Journal of King Saud University - Computer and Information Sciences 36 (2024) 101959

HOSTED BY

ELSEVIER

Contents lists available at ScienceDirect

Journal of King Saud University - Computer and Information Sciences

journal homepage: www.sciencedirect.com

1D-CapsNet-LSTM: A deep learning-based model for multi-step stock index forecasting

Cheng Zhang *, Nilam Nur Amir Sjarif, Roslina Ibrahim

Advanced Informatics Department, Razak Faculty of Technology and Informatics, Universiti Teknologi Malaysia, 54100 Kuala Lumpur, Malaysia

ARTICLE INFO

Keywords:
1D-CapsNet-LSTM
Deep learning
Time series
Stock index
Multi-step forecasting

ABSTRACT

Multi-step stock index forecasting is vital in finance for informed decision-making. Current forecasting methods for this task frequently produce unsatisfactory results due to the inherent randomness and instability of the data, thereby underscoring the demand for advanced forecasting models. Given the superiority of the capsule network (CapsNet) over CNNs in various forecasting and classification tasks, this study investigates the potential of integrating a 1D CapsNet with an LSTM network for multi-step stock index forecasting. To this end, a hybrid 1D-CapsNet-LSTM model is introduced, which utilizes a 1D CapsNet to generate high-level capsules from sequential data and an LSTM network to capture temporal dependencies. To maintain stochastic dependencies over different forecasting horizons, a multi-input multi-output (MIMO) strategy is employed. The model's performance is evaluated on real-world stock market indices, including S&P 500, DJIA, IXIC, and NYSE, and compared to baseline models, including LSTM, RNN, and CNN-LSTM, using metrics such as RMSE, MAE, MAPE, and TIC. The proposed 1D-CapsNet-LSTM model consistently outperforms the baseline models in two key aspects. It shows notable reductions in forecasting errors when compared to the baseline models. Additionally, it displays a slower rate of error escalation as forecast horizons lengthen, suggesting enhanced robustness for multi-step forecasting tasks.

1. Introduction

Stock market indices serve as vital indicators of financial market health and performance. Accurately forecasting future stock index values is crucial in the financial sector because it helps investors, traders, and financial institutions make well-informed decisions, manage risks, and optimize their investment strategies (Cavalcante et al., 2016; Tang et al., 2022). Rather than a one-step forecasting approach, multi-step forecasting, which predicts the price of a target variable at multiple consecutive time steps in the future, provides decision-makers with valuable insights into future price fluctuations over a specific time horizon (Aryal et al., 2020; Duan & Kashima, 2021; Zhang et al., 2023b).

One effective method for multi-step forecasting of stock indices is the Multiple-Input Multiple-Output (MIMO) strategy. This strategy involves generating several consecutive predicted values in a single step, without incurring high computational costs (Bontempi, 2008; Bontempi & Taieb, 2011). The MIMO strategy retains the stochastic dependencies between predicted values and is generally more effective than single-output approaches (Taieb et al., 2012). It has gained widespread adoption in long short-term memory (LSTM) based forecasting models for multi-step time series forecasting tasks (Deng et al., 2022; Nguyen et al., 2021; Zhang et al., 2023b). This is because LSTM layers are particularly suitable for capturing temporal dependencies within data and are a preferred choice for building time series forecasting models (Durairaj & Mohan, 2019; Hu et al., 2021; Kumar et al., 2021; Lara-Benítez et al., 2021; Nosratabadi et al., 2020; Sezer et al., 2020).

However, multi-step forecasting of stock indices using LSTM networks often yields unsatisfactory results due to the stochastic and volatile nature of the data. The Efficient Market Hypothesis (EMH) posits that asset prices reflect all available information (Fama, 1970), and in an efficient market, price behavior resembles a random walk, making it difficult to discern patterns in historical data (Fama, 1995). Stock market indices share characteristics with most asset price series, leading to a rapid deterioration in forecasting accuracy as the forecasting horizon increases. Ongoing challenges in this area involve improving forecasting accuracy and understanding the reasons behind such improvements (Makridakis et al., 2018).

One approach to enhancing multi-step forecasting accuracy is to

* Corresponding author.
E-mail addresses: zcheng582dx@gmail.com (C. Zhang), nilamnur@utm.my (N.N.A. Sjarif), iroslina.kl@utm.my (R. Ibrahim).

https://doi.org/10.1016/j.jksuci.2024.101959
Received 2 November 2023; Received in revised form 5 January 2024; Accepted 29 January 2024
Available online 13 February 2024

integrate time–frequency decomposition technologies, such as multivariate empirical mode decomposition (MEMD) and multivariate variational mode decomposition (MVMD), into LSTM-based forecasting models. These techniques decompose stock index series into a set of intrinsic mode functions (IMFs), which are subsequently input to the LSTM model for prediction (Deng et al., 2022; Zhang et al., 2023b). However, while decomposition techniques effectively capture time series components such as trends, seasonality, and residuals, they incorporate out-of-sample data into the training data, leading to overly optimistic forecasting accuracy (Hewamalage et al., 2023; Wu et al., 2022; Zhang et al., 2023a; Zhang et al., 2015). Nevertheless, these techniques underscore the importance of sophisticated feature extraction, mitigating the drawbacks of an LSTM-based forecasting model with univariate input (Altan et al., 2021). Proper feature extraction expands the model's input to multivariate data, which can reveal patterns or relationships that are not readily apparent in the raw data, enhancing the learning capability of the forecasting model.

Another way to enhance feature extraction is by employing a neural network capable of automatically extracting features from raw data. Inspired by the effectiveness of convolutional operations in feature extraction for image recognition, Waibel et al. (1989) designed a one-dimensional convolutional neural network (1D CNN) that performed convolutional operations on 1D sequences. The core concept behind 1D CNNs is the use of one-dimensional convolutional filters that slide over the input sequence to capture local patterns or features at different positions (Li et al., 2022). Filters, also known as kernels, are the key components of the convolutional layer and perform convolutions by computing the weighted sums of the neighboring elements. Activation functions, such as rectified linear units (ReLUs), are often applied to introduce nonlinearity. Additional layers such as pooling layers are often incorporated for downsampling. Furthermore, by leveraging the ability of 1D CNNs to extract univariate time series features, a type of hybrid neural architecture, namely, the convolutional neural network long short-term memory (CNN-LSTM) network, was designed (Shi et al., 2017; Tsantekidis et al., 2020; Zhan et al., 2018) and is widely used for forecasting financial time series prices (Aldhyani & Alzahrani, 2022; Livieris et al., 2020b; Lu et al., 2020). In these studies, CNN-LSTMs have demonstrated superior performance over LSTMs in various time series forecasting tasks, indicating the effectiveness of using a feature extraction component, such as a 1D CNN, to enhance the performance of an LSTM-based forecasting model.

Furthermore, given the superior feature extraction ability of capsule networks (CapsNets) over CNNs in various image classification tasks (Choudhary et al., 2023; Pawan & Rajan, 2022), it is reasonable to wonder whether the performance of an LSTM-based model in multi-step time series forecasting can be further enhanced by implementing CapsNet as the feature extraction component. CapsNets implement the concept of capsules within a feature map, adopt a dynamic routing approach to process the capsules of data to overcome the limitations of the elementary pooling methods employed by CNNs (Sabour et al., 2017), and have been combined with LSTM networks for various forecasting tasks, such as transportation network speed forecasting and machinery remaining life estimation (Ma et al., 2021; Qin et al., 2022). However, akin to CNNs used for 2D image processing, direct incorporation of CapsNets into an LSTM-based model for time series forecasting presents challenges. Traditional forms of CapsNets are not inherently designed for sequentially extracting features from 1D sequences. (Butun et al., 2020; Jayasekara et al., 2019; Liang et al., 2022). As a result, the application of CapsNets in financial time series forecasting is currently uncommon. The potential of CapsNets as feature extraction components in deep learning models for multi-step stock index forecasting has yet to be explored.

To investigate this potential, in our study, we propose a novel hybrid model, 1D-CapsNet-LSTM, in which a 1D CapsNet generates high-level features for each data point in the univariate sequence, represented as high-level capsules, and an LSTM layer captures the temporal dependencies between these high-level features. With the MIMO approach, the 1D-CapsNet-LSTM model produces multiple forecasts at different forecasting horizons in a single step. To assess the performance of this model for multi-step forecasting of real-world stock indices, we selected four prominent stock market indices—Standard and Poor's 500 index (S&P 500), Dow Jones Industrial Average (DJIA), Nasdaq Composite Index (IXIC), and New York Stock Exchange (NYSE)—for our experiments. We evaluated the model's performance using four metrics: the root mean squared error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE), and Theil inequality coefficient (TIC). Additionally, we included three baseline models (LSTM, RNN, and CNN-LSTM) for performance comparison.

This study makes two significant contributions to the literature. First, we introduce a novel architecture of 1D CapsNet that can generate high-level capsules for each data point from 1D sequential data. The time-distributed dynamic routing method used in 1D CapsNet at each time step represents a distinct and innovative methodology. Second, to the best of our knowledge, this study is the first to apply a hybrid model that combines a CapsNet with an LSTM network for financial time series forecasting, especially for multi-step forecasting tasks. In summary, this study extends the application of CapsNets in the field of time series forecasting.

The remaining sections of this paper are organized as follows. Section 2 discusses related studies on multi-step forecasting strategies, convolutional-recurrent neural networks, and applications of CapsNets in various domains. In Section 3, we explain the MIMO strategy for multi-step forecasting and provide a detailed explanation of the proposed 1D-CapsNet-LSTM architecture. Section 4 outlines the experimental setup and presents the results of the performance comparison between the proposed model and the baseline models. Finally, Section 5 concludes the paper.

