Expert Systems With Applications 194 (2022) 116506

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Forecasting the overnight return direction of stock market index combining global market indices: A multiple-branch deep learning approach

Ruize Gao, Xin Zhang, Hongwu Zhang, Quanwu Zhao, Yu Wang [*]

School of Economics and Business Administration, Chongqing University, Chongqing 400030, China
Chongqing Key Laboratory of Logistics, Chongqing University, Chongqing 400030, China

ARTICLE INFO

Keywords:
Stock market index
Overnight return
Deep learning
Genetic algorithm

ABSTRACT

Forecasting the overnight (close-to-open) return direction of a stock market index has recently attracted great attention. Owing to the strong interactions among stock markets around the globe, one stock market would be inevitably affected by others. In this study, we take global stock market indices as an informative source and propose a deep learning approach combining genetic algorithm to forecast the overnight return direction of a target stock market index. Starting from the multiple-branch input layers representing stock market indices from various regions worldwide, we use multiple convolution units to extract the features from each region. These features are then concatenated and connected with fully connected layers to forecast the daily direction of the overnight return. To optimize the deep neural network, genetic algorithm is used to determine the optimal network architecture and parameters. In the experimental study, we apply the proposed model to forecasting the overnight return directions of nine target indices from Asia, Americas and Europe markets. The experimental results indicate that compared with other competing methods, the proposed model is superior in terms of the accuracy, F-measure and Sharpe ratio.

1. Introduction

Forecasting the direction of stock market index (SMI) movements is an important issue in financial markets since it provides the investors and regulators with useful information for decision-making. However, because a stock market is essentially a dynamic, nonlinear, noisy and chaotic system, forecasting the direction of SMI movements is regarded as a challenging task for investors and researchers (Deboeck, 1994; Zhong & Enke, 2017).

In previous studies, researchers have investigated the predictive power of different factors such as historical prices or returns (Constantinou et al., 2006; Kara et al., 2011; Orimoloye et al., 2020; Wang et al., 2021), technical indicators (Kara et al., 2011; Zhong & Enke, 2017), news text (Kelly & Ahmad, 2018; Li et al., 2014; Tetlock, 2007), and public mood (Bollen et al., 2011; Oliveira et al., 2017; Wang, Lu, et al., 2019; Xing et al., 2019). These studies, among others, mainly focus on the effect of domestic market data on an SMI, paying little attention to the influence of international stock markets. As the trend of globalization becomes increasingly evident, many studies have investigated the cross-correlations among global SMIs. Marschinski and Kantz (2002) and Kwon and Yang (2008) employ the transfer entropy to investigate the relationships among different SMIs and confirm the existence of information transfer. Furthermore, forecasting the SMI movements based on international SMIs has become a hot research topic in recent years, and there is a large body of literature that uses international SMIs as the inputs of machine learning techniques to forecast the target SMI's movements (Hoseinzade & Haratizadeh, 2019; Karaca et al., 2020; Kia et al., 2018; Malagrino et al., 2018).

Since it is difficult to forecast the exact value of an SMI, most studies focus on the direction of SMI movements (Kia et al., 2018; Malagrino et al., 2018; Qiu & Song, 2016). The direction of SMI movements is generally calculated based on returns, which consists of three types, i.e., *close-to-close* (R_{C-C}), *close-to-open* (R_{C-O}) and *open-to-close* (R_{O-C}). Past research found that stock returns are more volatile during trading hours than during nontrading hours (R.French & Roll, 1986), and most studies focused on forecasting the direction of R_{C-C} (Hoseinzade & Haratizadeh, 2019; Kia et al., 2018) and R_{O-C} (Li et al., 2014; Shynkevich et al., 2016). In fact, many stock markets conduct open trading with call auctions that batch orders together and match them in predetermined time intervals to form the price. Consequently, the opening price of a trading day is different from the closing price of the previous trading day, and the return of the closing price compared to the opening price is called the overnight return or non-trading

[*] Corresponding author at: School of Economics and Business Administration, Chongqing University, Chongqing 400030, China.
E-mail addresses: gaoruizecqu@163.com (R. Gao), zhangxin_cq@163.com (X. Zhang), zhw@cqu.edu.cn (H. Zhang), zhaoquanwumx@cqu.edu.cn (Q. Zhao), yuwang@cqu.edu.cn (Y. Wang).

https://doi.org/10.1016/j.eswa.2022.116506
Received 27 July 2021; Received in revised form 30 November 2021; Accepted 1 January 2022
Available online 19 January 2022

return. Kelly and Clark (2011) find that the average returns of *close-to-open* are higher than those of *open-to-close*. Lou et al. (2019) find remarkable persistence in the overnight and intraday components of firm-level returns. From these points of view, forecasting the direction of overnight returns is helpful for investors to make decisions and increase their profits during the periods of call auction and trading.

In this study, we focus on forecasting the daily direction of R_{C-O} of a target SMI by considering three main factors: (1) the historical returns of the target SMI, (2) the historical returns of other SMIs, and (3) the current returns of stock markets that open before the target market on the same trading day. We illustrate the third factor by taking the SSEC (Shanghai Stock Exchange Composite Index, China) as an example. The SSEC opens at 1:30 (UTC) while most stock markets in the Americas open at 14:30 and close at 21:00 (Daylight Savings Time, UTC), and most stock markets in the Europe open at 7:00 and close at 19:00 (UTC). Consequently, the current returns of SMIs in the above two regions would inevitably have impact on the R_{C-O} of the SSEC and hence should be used as an important informative source, in addition to the historical returns of the target SMI and other SMIs.

There are generally three types of methods for forecasting a stock market with these multisource data: (1) conventional machine learning techniques (Zhong & Enke, 2017), (2) ensemble learning techniques (Weng et al., 2018), and (3) deep learning techniques (Long et al., 2019). Furthermore, the input variables (features) are of critical importance to the model performance in forecasting. However, conventional machine learning, such as artificial neural network (ANN) and support vector machine (SVM), and ensemble learning techniques face two challenges when they are applied to forecasting the direction of a target SMI. The first challenge is that in practice, SMIs within a region commonly have a strong correlation, which results in a number of correlation clusters (i.e., intraregional SMIs exhibit similar fluctuation trends) (Giese & Kouzmenko, 2019). If each SMI is used as an input feature for learning, there would be a large number of features with high correlations, which may lead to a biased forecasting model. The second challenge is that learning using all SMIs would cause overfitting, which means that the generalization ability is degraded, since many SMIs are redundant due to the high correlations. The third type of method, i.e., deep learning techniques, is beneficial for handling multisource data and has superior performance (Carta et al., 2021; Long et al., 2020, 2019), especially those with multiple branches or multiple filters (Gaetano et al., 2018; Long et al., 2019). Compared with a single-structure deep learning model (e.g., recurrent units, convolutional units, and long short-term memory units), deep learning models with multiple branches or multiple filters are able to adequately consider the characteristics of samples. For example, Long et al. (2019) employ a multi-filter neural network to forecast the 1-minute direction of stock price, and the results show that their model outperforms traditional machine learning models and single-structure deep learning methods. Gaetano et al. (2018) propose a two-branch convolutional neural network to integrate multi-resolution data, and their model achieves better results than recent classification methods. Although deep learning methods with multiple branches or multiple filters have certain advantages, the architecture and parameters of a deep neural network commonly have significant influences on the predictive performance (Sun et al., 2020). In other words, inappropriate architectural designs and parameters might deteriorate the model performance.

To address the above issues, we propose a Multiple Branch Convolutional Neural Network based on Genetic Algorithm (MBCNN-GA) for forecasting the daily direction of R_{C-O}. In the MBCNN-GA, the convolutional unit of each branch extracts features from intraregional SMIs by parameter sharing, which represents the overall features of intraregional SMIs. The overall features of different regions are then concatenated and put into the fully connected layer to forecast the movements of the target SMI. There are two novelties of the proposed model: (1) a multi-branch structure, where each branch extracts the overall features of specific intraregional SMIs that describe the overall influence of a region on the target SMI, is used in the proposed model; and (2) given the predefined multi-branch blocks of the model, GA is employed to automatically determine the optimal network architecture and combination of parameters for forecasting the R_{C-O} direction. To evaluate the effectiveness of the proposed model, we compare it with some other methods on nine target SMIs. The experimental results show that the proposed approach is superior to the competing methods in terms of predictive performance.

The remainder of the paper is organized as follows. In Section 2, we review the literature that is closely related to our study and summarize our contributions. In Section 3, we present the data, techniques, and proposed model. Section 4 reports the experimental study and discussion. In the last section, we end this paper with some concluding remarks.

2. Related work

2.1. Input variables for SMI forecasting

In general, the input variables (features) for SMI forecasting can be divided into three types: historical price data, technical analysis indicators, and market sentiment. In terms of the historical price data, Constantinou et al. (2006) employ lagged stock returns to forecast the stock return series of the Cyprus Stock Exchange general price index. Orimoloye et al. (2020) employ the opening prices, closing prices, highest prices and lowest prices to forecast the direction of the closing stock price. In addition to historical price data, many researchers employ technical indicators to forecast the SMI. For example, Kara et al. (2011) employ ten technical indicators to forecast the direction of daily closing price movements in the Istanbul Stock Exchange National 100 Index. Zhong and Enke (2017) use a number of features, including technical indicators, to forecast the closing prices of the S&P 500 Index ETF. With the rapid development of text mining techniques, forecasting SMI movements based on textual data, such as news text and investor sentiment, has greatly attracted researchers' attention (Cambria, 2016; Kelly & Ahmad, 2018; Oliveira et al., 2017; Tetlock, 2007; Xing et al., 2019). For example, Bollen et al. (2011) use Twitter emotion as the public mood to forecast an SMI. Li et al. (2014) employ sentiment polarity to predict the direction of individual stock, sector, and index levels.

The above studies focus on the effect of indicators coming from a single market only, which does not consider the influence of cross-market data. With the development of globalization, many studies employ cross-market indicators to forecast a target SMI. Marschinski and Kantz (2002) and Kwon and Yang (2008) employ the transfer entropy to investigate the relationships between different SMIs and confirm the existence of information transfer. In addition, there are many studies attempting to forecast a target SMI based on international SMIs. Kia et al. (2018) employ a hybrid supervised semi-supervised graph-based model to predict the daily movement of a target SMI based on international SMIs. Malagrino et al. (2018) use 12 indices worldwide to forecast IBOVESPA's next day closing direction based on a Bayesian network. Hoseinzade and Haratizadeh (2019) design a convolutional neural network for extracting features from multiple variables of relevant markets to forecast the daily direction of a target SMI. Karaca et al. (2020) employ the max-relevance and min-redundancy (mRMR) to select the features, and use the multi-linear regression, SVM, and ANN to forecast a target SMI.

2.2. SMI forecasting techniques

There are two main categories of forecasting techniques: statistical techniques and machine learning techniques. Statistical techniques used in SMI movement forecasting include the autoregressive moving average (ARMA), autoregressive integrated moving average

Table 1
Summary of key related works.

Authors	Features	Forecast target	Forecast techniques	Forecast type	Evaluation metric
Kia et al. (2018)	International stock indices	Direction of R_{C-C} (daily)	Hybrid supervised semi-supervised graph-based model	Classification	Accuracy
Malagrino et al. (2018)	International stock indices	Direction of R_{C-C} (daily)	Bayesian Network	Classification	Accuracy
Hoseinzade and Haratizadeh (2019)	Historical records of relevant markets	Direction of R_{C-C} (daily)	CNN	Classification	F-measure
Karaca et al. (2020)	Historical records of relevant markets	Daily price	Multi-linear regression, Support Vector Regression (SVR), ANN	Regression	Mean squared error
Long et al. (2019)	Market data (opening, closing, highest, lowest prices, etc.)	Direction of return (1-min frequency)	Multi-filter neural network	Classification	Accuracy

(ARIMA), and generalized autoregressive conditional heteroskedasticity (GARCH) (Abul Basher & Sadorsky, 2016; Franses & Ghijsels, 1999), among others. Since all of them are essentially linear models, their performance tends to degrade in non-linear financial markets (Zhong & Enke, 2017).

Recently, machine learning techniques, such as SVM (Li et al., 2014; Pan et al., 2017; Xiao et al., 2020), ANN (Chen et al., 2003, 2017) and random forest (RF) (Weng et al., 2018), have been successfully applied to forecast financial markets owing to their better generalization ability. However, the performance of the above techniques is highly affected by the input variables. As deep learning techniques, such as convolutional neural network (CNN), recurrent neural network (RNN) and long short-term memory (LSTM), have developed, they have been widely used in many fields, such as marketing (Yilmazer & Birant, 2021), transportation (Wang, Zhang, et al., 2019), and financial market prediction (Carta et al., 2021; Nabipour et al., 2020; Pang et al., 2020). For example, Hoseinzade and Haratizadeh (2019) employ the CNN to forecast the daily direction of a target SMI based on international SMIs. Long et al. (2019) adopt the multi-filter deep learning model to forecast the direction of 1-minute returns.

To provide an overview of the most related studies on machine learning techniques for SMI forecasting, we summarize the representative studies in Table 1.

Table 1 indicates that current studies pay more attention to the influence of international stock market on the target stock market, and most of them focus on the classification task rather than the regression task. The reason might be that it is difficult to forecast the exact value of SMI (Qiu & Song, 2016). In addition, the accuracy and F-measure are common metrics to evaluate the performance. Moreover, the common methods used in the above studies include traditional machine learning methods such as ANN and SVM, and deep learning methods such as CNN and multi-filter neural network.

In machine learning techniques, hyperparameters have significant impact on the predictive performance. Therefore, various evolutionary algorithms, such as particle swarm optimization (PSO) (Hung, 2011; Liu et al., 2018), firefly algorithm (FFA) (Kazem et al., 2013), and genetic algorithm (GA) (Chang & Lee, 2017; Pan et al., 2017), have been used to optimize these parameters. Among them, the GA inspired by the Darwin's theory is an effective optimization method because of its parallel search mechanism. For instance, Pan et al. (2017) adopt the GA to optimize the parameters of a multi-output SVM and achieve better predictive performance than the competing methods.

2.3. Our contributions

Although previous related works have employed international SMIs to forecast a target SMI, most of them have focused on the daily direction of R_{C-C} and ignored the effect of international stock markets on the daily direction of R_{C-O}. In addition, these works mainly take all international SMIs into the machine learning model rather than considering the overall effect of intraregional stock markets. Therefore, we contribute to the literature in the following aspects:

(1) We develop a novel deep learning framework based on a multi-branch convolutional neural network and genetic algorithm to forecast the daily direction of R_{C-O} using international SMIs.
(2) The multi-branch convolutional neural network extracts overall features from each region to improve the predictive performance.
(3) We utilize the GA to automatically determine the optimal network architecture and parameters.

3. Materials and methods

3.1. Data

3.1.1. Calculation and distribution of SMI returns

In this study, we collect 30 SMIs worldwide from Investing.com, one of the top three global financial websites. To avoid any possible bias caused by a short time period, we set a long time frame from October 21, 2013 to December 31, 2020. Since a stock market does not open on every working day (e.g., due to local holidays), there exists a synchronization problem: on some working days, some markets are open while others are not. To address this problem, we remain consistent with (Malagrino et al., 2018) by removing these working days with missing values from the data set. Consequently, there are a total of 1041 days in the data set. The reason we do not replace the missing values with other values is as follows. Suppose the R_{C-C} of SPX increased 3% on Monday, but the Americas market would be closed on Tuesday due to a holiday. Consequently, the missing value on Tuesday will have no influence on other international SMIs the next day, i.e., Wednesday; and it is unreasonable to replace the missing value with other values.

As mentioned before, there are three types of returns, i.e., *close-to-close* ($R_{C\text{-}C,t}$), *close-to-open* ($R_{C\text{-}O,t}$), and *open-to-close* ($R_{O\text{-}C,t}$), for each SMI on trading day t. These types of returns are illustrated in Fig. 1.

