent events, denoted as a sequence of vectors ${}_{t-\delta}\boldsymbol{X}_t = [\boldsymbol{x}_{t-\delta}, \ldots, \boldsymbol{x}_t]^T \in \mathbb{R}^{(\delta+1)\times 4n}$, where δ is the length of historical tick information. We set this to 49 in our experiments, which means that 50 historical ticks are used to predict the future signal at time t. While the prediction improves with the number of historical ticks, the improvement becomes mild for the length exceeding 50 historical ticks with our empirical dataset. The higher the number of ticks, the longer the time for normalization and prediction during the execution phase of the system. To complete order submission within 0.5 s, the use of 50 ticks strikes the best balance between performance and speed in our testing samples.

To normalize the range of features for easier deep learning, we use feature scaling (also known as data normalization). Normalization schemes include: z-score, min–max, decimal precision normalization (Ntakaris et al., 2018), and deep adaptive (learnable) input normalization (Passalis et al., 2019). We test all of them and find no significant difference among them in terms of profitability. We adopt the basic common standardization method (z-score) in the data preprocessing procedure in Section 4 for its advantages in storage saving, simplicity and a little faster computation. Afterward, we use the mean and standard deviation of the training and validation data to normalize the testing data for each feature. Future information is carefully avoided.

Then, the scaled features are used to predict trading actions, including *long* (+1), *short* (−1), and *none* (0). *long* means that we expect an upward trend of the stock price and buy before selling. *short* means the opposite, that is, selling before buying. If the stock is too stable

[3] For Shanghai Stock Exchange, http://english.sse.com.cn/start/trading/mechanism/, and for Shenzhen Stock Exchange, https://www.szse.cn/English/rules/siteRule/index.html.

[4] https://github.com/hkgsas/LOB

[5] https://www.trademastertech.com

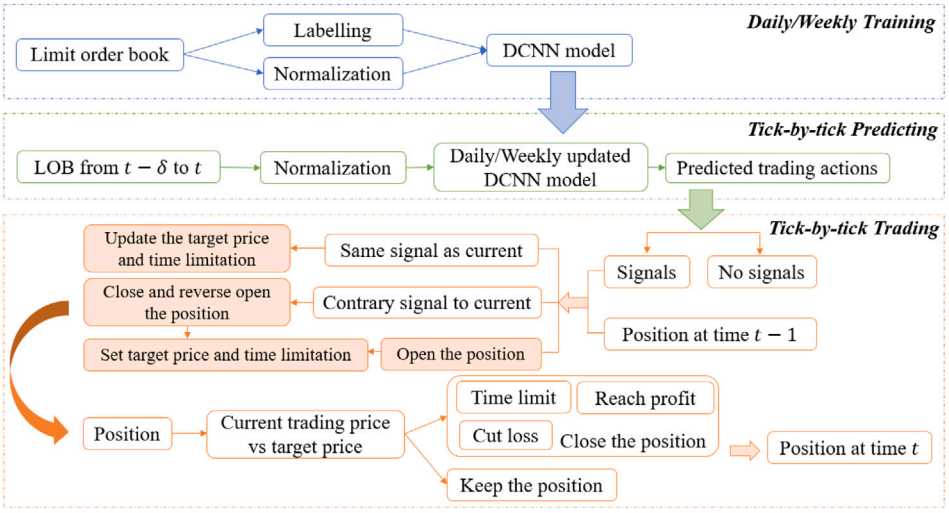

Fig. 4. Framework of our deep LOB trading system without optimization for selected stocks.

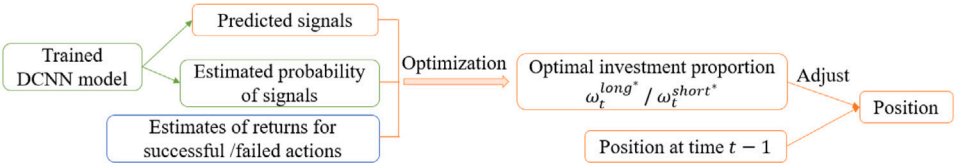

Fig. 5. Trading with optimization.

for a trend to be captured, we denote it as the third type, *none*. In other words, this scenario represents a multi-classification problem. To determine the labels, we first calculate the approximate wealth changes for signals. Specifically, for a *short* signal starting from time t, considering market orders, the stop-loss price is $\bar{\alpha}p_b(t)$ and the take-profit price is $\underline{\beta}p_b(t)$. If the upper and lower limits are not triggered, we use the best existing closing price $\min_{s\in[t,t+\Delta]} p_a(s)$ to measure the downward strength of the price. Because the corresponding wealth return with a closing price $p(s)$ is $1+(p_b(t)-p(s))/p_b(t)$, the approximate wealth change for a *short* signal is further simplified as

$$c_t^{short} \triangleq \begin{cases} 2-\underline{\beta} & \text{if } t < \min \underline{S} < \min\{\bar{S}, t+\Delta\} \\ 2-\bar{\alpha} & \text{if } t < \min \bar{S} < \min\{\underline{S}, t+\Delta\} \\ \frac{2p_b(t)-\min_{s\in[t,t+\Delta]} p_a(s)}{p_b(t)} & \text{if } \underline{S} = \bar{S} = \emptyset \end{cases} \tag{2.1}$$

where $\underline{S} := \{s \in [t, t+\Delta] : p_a(s) < \underline{\beta}p_b(t)\}$, $\bar{S} := \{s \in [t, t+\Delta] : p_a(s) > \bar{\alpha}p_b(t)\}$, $\bar{\alpha}$, and $\underline{\beta}$ are hyperparameters for stopping early. Δ is the length of the limited holding time and can be set at 1, 2, or more minutes by traders.

In cases where we expect a price trend in the opposite direction, we first buy and then sell. For such *long* signals, the corresponding wealth return with a closing price $p(s)$ is $p(s)/p_a(t)$. The early stopping setting is similar in slightly different formulas. The approximate wealth change for a *long* signal is given as

$$c_t^{long} \triangleq \begin{cases} \bar{\beta} & \text{if } t < \min \underline{S}' < \min\{\bar{S}', t+\Delta\} \\ \underline{\alpha} & \text{if } t < \min \bar{S}' < \min\{\underline{S}', t+\Delta\} \\ \frac{\max_{s\in[t,t+\Delta]} p_b(s)}{p_a(t)} & \text{if } \underline{S}' = \bar{S}' = \emptyset \end{cases} \tag{2.2}$$

where $\underline{S}' := \{s \in [t, t+\Delta] : p_b(s) > \bar{\beta}p_a(t)\}$ and $\bar{S}' := \{s \in [t, t+\Delta] : p_b(s) < \underline{\alpha}p_a(t)\}$ with hyperparameters $\underline{\alpha}$ and $\bar{\beta}$. We write the above in the form of sets for the simplicity of the respective situations. The conditions in the sets $\underline{S}$ and $\underline{S}'$ can be interpreted as closing the position after reaching a certain profit regularized by $\bar{\beta}$ and $\underline{\beta}$. $\bar{S}$ and $\bar{S}'$ with $\bar{\alpha}$ and $\underline{\alpha}$ are for cutting extreme loss.

Based on the approximate wealth returns for *long* and *short* signals, the labels of the corresponding three trading actions are

$$l_t = \begin{cases} +1 & \text{if } c_t^{long} > \max\{c_t^{short}, \epsilon\} \\ -1 & \text{if } c_t^{short} > \max\{c_t^{long}, \epsilon\} \\ 0 & \text{o.w.} \end{cases} \tag{2.3}$$

where ϵ is the threshold for checking the strength of signals. In the above equation, +1 denotes taking the long position, −1 denotes taking the short position, and 0 denotes doing nothing at the current time t. Thus, a larger ϵ denotes more stringent signal screening.