2. Related work

This section introduces studies related to multi-step forecasting strategies, convolutional-recurrent neural networks (CRNNs), and applications of CapsNets in different domains. The insights drawn from these studies imply that the emerging use of CapsNets in time series analysis, particularly in financial time series forecasting, presents an intriguing frontier.

2.1. Multi-step forecasting strategies

There are approximately five main strategies for generating multi-step forecasts, as outlined by Taieb et al. (2012): recursive, direct, DirRec, MIMO, and DIRMO. The first strategy is the recursive or iterative approach, which refers to training a one-step forecasting model and using the previous time step's forecast as its input for the subsequent forecast (Taieb & Hyndman, 2012). The limitation of this strategy is that the forecasting error accumulates rapidly after a few steps. The second is the direct strategy, which generates multiple forecasts using multiple corresponding models (Cheng et al., 2006). The problem with using the direct strategy is that the computational cost is high and the dependencies among consecutive predicted values are broken. The third is the DirRec strategy, which is a combination of the recursive and direct strategies (Taieb & Hyndman, 2012). Under this strategy, the forecast horizon is divided into several groups, after which the first group of forecasts is generated directly using a set of models, and the following groups of forecasts are recursively produced.

The multiple-input multiple-output (MIMO) strategy was introduced to preserve the stochastic dependencies between predicted values (Bontempi, 2008; Bontempi & Taieb, 2011). Unlike in previous approaches, a forecasting model under the MIMO strategy generates the predicted values for all forecasting horizons in a single step. Another strategy, namely, DIRMO, was developed to merge the best aspects of DirRec with those of MIMO (Taieb et al., 2009). DIRMO aims to balance the stochastic dependence between the forecasted values and maintain model flexibility. Although these five forecasting strategies have been described separately in the literature and sometimes with different terminologies, multiple-output approaches are generally more effective than single-output approaches (Taieb et al., 2012).

In the domain of financial time series forecasting, a direct strategy is commonly used to predict a specific value several time steps away from the current time step rather than several values at consecutive future time steps (Lin et al., 2022; Paquet & Soleymani, 2022; Tripathi & Sharma, 2022; Wang & Wang, 2020). In contrast, the MIMO strategy is often employed when multi-step forecasting results are needed (Aryal et al., 2020; Deng et al., 2022; Staffini, 2022).

2.2. Convolutional-recurrent neural networks

Convolutional-recurrent neural networks (CRNNs) are a type of hybrid neural network developed by combining the complementary strengths of CNNs and RNNs to capture spatiotemporal patterns in various tasks, including time series forecasting (Shi et al., 2017; Tsantekidis et al., 2020; Zhan et al., 2018). In a CRNN model for time series forecasting, a 1D CNN extracts spatial features from the input sequence and outputs a sequence of feature vectors, which are subsequently fed into a recurrent layer that captures temporal dependencies and long-term patterns in the feature map. Therefore, the use of a 1D CNN mitigates the drawbacks of a single RNN model with a univariate input. In the domain of forecasting financial time series prices, a 1D CNN can be combined with different types of RNN layers to construct a forecasting model. For example, the combination of a 1D CNN and an LSTM layer, known as CNN-LSTM, has found extensive applications in financial time series forecasting (Aldhyani & Alzahrani, 2022; Livieris et al., 2020a; Livieris et al., 2020b; Lu et al., 2020). In addition, combinations of a 1D CNN and a bidirectional long short-term memory (BiLSTM) layer, known as CNN-BiLSTM (Chen et al., 2021; Wang et al., 2021; Zheng, 2021), and combinations of a 1D CNN and a gated recurrent unit (GRU) layer, known as CNN-GRU (Jaiswal & Singh, 2022; Kang et al., 2022), are also commonly used. Nonetheless, LSTM layers continue to play a predominant role in CRNN models, as LSTM cells incorporate a memory mechanism that enables them to store information for long durations, selectively forget irrelevant information, and update content based on new input. Therefore, the LSTM layer provides the model with the ability to address the "vanishing gradient" problem inherent in simple RNNs (Hochreiter & Schmidhuber, 1997).

2.3. CapsNets and their applications

CapsNets are a type of neural network designed to overcome certain limitations of traditional CNNs in processing hierarchical relationships within data. These neural networks improve the recognition of complex patterns by considering the spatial relationships between features in an image (Sabour et al., 2017). Compared with CNNs, which implement elementary pooling that often assigns the same values to adjacent data points in a region within the feature map, possibly discarding useful information, CapsNets use an implementation of the concept of capsules within the feature map and adopt a more complex approach than the pooling operation to process the capsules of data. CapsNets first perform the same convolutions as CNNs and then create primary capsules from the previous convolutional results. Subsequently, high-level capsules in vector form, rather than scalar form, are generated by routing all the primary capsules. Through the routing operation, the positional information of the primary capsules is "rate coded" in the real-valued components of the high-level capsules. These high-level capsules, which are groups of neurons that represent specific features, can encode information regarding the presence, orientation, and various properties of a feature, allowing the network to better understand the spatial arrangement of objects within an image. CapsNets have been widely applied in various classification tasks, such as image recognition, pose estimation, and understanding complex visual patterns (Afshar et al., 2018; LaLonde & Bagci, 2018; Ragab et al., 2022; Tampubolon et al., 2019; Xiang et al., 2021).

Moreover, the CapsNet and LSTM networks can be combined into a hybrid neural network, where CapsNet often serves as a metaclassifier that classifies the features learned by LSTM. This combination has been adopted for a series of image classification tasks, such as compound fault diagnosis (Ke et al., 2022), emotion recognition (Shahin et al., 2022), and fake news detection (Sridhar et al., 2021). In regression tasks, the CapsNet and LSTM networks can be combined in a manner similar to the structure of CNN-LSTM so that CapsNet is used to sequentially extract high-level capsules from a series of images, and the LSTM layer is then used to generate an estimation of the target variable from these high-level capsules (Ma et al., 2021; Qin et al., 2022). However, when the input is a 1D sequence, CapsNet in the hybrid CapsNet-LSTM neural network cannot sequentially generate high-level capsules corresponding to every data point in the input sequence; therefore, its output is not compatible with the recurrent layer, which limits the application of CapsNet-LSTM networks to time series forecasting tasks.

It is worth noting that the concept of "1D CapsNet" has appeared in a few studies. Jayasekara et al. (2019) proposed TimeCaps, which can generate capsules along the temporal dimension to classify electrocardiogram signal beat categories. Butun et al. (2020) used a 1D version of CapsNet for automated coronary artery disease (CAD) detection. Berman (2019) achieved a 1D application of CapsNets for domain generation algorithm (DGA) detection. Similarly, while these studies jointly suggest that CapsNets have potential in time series analysis, these models are incapable of generating high-level capsules corresponding to every data point in the 1D sequence and are therefore incompatible with the recurrent layer for time series forecasting tasks. Overall, the application of CapsNets to financial time series forecasting is not yet common.

3. Method

This section introduces the key methods and neural architectures adopted in the proposed 1D-CapsNet-LSTM network: the MIMO strategy for generating multi-step forecasts, the 1D CapsNet for feature extraction, and the LSTM network for capturing sequential dependencies within the data. The proposed 1D-CapsNet-LSTM model integrates these components to serve as a forecasting model for multi-step stock index forecasting.

3.1. MIMO strategy

Given a time series y, whose time steps are denoted as $t \in [1, T]$, a multi-step time series forecasting model $f(\cdot)$ under the MIMO strategy estimates several values $[\widehat{y}_{t+1}, \cdots, \widehat{y}_{t+H}]$ at consecutive H time steps into the future in one step, utilizing the historical values (from time step t to $t-d$, with d being the time lags) of the desired time series, which are denoted as $y_{t-d}, \cdots, y_t$ (Bontempi, 2008; Bontempi & Taieb, 2011). This functional relationship is defined as follows:

$$[\widehat{y}_{t+1}, \cdots, \widehat{y}_{t+H}] = f([y_{t-d}, \cdots, y_t]; \boldsymbol{\theta}); t > d; H > 1, \quad (1)$$

where vector $\boldsymbol{\theta}$ denotes the model parameters that are adjusted through model training.

The objective of the MIMO approach is to maintain the stochastic dependencies between the predicted values, which helps to avoid the conditional independence assumption made by the direct approach and prevents the accumulation of errors associated with the recursive approach. By generating a vector of future values in one step instead of returning a scalar value, MIMO ensures that the correlations between future observations are captured during model training and utilized in the forecasting process.