In Fig. 1, $Close_t$ and $Open_t$ denote the closing price and opening price of an SMI on trading day t, respectively. The three types of returns are calculated by the following:

$$R_{C\text{-}O,t} = \frac{Open_t - Close_{t\text{-}1}}{Close_{t\text{-}1}}, \tag{1}$$

$$R_{O\text{-}C,t} = \frac{Close_t - Open_t}{Open_t}, \tag{2}$$

$$R_{C\text{-}C,t} = \frac{Close_t - Close_{t\text{-}1}}{Close_{t\text{-}1}}. \tag{3}$$

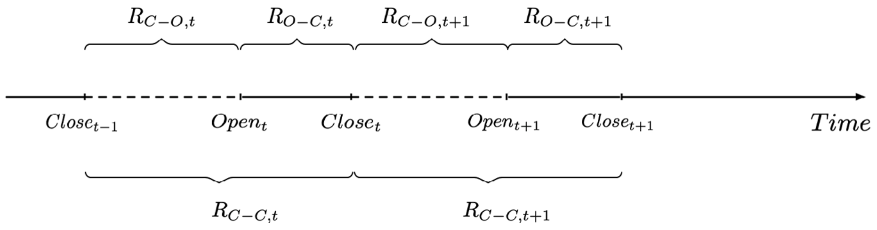


Fig. 1. Three types of SMI returns.

Table 2
Brief description of the SMI returns.

Region	Market	Index Code	% of up labels	% of down labels	Imbalance ratio
Asia	China	SSEC	40.193	59.807	1.488
	Australia[d]	AXJO	3.314	96.686	29.177[c]
	Japan	N225	52.378	47.622	1.100
	South Korea	KS11	59.910	40.090	1.494
	Taiwan	TWII	59.887	40.113	1.493
	Hong Kong	HSI	58.582	41.418	1.414
	India	BSESN	75.998	24.002	3.166[b]
	Indonesia	JKSE	55.290	44.710	1.237
	Philippines	PSI	56.971	43.029	1.324
Americas	United States	SPX	55.543	44.457	1.249
	Canada	GSPTSE	50.360	49.640	1.014
	Brazil	BVSP	22.216	77.784	3.501[b]
	Colombia	COLCAP	0.228	99.772	437.0[c]
	Mexico	MXX	44.524	55.476	1.246
	Peru	SPBLPGPT	44.002	55.998	1.273
Europe	Belgium	BEL 20	53.859	46.141	1.167
	Denmark	OMXC20	51.672	48.328	1.069
	Finland	OMXH25	47.366	52.634	1.111
	France	FCHI	52.391	47.609	1.10
	Germany	GDAXI	53.990	46.010	1.173
	Greece	ATG	54.989	45.011	1.222
	Italy	FTMIB	56.848	43.152	1.317
	Netherlands	AEX	55.652	44.348	1.255
	Norway	OBX	33.611	66.389	1.975[a]
	Poland	WIG30	58.919	41.081	1.434
	Portugal	PSI20	55.326	44.674	1.238
	Russia	IMOEX	48.122	51.878	1.078
	Spain	IBEX 35	54.807	45.193	1.213
	Turkey	BIST 100	69.464	30.536	2.275[a]
	United Kingdom	FTSE	0.055	99.945	1820.0[c]

[a] low imbalance.
[b] medium imbalance.
[c] high imbalance.
[d] The reason we put Australia in Asia market is that their opening time is close.

Since it is difficult to forecast the exact value of an SMI, we divide the $R_{C\text{-}O,t}$ into two classes, i.e., up ($R_{C\text{-}O,t} > 0$) and down ($R_{C\text{-}O,t} \leq 0$), which is consistent with previous studies (Bollen et al., 2011; Nam & Seong, 2019; Shynkevich et al., 2016). Consequently, the forecasting becomes a binary classification problem.

For each SMI in the data set, we calculate the percentages of the up and down labels and the imbalance ratio ($Imbalance\ ratio = \frac{number\ of\ majority\ class\ samples}{number\ of\ minority\ class\ samples}$). The results are shown in Table 2.

According to Fernández et al. (2008), imbalanced data can be divided into three categories: low imbalance (the imbalance ratio is between 1.5 and 3), medium imbalance (the imbalance ratio is between 3 and 9), and high imbalance (the imbalance ratio is higher than 9). We can see from Table 2 that the entire data set is a mixture of balanced data (23 out of 30), low imbalanced data (2 out of 30), medium imbalanced data (2 out of 30), and high imbalanced data (3 out of 30).

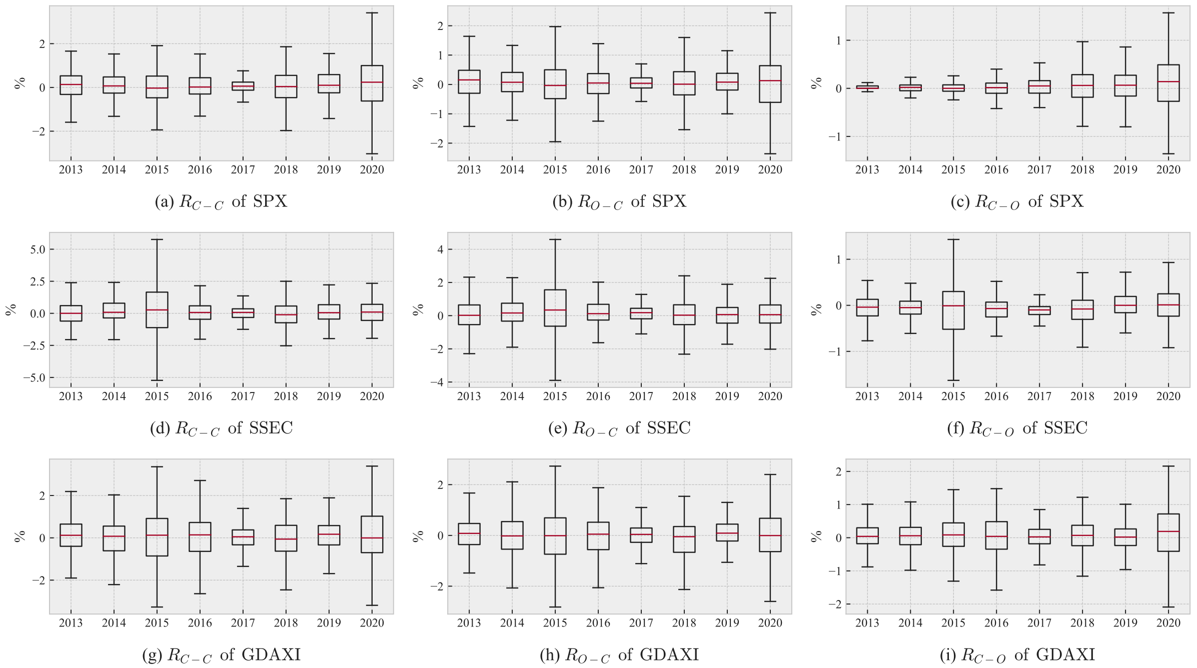
3.1.2. Statistical analysis of SMI returns

To illustrate the trend of SMI volatility, we present box plots of the returns of SPX, SSEC, and GDAXI (DAX 30 Index, Germany) in Fig. 2. The reason we choose the above three SMIs is that they are typical and important SMIs in the Americas, Asia, and Europe markets.

Fig. 2 shows that the volatility (standard deviation) of $R_{C\text{-}O}$ has increased over the six year. Specifically, the SPX increased from 0.104 in 2013 to 0.426 in 2018. It is noteworthy that the volatility of the SSEC suddenly increased in 2015 because of the Chinese stock market turbulence, which is an extreme event.

To illustrate the correlations among international SMIs, we present the correlation matrix of the daily $R_{C\text{-}C}$ and $R_{C\text{-}O}$ from October 21, 2013 to December 31, 2020 in Fig. 3.

In Fig. 3, the SMIs from top to bottom along the y-axis come from Asia, Europe, and Americas markets (corresponding to the x-axis from left to right), and a darker lattice indicates a higher correlation. Fig. 3 shows that the overall color of graph (b) is darker than that of graph (a), which means that the correlations of $R_{C\text{-}O}$ among international SMIs are greater than those of $R_{C\text{-}C}$. Besides, we can find from graph (a) and graph (b) that the colors among intraregional SMIs are darker than those among interregional SMIs, especially in the Asia and Europe markets, which implies stronger correlations.

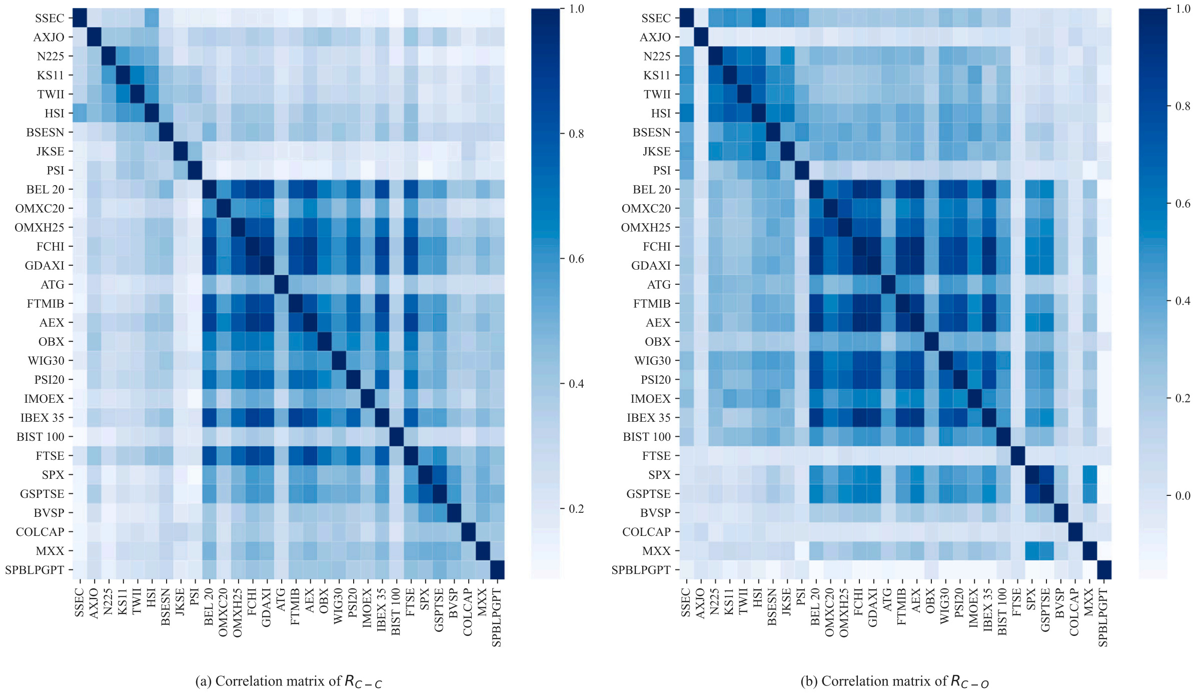


(a) R_{C-C} of SPX (b) R_{O-C} of SPX (c) R_{C-O} of SPX

(d) R_{C-C} of SSEC (e) R_{O-C} of SSEC (f) R_{C-O} of SSEC

(g) R_{C-C} of GDAXI (h) R_{O-C} of GDAXI (i) R_{C-O} of GDAXI

Fig. 2. Box plots of three kinds of returns of SPX, SSEC, and GDAXI.



(a) Correlation matrix of R_{C-C}

(b) Correlation matrix of R_{C-O}

Fig. 3. Correlation matrix of daily $R_{C\text{-}C}$ and $R_{C\text{-}O}$ among international SMIs from October 21, 2013 to December 31, 2020.

3.2. Forecasting techniques

3.2.1. Convolutional neural network

Although ANN has remarkable nonlinear mapping and fitting capability and has been widely applied to various forecasting tasks, its performance depends heavily on the quality of the input features. With the introduction of deep neural network (DNN) by Hinton et al. (2006), DNN has gradually become a prevalent forecasting method. Compared to ANN, DNN has various types of units, such as CNN (Lecun et al., 1998), RNN (Williams & Zipser, 1989), and LSTM (Hochreiter & Schmidhuber, 1997), which enable the network to extract useful features for specific objective tasks.

Among the above kinds of units, CNN works well in tasks such as image recognition (Sun et al., 2020) and financial forecasting (Hoseinzade & Haratizadeh, 2019). The convolutional layer in CNN uses the filters to perform convolutional operations on the input data. During the convolutional operation, each filter utilizes shared weights to horizontally slide and then vertically moves for the next horizontal slide until the whole data set has been scanned. Based on the convolutional operation, a new matrix called the feature map is formed. In this study, inspired by the parameter (weight) sharing of CNN, we employ it to extract the feature from intraregional SMIs for forecasting the target SMI. For the convolutional unit, the size of a filter and the number of filters are important parameters for the model performance. Many tricks, such as dropout (Srivastava et al., 2014), batch normalization (Ioffe & Szegedy, 2015), and residual (He et al., 2016), can be employed to improve the model performance.

3.2.2. Genetic algorithm

GA is a metaheuristic optimization algorithm developed by Holland (1992). It is based on a direct analogy to Darwinian natural selection, and initially works with a set of candidate solutions (individuals) called a population. Combining the selection, crossover, and mutation operations in successive generations, GA obtains an approximate optimal solution by iteratively improving the fitness function value of the individuals. Because of its parallel search mechanism, GA has a lower probability of obtaining a local optimal solution than other algorithms and has been widely used in stock market movement prediction (Kaboudan, 2000; Nuij et al., 2014).

3.3. The proposed model

In this study, we use multiple SMIs from different regions to forecast the target SMI. In practice, intraregional SMIs present similar fluctuation trends, which usually have a similar influence on the target SMI. To address the above issue and improve the predictive performance, we employ the multiple-branch convolutional neural network (MBCNN) to forecast the direction of target SMI based on international SMIs. In the MBCNN, each single branch convolutional unit shares the same weight for the intraregional SMIs, which describes the overall influence of intraregional SMIs on the target SMI. To improve the predictive performance of the MBCNN, we employ the GA to optimize the network architecture and parameters of the model. The framework of the proposed model MBCNN-GA is shown in Fig. 4.

(1) Multiple-branch input block

In Fig. 4, the first block is the multiple-branch input block. Specifically, there are 5 branches. The inputs of Branch (1), Branch (2), and Branch (3) are the returns ($R_{O\text{-}C,t\text{-}1}$, $R_{C\text{-}C,t\text{-}1}$) of SMIs in the Asia, Europe, and Americas markets on trading day *t-1*, respectively. These three regions are chosen because they are the major financial markets around the world.

The input of Branch (4) is the returns ($R_{O\text{-}C,t\text{-}1}$, $R_{C\text{-}C,t\text{-}1}$) of the target SMI. The reason we take the target index returns on trading day t-1 as a separate branch, rather than putting them in the former three branches, is that the historical prices of the target market can provide more insights into the future returns than other markets.

The input of Branch (5) is the returns ($R_{C\text{-}O,t}$) of the *Pre-Open Market*, i.e., stock markets that open before the target market on trading day *t*. According to the time zones, the sequence in which the stock market opens is the Asia, Europe, and Americas markets. Hence, there are three types of structures in Branch (5), depending on which region the target SMI belongs to. These types of structures are shown in Fig. 5.