When labeling in practice, the values of the hyperparameters are difficult to determine. Here, we outline our scheme used to assign their values. For different stocks, the bid–ask spread varies from negligible to huge. Therefore, to achieve better performance and more signals, we estimate the values of $\bar{\alpha}$, $\underline{\alpha}$, $\bar{\beta}$, and $\underline{\beta}$ based on historical bid–ask spreads from the training and validation datasets. More specifically, by utilizing the average ratio of the best ask to the best bid price, we calculate

$$\bar{\phi} = \max\{\frac{1}{n}\sum_{s=1}^{n}\frac{p_a(s)}{p_b(s)} - 1, \psi\} \tag{2.4}$$

where the constraint $\bar{\phi} \geq \psi$ is used to adjust extremely small values and prevent stopping too early for loss cutting. Here, ψ is a constant and satisfies $4\psi >$ *transaction cost* at a minimum to ensure that the target profit is higher than the transaction cost. With a customized value of ψ, the limits of profit can be regulated. Finally, the hyperparameters for target prices are given as

$$\bar{\alpha} = 1 + 2\bar{\phi}, \quad \underline{\alpha} = 1 - 2\bar{\phi}, \quad \bar{\beta} = 1 + 4\bar{\phi}, \quad \underline{\beta} = 1 - 4\bar{\phi}. \tag{2.5}$$

We then input these values into (2.1)–(2.2) and obtain the labels for subsequent classification.

3. The algo-trading system design

To design profitable and fully aggressive strategies by taking market orders, we propose our deep LOB trading system for selected stocks.

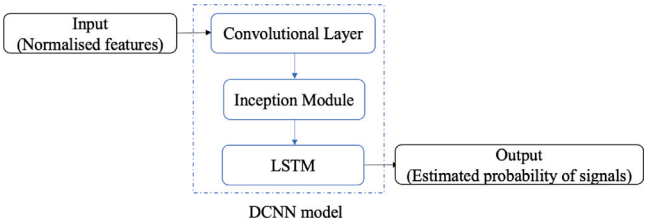

Fig. 6. Deep convolutional neural network (DCNN) model architecture.

Fig. 4 shows the three main components of the framework: training, predicting, and trading. We train a deep convolutional neural network (DCNN) model using features from the LOB daily or weekly. Then, at every tick of the testing period, the normalized features are added to the trained model to yield the probability for each classification. The predicted results can then be used in the trading strategy. We open the position to start a signal and close the position when faced with a contrary signal or when the holding time expires. If the same signal persists, we maintain the position and extend the holding time. We do not allow any overlapping transactions.

When trading without optimization, we invest the same amount for every signal. However, the predicted signals can be strong or weak, which suggests that different weights should be placed on them. Fig. 5 illustrates the trading strategy incorporating optimization using the fractional Kelly growth criterion under risk control. Experiments conducted on both the simulated data and empirical data strongly support the additional optimization step. Below, we explain some of the techniques in detail.

3.1. DCNN model for LOB

To derive the universal patterns of the LOB, we adopt a similar DCNN model architecture as in Zhang et al. (2019). The general design is shown in Fig. 6, and the detailed description of the topology, along with its hyperparameters, are given in Table A.9 of the Appendix.

- Convolutional Layer: The size of the first convolutional filter is (1×2) with a stride of (1×2), which summarizes information between price $p^{(i)}$ and volume $v^{(i)}$ at each order book level. By utilizing another convolutional layer with filter size (1×2) and stride (1×2), we measure the imbalance between the ask and bid sides. All information across different n levels is integrated using a large filter of size $(1 \times n)$. We also add some layers with filter size (5×1) to capture local interactions between the data over five time steps.
- Inception Module: To capture dynamic behavior over multiple timescales, the inception module is formed by wrapping several convolutions together (Szegedy et al., 2015). The use of varying convolutional filter sizes is beneficial. The 1×1 convolutional filters learn the cross-channel patterns, thus contributing to the overall feature extraction capabilities of the network.
- LSTM Module and Output: With feedback connections, the LSTM module is well-suited to time series data with lags of unknown durations between important events. After capturing the temporal relationships in the extracted features, the last output layer uses a softmax activation function. Hence, the final output elements represent the probability of each trading action at each time step.

Because the number of signals is relatively small and the proportion of *long* or *short* signals deviates significantly from 33%, we encounter imbalanced classification. When training using this type of data, the classic neutral network model is likely to overlook the minority cases. Undersampling, oversampling, and generating synthetic data are commonly discussed methods to balance the dataset (He & Garcia, 2009). However, modifying the dataset with resampling-like methods is in essence changing the reality of the financial markets. Another popular method is to change the loss function to inform the model to "pay more attention" to samples from an under-represented class. Before stating the specific loss function, we transform each label to a one-hot encoding variable $y_1, \ldots, y_M$ that satisfies

$$y_i = \begin{cases} 1 & \text{if } i \text{ is the true label} \\ 0 & \text{o.w.} \end{cases} \tag{3.1}$$

In general, the categorical cross-entropy loss function is the common choice for multi-class classification problems and softmax output units. It is defined as

$$CE = -\sum_{i=1}^{M} y_i \log \hat{y}_i \tag{3.2}$$

where y_i is the target value, $\hat{y}_i$ is the softmax probability for the ith class, and M is the number of classes. This loss function gives different classes the same weights. Zhang et al. (2019) use this loss function because the price movement classes in their data are roughly balanced. To better identify signals, we impose more penalties on the misclassification of *long* and *short* signals. Hence, the focal loss function originally proposed in Lin et al. (2017) is designed for imbalanced data, and is written as

$$FL = -\sum_{i=1}^{M} b_i (1-\hat{y}_i)^{\gamma} y_i \log \hat{y}_i \tag{3.3}$$

where y_i and $\hat{y}_i$ have the same meanings as in the categorical cross-entropy loss function. Importantly, b_i and γ are additional hyperparameters used to adjust the class weights. The effect of the modulating factor $(1-\hat{y}_i)^{\gamma}$ is increased with γ. Lin et al. (2017) find that $\gamma = 2$ works best in their experiments. Therefore, we adopt this approach to set the FL function. As the modulating factor related to γ helps to detect rare classes, we keep the weighting factor b_i the same for every class.

3.2. Risk measure

Value at risk (VaR) is a measure of the risk of loss of investments. VaR estimates how much a set of investments might lose, including the percentage loss and its probability. Risk managers use VaR to measure and control the level of risk exposure. For traders, it is beneficial to consider VaR when attempting to cut loss. Considering a single stock, we define the VaR of its trading signals for a given confidence level α as

$$P(c < VaR_{\alpha}) = 1 - \alpha \tag{3.4}$$

where c is the actual wealth return by implementing predicted *long* or *short* signals. We estimate the historical VaR using nonparametric methods as in Lai et al. (2020). In practice, if the training and validation period includes n *long* signals, the empirical wealth return distribution can be used to compute the VaR as follows:

$$\hat{P}(c^{long} < \widehat{VaR}_{\alpha}^{long}) = \frac{\#\{c_s^{long} : c_s^{long} < \widehat{VaR}_{\alpha}^{long}\}}{n} \approx 1 - \alpha \tag{3.5}$$

where $\#\{c_s^{long} : c_s^{long} < \widehat{VaR}_{\alpha}^{long}\}$ denotes the number of $\{c_s^{long}\}_{s=1}^{n}$ such that $c_s^{long} < \widehat{VaR}_{\alpha}^{long}$. More specifically, we sort $\{c_s^{long}\}_{s=1}^{n}$ in ascending order as $\{c_{(s')}^{long}\}_{s'=1}^{n}$ and write $\widehat{VaR}_{\alpha}^{long}$ as

$$k \triangleq \lfloor n(1-\alpha) \rfloor, \quad c_{(1)}^{long} \leqslant c_{(2)}^{long} \leqslant \cdots \leqslant c_{(k)}^{long} \leqslant \cdots \leqslant c_{(n)}^{long}$$
$$\widehat{VaR}_{\alpha}^{long} \triangleq c_{(k)}^{long}. \tag{3.6}$$

The same procedure can be used to obtain $\widehat{VaR}_{\alpha}^{short}$ for *short* signals.

Previously, when labeling signals, we cut the loss using hyperparameters $\bar{\alpha}$ and $\underline{\alpha}$. After estimating $\widehat{VaR}_{\alpha}$, we control the risk level and close the position if the current return $c_{t,t'} < \widehat{VaR}_{\alpha}$ to avoid extreme

Table 3
Performance of the deep convolutional neural network model on the simulation data and CS-20 dataset.