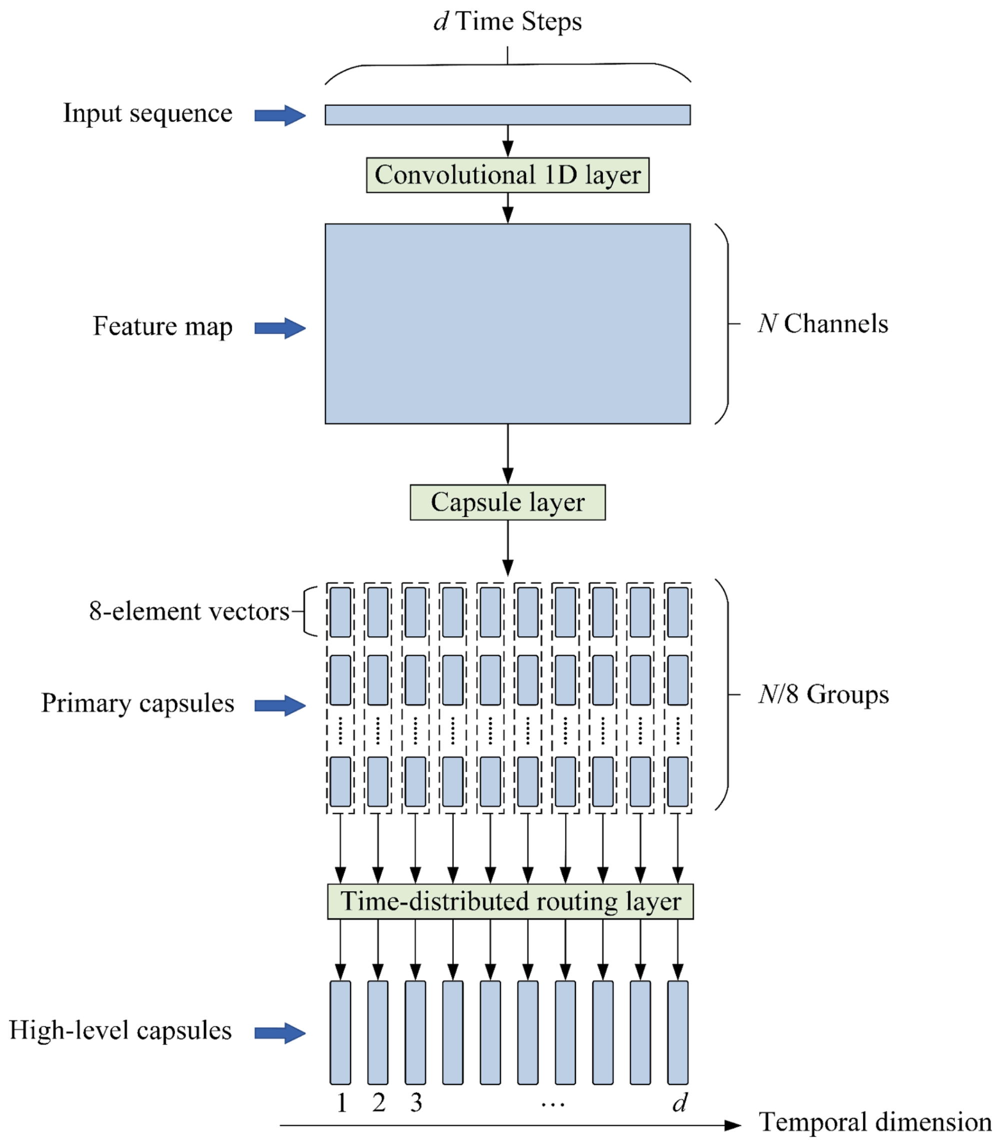

Fig. 1. The 1D CapsNet for 1D sequence processing.

3.2. 1D CapsNet

A CapsNet with a shallow architecture for image classification typically includes convolutional, capsule, and dynamic routing layers (Sabour et al., 2017). During the dynamic routing process, all primary capsules from the capsule layer are routed to generate high-level capsules. For the classification task, the number of high-level capsules is equal to the number of image categories. In the context of time series forecasting using an LSTM-based model, a feature extraction component should be able to extract features with respect to each data point in the input sequence so that the extracted feature map is compatible with the recurrent layer that produces the final prediction. To provide CapsNets with such feature extraction capabilities, we adopted a different approach to process the 1D sequence extracted from a time series. We assume that, for time series forecasting, each data point in the input sequence corresponds to one high-level capsule. Accordingly, the number of high-level capsules is equal to the sequence length. Another assumption is that a high-level capsule should be produced from a corresponding temporal slice of the primary capsules instead of being generated from all primary capsules. Each high-level capsule should be affected mainly by the primary capsules corresponding to the same time step. Therefore, there should be a "one-to-one correspondence" relationship between the original data points from the input sequence and high-level capsules. Consequently, the routing operation must be applied sequentially to each temporal slice of the primary capsules.

Fig. 1 shows a visual representation of the 1D CapsNet for 1D sequence processing. The 1D CapsNet includes a convolutional 1D layer, capsule layer, and time-distributed routing layer. Suppose the convolutional 1D layer has N filters with a size of two, a stride of one, and rectified linear unit (ReLU) activation. Ignoring the batch size, the 1D input sequence through this convolutional layer is converted into a feature map $X \in \mathbb{R}^{d \times N}$, where d is the sequence length. Given the feature map X, the capsule layer constructs the primary capsules through the following steps. Suppose that the primary capsules are 8-element vectors that are oriented along the channel axis, each of which contains the lowest level of features extracted from previous convolutions; then, the feature map X is divided into $n = N/8$ groups along the channel dimension and reshaped into a new tensor $X' \in \mathbb{R}^{d \times n \times 8}$ that contains $d \times n$ primary capsules.

All the primary capsules are then fed to a "squashing" function to make good use of the nonlinearity. The squashing function is given by Equation (2):

$$\boldsymbol{v}_{it} = \frac{\|\boldsymbol{s}_{it}\|^2}{1 + \|\boldsymbol{s}_{it}\|^2}\frac{\boldsymbol{s}_{it}}{\|\boldsymbol{s}_{it}\|}, \tag{2}$$

where $\boldsymbol{s}_{it}$ and $\boldsymbol{v}_{it}$ represent the i th primary capsule from temporal slice t before and after squashing, respectively. Subsequently, the primary capsule $\boldsymbol{v}_{it}$ is transformed into a new vector $\boldsymbol{u}_{it}$ using the transformation matrix $\boldsymbol{W}_{it}$ such that $\boldsymbol{u}_{it}$ has the same shape as the high-level capsule $\boldsymbol{x}_t$, which has a predefined number of elements. The transformation function is given by Equation (3):

$$\boldsymbol{u}_{it} = \boldsymbol{W}_{it} \cdot \boldsymbol{v}_{it} \tag{3}$$

After matrix transformation, all the transformed primary capsules from the temporal slice t, denoted as $\boldsymbol{u}_{it}, i = 1, 2, \cdots, n$, are routed to produce one high-level capsule $\boldsymbol{x}_t$. The overall procedure of the time-distributed dynamic routing is presented in **Algorithm 1**. For each iteration, the initial coupling coefficients $b_{it}, i = 1, 2, \cdots, n$, with initial values of zero are converted into coupling coefficients $c_{it}, i = 1, 2, \cdots, n$, using the softmax function given by Equation (4), such that the sum of the coupling coefficients c_{it} for temporal slice t is equal to one. Subsequently, a high-level capsule $\boldsymbol{x}_t$ is generated through the weighted sum of $\boldsymbol{u}_{it}$ with the coupling coefficients c_{it}. The degree of "agreement," or the dot product between $\boldsymbol{u}_{it}$ and $\boldsymbol{x}_t$, is added to b_{it} to update $\boldsymbol{x}_t$ in the following iteration.

$$c_{it} = \frac{\exp(b_{it})}{\sum_i \exp(b_{it})} \tag{4}$$

Overall, the 1D input sequence with length d, through the 1D CapsNet, is converted to d high-level capsules. Each high-level capsule corresponds to a data point in the input sequence, representing the high-level features extracted from this data point. Subsequently, all high-level capsules are fed to a recurrent layer with d cells, typically an LSTM layer, to generate predictions of the target variable.

3.3. LSTM

LSTM networks hold a significant position in time series forecasting and are often used as key components in forecasting models. An LSTM network consists of a sequence of LSTM cells, which, through a recurrent connection, enables information to flow from the previous steps to the current step, thereby capturing the temporal dependencies between data points at different positions in the input. LSTM cells incorporate a memory mechanism to alleviate the vanishing gradient problem in simple RNNs (Hochreiter & Schmidhuber, 1997). It can store information for long durations, selectively forget irrelevant information, and update content based on a new input.

As shown in Fig. 2, the LSTM cell architecture consists of four key components: a forget gate $\boldsymbol{f}_t$, an update gate $\boldsymbol{u}_t$, a candidate state $\widetilde{\boldsymbol{c}}_t$, and an output gate $\boldsymbol{o}_t$. Three inputs are received at each time step: the current

---

procedure ROUTING $(\mathbf{u}_{it}, d, n, r)$
    $\mathbf{u}_{it} \leftarrow$ The i th transformed primary capsule from temporal slice t
    $d \leftarrow$ The input sequence length
    $n \leftarrow$ The number of primary capsules from each temporal slice
    $r \leftarrow$ The iteration times of dynamic routing
    for $t = 1:d$ **do**
        for $i = 1:n$ **do**
            Initialize $b_{it}: b_{it} \leftarrow 0$
        for r iterations **do**
            for $i = 1:n$ **do**
                Generate $c_{it}: c_{it} \leftarrow \mathrm{softmax}(b_{it})$
            Generate $\mathbf{x}_t : \mathbf{x}_t \leftarrow \sum_i c_{it}\mathbf{u}_{it}$
            for $i = 1:n$ **do**
                Update $b_{it}: b_{it} \leftarrow b_{it} + \mathbf{u}_{it} \cdot \mathbf{x}_t$
        return $\mathbf{x}_t$
end procedure