In Fig. 5, three subgraphs correspond to three types of branch structures. If the target index is in the Asia market, the input of branch (5) is the returns ($R_{C\text{-}O,t}$) of the *Pre-Open Asia Market*, i.e., some Asia stock markets that open before the target market on trading day *t*. If the target index is in the Europe market, the input of branch (5) is the returns ($R_{O\text{-}C,t}$, $R_{C\text{-}C,t}$) of the Asia stock markets and the returns ($R_{C\text{-}O,t}$) of the *Pre-Open Europe Market*, i.e., some Europe stock markets that open before the target market on trading day *t*. If the target index is in the Americas market, the input of branch (5) is the returns ($R_{O\text{-}C,t}$, $R_{C\text{-}C,t}$) of the Asia stock markets, the returns ($R_{C\text{-}O,t}$) of the *Pre-Open Europe Market* and *Pre-Open Americas Market*, i.e., some Europe and the Americas stock markets that open before the target market on trading day *t*.

The multiple-branch input layer geographically segregates the markets to construct the multiple-branch input of the model. In what follows, the features of intraregional SMIs are extracted by the feature extraction block.

(2) Multiple-branch feature extraction block

To extract multiple features from the multiple-branch input block, 1-dimension (1D) convolutional unit is used. We choose the 1D convolutional unit to extract features for the following three reasons: (1) the local connectivity and shared weights of the convolutional unit are conducive to analyzing the correlation of intraregional SMIs, (2) 1D CNN is more suitable for handling time-series data, and (3) the computational complexity of 1D CNN is significantly lower than that of the 2-dimensional (2D) CNN.

Given the input matrix $1 \times N$ in input layer $l-1$ (corresponding to different returns of SMIs), the $1 \times F$ kernel function (filter) is used to calculate the output as:

$$V_j^l = \theta(\sum_{k=0}^{F-1} w_k V_{j+k}^{l-1}), \tag{4}$$

where V_j^l is the value in column j of the layer l, w_k is the weight in column k of the filter, and θ is the activation function:

$$\theta(x) = max(x, 0). \tag{5}$$

In what follows, the convolutional features are flattened to a single vector. In order to better regularize the neural network, a normalization layer is employed to normalize the distribution of each layer's inputs. For a layer with d-dimensional input $\boldsymbol{V} = \{V^{(1)}, V^{(2)}, \dots, V^{(d)}\}$, each dimension is normalized by:

$$\widehat{V}^{(k)} = \frac{V^{(k)} - \mu_{\boldsymbol{V}}}{\sigma_{\boldsymbol{V}}}, \tag{6}$$

$$y^{(k)} = \gamma^{(k)} \widehat{V}^{(k)} + \beta_{(k)}, \tag{7}$$

where $\mu_{\boldsymbol{V}}$ denotes the mean values, $\sigma_{\boldsymbol{V}}$ represents the standard deviation, and $y^{(k)}$ denotes the normalized value of the kth dimension. The pair of parameters $\gamma^{(k)}$and $\beta^{(k)}$ are determined along with the original model parameters.

(3) Multiple-branch feature fusion block

The extracted features of multiple branches are concatenated by a concatenation layer in the feature fusion block. And then, a fully connected layer is used to convert the concatenated features into the output. The reason we use the fully connected layer is that compared with the convolutional layer, the fully connected layer is more suitable for extracting the global features regardless of the dependence between two adjacent layers, which is helpful for improving the model performance.

In the fully connected layer, given the output of layer l and activation function θ, the output of the ith neuron at layer l+1 is obtained as follows:

$$V_i^{l+1} = \theta(\sum_k V_k^l w_{k,i}^l) \tag{8}$$

In Fig. 4, the last layer is the fully connected layer with a softmax function (normalized exponential function) which is used for classification. The probability p_i of the ith category is calculated as follows:

$$p_i = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}} \tag{9}$$

where n is the number of categories, and z_j denotes the output of the jth neuron in the previous fully connected layer.

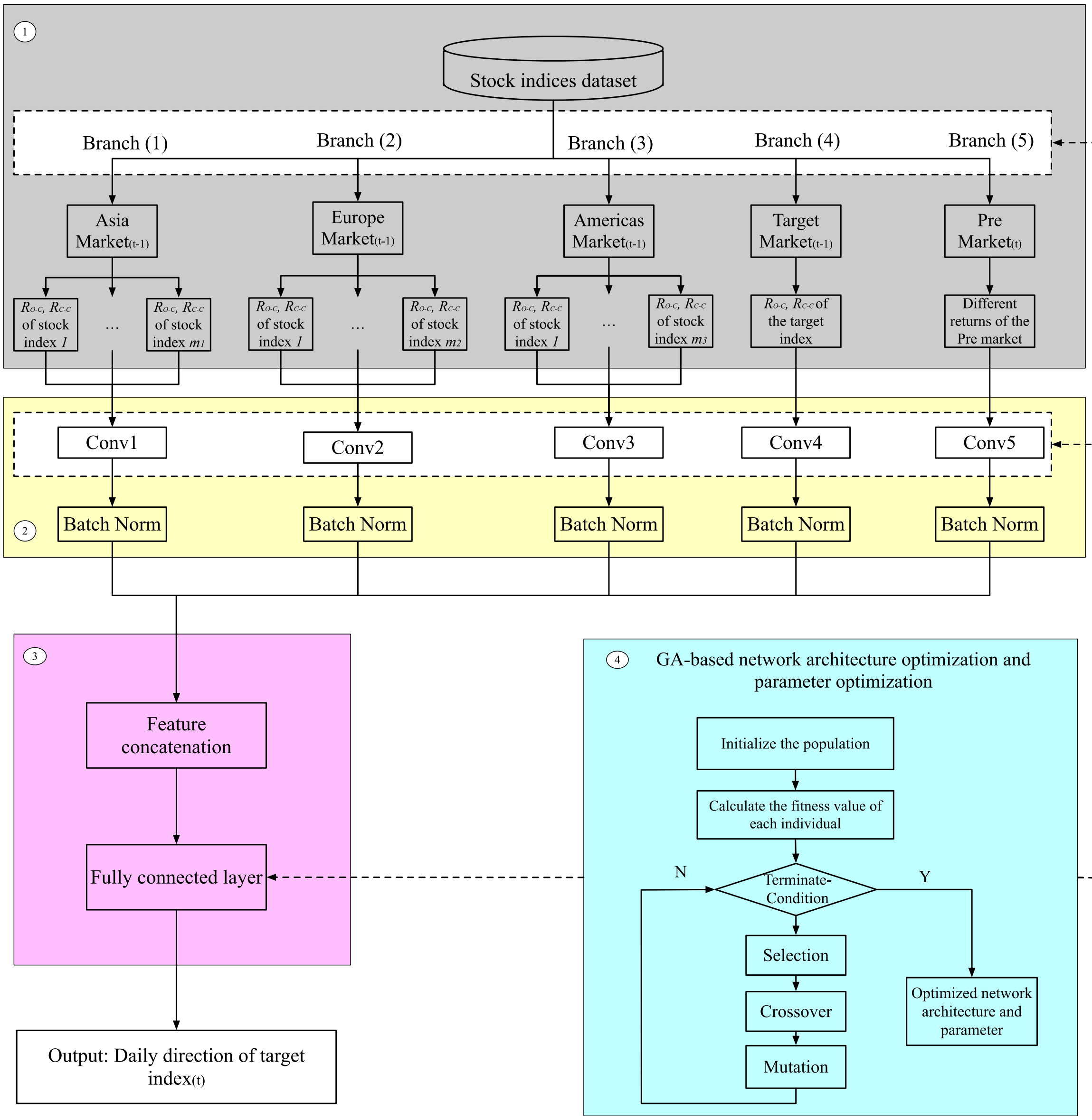

1
Stock indices dataset
Branch (1)
Branch (2)
Branch (3)
Branch (4)
Branch (5)
Asia Market(t-1)
Europe Market(t-1)
Americas Market(t-1)
Target Market(t-1)
Pre Market(t)
Ro-c, Rc-c of stock index 1
…
Ro-c, Rc-c of stock index m1
Ro-c, Rc-c of stock index 1
…
Ro-c, Rc-c of stock index m2
Ro-c, Rc-c of stock index 1
…
Ro-c, Rc-c of stock index m3
Ro-c, Rc-c of the target index
Different returns of the Pre market
Conv1
Conv2
Conv3
Conv4
Conv5
2
Batch Norm
Batch Norm
Batch Norm
Batch Norm
Batch Norm
3
Feature concatenation
Fully connected layer
Output: Daily direction of target index(t)
4
GA-based network architecture optimization and parameter optimization
Initialize the population
Calculate the fitness value of each individual
N
Terminate-Condition
Y
Selection
Crossover
Mutation
Optimized network architecture and parameter

Fig. 4. The framework of the proposed model.

(4) GA-based network architecture and parameter optimization

Different combinations of branches (each branch includes an input block and a feature extraction block) in Fig. 4 form neural networks with different architectures, which provide different predictive performances. In addition to the network architecture, CNN has several important hyperparameters such as the number of filters and filter size of the convolutional unit, and the number of neurons in the fully connected layer. In order to obtain a concise model with better generalization ability, GA is employed to optimize the network architecture and parameters.

Specifically, we take the SSEC (Shanghai Stock Exchange Composite Index) as the target SMI to illustrate how an individual is encoded. Suppose an individual is $\{0, 1, 1, 1, 0, 3, 3, 50\}$, which represents that (1) the Branch 2, Branch 3, Branch 4 are selected; and (2) the number of filters of the convolutional unit, the filter size of the convolutional unit and the number of neurons of the fully connected layer are set to 3, 3 and 50, respectively.

The main steps of the GA-based network architecture and parameter optimization are shown in Algorithm 1.

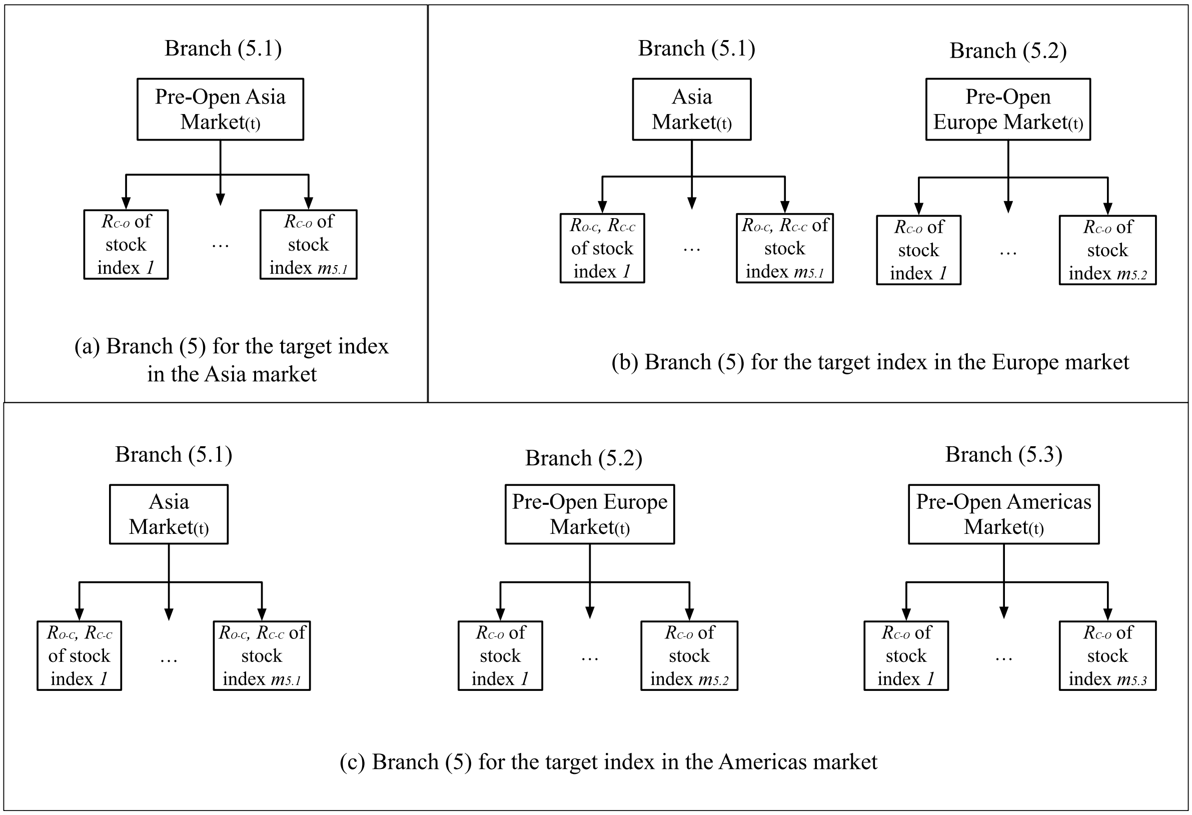
Given the population size, the maximal number of generations, the maximal number of stalled generations (the best fitness function value does not change over the limited number) and the data set, the proposed algorithm attempts to discover the best individual (the neural network with the optimal architecture and parameters) to classify the given data set using natural selection, crossover, and mutation over successive generations.

The detailed steps of Algorithm 1 are illustrated as follows.

Step 1: Initializing population and individuals. First, the counter for the current generation number and the current stalled generation number are initialized to zero (line 1). And then, a population with the predefined population size is randomly initialized (line 2), where $\boldsymbol{x}_p^{(t)}$ denotes the pth individual of the tth generation. Based on the initialized neural network architecture, samples in the training data set are used to train the model, and the fitness function value (loss value) of the individual on the validation set is obtained by line 3 in Algorithm 1. The loss value is obtained by binary cross-entropy:

$$Loss = -\frac{1}{m}\sum_{i=1}^{m}[y_i \cdot \ln p(y_i) + (1 - y_i) \cdot \ln(1 - p(y_i))], \tag{10}$$

where m denotes the number of samples, y_i represents the actual label of the ith sample and $p(y_i)$ denotes the predicted probability of the ith



(a) Branch (5) for the target index in the Asia market

(b) Branch (5) for the target index in the Europe market

(c) Branch (5) for the target index in the Americas market

Fig. 5. Three types of structures of Branch (5).

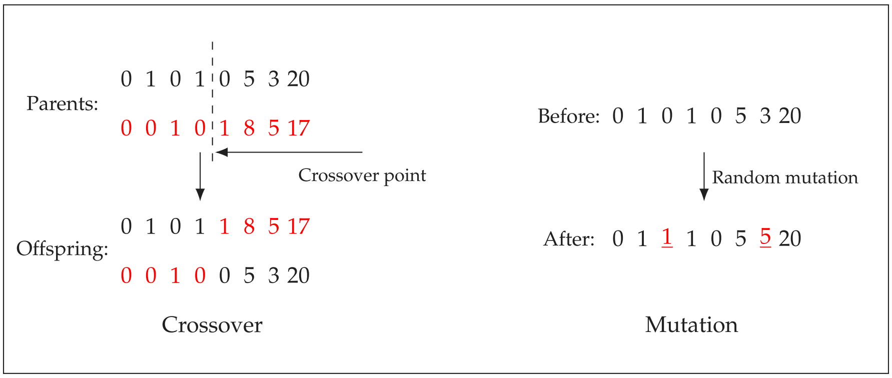


Fig. 6. An example of crossover and mutation.

sample. By comparing the fitness values among individuals in the first generation, the individual $\boldsymbol{x}_b^{(t)}(b \in \{1, 2, \dots, n\})$ with the best (minimum) fitness value is selected as the temporary optimal individual (line 4–5).

Step 2: Search for the optimal individual. During the successive generations, a specific proportion of individuals with top-ranked fitness values in the previous generation are selected as the parents (elites) (line 7). Crossover and mutation operators are then conducted to generate offspring based on the elites (line 8). The crossover operator is a mechanism for exchanging genes between the two individuals using a crossover point. The mutation operator randomly changes the genes of an individual. If the gene in the first five genes (or seven genes) is selected to mutate, 1 is changed to 0, and vice versa. If the gene in the last three genes is selected to mutate, a random value from the predefined range is chosen to replace the position. An example of crossover and mutation (taking the target SMI in the Asia markets as an example) is shown in Fig. 6.