	Uptrend		Downtrend		Flat		CS-20 group 1		CS-20 group 2	
	CE	FL	CE	FL	CE	FL	CE	FL	CE	FL
Accuracy	0.49	0.50	0.86	0.86	0.55	0.57	0.74	0.75	0.87	0.88
Precision	0.49	0.49	0.83	0.83	0.55	0.58	0.67	0.69	0.83	0.83
Recall	0.49	0.50	0.86	0.86	0.55	0.57	0.75	0.75	0.88	0.88
f1-score	0.49	0.49	0.82	0.83	0.55	0.56	0.69	0.70	0.84	0.84

loss. The current return at time t' for *long* and *short* signals starting from time t is calculated as

$$c_{t,t'}^{long} = 2 - \frac{p_a(t')}{p_b(t)}, \quad c_{t,t'}^{short} = \frac{p_b(t')}{p_a(t)}. \tag{3.7}$$

It is clear that $\widehat{VaR}_\alpha$ affects the training and validation performance for different stocks. We use it to increase the robustness of out-of-sample transactions. In addition, the estimated VaR contributes to risk control when we trade with optimization.

3.3. Optimization using the fractional Kelly growth criterion under risk control

Known as the scientific gambling method, the Kelly growth criterion is first described by Kelly (1956) and is widely applied in bet sizing. Suppose the whole investment involves T periods with initial wealth $S_0 = 1$, $\boldsymbol{c}_t$ is the wealth return, and $\boldsymbol{\omega}_t$ is the investment proportion. Then, the final cumulative wealth is

$$S_T = \prod_{t=1}^{T} \boldsymbol{c}_t^T \boldsymbol{\omega}_t \tag{3.8}$$

where $\boldsymbol{c}_t = (c_t^{short}, c_t^{long}, 1)^T$ and $\boldsymbol{\omega}_t = (\omega_t^{short}, \omega_t^{long}, \omega_t^{none})^T$. Because the signal type is known, only one signal can be obtained, which means that $\min\{\omega_t^{long}, \omega_t^{short}\} = 0$ for any t.

The main idea is to apply the Kelly growth criterion to determine the investment proportion to assign to the predicted signals. Kelly's criterion sometimes suffers from high risks because it does not consider the variance of returns. MacLean et al. (2004) discuss the optimal investment decisions under downside risk control. Based on the estimated VaR in Section 3.2, our objective function under VaR control is given by

$$\max_{\boldsymbol{\omega}_t \in \Omega} \sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{c} \sim D_{\boldsymbol{x}}} \left[\log(\boldsymbol{c}_t^T \boldsymbol{\omega}_t) | \boldsymbol{X} \right], \quad \text{s.t. } P(\log(\boldsymbol{c}_t^T \boldsymbol{\omega}_t) \le \log \underline{\omega}) \le 1 - \alpha \tag{3.9}$$

where $\Omega = \{\boldsymbol{\omega} \in \mathbb{R}_+^3 : \mathbf{1}^T \boldsymbol{\omega} = 1, \min\{\omega^{long}, \omega^{short}\} = 0\}$ and $\mathbf{1}$ denotes a 3-dimensional vector with all elements equal to 1. $\underline{\omega}$ is a constant that expresses risk preference.

(3.9) involves an online learning characteristic because the logarithmic returns are successively added to the objective function as time passes. Thus, we can perform the optimization by maximizing the expectation at time t. If a *long* signal is predicted, then $\omega_t^{short} = 0$ and the objective function is simplified as

$$\begin{aligned} &\max_{\boldsymbol{\omega}_t \in \Omega} \mathbb{E}_{\boldsymbol{c} \sim D_{\boldsymbol{x}}} \left[\log(\boldsymbol{c}_t^T \boldsymbol{\omega}_t) | \boldsymbol{X} \right] \\ &= \max_{0 \le \omega_t^{long} \le 1} \hat{p}_t^{long} \log(\hat{c}_+^{long} \omega_t^{long} + 1 - \omega_t^{long}) \\ &\quad + (1 - \hat{p}_t^{long}) \log(\hat{c}_-^{long} \omega_t^{long} + 1 - \omega_t^{long}) \end{aligned} \tag{3.10}$$

where $\hat{p}_t^{long}$ is the estimated probability of a *long* signal from the DCNN model. $\hat{c}_+^{long}$ denotes the estimated returns for successful *long* actions and $\hat{c}_-^{long}$ denotes the estimated returns for failed *long* actions. These values are estimated from empirical wealth returns in the training and validation period. The exact formulas are written as

$$\hat{c}_+^{long} = \frac{\sum_{j=1}^{n} c_j^{long} 1_{\{c_j^{long} > 1\}}}{\sum_{j=1}^{n} 1_{\{c_j^{long} > 1\}}}, \quad \hat{c}_-^{long} = \frac{\sum_{j=1}^{n} c_j^{long} 1_{\{c_j^{long} < 1\}}}{\sum_{j=1}^{n} 1_{\{c_j^{long} < 1\}}} \tag{3.11}$$

where $1_{\{\cdot\}}$ is the indicator function. Replacing *long* with *short* solves the optimization of *short* signals. The closed-form solutions to the objective function are provided in Theorem 3.1.

Theorem 3.1. *Using Kelly's exponential growth rate approach, under the objective function in (3.10) and notations in (3.9)–(3.11) for long signals, we optimize the trading strategy and deploy the investment proportion as*

$$\omega_t^{long*} = \max\left\{ \min\left(\tilde{\omega}_t^{long}, \frac{\underline{\omega} - 1}{\widehat{VaR}_\alpha^{long} - 1}, 1 \right), 0 \right\} \tag{3.12}$$

where $\tilde{\omega}_t^{long} = \frac{\hat{p}_t^{long}(\hat{c}_+^{long} - 1) + (1 - \hat{p}_t^{long})(\hat{c}_-^{long} - 1)}{(1 - \hat{c}_-^{long})(\hat{c}_+^{long} - 1)}$. $\widehat{VaR}_\alpha^{long}$ *is the VaR estimated by (3.6). By replacing long with short, we obtain the optimization result of short signals.*

Proof. See Appendix B. □

Maclean et al. (2010) indicate that the investment proportion suggested by the Kelly criterion may be very large and is relatively high-risk in the short term. This risk can be moderated using a fractional Kelly strategy, where a fraction f with $0 \le f \le 1$ of the total Kelly proportion is allocated to the high-risk asset and the remainder to cash. Therefore, the corresponding investment proportion becomes

$$\omega_t^{long*} = \max\left\{ \min\left(f\tilde{\omega}_t^{long}, \frac{\underline{\omega} - 1}{\widehat{VaR}_\alpha^{long} - 1}, 1 \right), 0 \right\} \tag{3.13}$$

where f depends on the investor's aversion to risk. The value can also be chosen according to the in-sample performance.

4. Performance of the algo-trading system

The system we propose mainly consists of training, predicting, and trading signals. We apply it to LOB datasets, which we split into three parts for training, validation, and testing. The out-of-sample results are given in two parts. To measure the forecasting performance of the DCNN model, we calculate the accuracy and weighted average of precision, recall, and f1-score. These values indicate how many signals are detected and how many of them are correct. Next, we enter the predicted signals into the trading strategy and produce transactions. The average profit per signal and accumulated profit per day are calculated and presented in various forms. Of all the results of our deep LOB trading system, we are most interested in the profit.

In this section, we compare the different experimental settings used for different datasets with respect to the choice of loss functions and optimization methods. First, we compare the effects of the categorical cross-entropy loss function and focal loss function on imbalanced data. Second, by comparing the results obtained using optimization to those obtained without optimization, we highlight the necessity and superiority of the optimization technique. In addition to the intuitive figures, a statistical hypothesis test (t-test) is used to test whether the average profit per signal of one setting is significantly greater than that of the other setting. More concrete performance results in the context of the previously introduced datasets are discussed.

4.1. Demonstration on simulation datasets

First, we conduct experiments on data containing three hypothetical market sentiments. The three cases are trained under two types of loss functions. If we simply look at the performance of the DCNN model for predicting signals in Table 3, the differences between the two loss functions are not obvious. However, the definitions of the approximate wealth changes for labeling signals are different from the actual returns obtained in the trading strategy. For trading signals, we have no future information. As such, it is impossible to obtain the optimal trading price

Table 4
Statistical descriptions of average profit per signal for the simulation datasets. "Qty" is the total number of signals, and *p*-value is derived from the *t*-test on the average profit between two settings.