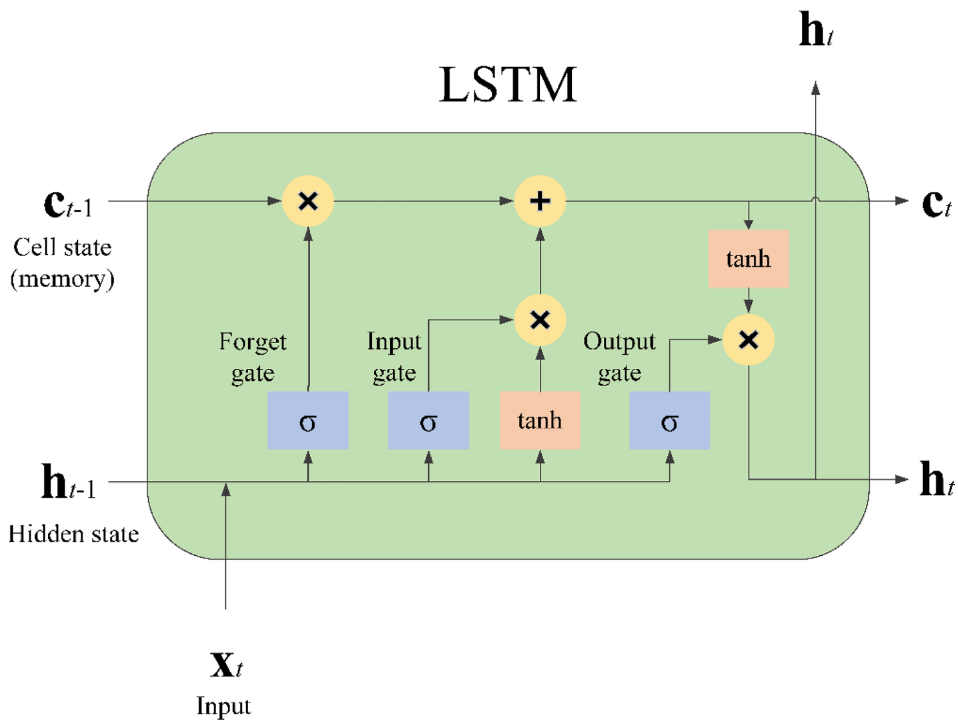

Fig. 2. The architecture of the LSTM cell.

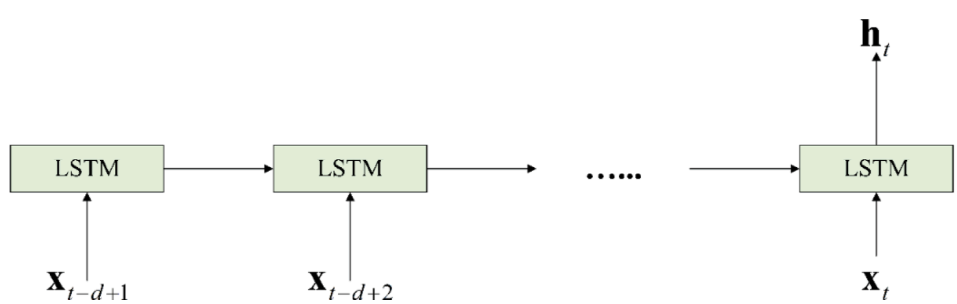

Fig. 3. The "many to one" LSTM layer.

cell input $\boldsymbol{x}_t$ corresponding to the current time step t, the previous cell output $\boldsymbol{h}_{t-1}$, and the previous cell state $\boldsymbol{c}_{t-1}$. These inputs are processed using the internal gates of the LSTM cell, resulting in the corresponding output $\boldsymbol{h}_t$ and cell state $\boldsymbol{c}_t$. The outputs of these gates were calculated using the following equations:

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_f \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_f) \tag{5}$$

$$\boldsymbol{u}_t = \sigma(\boldsymbol{W}_u \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_u) \tag{6}$$

$$\widetilde{\boldsymbol{c}}_t = \tanh(\boldsymbol{W}_c \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_c) \tag{7}$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t * \boldsymbol{c}_{t-1} + \boldsymbol{u}_t * \widetilde{\boldsymbol{c}}_t \tag{8}$$

$$\boldsymbol{o}_t = \tanh(\boldsymbol{W}_o \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_o) \tag{9}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t * \tanh(\boldsymbol{c}_t), \tag{10}$$

where tanh is the hyperbolic tangent function and σ is the sigmoid function. The symbol "·" represents the dot product of the matrices, and "∗" represents elementwise multiplication. The matrices $\boldsymbol{W}_f$ $\boldsymbol{W}_u$ $\boldsymbol{W}_c$ $\boldsymbol{W}_o$ and $\boldsymbol{b}_f$ $\boldsymbol{b}_u$ $\boldsymbol{b}_c$ $\boldsymbol{b}_o$ are the gate weights and bias, respectively.

For multi-step time series forecasting, a "many-to-one" LSTM layer is employed, which uses a sequence of vectors as input and generates one vector output at the last time step. Fig. 3 shows the LSTM layer used to process a sequence of vectors with d time steps. Each vector of the input sequence is a high-level capsule generated from 1D CapsNet. The output of this LSTM layer is usually passed through a dense layer with a suitable activation function to obtain multi-step predicted values.

3.4. 1D-CapsNet-LSTM

The proposed 1D-CapsNet-LSTM model, as depicted in Fig. 4, is a hybrid neural network that leverages the strengths of the multiple components mentioned above to effectively perform a multi-step forecasting task. It comprises two key components, a 1D CapsNet and an LSTM network, which work in concert to process 1D sequential data and predict multi-step future values of the target variable under the MIMO strategy.

Unlike traditional neural networks, which employ scalar neurons to represent learned features, CapsNet utilizes "capsules" to represent features in vector form, allowing for richer and more informative feature representations. This feature extraction approach is particularly beneficial when dealing with complex patterns and structures in sequential data. The LSTM network includes a "many to one" LSTM layer, which is responsible for capturing the temporal dependencies inherent in the high-level capsules generated by 1D CapsNet, and a dense (fully

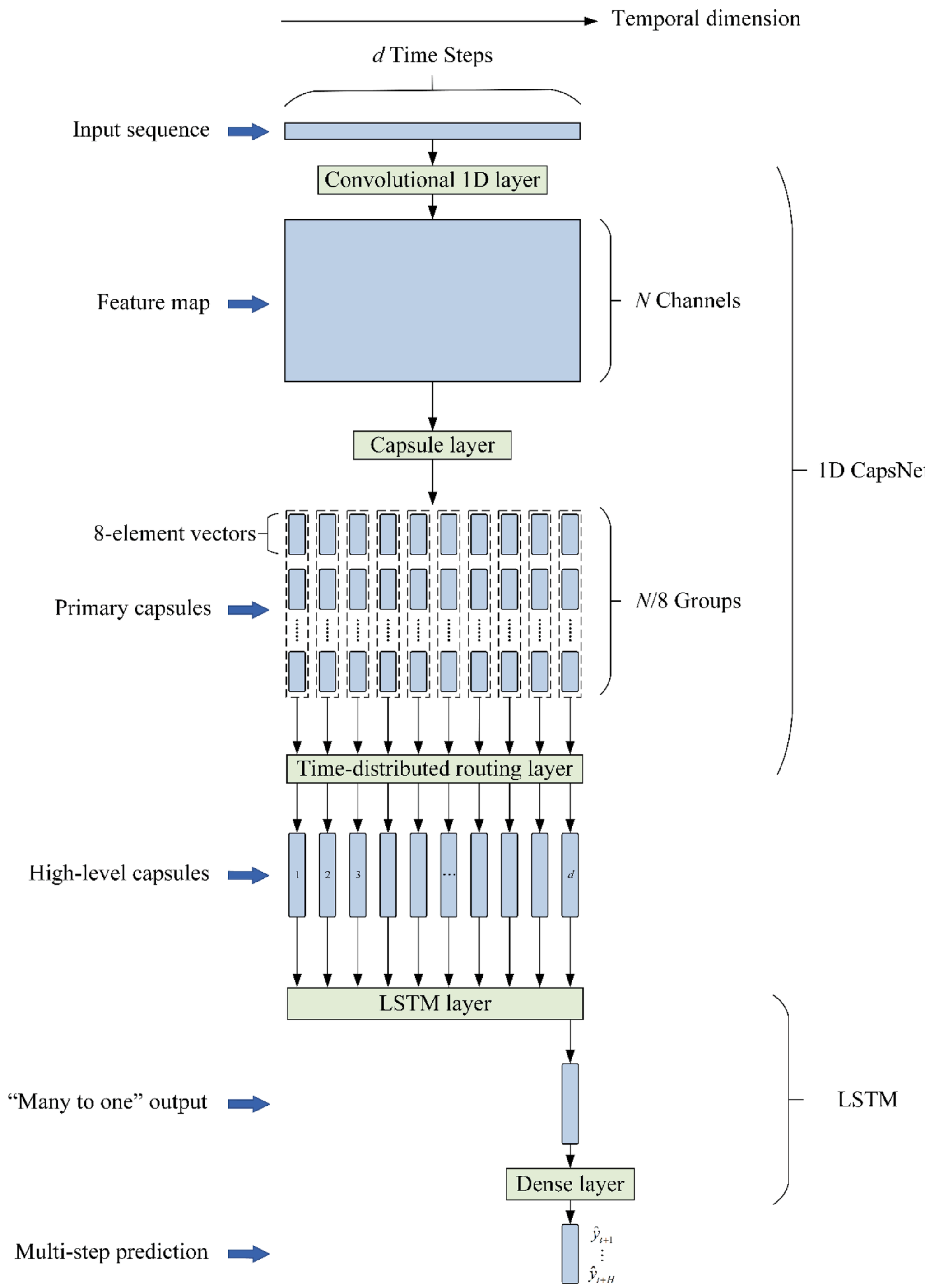

Fig. 4. The 1D-CapsNet-LSTM model for multi-step time series forecasting.

connected) layer, which maps the information extracted by the LSTM layer into a vector that represents the predicted values of the target variable at several consecutive future time steps. Through the 1D CapsNet and LSTM networks, meaningful predictions are obtained based on the learned features and temporal dependencies.