Consequently, a new population is composed of elites and offspring through crossover and mutation (line 9), and the fitness value of an individual is evaluated (line 10). The individual $\boldsymbol{x}_b^{(t)}$ with the best fitness value ($f_b^{(t)}$) is selected as the optimal individual in the new population (line 11). In what follows, $\boldsymbol{x}_b^{(t)}$ is compared with $\boldsymbol{x}_b$, and the one with a lower fitness value is selected as the current optimal individual $\boldsymbol{x}_b$.

Step 3: Stopping criteria. *Step 2* (line 7–18) is repeated until a termination condition has been reached. Termination conditions include (1) the algorithm has reached the maximum number of generations (T) and (2) the best fitness function value has reached a plateau for a given number of consecutive generations (H) (line 14). When Algorithm 1 terminates, it outputs the optimal individual representing the optimal architecture and parameters of the MBCNN.

Algorithm 1: GA-based network architecture and parameter optimization

```
Input  : The population size $n$, the maximal generation
         number $T$, the maximal stall generation number $H$,
         dataset $D$, learning algorithm of MBCNN $MB_{\mathcal{L}}$.
Output: The optimal individual (the neural network with the
         optimal architecture and parameters) $\boldsymbol{x}_b$.
$t \leftarrow 0$; $h \leftarrow 0$;
Initialize a population $P^{(t)} \leftarrow \{\boldsymbol{x}_1^{(t)}, \boldsymbol{x}_2^{(t)}, \ldots, \boldsymbol{x}_p^{(t)}, \ldots, \boldsymbol{x}_n^{(t)}\}$;
Evaluate each individual in the initial population
  $f_p^{(t)} \leftarrow f(MB_{\mathcal{L}}(D^{\boldsymbol{x}_p^{(t)}}))$;
Select individual $\boldsymbol{x}_b^{(t)}$ with the best $f_b^{(t)}$ in $P^{(t)}$;
$\boldsymbol{x}_b \leftarrow \boldsymbol{x}_b^t$; $f_b \leftarrow f_b^{(t)}$;
while $t < T$ do
    Select the elite from $P^{(t)}$;
    Generate offspring by crossover and mutation operation;
    Generate a new population $P^{(t)}$;
    Evaluate each individual in the new population
      $f_p^{(t)} \leftarrow f(MB_{\mathcal{L}}(D^{\boldsymbol{x}_p^{(t)}}))$;
    Select individual $\boldsymbol{x}_b^{(t)}$ with the best $f_b^{(t)}$ in $P^{(t)}$;
    if $f_b^{(t)} \leq f_b$ then
        $h \leftarrow h + 1$;
        if $h > H$ then break;
    else
        $h \leftarrow 0$; $\boldsymbol{x}_b \leftarrow \boldsymbol{x}_b^t$; $f_b \leftarrow f_b^{(t)}$;
    end
    $t \leftarrow t + 1$;
end
```

4. Experimental study

In this section, we design the experiment and evaluate the effectiveness of the proposed MBCNN-GA. Section 4.1 presents the evaluation metrics. Section 4.2 describes the experimental setting. Section 4.3 reports the predictive results of the proposed approach and other competing methods. Section 4.4 discusses the experimental results and presents some possible applications of the proposed approach.

4.1. Evaluation metrics

Traditional cross-validation is not applicable for a time series data set since the training set should only contain samples whose time labels are earlier than those in the testing set. Therefore, we adopt the time window slicing cross-validation strategy such that the training set consists of samples earlier than those in the validation and testing sets. Training periods with proportion of 40%, 45%, 50%, 55%, 60%, and 65% (testing periods with proportion of 40%, 35%, 30%, 25%, 20%, and 15%, respectively) are used for training (testing) the model, and 20% of the data set is used as the validation set to prevent overfitting. In this way, the testing sets in the time windows include samples from 2019 and 2020 and most samples from 2018, which considers the bullish and bearish periods (2020 pandemic selloff). The slicing time windows are shown in Fig. 7.

To evaluate the performance of the proposed model and other methods, we employ accuracy as the first evaluation metric. Because the data set consists of imbalanced data (cf. Table 2), the F-measure (Gunduz et al., 2017; Hoseinzade & Haratizadeh, 2019) is employed as the second evaluation metric.

Table 3
Confusion matrix for binary classification.

Actual/Predicted	Positive	Negative
Positive	TP	FN
Negative	FP	TN

Table 4
Parameters of the competing methods.

Competing methods	Parameters description
ANN	Epoch number: 50; Hidden layer neuron number: 20; Optimizer: lbfgs.
SVM	Kernel function: poly; C: 1; degree: 3.
RF	The number of trees in the forest: 5.
CNN	Epoch number: 50; Number and size of convolutional unit: 30,3; Optimizer: adam.
RNN	Epoch number: 50; Number of recurrent unit: 50; Optimizer: adam.
LSTM	Epoch number: 50; Number of LSTM unit: 50; Optimizer: adam.

Table 5
Parameters of the proposed model.

MBCNN-GA	Parameters description

MBCNN	Optimizer: Adam;
	Learning rate: 0.001;
	Activation function of the convolutional layers and the fully connected layer: the Rectified Linear Unit (ReLU);
	Activation function of the last layer: Softmax.

GA	Population size: 50; Maximal generation number: 30; Maximal stall generation number: 5; Selection rate of population: 0.3; Crossover rate of population(except for the elite): 0.8; Mutation mechanism: random change a gene value of the first five genes and a gene value of the last three genes.
Parameters space	Filter number of convolutional units: $[1, 30]$; Number of neurons in the fully connected layer: $[10, 99]$; Filter size of convolutional unit: $\{1, 3, 5, 7\}$.

The accuracy and F-measure are calculated based on the confusion matrix, which stores the counts of correctly and incorrectly classified instances, as shown in Table 3. In the confusion matrix, *TP* (true positive) means that the actual result and prediction are both ‘up’, and *TN* (true negative) means that the actual result and prediction are both ‘down’. If the actual result is ‘down’ and the prediction is ‘up’, it is denoted as *FP* (false positive). Similarly, if the actual result is ‘up’ and the prediction is ‘down’, it is denoted as *FN* (false negative).

Based on Table 3, the precision, recall, F-measure, and accuracy are computed by the following equations:

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{11}$$

$$\text{Recall} = \frac{TP}{TP + FN}, \tag{12}$$

$$\text{F-Measure} = \frac{2 \times Precision \times Recall}{Precision + Recall}, \tag{13}$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}. \tag{14}$$

4.2. Experimental setting

First, we compare the performance of the model combining international SMIs with that based only on historical data of the target SMI

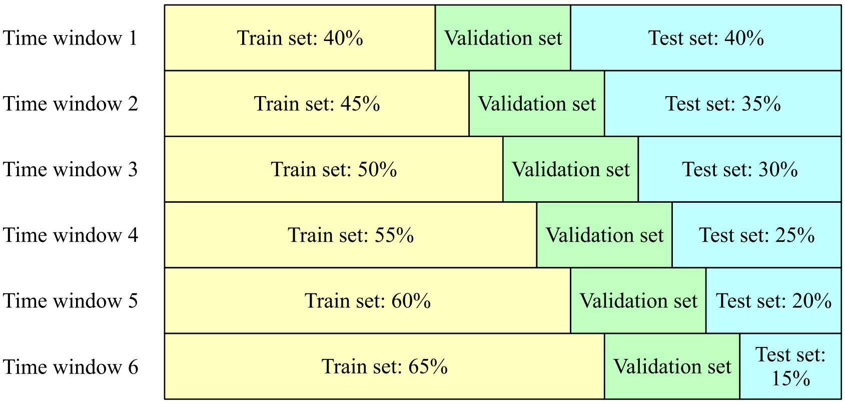


Fig. 7. Illustration of the slicing time windows.

Table 6
Average accuracy and F-measure of each model on multiple SMIs.

	Method	Input	SPX	GDAXI	N225	FCHI	KS11	FTMIB	HSI	SSEC	AEX
Accuracy	ANN	T	0.5363	0.5125	0.5143	0.5218	0.6004	0.5400	0.5839	0.5582	0.5314
		T+I	0.6368	0.7112	0.6845	0.7077	0.7388	0.6983	0.6588	0.6150	0.7085
	SVM	T	0.5635	0.5328	0.5378	0.5067	0.6098	0.5525	0.5875	0.4861	0.5418
		T+I	0.6068	0.6645	0.7122	0.7141	0.6854	0.6712	0.6670	0.5667	0.6565
	RF	T	0.4782	0.5422	0.4813	0.4998	0.5386	0.5207	0.5397	0.5362	0.5165
		T+I	0.6289	0.6722	0.6813	0.6490	0.7216	0.6468	0.6906	0.6262	0.6555
	CNN	T	0.5356	0.4943	0.5489	0.5256	0.6044	0.5548	0.5702	0.5459	0.5336
		T+I	0.6582	0.7129	0.7227	0.7227	0.7445	0.7431	0.7271	0.6477	0.6929
	RNN	T	0.5358	0.5208	0.5451	0.5121	0.6157	0.5458	0.5659	0.5554	0.5465
		T+I	0.6910	0.7409	0.7409	0.7473	0.7699	0.7604	0.7236	0.6434	0.7280
	LSTM	T	0.5160	0.5260	0.5459	0.5170	0.6215	0.5469	0.5884	0.5328	0.5455
		T+I	0.6923	0.7406	0.7259	0.7515	0.7510	0.7592	0.7250	0.6674	0.7105
F-measure	ANN	T	0.5851	0.5981	0.5427	0.4355	0.7386	0.6482	0.7286	0.4362	0.6526
		T+I	0.6670	0.7288	0.6997	0.7030	0.7825	0.7257	0.7187	0.6038	0.7282
	SVM	T	0.7196	0.6925	0.6993	0.6607	0.7573	0.7095	0.7401	0.0055	0.7028
		T+I	0.7351	0.7301	0.7662	0.6925	0.7903	0.7596	0.7733	0.2742	0.7454
	RF	T	0.5426	0.5780	0.5203	0.5063	0.6390	0.5799	0.6257	0.4735	0.5623
		T+I	0.6652	0.6954	0.6997	0.6595	0.7750	0.6865	0.7445	0.5949	0.6912
	CNN	T	0.6285	0.6072	0.5961	0.4592	0.7467	0.6764	0.7152	0.3704	0.6753
		T+I	0.6845	0.7303	0.7167	0.7167	0.7933	0.7694	0.7759	0.6162	0.7156
	RNN	T	0.6301	0.6155	0.6143	0.5190	0.7586	0.6521	0.7128	0.3780	0.6817
		T+I	0.7113	0.7543	0.7400	0.7437	0.8165	0.7798	0.7724	0.5980	0.7466
	LSTM	T	0.6187	0.6187	0.6232	0.4898	0.7628	0.6553	0.7346	0.3094	0.6875
		T+I	0.7136	0.7547	0.7527	0.7506	0.7981	0.7856	0.7736	0.6358	0.7337

itself. Traditional machine learning models including the ANN (Hinton, 1989), SVM (Chang & Lin, 2011) and RF (Breiman, 2001), and deep learning models including the CNN (Krizhevsky et al., 2012), RNN (Abadi et al., 2016), and LSTM (Hochreiter & Schmidhuber, 1997) are used as the competing methods. The parameters of the above methods are shown in Table 4.

Second, we compare the performance of different methods combining international SMIs. In addition to the above competing methods shown in Table 4, we add a series of other competing methods.

(1) Naive baselines: Random prediction and label t-1 (using label t-1 as the predicted label of trading day t).
(2) Standard machine learning and deep learning methods shown in Table 4 with all international SMIs.
(3) Machine learning and deep learning methods combined with the clustering method: ANN+Cluster, SVM+Cluster, RF+Cluster, CNN+Cluster, RNN+Cluster, and LSTM+Cluster. In these methods, we first employ the K-means clustering method (Sculley, 2010) to form two clusters. And then, the SMIs in the same category of the target index are chosen as the inputs.
(4) MBCNN: the standard multiple branch convolutional neural network that is not optimized by GA.

The details of the architecture and parameters of the proposed MBCNN-GA are shown in Table 5.

In this work, Scikit-learn 0.22.1 (Pedregosa et al., 2011) and Tensorflow 2.3.0 (Abadi et al., 2016) in Python 3.7 are utilized to implement Algorithm 1 and the competing methods.

4.3. Experimental results

4.3.1. Predictive results combining international SMIs

First, we compare the performance of the model combining international SMIs with that based only on historical data on nine target SMIs, i.e., the SPX, GDAXI, N225, FCHI, KS11, FTMIB, HSI, SSEC, and AEX. The above indices are chosen based on the following three rules: (1) indices from Asia, Americas and Europe markets should be considered; (2) the imbalanced ratio of the R_{C-O} should be low; and (3) the target indices should be important in international stock markets.

Table 7
Accuracy of each model on multiple target SMIs over different testing periods.