	Uptrend			
	non-op-CE	non-op	op	adjusted op
Avg profit (std)	2.18 (0.002888)	3.88 (0.002950)	6.91 (0.002464)	8.67 (0.002732)
Qty	1842	1586	1586	1265
p-value	–	0.0909	–	–
	–		4.02e−14	–
	Downtrend			
	non-op-CE	non-op	op	adjusted op
Avg profit (std)	56.51 (0.0341)	55.77 (0.0294)	67.42 (0.0283)	73.8 (0.0295)
Qty	173	163	163	149
p-value	–	0.983	–	–
	–		0.0287	–
	Flat			
	non-op-CE	non-op	op	adjusted op
Avg profit (std)	1.77 (0.003856)	9.68 (0.003871)	12.67 (0.003475)	15.17 (0.003752)
Qty	2249	1944	1944	1624
p-value	–	4.21e−11	–	–
	–		4.92e−16	–

during the limited holding period. Hence, profit is the key to assessing the performance of the system.

The profits obtained from the predicted signals are reported in Table 4. We present the mean (avg), standard deviation (std), and *p*-value yielded by the *t*-test. The following abbreviations represent different settings: *non-op-CE* denotes a trade without optimization based on the signals predicted from the categorical cross-entropy loss model; *non-op* denotes a trade without optimization based on the signals predicted from the focal loss model; and *op* denotes a trade with optimization based on the signals predicted from the FL model (the same DCNN model as *non-op*). Because we sometimes do not invest after optimization, *adjusted op* is more accurate for evaluating the average profit of *op* by excluding transactions with an investment proportion of zero.

Comparing *non-op-CE* with *non-op*, Table 4 shows that *uptrend* and *flat* have signals with much higher average profits when the model is trained by FL. Although the average is not improved over that of *downtrend*, the variance is smaller. It is worth mentioning that the results of *uptrend* and *flat* are more convincing than those of *downtrend* when more signals are executed. In terms of accumulated profit, Fig. 7 clearly demonstrates that FL works better than CE. Furthermore, by splitting the testing periods into several intervals, we compare the cumulative profit over equal intervals and produce the boxplot shown in Fig. 8. This finding is consistent with the results for average profit in Table 4. Another comparison between *non-op* and *op* (whether looking at average profit or accumulated profit) reveals that optimization further improves the results (higher returns with lower risks and p-values close to zero). Overall, the findings show that profit is greatly enhanced by our design schemes under all market sentiments.

Fig. 7 shows that profits in *flat* with more signals (larger "qty" in Table 4) are higher. In this situation, the model can earn profit stably and continuously with low risk. This leads us to speculate on the relationship between profitability and the number of signals. When there are more signals, the probability of encountering correct signals is higher. Therefore, it should be beneficial to add a step that considers this information when choosing which stocks to trade. In the following empirical study, we verify this assumption. By considering the number of signals combined with in-sample performance, we select more profitable stocks and compose the proprietary CS-100 dataset.

To analyze the data over a long-term period encompassing mixed sentiments, we deploy our system on 21 periods using the rolling window method. When counted according to the average number of ticks per stock per day in the Chinese A-share market, each period contains approximate 6 days. Hence, the total number of testing days is equivalent to more than half a year. Fig. 9 plots the estimates of the daily profits for a single period with a 50% confidence interval for all periods. To better match the data to the real-world market situation, the transaction cost is taken as 10 bp (basis points). Based on the performance over 126 testing days, our model, when using the FL function and optimization technique, consistently achieves better daily profits than the other two settings.

4.2. Empirical study of the Chinese A-share market

4.2.1. CS-20 dataset

Using the CS-20 dataset, we evaluate our model on all 20 stocks from June 2020 to September 2020. The four months of data are divided into training, validation, and testing subsets. Because of the small sample size, the testing period is quite short: a total of 80 days of trading results for all 20 stocks. Furthermore, we do not select the stocks in the CS-20 dataset. Therefore, not all the stocks are suitable for our trading system. Some of the stocks are too stable and lack price fluctuations over a short period (e.g., 40 ticks). Some of the stocks have too large bid–ask spreads, which are very hard to cover when making market orders. As such, we do not expect to obtain high profits by trading signals from this stock pool, and we put the focus on relative profits in our comparisons.

We rank the stocks by the number of signals during the training period and group the top 10 as *CS-20 group 1*. The remaining stocks are denoted as *CS-20 group 2*. Statistical descriptions of the average profit per signal for the two groups are shown in Table 5. The conclusions are that the signals from models trained by FL have higher profits than those trained by CE and that optimization significantly increases profits. For these stocks, trading based on the signals from models trained by CE would be undesirable. Thanks to the use of FL, severe loss can be avoided and small gains can be made. After adding the optimization step, profits are greatly increased to cover the spread and remain positive. Although the average profits are moderate and sometimes do not cover the transaction costs, our deep LOB trading system reflects the value of using FL and optimization.

Apart from the average profit, we display the daily profit for all stocks for the testing days in Fig. 10. Here, the 50% confidence interval helps to reveal the discrepancies between stocks in the same group. The ranking of these three settings supports our previous findings. Furthermore, the results of group 1 in the CS-20 dataset are more promising than those of group 2 in terms of both average profit and

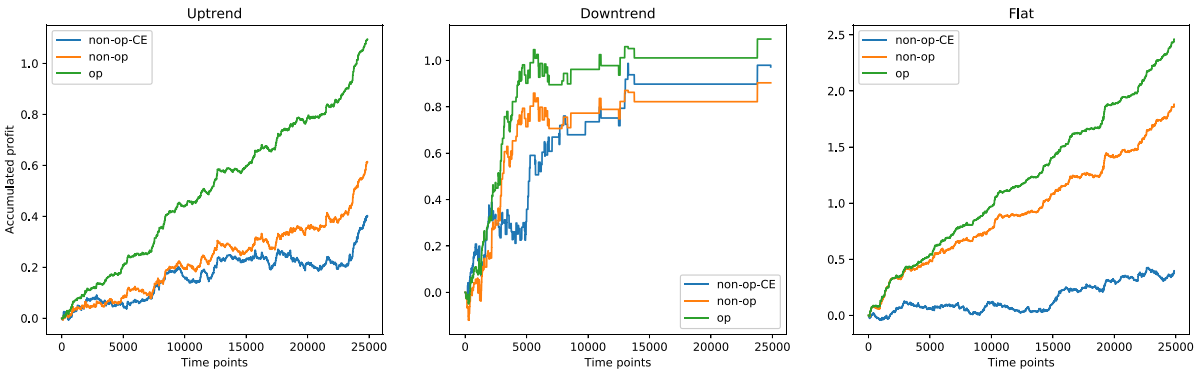

Fig. 7. Accumulated profit for the simulation datasets.

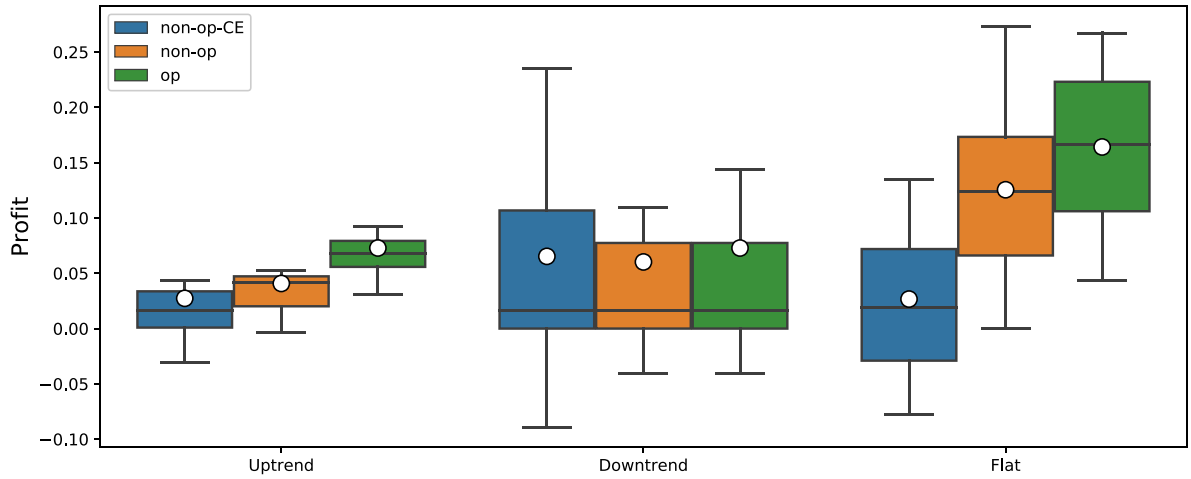

Fig. 8. Boxplot of cumulative profit over fixed time intervals. The white hollow dots denote the mean and the black horizontal lines denote the median.