4. Experimental setup and performance comparison

Assessing the performance of the proposed 1D-CapsNet-LSTM model for multi-step forecasting of a stock index involves several steps. First, the raw data were extracted, cleaned, split, and normalized. A sliding window was subsequently used to extract fixed-length segments of the data to construct the input sequences and labels. Second, the proposed and baseline models (LSTM, RNN, and CNN-SLTM) were constructed and subsequently trained with input sequences and labels. Model performance was determined by comparing the predicted and actual values of the test set using a set of performance metrics. Third, the model performances were compared to demonstrate the effectiveness and superiority of the proposed model.

4.1. Data description and preprocessing

Four stock indices, namely, the S&P 500, DJIA, IXIC, and NYSE, were selected for the experiments. The raw data of these indices, covering ten

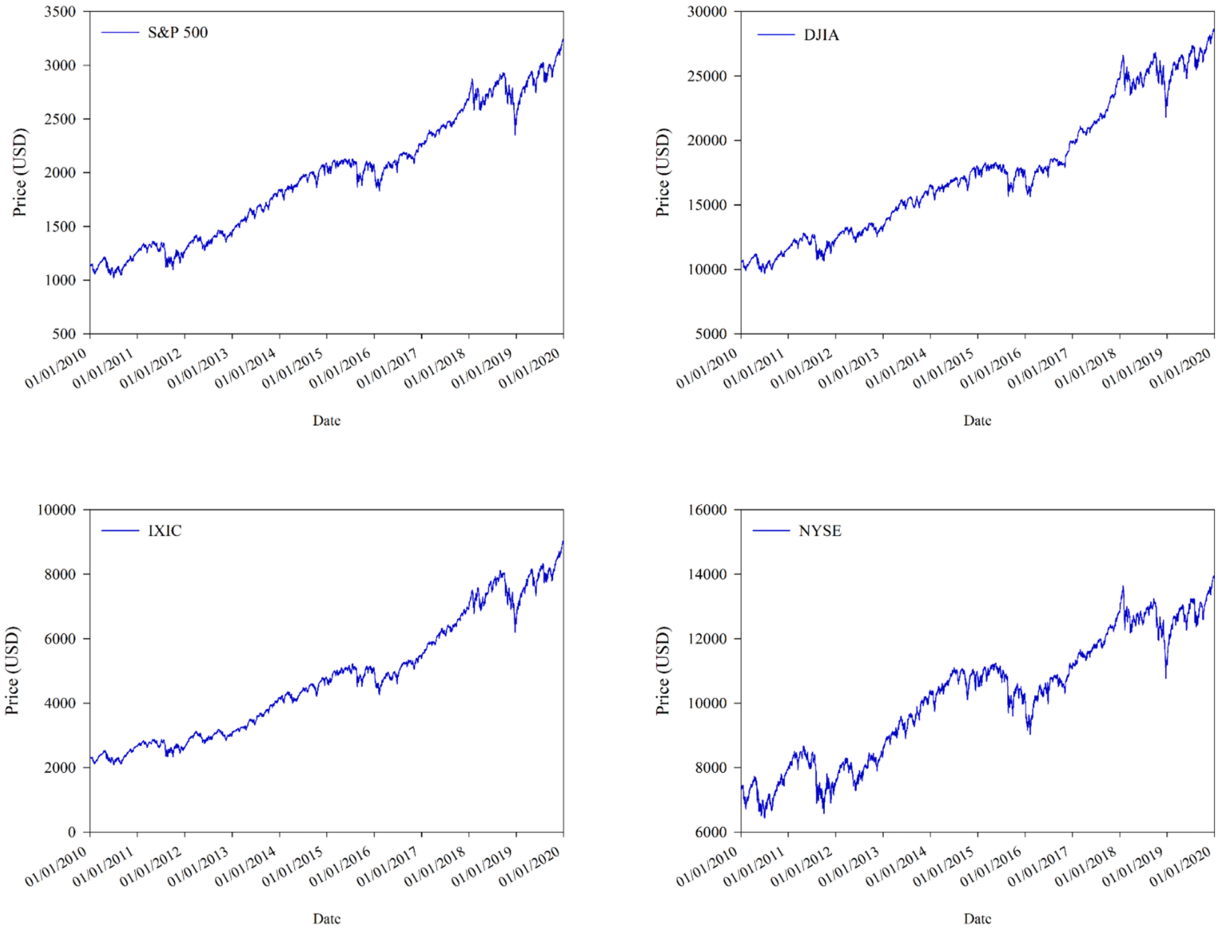

Fig. 5. Close price series of four stock indices.

years of historical daily closing prices from January 1, 2010, to December 31, 2019, were retrieved from the Yahoo Finance website. The four univariate financial time series are visualized in Fig. 5 and are briefly described in Table 1.

For each stock index, the raw data were split into training, validation, and test sets, following an 8:1:1 ratio. This data splitting serves the purposes of training, fine-tuning, and evaluating models, ensuring their ability to generalize to new data while preventing overfitting to the training set. The training set consisted of 2014 observations, whereas the validation and test sets consisted of 251 observations. After data splitting, min—max normalization was performed on the training set to improve the convergence of the deep learning algorithms. The min—max normalization is given by Equation (11):

$$y_t^{normalized} = \frac{y_t - \min(y_t)}{\max(y_t) - \min(y_t)}, \tag{11}$$

where y_t represents the data points in the training set, $y_t^{normalized}$ represents the normalized data points in the training set, $\min(y_t)$ is the minimum value of y_t, and $\max(y_t)$ is the maximum value of y_t. In addition, the values of $\min(y_t)$ and $\max(y_t)$ were used to normalize the validation and test sets.

After data normalization, the entire time series was transformed into short sequences using a sliding window approach in a supervised learning scheme. Fixed-length segments of the data were extracted as the window (time lag) was moved over the entire series. The segments from the training set and corresponding observations of the target variable construct pairs of sequences and labels for model training. The sliding window approach for preparing the input sequences and labels is shown in Fig. 6. In particular, the data segment length, or the input sequence length d, was set to 50 to efficiently process a context with sufficient information while avoiding excessive computational and memory requirements. The following five data points were regarded as the corresponding label of the input sequence, covering the forecasting horizon that would not lead to extremely high prediction error. This operation was conducted sequentially by shifting one time step to the future each time to produce all the input sequences and labels. Labels were not needed for the validation and test sets, and only input sequences were prepared. The predictions provided by the forecasting model were denormalized using the same parameters given by Equation (11) to return the output to the original scale.

4.2. Model configuration

To ensure that the difference in the models' performances is primarily due to the specific part of each neural architecture rather than other sources, such as hyperparameter optimization, common settings were used for the convolutional and LSTM layers in all the models. The hyperparameters that need tuning include the dimensionality of the high-level capsules, the number of iterations of dynamic routing for each temporal slice, and the learning rate, as presented in Table 2. This study adopted the hyperband method for hyperparameter tuning, which can efficiently balance the exploration of different hyperparameter configurations with the allocation of computational resources, allowing for the discovery of optimal hyperparameters more quickly and cost-effectively than traditional grid search or random search methods (Li et al., 2017). During the hyperparameter tuning process, numerous configurations with different hyperparameters were randomly sampled and trained for a small number of epochs. After this initial phase, only the best-performing configurations were selected to proceed to the next round,

Table 1
Statistics of four close price series.

Financial time series	Count	Mean	Standard Deviation	Minimum	Median	Maximum
S&P 500	2516	1962.60886	588.91025	1022.58	1986.48	3240.02
DJIA	2516	17606.74157	5147.05011	9686.48	17008.23	28645.26
IXIC	2516	4744.1578	1878.80328	2091.79004	4620.54517	9022.38965
NYSE	2516	10162.17652	1942.52197	6434.81006	10440.30518	13944.13965

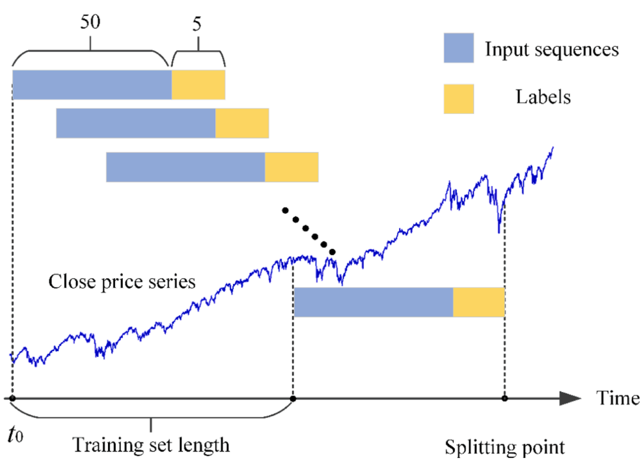

Fig. 6. Sliding windows for preparing the input sequences and labels.

where they were allocated more resources or training epochs. This process continued until only one configuration remained. In addition, the number of training epochs was set to 400 to ensure that each model was properly trained and converged. The learning rate decreased by 5 % if no improvement was observed in model performance after five epochs. The Adam optimizer (Kingma & Ba, 2014) was used to update the model parameters with the mean squared error (MSE) as the cost function. Table 3 provides an overview of the structures of the proposed and baseline models.

The proposed and baseline models were implemented using Python TensorFlow and the high-level API Keras. In addition, a distributed strategy[1] using eight tensor processing units[2] (TPUs) was employed for model training. Under the distributed strategy, each batch of training data comprising 32 samples was divided into eight groups and distributed across eight TPUs for code execution. The gradients produced by each TPU were aggregated to update the model parameters.