Method	Train rate						Train rate						Train rate					
	40%	45%	50%	55%	60%	65%	40%	45%	50%	55%	60%	65%	40%	45%	50%	55%	60%	65%
	SPX						GDAXI						N225					
Random Prediction	0.568	0.521	0.502	0.475	0.502	0.529	0.547	0.460	0.521	0.490	0.493	0.497	0.518	0.455	0.466	0.464	0.493	0.541
Label t-1	0.542	0.526	0.534	0.529	0.502	0.516	0.472	0.485	0.492	0.494	0.507	0.484	0.511	0.512	0.518	0.510	0.517	0.497
ANN	0.612	0.644	0.668	0.663	0.617	0.618	0.727	0.707	0.728	0.693	0.679	0.732	0.695	0.742	0.687	0.625	0.708	0.650
ANN+Cluster	0.655	0.597	0.658	0.636	0.632	0.567	0.686	0.668	0.674	0.598	0.656	0.573	0.727	0.715	0.706	0.670	0.689	0.694
SVM	0.588	0.575	0.601	0.625	0.622	0.631	0.650	0.666	0.728	0.648	0.646	0.650	0.671	0.677	0.728	0.705	0.746	0.745
SVM+Cluster	0.588	0.592	0.620	0.648	0.627	0.656	0.643	0.674	0.684	0.674	0.670	0.662	0.679	0.685	0.709	0.705	0.746	0.752
RF	0.628	0.592	0.594	0.667	0.617	0.675	0.674	0.679	0.687	0.670	0.679	0.643	0.700	0.679	0.671	0.705	0.689	0.643
RF+Cluster	0.609	0.616	0.620	0.636	0.636	0.656	0.640	0.701	0.626	0.613	0.589	0.631	0.621	0.647	0.671	0.709	0.670	0.694
CNN	0.640	0.633	0.658	0.674	0.675	0.669	0.736	0.718	0.719	0.709	0.689	0.707	0.712	0.718	0.725	0.716	0.713	**0.752**
CNN+Cluster	0.664	0.600	0.658	0.686	0.660	0.701	0.712	0.704	0.703	0.705	0.708	0.694	0.712	0.704	0.703	0.713	0.703	0.707
RNN	0.662	0.660	0.677	0.701	0.713	0.732	0.751	0.734	0.728	0.736	0.751	0.745	0.739	0.742	**0.748**	0.739	0.751	0.726
RNN+Cluster	0.659	0.668	0.684	0.709	0.675	0.720	0.739	0.732	0.725	0.739	0.737	0.758	0.703	0.707	0.712	0.705	0.718	0.739
LSTM	0.671	**0.677**	0.671	0.705	0.703	0.726	0.760	0.748	0.738	0.720	0.732	0.745	0.741	0.732	0.744	0.728	0.703	0.707
LSTM+Cluster	0.667	0.658	0.684	0.686	0.679	0.682	0.736	0.751	0.722	0.736	0.722	0.752	0.739	0.726	0.741	0.713	0.722	0.713
MBCNN	0.664	0.660	0.671	0.693	0.675	0.707	0.727	0.756	0.754	0.747	0.761	0.707	0.722	0.721	0.738	0.724	0.708	0.732
GA-MBCNN	**0.715**	**0.677**	**0.732**	**0.747**	**0.718**	**0.764**	**0.763**	**0.762**	**0.767**	**0.751**	**0.785**	**0.771**	**0.770**	**0.759**	0.741	**0.770**	**0.766**	**0.752**
	FCHI						KS11						FTMIB					
Random Prediction	0.547	0.482	0.530	0.483	0.488	0.503	0.532	0.499	0.492	0.467	0.474	0.529	0.556	0.482	0.524	0.498	0.493	0.484
Label t-1	0.501	0.501	0.505	0.494	0.507	0.510	0.535	0.534	0.543	0.548	0.555	0.548	0.489	0.488	0.489	0.490	0.483	0.478
ANN	0.691	0.704	0.703	0.686	0.756	0.707	0.763	0.723	0.728	0.739	0.708	0.771	0.703	0.710	0.671	0.701	0.699	0.707
ANN+Cluster	0.659	0.682	0.639	0.655	0.675	0.637	0.755	0.712	0.706	0.724	0.737	0.656	0.712	0.696	0.684	0.686	0.694	0.675
SVM	0.679	0.726	0.709	0.720	0.699	0.752	0.669	0.666	0.674	0.670	0.713	0.720	0.681	0.655	0.681	0.670	0.665	0.675
SVM+Cluster	0.686	0.690	0.706	0.686	0.718	0.720	0.681	0.674	0.671	0.667	0.713	0.771	0.695	0.677	0.681	0.667	0.665	0.656
RF	0.647	0.660	0.658	0.644	0.660	0.624	0.727	0.682	0.712	0.766	0.742	0.701	0.652	0.633	0.655	0.625	0.641	0.675
RF+Cluster	0.604	0.647	0.633	0.674	0.608	0.599	0.739	0.712	0.687	0.678	0.689	0.752	0.604	0.627	0.658	0.651	0.689	0.675
CNN	0.712	0.718	0.725	0.716	0.713	0.752	0.739	0.721	0.741	0.732	0.732	**0.803**	0.712	0.751	0.754	0.728	0.718	0.796
CNN+Cluster	0.703	0.696	0.716	0.716	0.727	0.732	0.743	0.721	0.716	0.724	0.766	0.777	0.691	0.688	0.754	0.701	0.718	0.764
RNN	0.739	0.742	0.748	0.739	0.751	0.764	**0.770**	0.775	0.757	0.739	0.794	0.783	0.753	0.753	0.760	0.736	0.770	0.790
RNN+Cluster	0.703	0.707	0.712	0.705	0.718	0.739	0.767	0.751	0.735	0.743	0.770	0.777	0.717	0.721	0.732	0.728	0.751	0.783
LSTM	0.741	0.734	**0.751**	0.755	0.751	**0.777**	0.739	0.756	0.744	0.724	0.766	0.777	0.772	0.742	0.748	0.739	0.751	**0.803**
LSTM+Cluster	0.710	0.732	0.744	0.732	0.742	0.752	0.758	0.762	0.741	0.724	0.766	0.796	0.719	0.723	0.732	0.732	0.766	0.752
MBCNN	0.734	0.715	0.732	0.747	0.746	0.732	0.763	0.740	0.728	0.739	0.751	0.758	0.719	0.737	0.738	0.732	0.722	0.745
GA-MBCNN	**0.765**	**0.751**	**0.751**	**0.774**	**0.780**	0.771	**0.770**	**0.792**	**0.767**	**0.801**	**0.823**	**0.803**	**0.775**	**0.786**	**0.796**	**0.770**	**0.789**	0.783
	HSI						SSEC						AEX					
Random Prediction	0.554	0.496	0.476	0.475	0.488	0.541	0.492	0.474	0.505	0.502	0.555	0.529	0.564	0.485	0.540	0.513	0.493	0.478
Label t-1	0.523	0.526	0.534	0.544	0.560	0.605	0.516	0.512	0.505	0.510	0.498	0.510	0.518	0.526	0.534	0.536	0.522	0.516
ANN	0.669	0.663	0.620	0.651	0.675	0.675	0.609	0.619	0.639	0.605	0.593	0.624	0.707	0.690	0.725	0.674	0.727	0.726
ANN+Cluster	0.659	0.655	0.661	0.670	0.727	0.643	0.647	0.649	0.613	0.579	0.550	0.643	0.669	0.696	0.696	0.670	0.665	0.656
SVM	0.640	0.644	0.652	0.659	0.694	0.713	0.588	0.573	0.559	0.567	0.541	0.573	0.650	0.649	0.661	0.667	0.656	0.656
SVM+Cluster	0.657	0.655	0.649	0.655	0.699	0.726	0.616	0.603	0.588	0.598	0.560	0.605	0.650	0.641	0.636	0.628	0.646	0.637
RF	0.686	0.699	0.722	0.713	0.694	0.631	0.655	0.644	0.633	0.617	0.560	0.650	0.693	0.696	0.652	0.667	0.589	0.637
RF+Cluster	0.698	0.693	0.629	0.693	0.718	0.732	0.645	0.696	0.623	0.602	0.555	0.561	0.703	0.622	0.626	0.674	0.694	0.707
CNN	0.743	0.723	0.719	0.732	0.713	0.732	0.657	0.644	0.633	0.628	0.656	0.669	0.698	0.701	0.700	0.690	0.656	0.713
CNN+Cluster	0.731	0.715	0.719	0.736	0.713	0.726	0.691	0.674	0.649	0.670	0.641	0.682	0.715	0.677	0.712	0.667	0.699	0.739
RNN	0.731	0.715	0.709	0.728	0.732	0.726	0.693	0.660	0.639	0.636	0.589	0.643	0.727	0.723	0.728	0.720	0.718	0.752
RNN+Cluster	0.731	0.723	0.712	0.736	0.732	**0.758**	**0.722**	0.704	0.674	0.651	0.646	0.624	0.700	0.699	0.719	0.720	0.727	0.752
LSTM	0.724	0.726	0.728	0.724	0.727	0.720	0.679	0.655	0.665	0.644	0.675	0.688	0.724	0.732	0.709	0.705	0.679	0.713
LSTM+Cluster	0.719	0.712	0.712	0.724	0.708	0.707	0.712	0.679	0.655	0.651	0.636	0.662	0.688	0.718	0.690	0.724	0.675	0.726
MBCNN	0.741	0.732	0.722	0.693	0.694	0.732	0.676	0.666	0.661	0.667	0.646	0.682	0.722	0.726	0.722	0.716	0.722	0.726
GA-MBCNN	**0.784**	**0.767**	**0.767**	**0.743**	**0.751**	0.739	0.683	**0.718**	**0.722**	**0.674**	**0.737**	**0.701**	**0.741**	**0.751**	**0.732**	**0.728**	**0.737**	**0.758**

The average accuracy and *F*-measure of each model are shown in Table 6, where the inputs *T* and I denote the target index data (T) and the international index data (I), respectively.

Table 6 shows that models with target index data and international index data outperform those with only target index data. The results illustrate that combining international SMIs is helpful to improve the model performance rather than adding noise to the model. Therefore, we set T+I as the input of the model in the following experiments.

4.3.2. *Predictive results using different forecasting techniques*

We further compare the performance of different methods combining international SMIs. The inputs of the competing methods (except for random prediction and label *t*-1) and the proposed model are all T+I. The accuracy of each model on multiple target SMIs over different testing periods is shown in Table 7.

Table 7 shows that machine learning-based techniques outperform the random prediction and label *t*-1 methods. Moreover, we find that the MBCNN-GA is superior to most of the competing models over different proportions of testing periods.

To test the statistical significance of the proposed model and the competing methods in terms of accuracy, a group of paired *t* tests is conducted. The results are shown in Table 8.

In Table 8, the accuracy in boldface of each column represents the highest result of each column, and the symbol $*$ represents the statistical significance of difference between our model and the competing model with the highest accuracy. Table 8 shows that the average accuracy of the proposed model is higher than that of other models on all target indices. In addition, we find that there is a statistically significant difference between the proposed model and the competing methods, which implies that our model can effectively integrate international SMIs to improve the predictive performance.

For some SMIs, the data sets are imbalanced (cf. Table 2). Therefore, we use the F-measure to evaluate the performance of the proposed model and the competing models. The results are shown in Table 9.

Table 8
Average accuracy of each model on multiple SMIs.

	SPX	GDAXI	N225	FCHI	KS11	FTMIB	HSI	SSEC	AEX
Random Prediction	0.5161	0.5013	0.4895	0.5055	0.4988	0.5062	0.5051	0.5093	0.5121
Label t-1	0.5248	0.4891	0.5106	0.5031	0.5438	0.4862	0.5486	0.5082	0.5252
ANN	0.6368	0.7112	0.6845	0.7077	0.7388	0.6983	0.6588	0.6150	0.7085
ANN+Cluster	0.6241	0.6425	0.7003	0.6579	0.7151	0.6911	0.6694	0.6137	0.6755
SVM	0.6068	0.6645	0.7122	0.7141	0.6854	0.6712	0.6670	0.5667	0.6565
SVM+Cluster	0.6216	0.6678	0.7126	0.7009	0.6960	0.6734	0.6734	0.5949	0.6397
RF	0.6289	0.6722	0.6813	0.6490	0.7216	0.6468	0.6906	0.6262	0.6555
RF+Cluster	0.6290	0.6333	0.6686	0.6274	0.7094	0.6509	0.6940	0.6135	0.6710
CNN	0.6582	0.7129	0.7227	0.7227	0.7445	0.7431	0.7271	0.6477	0.6929
CNN+Cluster	0.6615	0.7044	0.7070	0.7151	0.7411	0.7192	0.7233	0.6677	0.7013
RNN	0.6910	0.7409	0.7409	0.7473	0.7699	0.7604	0.7236	0.6434	0.7280
RNN+Cluster	0.6858	0.7383	0.7139	0.7139	0.7573	0.7386	0.7321	0.6703	0.7195
LSTM	0.6923	0.7406	0.7259	0.7515	0.7510	0.7592	0.7250	0.6674	0.7105
LSTM+Cluster	0.6758	0.7364	0.7257	0.7351	0.7578	0.7372	0.7139	0.6661	0.7035
MBCNN	0.6784	0.7419	0.7242	0.7344	0.7466	0.7323	0.7191	0.6662	0.7225
MBCNN-GA	**0.7254*****	**0.7662****	**0.7595*****	**0.7652****	**0.7924****	**0.7832***	**0.7586****	**0.7059***	**0.7410*****

*: 0.1; **:0.05; ***:0.01.

Table 9 shows that, similarly, machine learning-based techniques outperform the random prediction and label t-1 methods. In addition, we can see that the MBCNN-GA is superior to most of the competing models over different proportions of training periods in terms of the F-measure.

To test the statistical significance of the proposed model and the competing methods in terms of the F-measure, a group of paired t tests is conducted. The results are shown in Table 10.

In Table 10, the F-measure in boldface represents the highest result of each column, and the symbol $*$ represents the statistical significance of difference between our model and the competing model with the highest accuracy. Table 10 shows that the proposed MBCNN-GA achieves the highest mean F-measure compared to the other models on all nine target SMIs. Also, there is a statistically significant difference between MBCNN-GA and the competing methods over eight target SMIs, i.e., the SPX, GDAXI, FCHI, KS11, FTMIB, HSI, SSEC, and AEX. Based on the results in Tables 7 and 10, we find that our model has a better performance than other competing models in terms of the accuracy and F-measure.

4.3.3. Economic evaluation

Since different predicted labels have different importance, we evaluate the model performance in terms of the percent profit (i.e., predicted daily return) and Sharpe ratio, which accounts for the profits over risk. First, we calculate the daily return as follows. If the predictive label is positive, R_{C-O} on trading day t is used as the daily return. If the predictive label is negative, the daily return of trading day t is 0. Second, we calculate the average daily return. Last, we employ the ratio of the average daily return to the standard deviation (Johnman et al., 2018; Kelly & Ahmad, 2018) to calculate the Sharpe ratio of each model over multiple target SMIs. In the performance evaluation, we use the data split method shown in Fig. 7, and calculate the average daily return and Sharpe ratio in each testing data set. And then, we calculate the means and standard deviations of the average daily return and Sharpe ratio over all testing data sets. In this experiment, we do not consider the transaction costs and taxes.

The results of the percent profits (average daily return (%)) of each model on multiple SMIs are shown in Table 11.

Table 11 shows that the average daily return of the MBCNN-GA is superior to those of the competing methods. Furthermore, the MBCNN-GA outperforms the CNN and CNN+Cluster in most SMIs. In addition, we observe that the average daily returns of deep learning methods (CNN, RNN and LSTM) are higher than those of traditional machine learning methods (ANN, SVM and RF) and the simple prediction methods (random prediction and label t-1 method).

The results of the Sharpe ratio of each model on multiple SMIs are shown in Table 12.

Table 12 shows that the MBCNN-GA outperforms other methods in terms of the Sharpe ratio. Moreover, we observe that the deep learning methods (CNN, RNN, and LSTM) are superior to the traditional machine learning methods (ANN, SVM and RF). We can also find that the Sharpe ratios obtained by machine learning techniques are higher than those obtained by simple prediction methods (random prediction and label t-1 method).

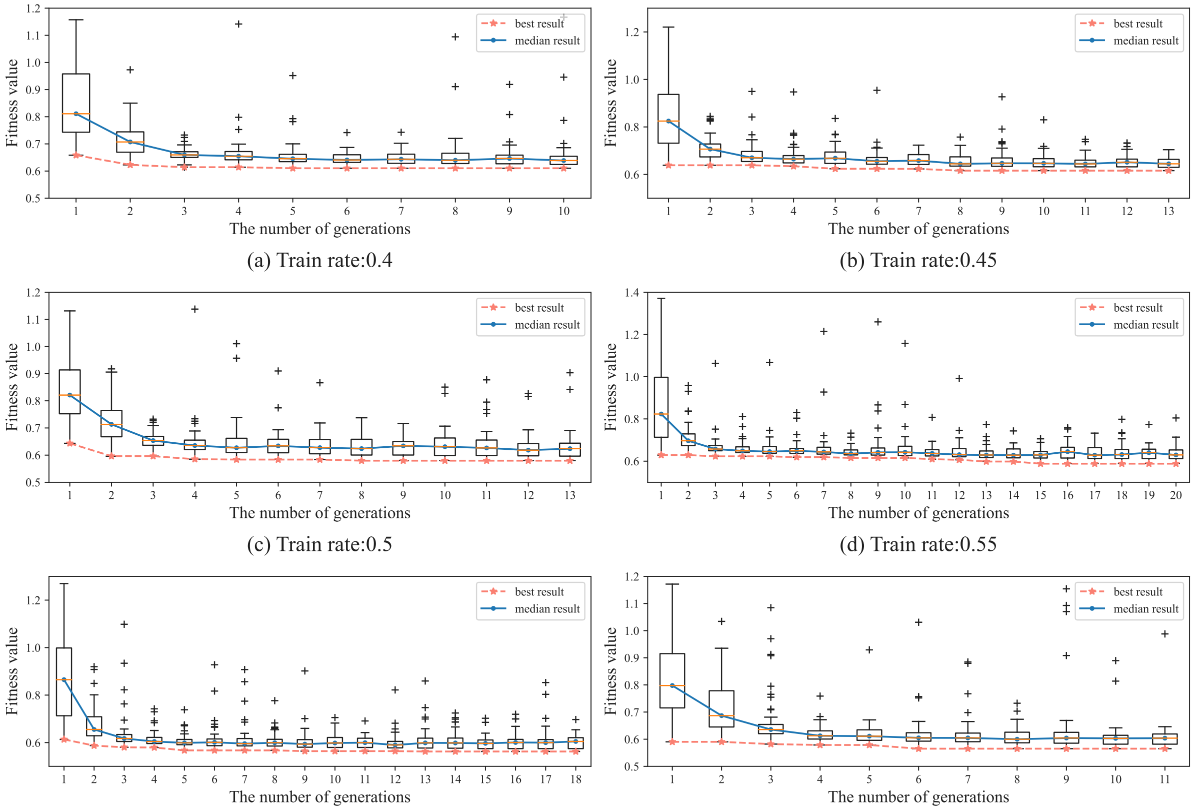
4.3.4. Evolutionary trajectories

Because the optimal solution of the GA is obtained by iterative processes, it is necessary to investigate the robustness and convergence of the MBCNN-GA to discover the optimal architectures and parameters. Therefore, we summarize the fitness function values of the individuals in each generation using box plots and connect the best and median fitness values using a dashed line and a solid line, respectively. Taking the SPX index as an example, we plot the evolutionary trajectories over different training rates, as shown in Fig. 8, where the horizon axis denotes the number of generations, and the vertical axis denotes the fitness function value.