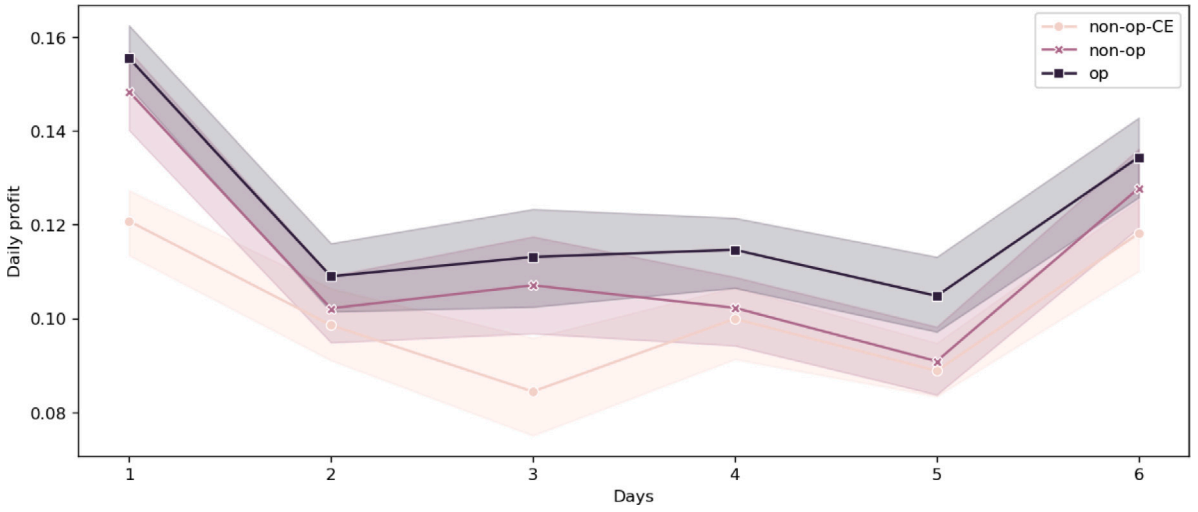

Fig. 9. Estimates of the daily average profit with a 50% confidence interval for 21 testing periods using the rolling window method. "Daily" is derived from artificial segmentation of the long-term simulation dataset and the transaction fee (10 basis points) is deducted.

daily profit, which also demonstrates that our ranking of stocks has a significant effect on the results. If we rank all of the stocks in the entire Chinese market, we can obtain more profitable signals. Definite losses resulting from spreads and transaction fees make it difficult to yield high returns. High returns are therefore unrealistic for all 4000 stocks. For this reason, the stock selection process is necessary to identify stocks that are workable in our deep LOB trading. We use this selection process to assemble our proprietary CS-100 dataset.

4.2.2. *Proprietary CS-100 dataset*

In contrast to the CS-20 dataset, which includes stocks we did not select, the proprietary CS-100 dataset includes stocks selected according to our defined ranking scheme. Based on the number of signals and in-sample performance, we choose 100 stocks and conduct experiments to

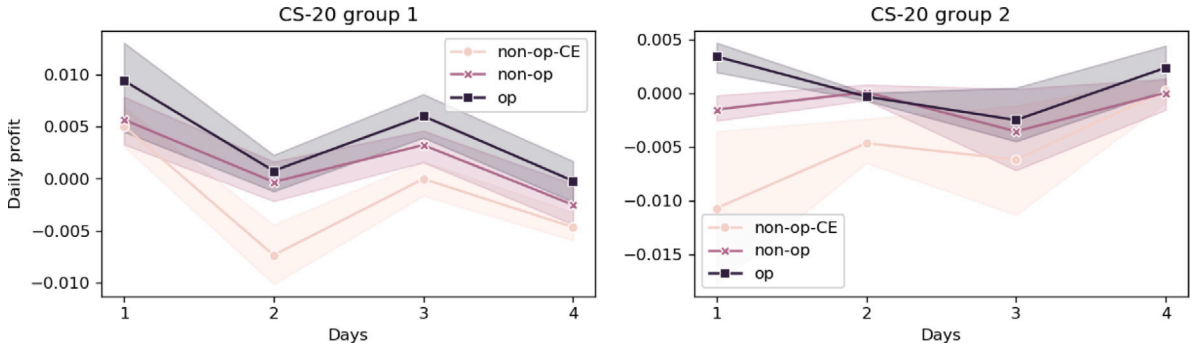

Fig. 10. Estimates of the daily average profit with a 50% confidence interval for all stocks in the CS-20 dataset.

Table 5
Statistical descriptions of average profit per signal for the CS-20 dataset. "Avg qty" is the average number of signals per stock per day with the standard deviation ("std"). The *p*-value is derived from the *t*-test on the average profit between two settings.

	CS-20 group 1				CS-20 group 2			
	non-op-CE	non-op	op	adjusted op	non-op-CE	non-op	op	adjusted op
Avg profit	−0.53	0.50	1.32	1.57	−5.57	−1.55	0.89	1.15
Std	0.0016	0.0016	0.0015	0.0017	0.0027	0.0026	0.0021	0.0024
Avg qty	32.93	30.38	30.38	25.55	9.58	8.08	8.08	6.28
Std	52.57	46.14	46.14	38.16	15.91	13.70	13.70	10.92
p-value	0.11		–	–	0.048		–	–
	–	1.62e−6		–	–	0.0013		–

demonstrate the performance of our deep LOB trading system. In terms of profit, the transaction fee (an additional cost) is of great concern to traders involved in HFT (Menkveld, 2013). For transactions in the Chinese A-share market, we set the transaction fee as 10 bp of trading value, covering stamp duty[6] (the main fee). In our experiments on the long-term simulation data, the 10 bp transaction cost is considered and the results are shown in Fig. 9. Similarly, all of the profits we present from this dataset are shown after deducting the transaction fee.

We apply the same group training technique as used for the CS-20 dataset. Four groups are formed and displayed in Table 1. Our system trains stocks within the same group together and then runs the tests on them. Because one group contains 25 stocks, the length of the historical data for training can be appropriately reduced. To guarantee both the adequacy of samples and the efficiency of calculations, we use 40 days for training, 1 day for validation, and 5 days for testing, with all of this regarded as one period. To make the results more convincing and to avoid random effects, we test the out-of-sample performance of four periods covering one month, from July 26 to August 20, 2021. We retrain the DCNN model once per week using the rolling window method. Specifically, we retrain the model every weekend and predict the signals for the next week (consisting of five weekdays). This approach is easy to implement and does not incur much computational overhead. Hence, a total of 2000 days of trading results for all 100 stocks are analyzed.

Displayed for the groups separately, the performance of the DCNN model is given in Table 6, and the statistical descriptions of the average profit per signal are presented in Table 7. We verify the results and conclude that when the FL and optimization steps are included, our system shows excellent performance. "Avg qty", the average number of signals per stock per day for the different groups, is highly correlated with the ratio of in-sample signals, as shown in Table 1. The average number of transactions completed by stocks in group 1 every day is very large (more than 90). Therefore, if we execute all of these transactions, the accumulated daily profit is impressive (Fig. 11). Furthermore, the daily profit for stocks in group 1 reaches approximately 8% after the loss of bid–ask spreads and the deduction of transaction fees.

[6] http://english.sse.com.cn/start/taxes/

Table 6
Performance of the deep convolutional neural network model in the proprietary CS-100 dataset.