4.3. Evaluation metrics

After model training, the performances of the proposed and baseline models were evaluated using four evaluation metrics, namely, the root mean squared error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE), and Theil inequality coefficient (TIC), based on the predictions in the test set. The formulas for these metrics are as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{12}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{13}$$

1 https://www.tensorflow.org/tutorials/distribute/keras
2 https://cloud.google.com/tpu/docs/system-architecture-tpu-vm.

Table 2
Hyperparameter tuning list.

Hyperparameter	Range
Dimension of high-level capsule	(256,512,768,1024)
Iteration times of dynamic routing for each temporal slice	(2,3,4,5)
Learning rate	(0.0001 ~ 0.01)

$$MAPE(\%) = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{14}$$

$$TIC = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}y_i^2} + \sqrt{\frac{1}{n}\sum_{i=1}^{n}\hat{y}_i^2}}, \tag{15}$$

where n denotes the test set size, $\hat{y}_i$ denotes the predicted value, and y_i denotes the actual stock index value. The RMSE places greater emphasis on the highest error and is therefore more sensitive to outliers, whereas the MAE is more robust to outliers. The MAE and MAPE were used to determine the average difference between the predicted and actual values, and the TIC provided insight into how closely the estimated values tracked the actual values over time. In general, smaller values of these metrics indicate more accurate and reliable forecasts (Gilliland, 2010).

Table 3
Structures of the forecasting models.

Forecasting model	Layer	Parameters	Output Shape	Parameter scale
CapsNet-LSTM	InputLayer		(50, 1)	0
	Conv1D	filters = 256kernel_size = 2strides = 1	(50, 256)	768
	Reshape	dimension of primary capsule = 8	(50, 32, 8)	0
	Lambda (Squashing)		(50, 32, 8)	0
	Time-distributed Routing	dimension of high-level capsule = 256iteration times = 3	(50, 256)	65,536
	LSTM	hidden unit = 200	(200)	365,600
	Dense		(5)	1005
LSTM	InputLayer		(50, 1)	0
	LSTM	hidden unit = 200	(200)	161,600
	Dense		(5)	1005
RNN	InputLayer		(50, 1)	0
	SimpleRNN	hidden unit = 200	(200)	40,400
	Dense		(5)	1005
CNN-LSTM	InputLayer		(50, 1)	0
	Conv1D	filters = 256kernel_size = 2strides = 1	(50, 256)	768
	MaxPooling1D	pool_size = 2strides = 1	(50, 256)	0
	LSTM	hidden unit = 200	(200)	365,600
	Dense		(5)	1005

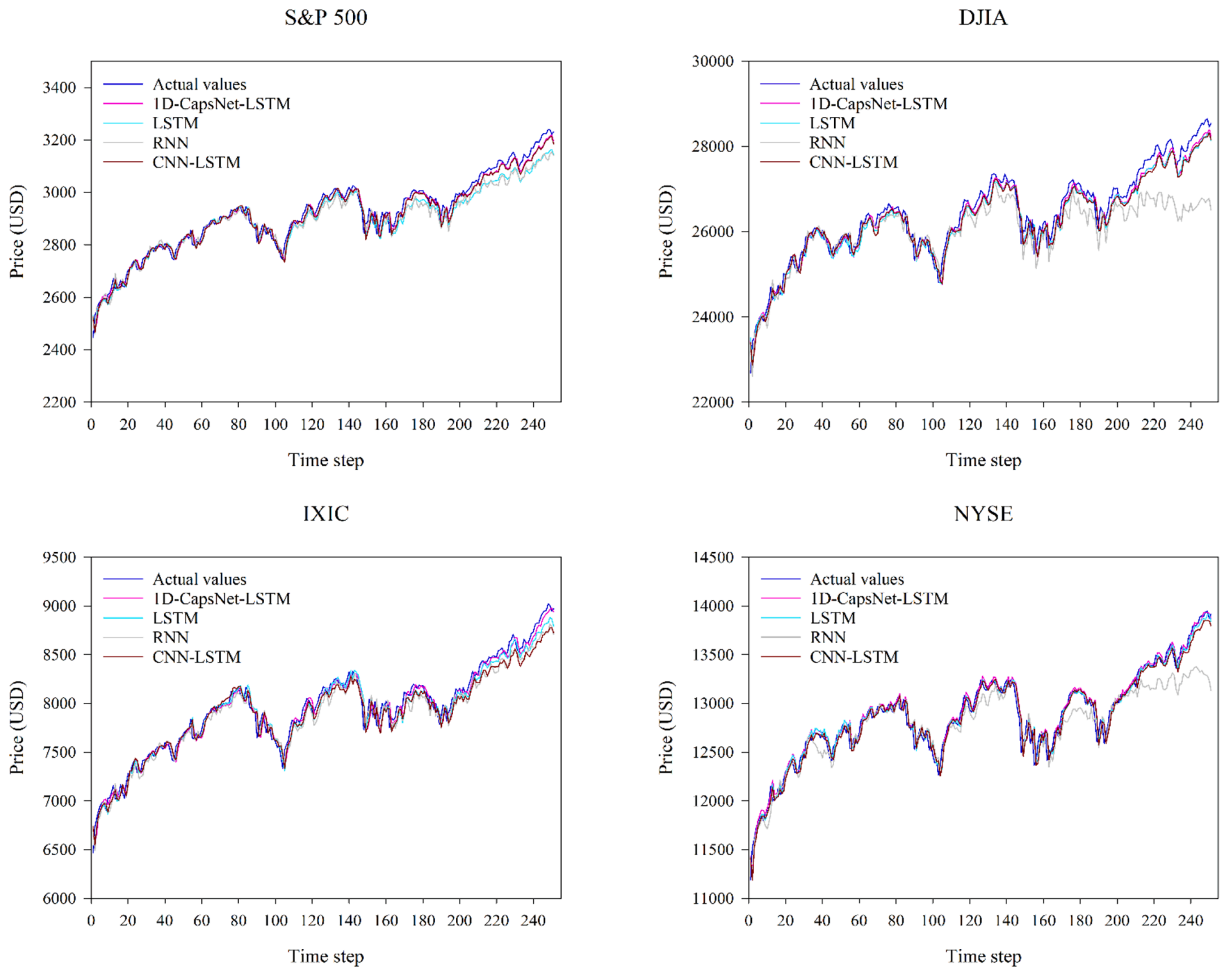

Fig. 7. One-step-ahead forecasting results for four stock indices.

4.4. Performance comparison

This section compares the performances of all the models based on the predictions in the test set in terms of the RMSE, MAE, MAPE, and TIC. Figs. 7, 8, and 9 present the one-step-ahead, three-step-ahead, and five-step-ahead forecasting results for the four stock indices, respectively. In general, as the forecasting horizon extends, forecasting errors accumulate, resulting in a gradual decline in forecasting accuracy. According to our observations, the 1D-CapsNet-LSTM model exhibited better prediction accuracy than did the baseline models across all the forecast horizons. This enhancement can be attributed to the remarkable feature extraction capabilities of 1D CapsNet, as all the models adopted the same settings for the common parts. Therefore, the 1D CapsNet plays a pivotal role in enhancing the performance of the proposed model in multi-step stock index forecasting. It is noteworthy that the RNN model delivers subpar forecasting results for the DJIA and NYSE stock indices. This suboptimal performance can be attributed to its simplistic structure and relatively fewer parameters than those of other models. This simplicity renders the RNN model more susceptible to overfitting, which leads to diminished model performance.

Tables 4, 5, 6, and 7 provide the performance comparisons of all the forecasting models in the five-step forecasting of the S&P 500, DJIA, IXIC, and NYSE, respectively. Table 4 shows the performances of all the models in terms of S&P 500 forecasting. The 1D-CapsNet-LSTM model, which yielded the lowest values of RMSE, MAE, MAPE, and TIC for one-step-ahead, three-step-ahead, four-step-ahead, and five-step-ahead forecasting, respectively, consistently outperformed the other baseline models, demonstrating its superior predictive accuracy. In contrast, the RNN and LSTM models tend to exhibit higher prediction errors, particularly when the forecasting horizon is long. The CNN-LSTM model performed competitively, and its performance fell between that of the 1D-CapsNet-LSTM and LSTM models. It was also observed that for the proposed model, the forecast error was greater over a long horizon than over a short horizon, and the four-step-ahead forecast error was approximately twice that of the one-step-ahead forecast. Similar trends are observed in Tables 5, 6, and 7. The proposed 1D-CapsNet-LSTM model consistently outperformed the other models in terms of all the evaluation metrics, except for a few cases in which LSTM and CNN-LSTM performed better for one or two specific forecast horizons. This finding implies that the 1D-CapsNet-LSTM model achieves superior performance in most cases. Nonetheless, it is worth highlighting that stock market indices are highly volatile and stochastic and that a single type of deep learning model is unlikely to generate accurate predictions across all scenarios.