Fig. 8 indicates that the best and median fitness values decrease as the evolution progresses. According to the height of each box, we can observe that the variation of the fitness value decreases as the number of generations increases, which implies that the results tend to reach a steady state when discovering the optimal neural network architectures and parameters. We can also find that the algorithms terminate around generation 10 when the train rate is less than 0.65, which indicates that our model converges quickly, and the setting of 30 generations is reasonable.

To further determine which branches are helpful for improving the model performance, we plot the evolutionary heat-maps over different training periods, which are shown in Fig. 9. In Fig. 9, the symbols on the y-axis represent different branches: A: $Target\ Market_{(t\text{-}1)}$, B: $Pre\text{-}Open\ Americas\ Market_t$, C: $Pre\text{-}Open\ Europe\ Market_t$, D: $Asia\ Market_t$, E: $Americas\ Market_{(t\text{-}1)}$, F: $Europe\ Market_{(t\text{-}1)}$, and G: $Asia\ Market_{(t\text{-}1)}$ (cf. Fig. 4 and Fig. 5). The dark blue represents that the branch is selected while the light color means that the branch is not selected.

Fig. 9 shows that GA gradually achieves a better branch structure over iterative processes for improving the model performance. In addition, we observe that branches C ($Pre\text{-}Open\ Europe\ Market_t$) and D ($Asia\ Market_t$) appear five times (5/6) in the optimal branch structure, which means that the performance of the Europe and Asia markets on trading day t is more helpful for improving the predictive performance of the SPX.



(a) Train rate:0.4

(b) Train rate:0.45

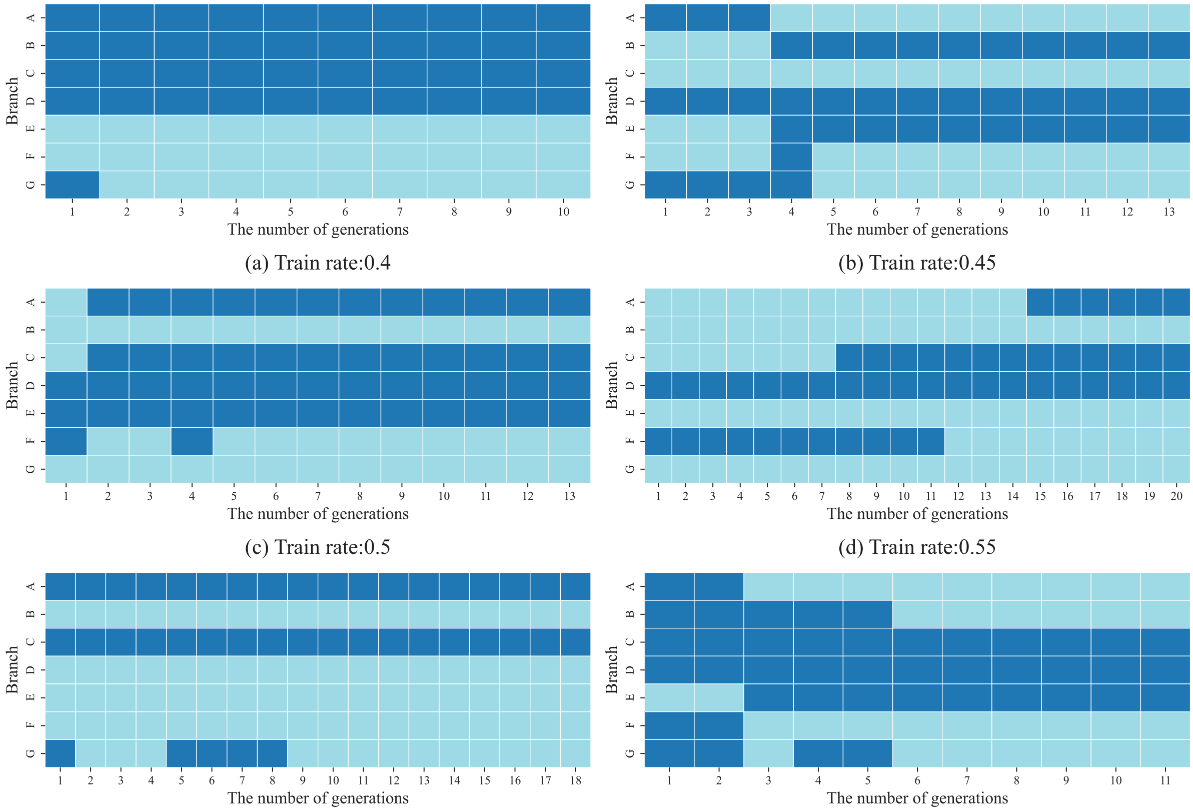
(c) Train rate:0.5

(d) Train rate:0.55

(e) Train rate:0.6

(f) Train rate:0.65

Fig. 8. The evolutionary trajectory of MBCNN-GA in discovering the best neural network architecture and parameters on the SPX index.



(a) Train rate:0.4

(b) Train rate:0.45

(c) Train rate:0.5

(d) Train rate:0.55

(e) Train rate:0.6

(f) Train rate:0.65

Fig. 9. The evolutionary trajectory of selected branch structure.

Table 9
F-measure of each model on multiple target SMIs over different training periods.

Method	Train rate						Train rate						Train rate					
	40%	45%	50%	55%	60%	65%	40%	45%	50%	55%	60%	65%	40%	45%	50%	55%	60%	65%
	SPX						GDAXI						N225					
Random Prediction	0.589	0.550	0.541	0.509	0.540	0.543	0.559	0.488	0.548	0.506	0.518	0.490	0.531	0.491	0.498	0.474	0.523	0.538
Label t-1	0.591	0.564	0.580	0.594	0.563	0.568	0.511	0.518	0.523	0.535	0.546	0.503	0.547	0.557	0.549	0.540	0.563	0.521
ANN	0.655	0.675	0.690	0.697	0.646	0.639	0.746	0.718	0.740	0.726	0.700	0.744	0.716	0.756	0.698	0.620	0.734	0.675
ANN+Cluster	0.696	0.637	0.686	0.686	0.698	0.622	0.723	0.686	0.685	0.615	0.673	0.573	0.755	0.741	0.714	0.684	0.703	0.696
SVM	0.728	0.713	0.730	0.751	0.744	0.743	0.747	0.740	0.758	0.716	0.713	0.706	0.751	0.756	**0.773**	0.767	0.787	0.762
SVM+Cluster	0.726	**0.719**	0.736	0.759	0.745	0.755	0.734	0.740	0.729	0.732	0.727	0.697	0.761	0.763	0.764	0.769	**0.791**	0.769
RF	0.674	0.623	0.634	0.705	0.661	0.695	0.708	0.699	0.708	0.695	0.707	0.654	0.725	0.693	0.689	0.737	0.709	0.646
RF+Cluster	0.649	0.622	0.653	0.674	0.658	0.663	0.668	0.720	0.653	0.643	0.616	0.670	0.643	0.677	0.685	0.719	0.699	0.704
CNN	0.678	0.651	0.681	0.702	0.691	0.705	0.760	0.737	0.732	0.730	0.700	0.723	0.706	0.722	0.713	0.704	0.706	0.748
CNN+Cluster	0.698	0.612	0.682	0.715	0.679	0.725	0.729	0.714	0.705	0.714	0.721	0.704	0.736	0.726	0.717	0.737	0.716	0.720
RNN	0.697	0.674	0.697	0.729	0.725	0.747	0.769	0.744	0.740	0.751	0.766	0.756	0.746	0.737	0.744	0.736	0.740	0.736
RNN+Cluster	0.690	0.687	0.704	0.738	0.696	0.741	0.756	0.741	0.728	0.744	0.751	0.762	0.705	0.708	0.696	0.686	0.698	0.725
LSTM	0.699	0.693	0.693	0.724	0.728	0.746	0.777	0.758	0.745	0.742	0.750	0.756	0.748	0.736	0.745	0.763	0.745	**0.779**
LSTM+Cluster	0.696	0.672	0.699	0.709	0.691	0.699	0.757	0.762	0.729	0.753	0.743	0.764	0.710	0.712	0.722	0.705	0.713	0.752
MBCNN	0.714	0.679	0.687	0.709	0.699	0.742	0.736	0.776	0.771	0.765	0.773	0.726	0.758	0.742	0.759	0.743	0.721	0.731
GA-MBCNN	**0.738**	0.712	**0.757**	**0.780**	**0.755**	**0.796**	**0.778**	**0.776**	**0.777**	**0.772**	**0.800**	**0.780**	**0.789**	**0.779**	0.760	**0.789**	0.788	0.761
	FCHI						KS11						FTMIB					
Random Prediction	0.547	0.488	0.539	0.471	0.488	0.480	0.570	0.550	0.547	0.505	0.542	0.580	0.579	0.517	0.566	0.524	0.527	0.484
Label t-1	0.512	0.492	0.495	0.484	0.498	0.497	0.610	0.608	0.615	0.617	0.652	0.654	0.546	0.534	0.548	0.549	0.538	0.506
ANN	0.673	0.695	0.699	0.692	0.746	0.712	0.798	0.761	0.761	0.783	0.768	0.824	0.734	0.735	0.700	0.725	0.734	0.726
ANN+Cluster	0.645	0.646	0.622	0.637	0.653	0.632	0.795	0.763	0.755	0.769	0.781	0.710	0.747	0.729	0.735	0.727	0.722	0.691
SVM	0.731	0.727	0.637	0.695	0.613	0.752	0.780	0.778	0.781	0.777	0.811	0.814	0.770	0.75	0.769	0.762	0.755	0.751
SVM+Cluster	0.734	0.710	0.652	0.682	0.688	0.722	0.787	0.780	0.776	0.772	0.808	0.841	0.776	0.763	0.767	0.758	0.750	0.727
RF	0.668	0.670	0.669	0.635	0.664	0.651	0.768	0.739	0.772	0.804	0.797	0.771	0.693	0.678	0.693	0.667	0.686	0.702
RF+Cluster	0.624	0.660	0.639	0.677	0.624	0.613	0.782	0.760	0.751	0.732	0.758	0.808	0.639	0.676	0.693	0.687	0.730	0.702
CNN	0.706	0.722	0.713	0.704	0.706	0.748	0.791	0.769	0.790	0.781	0.783	0.846	0.741	0.774	0.787	0.758	0.745	0.812
CNN+Cluster	0.699	0.680	0.704	0.706	0.711	0.737	0.788	0.774	0.774	0.771	0.812	0.824	0.710	0.712	0.769	0.717	0.738	0.789
RNN	0.746	0.737	0.744	0.736	0.740	0.758	0.812	0.819	0.805	0.788	0.840	0.835	0.771	0.772	0.783	0.763	0.791	0.800
RNN+Cluster	0.705	0.708	0.696	0.686	0.698	0.725	0.808	0.796	0.782	0.789	0.814	0.824	0.732	0.741	0.751	0.750	0.770	0.795
LSTM	0.748	0.736	0.745	0.750	0.745	0.780	0.782	0.799	0.791	0.769	0.816	0.831	0.795	0.772	0.775	0.772	0.780	**0.821**
LSTM+Cluster	0.718	0.728	0.735	0.729	0.738	0.748	0.799	0.804	0.786	0.772	0.816	0.842	0.741	0.749	0.753	0.760	0.790	0.772
MBCNN	0.742	0.711	0.734	0.736	0.734	0.724	0.800	0.789	0.777	0.788	0.806	0.821	0.741	0.760	0.770	0.764	0.748	0.770
GA-MBCNN	**0.772**	**0.757**	**0.755**	**0.767**	**0.781**	**0.786**	**0.807**	**0.830**	**0.813**	**0.838**	**0.864**	**0.852**	**0.802**	**0.818**	**0.821**	**0.792**	**0.818**	0.809
	HSI						SSEC						AEX					
Random Prediction	0.583	0.542	0.526	0.505	0.541	0.576	0.470	0.478	0.523	0.496	0.583	0.532	0.581	0.515	0.569	0.528	0.518	0.481
Label t-1	0.588	0.591	0.594	0.602	0.635	0.677	0.485	0.500	0.514	0.511	0.549	0.544	0.562	0.562	0.565	0.572	0.558	0.548
ANN	0.714	0.713	0.676	0.707	0.746	0.756	0.607	0.617	0.622	0.565	0.573	0.638	0.726	0.717	0.739	0.704	0.737	0.746
ANN+Cluster	0.695	0.699	0.706	0.707	0.769	0.714	0.644	0.626	0.593	0.522	0.552	0.627	0.668	0.734	0.715	0.699	0.682	0.679
SVM	0.757	0.758	0.761	0.765	0.792	0.805	0.246	0.228	0.242	0.252	0.304	0.374	0.753	0.748	0.745	**0.752**	0.741	0.733
SVM+Cluster	0.767	0.766	0.760	0.763	0.795	0.807	0.339	0.332	0.345	0.371	0.387	0.466	0.746	0.735	0.722	0.722	0.724	0.708
RF	0.733	0.737	0.765	0.760	0.761	0.710	0.623	0.608	0.599	0.569	0.516	0.654	0.729	0.719	0.686	0.703	0.648	0.663
RF+Cluster	0.739	0.740	0.691	0.745	0.776	0.786	0.606	0.667	0.569	0.548	0.528	0.531	0.729	0.658	0.661	0.706	0.706	0.726
CNN	0.778	0.769	0.765	0.778	0.773	0.792	0.623	0.633	0.582	0.573	0.625	0.662	0.716	0.726	0.719	0.708	0.687	0.738
CNN+Cluster	0.774	0.759	0.767	0.781	0.774	0.786	0.670	0.645	0.604	0.639	0.603	0.667	0.733	0.699	0.727	0.697	0.727	0.773
RNN	0.773	0.758	0.756	0.773	0.788	0.786	0.661	0.629	0.586	0.554	0.552	0.606	0.750	0.743	0.746	0.742	0.738	0.761
RNN+Cluster	0.773	0.770	0.762	0.784	0.788	**0.816**	0.693	0.675	0.614	0.588	0.615	0.587	0.719	0.721	0.732	0.735	0.740	0.769
LSTM	0.767	0.770	0.768	0.774	0.783	0.780	0.647	0.629	0.621	0.594	0.657	0.667	0.746	0.754	0.728	0.732	0.702	0.740
LSTM+Cluster	0.760	0.759	0.754	0.772	0.770	0.768	**0.697**	0.659	0.621	0.603	0.604	0.654	0.714	0.738	0.703	0.745	0.702	0.743
MBCNN	0.777	0.777	0.773	0.744	0.758	0.796	0.652	0.628	0.629	0.642	0.631	0.688	0.738	0.747	0.731	0.736	0.724	0.746
GA-MBCNN	**0.821**	**0.805**	**0.807**	**0.786**	**0.805**	0.798	0.690	**0.742**	**0.755**	**0.698**	**0.779**	**0.756**	**0.773**	**0.779**	**0.753**	0.751	**0.772**	**0.789**

4.4. Discussion

First, we can conclude from the above experimental results that a forecasting model with the target SMI and international SMIs is better than the one with only target index, which is in line with previous research (Malagrino et al., 2018). This observation demonstrates that adding international SMIs is helpful for improving the performance. The possible reason is that cross-market dependencies have become an important factor affecting the target stock market.