	Group 1		Group 2		Group 3		Group 4	
	CE	FL	CE	FL	CE	FL	CE	FL
Accuracy	0.49	0.51	0.57	0.58	0.59	0.60	0.63	0.64
Precision	0.48	0.50	0.51	0.51	0.50	0.54	0.52	0.52
Recall	0.42	0.43	0.41	0.41	0.39	0.40	0.35	0.38
f1-score	0.41	0.42	0.40	0.41	0.37	0.38	0.31	0.35

It is worth noting that trading on the Chinese stock exchanges is subject to multiple restrictions. As such, the cumulative profit achieved by executing all signals is not realistic in practice. The Chinese A-share market has a number of specific rules and regulations. First, the "T+1 trading rule" requires an investor to sell only stocks purchased at least one day prior and forbids the investor from selling stocks bought on the same day. Second, shorting is not permitted. However, if the stocks already exist in the account, they can be sold to overcome both constraints. Although the number of stock shares in the account determines the limited amount of intraday trading allowed, we can trade a part of the signals. This motivates us to calculate another index, the profit per signal per day, which describes the situation in which only one transaction is executed per day. A boxplot for all of these values across different settings and groups is shown in Fig. 12. The profits per signal per day of group 1 and group 2 are satisfactory and significantly improved by our system. If we trade only one signal from the first stock group every day continuously, the annual return can reach approximately 25%. The performance of our deep LOB trading system is reasonable and acceptable to secure additional profit for stockholders.

To understand the profitability yielded in our empirical study, it is necessary to note that the Chinese A-share market is an immature financial market. The fact that the LOB updates every 3 s means that the market is not completely transparent. Some hidden information and features are not revealed, and investors cannot absorb and act at short horizons. These behaviors make statistical arbitrage possible. Other regulations, such as "T+1", "no short", and price limit, discourage

Table 7
Statistical descriptions of the average profit per signal for the proprietary CS-100 dataset after deducting the transaction fee (10 basis points). "Avg qty" is the average number of signals per stock per day with the standard deviation ("std"). The *p*-value is derived from the *t*-test on the average profit between two settings.

	Group 1				Group 2			
	non-op-CE	non-op	op	adjusted op	non-op-CE	non-op	op	adjusted op
Avg profit	6.00	6.43	8.74	9.64	0.91	1.92	4.69	5.34
Std	0.0046	0.0045	0.0042	0.0043	0.0040	0.0041	0.0038	0.0040
Avg qty	90.75	93.52	93.52	84.83	75.86	73.94	73.94	64.97
Std	49.10	51.68	51.68	44.04	74.31	68.88	68.88	54.48
p-value	0.15		–	–	0.0008		–	–
	–	0.0		–	–	0.0		–
	Group 3				Group 4			
	non-op-CE	non-op	op	adjusted op	non-op-CE	non-op	op	adjusted op
Avg profit	1.97	2.70	4.98	5.27	3.02	4.76	5.43	5.45
Std	0.0053	0.0054	0.0050	0.0052	0.0064	0.0060	0.0056	0.0056
Avg qty	47.87	49.75	49.75	47.03	22.53	31.80	31.80	31.73
Std	39.05	38.64	38.64	35.90	12.13	15.01	15.01	14.99
p-value	0.13		–	–	0.02		–	–
	–	2.08e−171		–	–	5.08e−15		–

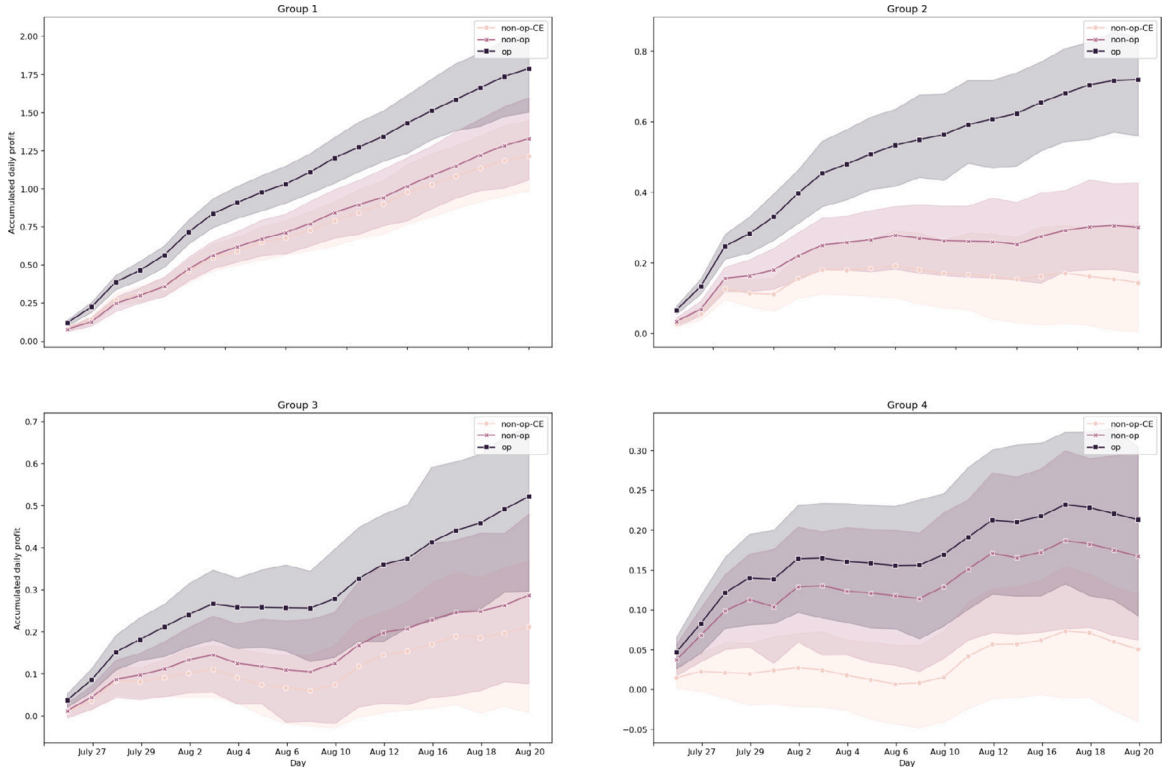

Fig. 11. Estimates of accumulated daily average profit with a 50% confidence interval for all stocks in the proprietary CS-100 dataset. The testing period is four weeks and is achieved using the rolling window method. The transaction fee (10 basis points) is deducted.

Table 8
Average computation time of each stage.

	Data normalization	One step prediction	Optimization
Time (ms)	0.998	25.7	0.0047

people from intraday trading and make the market inefficient. As Chordia et al. (2008) evidence, the predictability of short-horizon returns based on order flows is an inverse indicator of market efficiency. As such, we have reasons to expect profits by making use of the tick-time interval of the Chinese A-share market. We analyze whether the time constraint is problematic in practice, and we find that a gap of longer than or close to 0.5 s is adequate to implement our system. When predicting and trading tick by tick out of sample, three tasks must be completed: data normalization, signal prediction, and optimization. The time required for each task is recorded in Table 8 without optimized code and advanced computational processing power. The total time is less than 27 ms, leaving enough time for additional order-related operations such as order submission and execution in exchanges. We are confident in implementing this system and obtaining considerable returns from eligible markets. Notably, the Chinese A-share market updates the snapshot, 10-level LOB data every 3 s, which is much slower than other stock markets and provides us with an opportunity to implement our system successfully.