Fig. 10 presents a visual comparison of the RMSE values for the multi-step forecasting of the four stock indices using different models. Although all the RMSE values increase as the forecast horizon increases, implying a decrease in forecasting accuracy, the proposed 1D-CapsNet-LSTM model still outperforms the baseline models in two aspects. First, substantial decreases in the RMSE values were observed when the proposed and the baseline models were compared. For instance, in five-step S&P 500 forecasting, when comparing the 1D-CapsNet-LSTM model with the LSTM model, the RMSE values decreased by 33.9 %, 15.1 %, 28.8 %, 12.9 %, and 21.7 % for each forecasting horizon. In contrast, when the 1D-CapsNet-LSTM model was compared with the CNN-LSTM model, the RMSE decreased by 5.0 %, −4.1 %, 1.2 %, 11.8 %, and

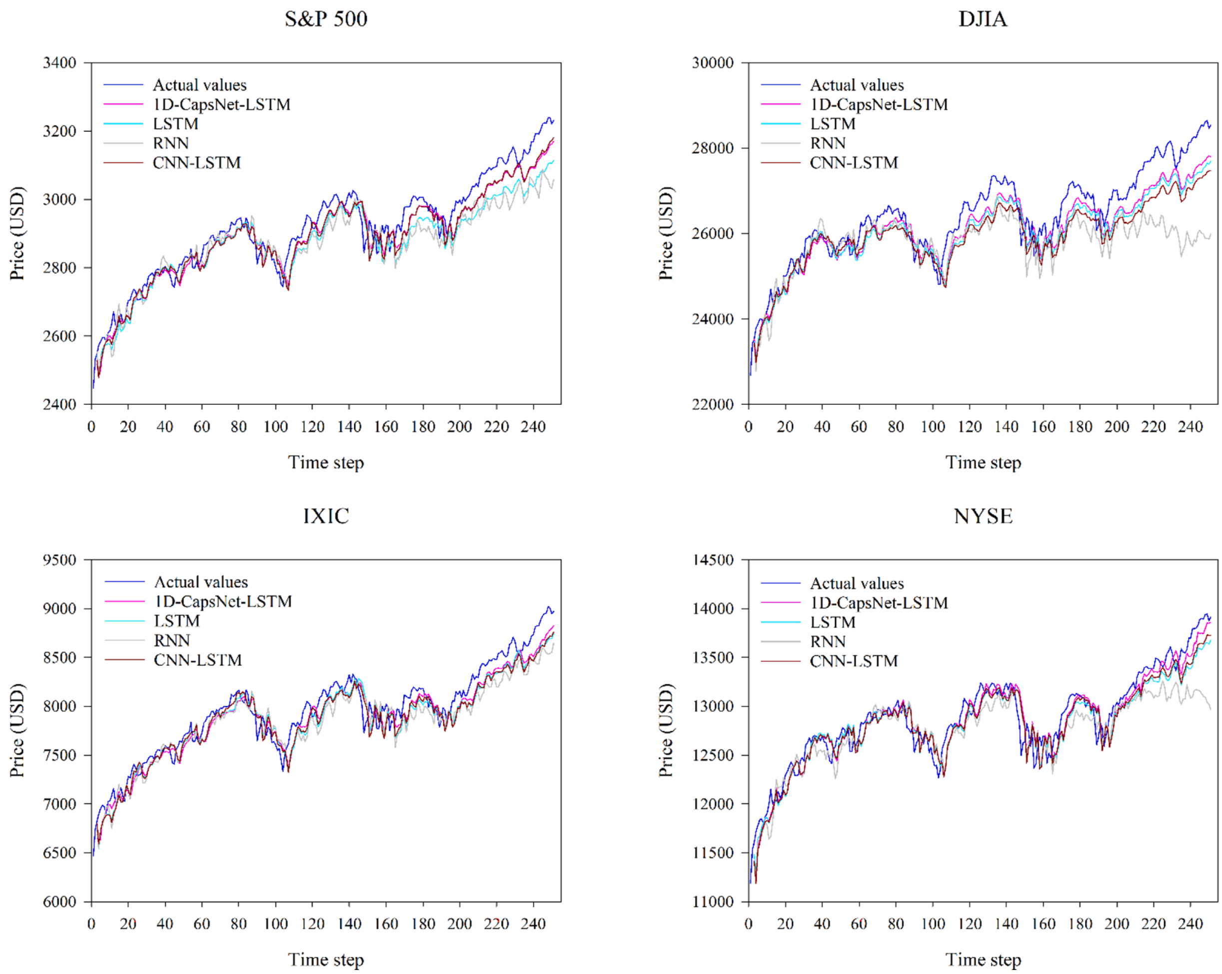

Fig. 8. Three-step-ahead forecasting results for four stock indices.

14.5 %, respectively. Similarly, in five-step DJIA forecasting, when the 1D-CapsNet-LSTM model was compared with the LSTM model, the RMSE decreased by 7.7 %, 9.4 %, 11.6 %, 3.9 %, and 9.0 %, respectively. In contrast, when the 1D-CapsNet-LSTM model was compared with the CNN-LSTM model, the RMSE decreased by 8.5 %, 30.8 %, 24.4 %, 23.2 %, and 31.3 %, respectively. Similar phenomena were observed for the remaining forecasting tasks. Second, as the forecast horizon increased, the RMSE values of the 1D-CapsNet-LSTM model increased at a slower rate than did those of the baseline models. When the forecasting horizon was short, the disparity in the RMSE values between the 1D-CapsNet-LSTM model and baseline models was relatively modest in comparison with the disparity observed when dealing with longer forecasting horizons. This result indicates that the 1D-CapsNet-LSTM model is more robust for multi-step forecasting. A comparison of the other metric values also showed a similar trend.

Based on the experimental results, we made the following key observations. First, the forecast errors tend to increase as the forecasting horizon increases, regardless of the forecasting model. In other words, it is generally more challenging to make accurate predictions for longer time horizons. This is a common phenomenon in financial time series forecasting, suggesting that uncertainty in predictions increases over time. In particular, the four-step-ahead forecast error is approximately twice as large as the one-step-ahead forecast error. This indicates that the error does not increase linearly with the forecasting horizon; instead, it exhibits a certain level of exponential or nonlinear growth. Second, multi-step forecasts are often subject to complex and unpredictable factors, which can increase the difficulty of accurate predictions. Factors such as changing market conditions, economic shifts, and unforeseen events may have greater impacts on multi-step predictions. This observation highlights the importance of selecting appropriate forecasting models and methods, particularly for long forecasting horizons. Models that perform well for single-step predictions may not necessarily perform well for multi-step predictions, and vice versa.

Overall, the proposed 1D-CapsNet-LSTM model outperformed the baseline deep learning models for multi-step stock index forecasting in terms of the RMSE, MAE, MAPE, and TIC. The 1D-CapsNet-LSTM model achieved accurate forecasting results for one-step-ahead forecasting while maintaining stable performance in multi-step-ahead forecasting as the forecasting horizon increased; therefore, this model is a reliable and robust option for similar forecasting tasks.

4.5. Practical considerations

Apart from evaluating the accuracy of the forecasts given by different models, several practical aspects should be considered when implementing the 1D-CapsNet-LSTM model in real-world applications. First, the training speed of the 1D-CapsNet-LSTM model is a crucial factor in determining its suitability for a specific task. The training speed was measured by recording and comparing the time taken by each model to train a batch of training sets. The implementation of a 1D-CapsNet-LSTM model often requires a longer training time compared to implementing a CNN-LSTM model because the nested routing operation in 1D CapsNet is slower than the pooling operation in the CNN (Ma et al., 2021). Nonetheless, the distributed training strategy is effective at reducing the training time of the 1D-CapsNet-LSTM model.

Fig. 11 shows a comparison of the training speeds of the proposed

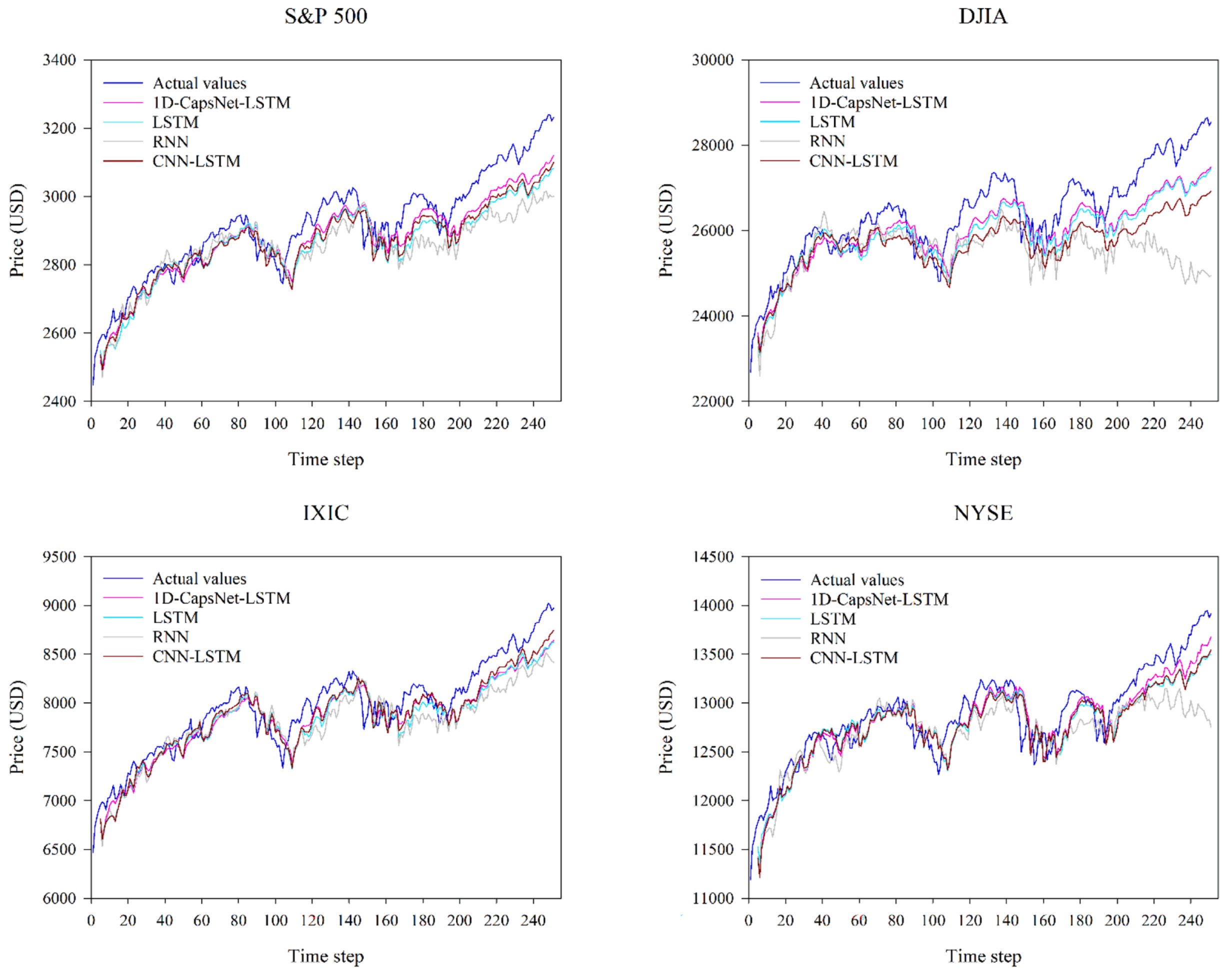

Fig. 9. Five-step-ahead forecasting results for four stock indices.