Second, the MBCNN-GA statistically outperforms the competing methods on multiple target SMIs over different testing periods in terms of the accuracy and *F*-measure. We attribute this to its advantages over other models as follows: (1) The multiple-branch structure of the proposed model extracts features of different intraregional SMIs, which decreases the correlations existing in these indices. (2) Extracted convolutional features from multiple intraregional SMIs prevent the model from learning all the SMIs for forecasting the target SMI, which decreases the risk of over-fitting. (3) The GA is conducive to determining the optimal architecture and parameters of deep neural network.

Third, the MBCNN-GA has the following advantages. Compared with the network-based model (Kia et al., 2018; Malagrino et al., 2018), the major advantage of the MBCNN-GA is that it does not need to consider the rigid topological structure of the network before building the models, which is beneficial for applying the proposed model to other predictive tasks. Compared with the model (mRMR + machine learning method) in (Karaca et al., 2020), the proposed model directly learns patterns from input to output without extra handcrafted feature selection, which is helpful to simplifying the data analysis work. Compared with the CNN model in (Hoseinzade & Haratizadeh, 2019), the multiple-input design in the MBCNN-GA considers the correlations among intraregional SMIs, and the experimental results show that the performance of the MBCNN-GA is superior to that of the CNN.

Table 10
Average F-measure of each model on multiple SMIs.

	SPX	GDAXI	N225	FCHI	KS11	FTMIB	HSI	SSEC	AEX
Random Prediction	0.5454	0.5183	0.5093	0.5021	0.5489	0.5326	0.5457	0.5137	0.5320
Label t-1	0.5768	0.5227	0.5461	0.4962	0.6259	0.5369	0.6146	0.5173	0.5612
ANN	0.6670	0.7288	0.6997	0.7030	0.7825	0.7257	0.7187	0.6038	0.7282
ANN+Cluster	0.6710	0.6592	0.7157	0.6393	0.7622	0.7249	0.7151	0.5939	0.6961
SVM	0.7351	0.7301	0.7662	0.6925	0.7903	0.7596	0.7733	0.2742	0.7454
SVM+Cluster	0.7401	0.7265	0.7693	0.6978	0.7937	0.7569	0.7762	0.3732	0.7261
RF	0.6652	0.6954	0.6997	0.6595	0.7750	0.6865	0.7445	0.5949	0.6912
RF+Cluster	0.6530	0.6617	0.6875	0.6396	0.7654	0.6880	0.7460	0.5748	0.6978
CNN	0.6845	0.7303	0.7167	0.7167	0.7933	0.7694	0.7759	0.6162	0.7156
CNN+Cluster	0.6853	0.7144	0.7251	0.7063	0.7905	0.7391	0.7737	0.6379	0.7261
RNN	0.7113	0.7543	0.7400	0.7437	0.8165	0.7798	0.7724	0.5980	0.7466
RNN+Cluster	0.7096	0.7471	0.7030	0.7030	0.802	0.7566	0.7820	0.6286	0.7359
LSTM	0.7136	0.7547	0.7527	0.7506	0.7981	0.7856	0.7736	0.6358	0.7337
LSTM+Cluster	0.6944	0.7513	0.7189	0.7326	0.8034	0.7608	0.7637	0.6396	0.7240
MBCNN	0.7050	0.7577	0.7424	0.7302	0.7968	0.7586	0.7707	0.6450	0.7368
MBCNN-GA	**0.7564*****	**0.7805****	**0.7776**	**0.7696*****	**0.8340****	**0.8099****	**0.8037****	**0.7367*****	**0.7694*****

*: 0.1; **:0.05; ***:0.01.

Table 11
The percent profits (average daily return (%)) of each model on multiple SMIs (mean (± std. Deviation))

	SPX	GDAXI	N225	FCHI	KS11	FTMIB	HSI	SSEC	AEX
Random Prediction	−0.0201 (±0.0435)	−0.0087 (±0.0405)	0.0160 (±0.0200)	−0.0180 (±0.0391)	0.0631 (±0.0336)	0.0222 (±0.0297)	0.0281 (±0.0419)	−0.0047 (±0.0161)	−0.0185 (±0.0357)
Label t-1	−0.0048 (±0.0143)	0.0003 (±0.0113)	0.0471 (±0.0090)	−0.0227 (±0.0124)	0.1197 (±0.0296)	0.0050 (±0.0086)	0.1020 (±0.0319)	0.0111 (±0.0128)	−0.0049 (±0.0145)
ANN	0.1067 (±0.0089)	0.1824 (±0.0188)	0.1874 (±0.0237)	0.1474 (±0.0142)	0.2157 (±0.0357)	0.1273 (±0.0127)	0.1906 (±0.0348)	0.0676 (±0.0074)	0.1397 (±0.0150)
ANN+Cluster	0.0985 (±0.0312)	0.1536 (±0.0377)	0.1927 (±0.0135)	0.1216 (±0.0116)	0.2045 (±0.039)	0.1506 (±0.0157)	0.2087 (±0.0370)	0.0711 (±0.0089)	0.1394 (±0.0165)
SVM	0.0935 (±0.0204)	0.1678 (±0.0430)	0.1601 (±0.0262)	0.1329 (±0.0462)	0.2190 (±0.0628)	0.1137 (±0.0205)	0.1701 (±0.0478)	0.0344 (±0.0127)	0.1056 (±0.0339)
SVM+Cluster	0.1219 (±0.0289)	0.1635 (±0.0270)	0.1657 (±0.0289)	0.1467 (±0.0402)	0.2265 (±0.0675)	0.1182 (±0.0232)	0.1815 (±0.0463)	0.0369 (±0.0111)	0.0963 (±0.0259)
RF	0.0763 (±0.0427)	0.1224 (±0.0396)	0.1746 (±0.0192)	0.0965 (±0.0204)	0.2197 (±0.0389)	0.1236 (±0.0145)	0.2263 (±0.0162)	0.0541 (±0.0101)	0.0899 (±0.0178)
RF+Cluster	0.0775 (±0.0161)	0.1159 (±0.0254)	0.1717 (±0.0209)	0.0778 (±0.0296)	0.2128 (±0.0450)	0.1240 (±0.0292)	0.2089 (±0.0400)	0.0599 (±0.0111)	0.1341 (±0.0251)
CNN	0.1205 (±0.0183)	0.2131 (±0.0198)	0.1875 (±0.0063)	0.1883 (±0.0187)	0.2371 (±0.0486)	0.1756 (±0.0133)	0.2423 (±0.0210)	0.0829 (±0.0032)	0.1677 (±0.0187)
CNN+Cluster	0.1292 (±0.0245)	0.2004 (±0.0164)	0.1908 (±0.0075)	0.1800 (±0.0198)	0.2427 (±0.0542)	0.1782 (±0.0326)	0.2424 (±0.0244)	0.0773 (±0.0086)	0.1728 (±0.0246)
RNN	0.1414 (±0.0257)	0.2462 (±0.0216)	0.2145 (±0.0071)	0.2121 (±0.0217)	0.2693 (±0.0520)	0.2048 (±0.0167)	0.2277 (±0.0218)	0.0795 (±0.0036)	0.1977 (±0.0171)
RNN+Cluster	0.1481 (±0.0250)	0.2329 (±0.0333)	0.2124 (±0.0098)	0.1873 (±0.0314)	0.2708 (±0.0549)	0.1877 (±0.0211)	0.2286 (±0.0203)	0.0813 (±0.0058)	0.1953 (±0.0264)
LSTM	0.1383 (±0.0273)	0.2235 (±0.0254)	0.2102 (±0.0043)	0.2072 (±0.0219)	0.2676 (±0.0507)	0.1899 (±0.0116)	0.2343 (±0.0180)	0.0796 (±0.0070)	0.1698 (±0.0129)
LSTM+Cluster	0.1405 (±0.0214)	0.2122 (±0.0285)	0.2093 (±0.0035)	0.1940 (±0.0244)	0.2695 (±0.0505)	0.1849 (±0.0142)	0.2268 (±0.0165)	0.0878 (±0.0050)	0.1768 (±0.0270)
MBCNN	0.1328 (±0.0178)	0.2415 (±0.0273)	0.2149 (±0.0078)	0.2009 (±0.0373)	0.2700 (±0.0485)	0.1860 (±0.0096)	0.2344 (±0.0306)	0.0895 (±0.0042)	0.1871 (±0.0205)
MBCNN- GA	**0.1608** (±0.0271)	**0.2585** (±0.0397)	**0.2248** (±0.0082)	**0.2175** (±0.0347)	**0.2772** (±0.0496)	**0.2138** (±0.0225)	**0.2635** (±0.0123)	**0.0960** (±0.0157)	**0.2087** (±0.0281)

Our model is readily applicable for investors and regulators. For investors, forecasting the direction of overnight returns can help them make decisions during call auctions. For example, if the output of the model is positive, they could buy or hold the investment product related to the target SMI (e.g., an exchange traded fund (ETF)). If the predicted label is negative, they could sell the investment products to decrease their investment risk. For regulators, forecasting the direction of R_{C-O} helps them monitor the volatility of the stock market. For instance, if the target SMI continuously decreases and the output of the model is still negative, regulators could make suitable policies to restore investors' confidence and maintain the stability of a stock market.

5. Conclusion and future work

Under the tendency of economic globalization, one stock market would be inevitably affected by others. The effects of other stock markets on the target stock market could be reflected in the overnight returns. Therefore, we propose a novel deep learning model to forecast the direction of the overnight returns of a target SMI by integrating international SMIs. The first contribution of the study is that we employ the deep learning techniques to forecast the direction of R_{C-O} of the target SMI, which is helpful for investors to make decisions and increase their profits during call auction and trading periods. Second, we propose the MBCNN-GA model in which the overall features of intraregional SMIs are extracted by multiple branches for forecasting the target SMI. We use nine target SMIs to compare our model and the competing methods, and the experimental results show that the proposed model is superior to the competing methods in terms of the predictive accuracy and F-measure.

There are some limitations of our study, which provide opportunities for future investigation. First, in addition to international SMIs, many other factors may affect the target SMI movements. These factors

Table 12
Sharp ratio of each model on multiple SMIs (mean (± std. Deviation)).

Method	SPX	GDAXI	N225	FCHI	KS11	FTMIB	HSI	SSEC	AEX
Random Prediction	−0.0167 (± 0.0692)	−0.0020 (± 0.0492)	0.0310 (± 0.0385)	−0.0139 (± 0.0516)	0.1044 (± 0.0491)	0.0454 (± 0.0578)	0.0449 (± 0.0603)	−0.0091 (± 0.0434)	−0.0187 (± 0.0516)
Label t-1	−0.0053 (± 0.0211)	0.0016 (± 0.0143)	0.0886 (± 0.0138)	−0.0340 (± 0.0140)	0.1909 (± 0.0314)	0.0097 (± 0.0141)	0.1773 (± 0.0575)	0.0326 (± 0.0364)	−0.0054 (± 0.0208)
ANN	0.2224 (± 0.0341)	0.3236 (± 0.0253)	0.3838 (± 0.0568)	0.2966 (± 0.0250)	0.3076 (± 0.0216)	0.3014 (± 0.0219)	0.3147 (± 0.0387)	0.2413 (± 0.0313)	0.2710 (± 0.0618)
ANN+Cluster	0.1869 (± 0.0711)	0.2010 (± 0.0716)	0.3984 (± 0.0318)	0.1800 (± 0.0562)	0.2944 (± 0.0392)	0.2871 (± 0.0423)	0.3891 (± 0.0726)	0.2193 (± 0.0378)	0.2490 (± 0.0754)
SVM	0.1605 (± 0.0219)	0.2112 (± 0.0366)	0.2667 (± 0.0490)	0.1977 (± 0.0490)	0.3373 (± 0.0749)	0.1766 (± 0.0217)	0.2415 (± 0.0743)	0.1866 (± 0.0373)	0.1477 (± 0.0339)
SVM+Cluster	0.2314 (± 0.0398)	0.2144 (± 0.0279)	0.2810 (± 0.0606)	0.2245 (± 0.0437)	0.3551 (± 0.0860)	0.1870 (± 0.0264)	0.2643 (± 0.0685)	0.1793 (± 0.0245)	0.1507 (± 0.0308)
RF	0.1318 (± 0.0773)	0.1584 (± 0.0857)	0.3392 (± 0.0529)	0.1196 (± 0.0316)	0.3246 (± 0.0344)	0.2124 (± 0.0248)	0.4125 (± 0.0291)	0.1559 (± 0.0467)	0.1194 (± 0.0361)
RF+Cluster	0.1389 (± 0.0462)	0.1319 (± 0.0427)	0.3414 (± 0.0463)	0.0944 (± 0.0437)	0.3417 (± 0.0854)	0.2158 (± 0.0446)	0.3656 (± 0.0568)	0.1850 (± 0.0456)	0.2115 (± 0.0327)
CNN	0.2202 (± 0.0285)	0.2777 (± 0.0155)	0.4272 (± 0.0164)	0.2711 (± 0.0244)	0.4068 (± 0.0405)	0.3294 (± 0.0207)	0.3906 (± 0.0203)	0.2292 (± 0.0214)	0.2396 (± 0.0145)
CNN+Cluster	0.2828 (± 0.0337)	0.2495 (± 0.0075)	0.4372 (± 0.0243)	0.1652 (± 0.0278)	0.4147 (± 0.0445)	0.2845 (± 0.0240)	0.3814 (± 0.0239)	0.2947 (± 0.0336)	0.2038 (± 0.0106)
RNN	0.3106 (± 0.0327)	0.3411 (± 0.0274)	0.4423 (± 0.0189)	0.3164 (± 0.0279)	0.4794 (± 0.0443)	0.4096 (± 0.0194)	0.4073 (± 0.0212)	0.2960 (± 0.0150)	0.3576 (± 0.0264)
RNN+Cluster	0.3099 (± 0.0256)	0.3362 (± 0.0237)	0.4387 (± 0.0167)	0.2819 (± 0.0181)	0.4770 (± 0.0509)	0.3458 (± 0.0081)	0.4113 (± 0.0287)	0.3089 (± 0.0096)	0.3289 (± 0.0185)
LSTM	0.2820 (± 0.0182)	0.2945 (± 0.0230)	0.4598 (± 0.0134)	0.3102 (± 0.0223)	0.4329 (± 0.0400)	0.3445 (± 0.0343)	0.3912 (± 0.0160)	0.2793 (± 0.0059)	0.2569 (± 0.0555)
LSTM+Cluster	0.2842 (± 0.0173)	0.2666 (± 0.0153)	0.4546 (± 0.0105)	0.2413 (± 0.0322)	0.4322 (± 0.0454)	0.3347 (± 0.0265)	0.3920 (± 0.0217)	0.3090 (± 0.0167)	0.2739 (± 0.0422)
MBCNN	0.2855 (± 0.0356)	0.3087 (± 0.0407)	0.4186 (± 0.0296)	0.2758 (± 0.0345)	0.4409 (± 0.0326)	0.3671 (± 0.0571)	0.4051 (± 0.0552)	0.2729 (± 0.0607)	0.3269 (± 0.0581)
MBCNN-GA	**0.3474** (± 0.0215)	**0.3592** (± 0.0412)	**0.4744** (± 0.0219)	**0.3264** (± 0.0144)	**0.5013** (± 0.0481)	**0.4217** (± 0.0242)	**0.4655** (± 0.0149)	**0.3196** (± 0.0254)	**0.3669** (± 0.0219)

include economic variables, industry specific variables, psychological variables of investors, political variables, and others. Therefore, how to integrate these factors into the forecasting model to improve its performance is left for future research. Second, we plan to improve the computational efficiency of the proposed methods. While the performance and robustness of the proposed model are superior to those of the competing methods, it does take more time to train the model. In terms of the future work, one possible attempt will be to employ the parallel computing and multiple processors for searching the optimal solution.