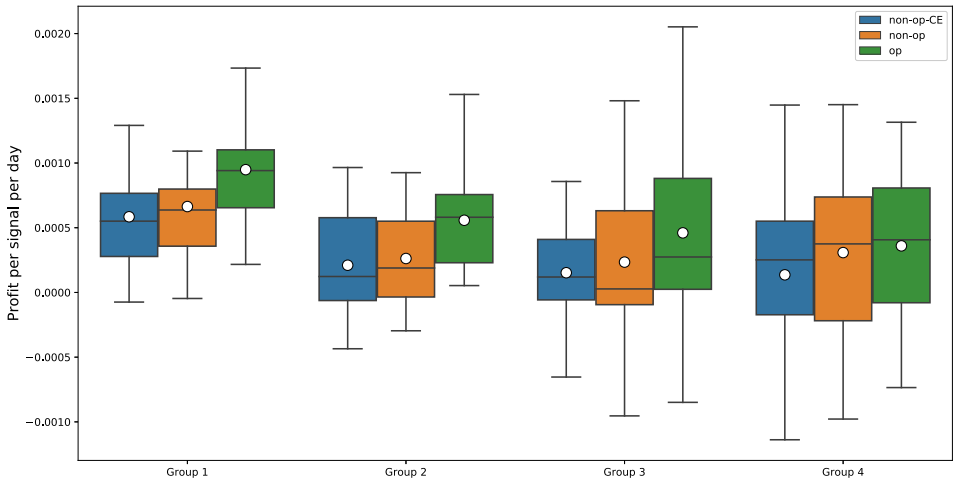

Fig. 12. Boxplot of profit per signal per day for four groups over the entire testing period after deducting the transaction fee (10 basis points). The white hollow dots refer to the mean and the black horizontal lines refer to the median.

Table A.9
Summary of the deep convolutional neural network model structure.

Module	Layer (options)	Output shape	Number of parameters
	Input	(None, 50, 40, 1)	–
Convolutional layers	Conv2D (1 × 2@16, stride = 1 × 2)	(None, 50, 20, 16)	48
	Conv2D (5 × 1@16)	(None, 50, 20, 16)	1296
	Conv2D (1 × 2@16, stride = 1 × 2)	(None, 50, 10, 16)	528
	Conv2D (5 × 1@16)	(None, 50, 10, 16)	1296
	Conv2D (1 × 10@16)	(None, 50, 1, 16)	2576
	Conv2D (5 × 1@16)	(None, 50, 1, 16)	1296
Inception module	Table A.10	(None, 50, 1, 48)	2896
LSTM module and output	LSTM (@32)	(None, 32)	10368
	Dense ('softmax')	(None, 3)	99

Table A.10
Inception module.

Conv2D (1 × 1@16)	Conv2D (1 × 1@16)	MaxPooling2D (3 × 1)
Conv2D (3 × 1@16)	Conv2D (5 × 1@16)	Conv2D (1 × 1@16)
Filter concatenation		

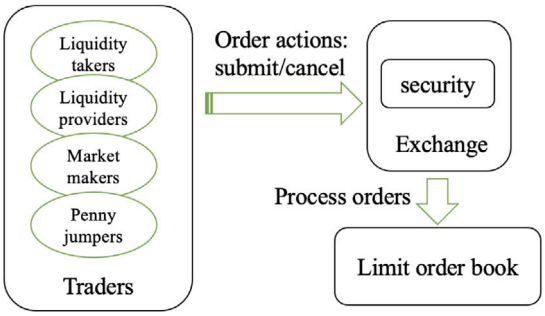

Fig. C.13. Illustration of the limit order book simulation model framework.

4.3. Discussion and future works

All algo-trading systems have limitations, including this one. An obvious limitation from ours is that we require 0.03 s to complete all of the computations (not including additional order-related operations) but the technology improvement will definitely increase the tick frequency in the near future. In mature markets, the tick frequency can reach an order of nanoseconds so that most (if not all) deep-learning based trading algorithms suffer from delayed decision-making compared to the tick frequency. To partially address the impact of such a delay, we conduct a back test with our Chinese A-share dataset. While the tick frequency is around 3 s, we test the change in profits for submitting orders in 6 s. Considering such a 3-second delay between signal generation and order execution, the average profit per signal for Group 1 under *op* setting changes from 8.74 to 8.22. The impact is mild. However, the backtesting ignores possible market interactions. If our competitors are aware of the delay, they may strategically submit orders to game us. Therefore, an interesting future study is the deep LOB trading system with a generative adversarial network architecture that replicates adversarial orders from competitors.

To the best of our knowledge, it is the first paper on deep LOB trading. To support the use of deep learning, our subsequent work (Yin & Wong, 2022) develops different metrics to compare LOB trading with different machine learning methods, including 'black-box' algorithms such as the one proposed here. In Yin and Wong (2022), we apply the metrics to compare this DCNN model against classical alternatives, such as logistic regression, support vector machine, and random forest. It is shown that DCNN significantly outperforms all of the competitive models in terms of prediction and consistency.

5. Conclusion

This paper explores an integral deep LOB trading system that takes advantage of tick-time intervals. From receiving the updated tick-report to executing the order, all of the steps in the system can be completed within 0.5 s and implemented in the Chinese A-share market. The basic features of the LOB are normalized and analyzed using a DCNN model to classify current trading actions: *long*, *short*, and *none*. The main contributions of this novel system are the application of the FL function to training to deal with imbalanced classification and optimization using the fractional Kelly growth criterion under risk

Table C.11
Model parameters used to simulate the limit order book dataset over a long-term period with mixed sentiments.

Description	Parameter	Value
Number of liquidity providers	N_{LP}	500
Number of market makers	N_{MM}	1
Number of market maker quotes		12
Market maker quote range		60 ticks
Number of liquidity takers	N_{LT}	500
Maximum order quantity	$Q_{\max}$	10
Number of time steps	T	1152000
Liquidity provider arrival rate parameter	α	0.0375
Liquidity taker arrival rate parameter	μ	0.001
Liquidity provider and market maker cancel rate	δ	0.025
Liquidity provider and market maker bid probability	q_{provide}	[0.45, 0.55]
Fixed scale parameter for the liquidity provider price distribution	λ_0	100
Adjustable scale parameter for the liquidity provider price distribution	C_λ	{1,3, …,97,99}
Variance parameter for the q_{take} mean-reverting random walk	Δs	0.001
Minimum price increment	MPI	1
Probability that the penny jumper will have the opportunity to participate	α_{PJ}	0

control to improve trading profits. Furthermore, our system considers factors in real trading circumstances, such as order choices, transaction costs, and extreme losses. The performance of the system on simulated and empirical datasets shows the benefits of our approach in terms of average profit per signal and accumulated profit. This innovative system is inspirational and will serve as a reference for financial firms or organizations engaging in LOB trading.

CRediT authorship contribution statement

Jie Yin: Methodology, Investigation, Data curation, Software, Visualization, Writing – original draft. **Hoi Ying Wong:** Conceptualization, Methodology, Investigation, Supervision, Validation, Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Hoi Ying Wong reports financial support was provided by Research Grants Council of Hong Kong. Hoi Ying Wong reports a relationship with TradeMaster Securities (Hong Kong) Limited that includes: consulting or advisory.

Data availability

The authors can share the simulated data but do not have the permission to share market data.

Acknowledgments

We would like to thank the anonymous referees and the editors for their careful reading and valuable comments, which have greatly improved the manuscript. Jie Yin is grateful to participants for their insightful comments at 2022 NVIDIA GPU Technology Conference, and the 11th World Congress of the Bachelier Finance Society. We are particularly grateful for the assistance and historical data provided by TradeMaster, a Fintech company in China. Special thanks are extended to the company's Hangzhou Quantitative Team members: Yang Yu, Jinwen Cheng, and Tongji Li, for all their helpful comments. H.Y. Wong acknowledges the support from the Research Grants Council of Hong Kong with project codes: RMG8601495 and GRF14308422.

Appendix. Details of the DCNN model

Details on the hyperparameters: the optimizer is adaptive moment estimation algorithm (ADAM) with default parameters $\beta_1 = 0.9, \beta_2 = 0.999$ from Kingma and Ba (2014); the batch size is 128; the learning rate is 0.01; the activation function for Conv2D is LeakyReLU with negative slope coefficient $\alpha = 0.01$.