CRediT authorship contribution statement

Ruize Gao: Methodology, Software, Validation, Investigation, Writing – original draft. **Xin Zhang:** Resources, Data curation, Visualization. **Hongwu Zhang:** Formal analysis, Writing – review & editing. **Quanwu Zhao:** Experimental design for revision, Proofreading. **Yu Wang:** Conceptualization, Validation, Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research is supported by the National Key R&D Program of China (Grant No. 2018YFB1403600), National Natural Science Foundation of China (Grant No. 71471022), and the Fundamental Research Funds for the Central Universities, China (Project No. 2021CD-JSKJC10).

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., ... Zheng, X. (2016). TensorFlow: a system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)* (pp. 265–283).

Abul Basher, S., & Sadorsky, P. (2016). Hedging emerging market stock prices with oil, gold, VIX, and bonds: A comparison between DCC, ADCC and GO-GARCH. *Energy Economics*, *54*, 235–247. http://dx.doi.org/10.1016/j.eneco.2015.11.022.

Bollen, J., Mao, H., & Zeng, X. (2011). Twitter mood predicts the stock market. *Journal of Computational Science*, *2*, 1–8. http://dx.doi.org/10.1016/j.jocs.2010.12.007.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*, 5–32. http://dx.doi.org/10.1023/A:1010933404324.

Cambria, E. (2016). Affective computing and sentiment analysis. *IEEE Intelligent Systems*, *31*, 102–107. http://dx.doi.org/10.1109/MIS.2016.31.

Carta, S., Corriga, A., Ferreira, A., Podda, A. S., & Recupero, D. R. (2021). A multi-layer and multi-ensemble stock trader using deep learning and deep reinforcement learning. *Applied Intelligence*, *51*, 889–905. http://dx.doi.org/10.1007/s10489-020-01839-5.

Chang, Y.-H., & Lee, M.-S. (2017). Incorporating Markov decision process on genetic algorithms to formulate trading strategies for stock markets. *Applied Soft Computing*, *52*, 1143–1153. http://dx.doi.org/10.1016/j.asoc.2016.09.016.

Chang, C.-C., & Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions On Intelligent Systems and Technology*, *2*, 1–27. http://dx.doi.org/10.1145/1961189.1961199.

Chen, A. S., Leung, M. T., & Daouk, H. (2003). Application of neural networks to an emerging financial market: Forecasting and trading the Taiwan stock index. *Computers & Operations Research*, *30*, 901–923. http://dx.doi.org/10.1016/S0305-0548(02)00037-0.

Chen, H., Xiao, K., Sun, J., & Wu, S. (2017). A double-layer neural network framework for high-frequency forecasting. *ACM Transactions On Management Information Systems*, *7*, 1–17. http://dx.doi.org/10.1145/3021380.

Constantinou, E., Georgiades, R., Kazandjian, A., & Kouretas, G. P. (2006). Regime switching and artificial neural network forecasting of the cyprus stock exchange daily returns. *International Journal of Finance & Economics*, *11*, 371–383. http://dx.doi.org/10.1002/ijfe.305.

Deboeck, G. J. (1994). *Trading on the edge: Neural, genetic, and fuzzy systems for chaotic financial markets*. Wiley.

Fernández, A., García, S., del Jesus, M. J., & Herrera, F. (2008). A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets. *Fuzzy Sets and Systems*, *159*, 2378–2398. http://dx.doi.org/10.1016/j.fss.2007.12.023, Theme: Information processing.

Franses, P. H., & Ghijsels, H. (1999). Additive outliers, GARCH and forecasting volatility. *International Journal of Forecasting*, *15*, 1–9. http://dx.doi.org/10.1016/S0169-2070(98)00053-3.

Gaetano, R., Ienco, D., Ose, K., & Cresson, R. (2018). A two-Branch CNN architecture for land cover classification of PAN and MS imagery. *Remote Sensing*, *10*, 1746. http://dx.doi.org/10.3390/rs10111746.

Giese, G., & Kouzmenko, R. (2019). Have regional equity-market correlations risen? https://www.msci.com/www/blog-posts/have-regional-equity-market/01644267058.

Gunduz, H., Yaslan, Y., & Cataltepe, Z. (2017). Intraday prediction of borsa Istanbul using convolutional neural networks and feature correlations. *Knowledge-Based Systems*, *137*, 138–148. http://dx.doi.org/10.1016/j.knosys.2017.09.023.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE conference on computer vision and pattern recognition* (pp. 770–778). http://dx.doi.org/10.1109/CVPR.2016.90.

Hinton, G. E. (1989). Connectionist learning procedures. *Artificial Intelligence*, *40*, 185–234. http://dx.doi.org/10.1016/0004-3702(89)90049-0.

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, *18*, 1527–1554. http://dx.doi.org/10.1162/neco.2006.18.7.1527.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*, 1735–1780. http://dx.doi.org/10.1162/neco.1997.9.8.1735.

Holland, J. H. (1992). *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. MIT Press.

Hoseinzade, E., & Haratizadeh, S. (2019). CNNpred: CNN-based stock market prediction using a diverse set of variables. *Expert Systems with Applications*, *129*, 273–285. http://dx.doi.org/10.1016/j.eswa.2019.03.029.

Hung, J.-C. (2011). Adaptive fuzzy-GARCH model applied to forecasting the volatility of stock markets using particle swarm optimization. *Information Sciences*, *181*, 4673–4683. http://dx.doi.org/10.1016/j.ins.2011.02.027.

Ioffe, S., & Szegedy, C. (2015). Batch normalization: accelerating deep network training by reducing internal covariate shift. arxiv:1502.03167 [cs].

Johnman, M., Vanstone, B. J., & Gepp, A. (2018). Predicting FTSE 100 returns and volatility using sentiment analysis. *Accounting & Finance*, *58*, 253–274. http://dx.doi.org/10.1111/acfi.12373.

Kaboudan, M. A. (2000). Genetic programming prediction of stock prices. *Computational Economics*, *16*, 207–236. http://dx.doi.org/10.1023/A:1008768404046.

Kara, Y., Acar Boyacioglu, M., & Baykan, Ö. K. (2011). Predicting direction of stock price index movement using artificial neural networks and support vector machines: the sample of the Istanbul stock exchange. *Expert Systems with Applications*, *38*, 5311–5319. http://dx.doi.org/10.1016/j.eswa.2010.10.027.

Karaca, Y., Zhang, Y.-D., & Muhammad, K. (2020). Characterizing complexity and self-similarity based on fractal and entropy analyses for stock market forecast modelling. *Expert Systems with Applications*, *144*, Article 113098. http://dx.doi.org/10.1016/j.eswa.2019.113098.

Kazem, A., Sharifi, E., Hussain, F. K., Saberi, M., & Hussain, O. K. (2013). Support vector regression with chaos-based firefly algorithm for stock market price forecasting. *Applied Soft Computing*, *13*, 947–958. http://dx.doi.org/10.1016/j.asoc.2012.09.024.

Kelly, S., & Ahmad, K. (2018). Estimating the impact of domain-specific news sentiment on financial assets. *Knowledge-Based Systems*, *150*, 116–126. http://dx.doi.org/10.1016/j.knosys.2018.03.004.

Kelly, M. A., & Clark, S. P. (2011). Returns in trading versus non-trading hours: the difference is day and night. *Journal of Asset Management*, *12*(2), 132–145. http://dx.doi.org/10.1057/jam.2011.2.

Kia, A. N., Haratizadeh, S., & Shouraki, S. B. (2018). A hybrid supervised semi-supervised graph-based model to predict one-day ahead movement of global stock markets and commodity prices. *Expert Systems with Applications*, *105*, 159–173. http://dx.doi.org/10.1016/j.eswa.2018.03.037.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097–1105).

Kwon, O., & Yang, J.-S. (2008). Information flow between stock indices. *EPL (Europhysics Letters)*, *82*, 68003. http://dx.doi.org/10.1209/0295-5075/82/68003.

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*, 2278–2324. http://dx.doi.org/10.1109/5.726791.

Li, X., Xie, H., Chen, L., Wang, J., & Deng, X. (2014). News impact on stock price return via sentiment analysis. *Knowledge-Based Systems*, *69*, 14–23. http://dx.doi.org/10.1016/j.knosys.2014.04.022.

Liu, J., Si, Y.-W., Zhang, D., & Zhou, L. (2018). Trend following in financial time series with multi-objective optimization. *Applied Soft Computing*, *66*, 149–167. http://dx.doi.org/10.1016/j.asoc.2018.02.014.

Long, J., Chen, Z., He, W., Wu, T., & Ren, J. (2020). An integrated framework of deep learning and knowledge graph for prediction of stock price trend: an application in Chinese stock exchange market. *Applied Soft Computing*, *91*, Article 106205. http://dx.doi.org/10.1016/j.asoc.2020.106205.

Long, W., Lu, Z., & Cui, L. (2019). Deep learning-based feature engineering for stock price movement prediction. *Knowledge-Based Systems*, *164*, 163–173. http://dx.doi.org/10.1016/j.knosys.2018.10.034.

Lou, D., Polk, C., & Skouras, S. (2019). A tug of war: Overnight versus intraday expected returns. *Journal of Financial Economics*, *134*(1), 192–213. http://dx.doi.org/10.1016/j.jfineco.2019.03.011.

Malagrino, L. S., Roman, N. T., & Monteiro, A. M. (2018). Forecasting stock market index daily direction: A Bayesian network approach. *Expert Systems with Applications*, *105*, 11–22. http://dx.doi.org/10.1016/j.eswa.2018.03.039.

Marschinski, R., & Kantz, H. (2002). Analysing the information flow between financial time series. *The European Physical Journal B - Condensed Matter and Complex Systems*, *30*, 275–281. http://dx.doi.org/10.1140/epjb/e2002-00379-2.

Nabipour, M., Nayyeri, P., Jabani, H., Mosavi, A., Salwana, E., & S., S. (2020). Deep learning for stock market prediction. *Entropy*, *22*, 840. http://dx.doi.org/10.3390/e22080840.

Nam, K., & Seong, N. (2019). Financial news-based stock movement prediction using causality analysis of influence in the Korean stock market. *Decision Support Systems*, *117*, 100–112. http://dx.doi.org/10.1016/j.dss.2018.11.004.

Nuij, W., Milea, V., Hogenboom, F., Frasincar, F., & Kaymak, U. (2014). An automated framework for incorporating news into stock trading strategies. *IEEE Transactions On Knowledge and Data Engineering*, *26*, 823–835. http://dx.doi.org/10.1109/TKDE.2013.133.

Oliveira, N., Cortez, P., & Areal, N. (2017). The impact of microblogging data for stock market prediction: using Twitter to predict returns, volatility, trading volume and survey sentiment indices. *Expert Systems with Applications*, *73*, 125–144. http://dx.doi.org/10.1016/j.eswa.2016.12.036.

Orimoloye, L. O., Sung, M.-C., Ma, T., & Johnson, J. E. V. (2020). Comparing the effectiveness of deep feedforward neural networks and shallow architectures for predicting stock price indices. *Expert Systems with Applications*, *139*, Article 112828. http://dx.doi.org/10.1016/j.eswa.2019.112828.

Pan, Y., Xiao, Z., Wang, X., & Yang, D. (2017). A multiple support vector machine approach to stock index forecasting with mixed frequency sampling. *Knowledge-Based Systems*, *122*, 90–102. http://dx.doi.org/10.1016/j.knosys.2017.01.033.

Pang, X., Zhou, Y., Wang, P., Lin, W., & Chang, V. (2020). An innovative neural network approach for stock market prediction. *The Journal of Supercomputing*, *76*, 2098–2118. http://dx.doi.org/10.1007/s11227-017-2228-y.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Qiu, M., & Song, Y. (2016). Predicting the direction of stock market index movement using an optimized artificial neural network model. *PLoS One*, *11*, Article e0155133. http://dx.doi.org/10.1371/journal.pone.0155133.

R.French, K., & Roll, R. (1986). Stock return variances:the arrival of information and the reaction of traders. *Journal of Financial Economics*, *17*, 5–26.

Sculley, D. (2010). Web-scale k-means clustering. In *Proceedings of the 19th international conference on world wide web* (p. 1177). Raleigh, North Carolina, USA: ACM Press, http://dx.doi.org/10.1145/1772690.1772862.

Shynkevich, Y., McGinnity, T., Coleman, S. A., & Belatreche, A. (2016). Forecasting movements of health-care stock prices based on different categories of news articles using multiple kernel learning. *Decision Support Systems*, *85*, 74–83. http://dx.doi.org/10.1016/j.dss.2016.03.001.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929–1958.

Sun, Y., Xue, B., Zhang, M., Yen, G. G., & Lv, J. (2020). Automatically designing CNN architectures using the genetic algorithm for image classification. *IEEE Transactions On Cybernetics*, *50*, 3840–3854. http://dx.doi.org/10.1109/TCYB.2020.2983860.

Tetlock, P. C. (2007). Giving content to investor sentiment: The role of media in the stock market. *The Journal of Finance*, *62*, 1139–1168. http://dx.doi.org/10.1111/j.1540-6261.2007.01232.x.

Wang, J., He, J., Feng, C., Feng, L., & Li, Y. (2021). Stock index prediction and uncertainty analysis using multi-scale nonlinear ensemble paradigm of optimal feature extraction, two-stage deep learning and Gaussian process regression. *Applied Soft Computing*, 107898. http://dx.doi.org/10.1016/j.asoc.2021.107898.

Wang, H., Lu, S., & Zhao, J. (2019). Aggregating multiple types of complex data in stock market prediction: A model-independent framework. *Knowledge-Based Systems*, *164*, 193–204. http://dx.doi.org/10.1016/j.knosys.2018.10.035.

Wang, Y., Zhang, D., Liu, Y., Dai, B., & Lee, L. H. (2019). Enhancing transportation systems via deep learning: A survey. *Transportation Research Part C (Emerging Technologies)*, *99*, 144–163. http://dx.doi.org/10.1016/j.trc.2018.12.004.

Weng, B., Lu, L., Wang, X., Megahed, F. M., & Martinez, W. (2018). Predicting short-term stock prices using ensemble methods and online data sources. *Expert Systems with Applications*, *112*, 258–273. http://dx.doi.org/10.1016/j.eswa.2018.06.016.

Williams, R. J., & Zipser, D. (1989). A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, *1*, 270–280. http://dx.doi.org/10.1162/neco.1989.1.2.270.

Xiao, C., Xia, W., & Jiang, J. (2020). Stock price forecast based on combined model of ARI-MA-LS-SVM. *Neural Computing and Applications*, *32*, 5379–5388. http://dx.doi.org/10.1007/s00521-019-04698-5.

Xing, F. Z., Cambria, E., & Zhang, Y. (2019). Sentiment-aware volatility forecasting. *Knowledge-Based Systems*, *176*, 68–76. http://dx.doi.org/10.1016/j.knosys.2019.03.029.

Yilmazer, R., & Birant, D. (2021). Shelf auditing based on image classification using semi-supervised deep learning to increase on-shelf availability in grocery stores. *Sensors*, *21*, 327. http://dx.doi.org/10.3390/s21020327.

Zhong, X., & Enke, D. (2017). Forecasting daily stock market return using dimensionality reduction. *Expert Systems with Applications*, *67*, 126–139. http://dx.doi.org/10.1016/j.eswa.2016.09.027.