Appendix B. Proof of Theorem 3.1

Proof. Below, we consider the case of a *long* signal. The proof for a *short* signal is equivalent. First, take the partial derivative of (3.10) with respect to ω_t^{long}, let it equal 0, and solve the equation to get

$$\tilde{\omega}_t^{long} = \frac{\hat{p}_t^{long}(\hat{c}_+^{long} - 1) + (1 - \hat{p}_t^{long})(\hat{c}_-^{long} - 1)}{(1 - \hat{c}_-^{long})(\hat{c}_+^{long} - 1)}. \tag{B.1}$$

The risk is controlled by $P(\log(\boldsymbol{c}_t^T \boldsymbol{\omega}_t) \le \log \underline{\omega}) \le 1 - \alpha$, which is equivalent to

$$\begin{aligned} P(\log(c_t^{long} * \omega_t^{long} + 1 - \omega_t^{long}) \le \log \underline{\omega}) \le 1 - \alpha, \\ P(c_t^{long} * \omega_t^{long} + 1 - \omega_t^{long} \le \underline{\omega}) \le 1 - \alpha, \\ P(c_t^{long} \le \frac{\omega_t^{long} + \underline{\omega} - 1}{\omega_t^{long}}) \le 1 - \alpha. \end{aligned} \tag{B.2}$$

Then, by the definition of VaR such that $P(c_t^{long} < VaR_\alpha^{long}) = 1 - \alpha$ leads to the VaR constraint as

$$\begin{aligned} \frac{\omega_t^{long} + \underline{\omega} - 1}{\omega_t^{long}} \le VaR_\alpha^{long}, \\ \omega_t^{long} \le \frac{\underline{\omega} - 1}{VaR_\alpha^{long} - 1} \end{aligned} \tag{B.3}$$

where VaR_α^{long} can be estimated by historical returns in (3.6). Combine the proportion $\tilde{\omega}_t^{long}$ with the VaR constraint $0 \le \omega_t^{long} \le 1$, and we solve the optimal investment proportion as

$$\omega_t^{long*} = \max\left\{\min\left(\tilde{\omega}_t^{long}, \frac{\underline{\omega} - 1}{\widehat{VaR}_\alpha^{long} - 1}, 1\right), 0\right\}. \quad \square \tag{B.4}$$

Appendix C. Details of the LOB simulation model

See Table C.11 and Fig. C.13.

References

Abergel, F., Huré, C., & Pham, H. (2020). Algorithmic trading in a microstructural limit order book model. *Quantitative Finance*, *20*(8), 1263–1283.

Abudy, M. M., & Wohl, A. (2018). Corporate bond trading on a limit order book exchange. *Review of Finance*, *22*(4), 1413–1440.

Avellaneda, M., Li, T. N., Papanicolaou, A., & Wang, G. (2021). Trading signals in VIX futures. arXiv preprint arXiv:2103.02016.

Avellaneda, M., & Stoikov, S. (2008). High-frequency trading in a limit order book. *Quantitative Finance*, *8*(3), 217–224.

Baldauf, M., & Mollner, J. (2020). High-frequency trading and market performance. *The Journal of Finance*, *75*(3), 1495–1526.

Brogaard, J., Hendershott, T., & Riordan, R. (2014). High-frequency trading and price discovery. *Review of Financial Studies*, *27*(8), 2267–2306.

Cartea, Á., Jaimungal, S., & Ricci, J. (2014). Buy low, sell high: A high frequency trading perspective. *SIAM Journal on Financial Mathematics*, *5*(1), 415–444.

Chordia, T., Roll, R., & Subrahmanyam, A. (2008). Liquidity and market efficiency. *Journal of Financial Economics*, *87*(2), 249–268.

Collver, C. (2017). An application of agent-based modeling to market structure policy: The case of the US Tick Size Pilot Program and market maker profitability. White Paper.

Cont, R., Stoikov, S., & Talreja, R. (2010). A stochastic model for order book dynamics. *Operations Research*, *58*(3), 549–563.

Guilbaud, F., & Pham, H. (2013). Optimal high-frequency trading with limit and market orders. *Quantitative Finance*, *13*(1), 79–94.

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, *21*(9), 1263–1284.

Huang, C., Ge, W., Chou, H., & Du, X. (2021). Benchmark dataset for short-term market prediction of limit order book in China markets. *The Journal of Financial Data Science*.

Kelly, J. L., Jr. (1956). A new interpretation of information rate. *The Bell System Technical Journal*.

Kercheval, A. N., & Zhang, Y. (2015). Modelling high-frequency limit order book dynamics with support vector machines. *Quantitative Finance*, *15*(8), 1315–1329.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kirilenko, A., Kyle, A. S., Samadi, M., & Tuzun, T. (2017). The flash crash: High-frequency trading in an electronic market. *The Journal of Finance*, *72*(3), 967–998.

Lai, Z. -R., Tan, L., Wu, X., & Fang, L. (2020). Loss control with rank-one covariance estimate for short-term portfolio optimization. *Journal of Machine Learning Research*, *21*(97), 1–37.

Lin, T. -Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. Proceedings of the IEEE international conference on computer vision. (pp. 2980–2988).

MacLean, L. C., Sanegre, R., Zhao, Y., & Ziemba, W. T. (2004). Capital growth with security. *Journal of Economic Dynamics and Control*, *28*(5), 937–954.

Maclean, L. C., Thorp, E. O., & Ziemba, W. T. (2010). Long-term capital growth: The good and bad properties of the kelly and fractional kelly capital growth criteria. *Quantitative Finance*, *10*(7), 681–687.

Menkveld, A. J. (2013). High frequency trading and the new market makers. *Journal of Financial Markets*, *16*(4), 712–740.

Ntakaris, A., Magris, M., Kanniainen, J., Gabbouj, M., & Iosifidis, A. (2018). Benchmark dataset for mid-price forecasting of limit order book data with machine learning methods. *Journal of Forecasting*, *37*(8), 852–866.

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

Obizhaeva, A. A., & Wang, J. (2013). Optimal trading strategy and supply/demand dynamics. *Journal of Financial Markets*, *16*(1), 1–32.

Passalis, N., Tefas, A., Kanniainen, J., Gabbouj, M., & Iosifidis, A. (2018). 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*, *4*(6), 774–785.

Passalis, N., Tefas, A., Kanniainen, J., Gabbouj, M., & Iosifidis, A. (2019). Deep adaptive input normalization for time series forecasting. *IEEE Transactions on Neural Networks and Learning Systems*, *31*(9), 3760–3765.

Pun, C. S., Wang, L., & Wong, H. Y. (2020). Financial thought experiment: A GAN-based approach to vast robust portfolio selection. Proceedings of the 29th international joint conference on artificial intelligence.

Roşu, I. (2009). A dynamic model of the limit order book. *Review of Financial Studies*, *22*(11), 4601–4641.

Sirignano, J., & Cont, R. (2019). Universal features of price formation in financial markets: Perspectives from deep learning. *Quantitative Finance*, *19*(9), 1449–1459.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., & Anguelov, D. (2015). Going deeper with convolutions. Proceedings of the IEEE conference on computer vision and pattern recognition. (pp. 1–9).

Tran, D. T., Iosifidis, A., Kanniainen, J., & Gabbouj, M. (2018). Temporal attention-augmented bilinear network for financial time-series data analysis. *IEEE Transactions on Neural Networks and Learning Systems*, *30*(5), 1407–1418.

Tsang, K. H., & Wong, H. Y. (2020). Deep-learning solution to portfolio selection with serially dependent returns. *SIAM Journal on Financial Mathematics*, *11*(2), 593–619.

Tsantekidis, A., Passalis, N., Tefas, A., Kanniainen, J., Gabbouj, M., & Iosifidis, A. (2017a). Forecasting stock prices from the limit order book using convolutional neural networks. *1*, In *2017 IEEE 19th conference on business informatics* (pp. 7–12). IEEE.

Tsantekidis, A., Passalis, N., Tefas, A., Kanniainen, J., Gabbouj, M., & Iosifidis, A. (2017b). Using deep learning to detect price change indications in financial markets. In *2017 25th E uropean signal processing conference* (pp. 2511–2515). IEEE.

Yin, J., & Wong, H. Y. (2022). The relevance of features to limit order book learning. Available at SSRN: https://ssrn.com/abstract=4226309.

Zhang, Z., Zohren, S., & Roberts, S. (2018). BDLOB: Bayesian deep convolutional neural networks for limit order books. arXiv preprint arXiv:1811.10041.

Zhang, Z., Zohren, S., & Roberts, S. (2019). Deeplob: Deep convolutional neural networks for limit order books. *IEEE Transactions on Signal Processing*, *67*(11), 3001–3012.