and baseline models. For one batch of data, the training time of the 1D-CapsNet-LSTM model improved from approximately 1000 ms to 30 ms, which was much closer to the training time of the baseline models. The use of a distributed strategy enhances the ability of the 1D-CapsNet-LSTM model to effectively handle large-scale prediction tasks. Notably, in this study, the training of the 1D-CapsNet-LSTM model was accelerated with the aid of the parallel processing power offered by the TPUs. In addition, 1D-CapsNet-LSTM models can be deployed on field-programmable gate arrays (FPGAs), which can speed up model training through massive parallel computations while consuming less energy than do graphics processing units (GPUs) or TPUs.

Furthermore, although deep learning models have the potential to enhance traders' understanding of market behavior by analyzing vast amounts of data, identifying complex patterns, and providing a competitive edge over traditional methods, integrating these models into real-time trading systems remains challenging because of factors such as model complexity, the need for sufficient training data, and the need to adapt to changing market conditions. Additional research and development are necessary to effectively incorporate the 1D-CapsNet-LSTM model into high-frequency trading systems.

5. Conclusion

Accurately predicting the multi-step future prices of a stock market index is crucial for profitable trading, risk management, and informed investment decision making. However, forecasting results are often unsatisfactory owing to the stochastic and volatile nature of the data. Researchers have made various attempts, and this process is ongoing.

Table 4
Performance comparison of different models for S&P 500 forecasting.

Forecasting horizon	Models	RMSE	MAE	MAPE	TIC
1-step ahead	1D-CapsNet-LSTM	**25.2**	**19.61**	**0.68**	**0.00432**
	RNN	40.39	32.09	1.08	0.00695
	LSTM	38.13	30.42	1.03	0.00656
	CNN-LSTM	26.52	20.67	0.71	0.00455
2-step ahead	1D-CapsNet-LSTM	36.6	30.68	1.04	0.00629
	RNN	59.44	47.82	1.61	0.01024
	LSTM	43.14	36.41	1.23	0.00742
	CNN-LSTM	**35.17**	**29.1**	**0.99**	**0.00604**
3-step ahead	1D-CapsNet-LSTM	**45.74**	**38.77**	**1.32**	**0.00786**
	RNN	74.18	59.49	2	0.0128
	LSTM	64.15	52.91	1.78	0.01106
	CNN-LSTM	46.31	38.83	1.32	0.00796
4-step ahead	1D-CapsNet-LSTM	**55.05**	**46.61**	**1.58**	**0.00947**
	RNN	96.69	76.75	2.56	0.01673
	LSTM	63.18	54.03	1.83	0.01089
	CNN-LSTM	62.41	52.69	1.78	0.01076
5-step ahead	1D-CapsNet-LSTM	**64.11**	**54.31**	**1.83**	**0.01104**
	RNN	103.25	84.36	2.82	0.01787
	LSTM	81.85	69.29	2.33	0.01414
	CNN-LSTM	74.95	63.27	2.13	0.01294

Table 5
Performance comparison of different models for DJIA forecasting.

Forecasting horizon	Models	RMSE	MAE	MAPE	TIC
1-step ahead	1D-CapsNet-LSTM	**233.07**	**186.4**	**0.71**	**0.00442**
	RNN	636.5	446.09	1.65	0.01214
	LSTM	252.5	207.25	0.78	0.00479
	CNN-LSTM	254.62	208.06	0.79	0.00483
2-step ahead	1D-CapsNet-LSTM	**306.97**	**256.52**	**0.97**	**0.00582**
	RNN	796.88	583.24	2.16	0.01522
	LSTM	338.98	292.92	1.1	0.00644
	CNN-LSTM	443.43	374.45	1.4	0.00844
3-step ahead	1D-CapsNet-LSTM	**461.87**	**392.13**	**1.47**	**0.00879**
	RNN	963.34	715.47	2.64	0.01844
	LSTM	522.32	444.12	1.66	0.00995
	CNN-LSTM	611.06	512.39	1.91	0.01166
4-step ahead	1D-CapsNet-LSTM	**582.55**	**493.69**	**1.84**	**0.0111**
	RNN	1183.11	871.35	3.21	0.02271
	LSTM	606.5	520.32	1.94	0.01156
	CNN-LSTM	759.05	639.93	2.38	0.01451
5-step ahead	1D-CapsNet-LSTM	**617.14**	**524.68**	**1.96**	**0.01176**
	RNN	1343.31	1019.67	3.77	0.02585
	LSTM	678.45	581.09	2.17	0.01295
	CNN-LSTM	898.95	760.34	2.82	0.01722

Table 6
Performance comparison of different models for IXIC forecasting.

Forecasting horizon	Models	RMSE	MAE	MAPE	TIC
1-step ahead	1D-CapsNet-LSTM	**80.89**	**61.72**	**0.79**	**0.00509**
	RNN	113.67	92.54	1.16	0.00717
	LSTM	90.23	71.58	0.9	0.00568
	CNN-LSTM	107.75	86.02	1.07	0.00679
2-step ahead	1D-CapsNet-LSTM	**112.36**	**92.01**	**1.16**	**0.00707**
	RNN	168.47	140.34	1.74	0.01064
	LSTM	117.42	96.25	1.21	0.00739
	CNN-LSTM	144.39	117.64	1.46	0.00911
3-step ahead	1D-CapsNet-LSTM	**140.99**	**120.01**	**1.5**	**0.00889**
	RNN	195.03	163.73	2.03	0.01233
	LSTM	160.01	134.17	1.67	0.0101
	CNN-LSTM	155.74	129.76	1.62	0.00983
4-step ahead	1D-CapsNet-LSTM	**168.32**	**141.57**	**1.76**	**0.01062**
	RNN	228.19	191.61	2.37	0.01444
	LSTM	185.4	155.69	1.94	0.01171
	CNN-LSTM	182.07	152.5	1.89	0.0115
5-step ahead	1D-CapsNet-LSTM	194.44	163.1	2.02	0.01228
	RNN	261.52	221.36	2.74	0.01656
	LSTM	213.25	180.02	2.24	0.01348
	CNN-LSTM	**183.85**	**155.51**	**1.94**	**0.0116**

Table 7
Performance comparison of different models for NYSE forecasting.

Forecasting horizon	Models	RMSE	MAE	MAPE	TIC
1-step ahead	1D-CapsNet-LSTM	91.37	66.41	0.52	0.00354
	RNN	188.79	135.2	1.03	0.00736
	LSTM	**90.09**	**66.03**	**0.52**	**0.0035**
	CNN-LSTM	93.34	70	0.55	0.00363
2-step ahead	1D-CapsNet-LSTM	**122.63**	**94.61**	**0.74**	**0.00476**
	RNN	230.59	175.37	1.34	0.00899
	LSTM	131.95	106.33	0.83	0.00513
	CNN-LSTM	132.75	106.56	0.83	0.00516
3-step ahead	1D-CapsNet-LSTM	**148.2**	**117.02**	**0.91**	**0.00576**
	RNN	260.7	197.17	1.51	0.01017
	LSTM	167.18	136.84	1.06	0.0065
	CNN-LSTM	162.07	131.52	1.02	0.0063
4-step ahead	1D-CapsNet-LSTM	**179.61**	**146.25**	**1.14**	**0.00698**
	RNN	310.07	232.95	1.78	0.0121
	LSTM	203.61	165.93	1.28	0.00792
	CNN-LSTM	197.17	160.46	1.24	0.00767
5-step ahead	1D-CapsNet-LSTM	**203.55**	**169.35**	**1.31**	**0.00792**
	RNN	337.03	257.51	1.97	0.01316
	LSTM	228.88	190.1	1.47	0.00891
	CNN-LSTM	225.41	188.11	1.45	0.00878

Inspired by CNN-LSTM networks, which employ a 1D CNN as a sophisticated feature extraction component to improve model performance, this study aims to investigate the potential of CapsNet as a more advanced feature extraction component in the LSTM-based forecasting model to improve the multi-step forecasting result. In this study, we propose a hybrid deep learning model, 1D-CapsNet-LSTM, that integrates a 1D CapsNet to extract high-level capsules from a 1D sequence and an LSTM layer to capture the temporal dependencies between these capsules.

Under the MIMO strategy, the performance of the proposed 1D-CapsNet-LSTM model was evaluated based on five-step forecasting of four real-world stock market indices, the S&P 500, DJIA, IXIC, and NYSE, using four evaluation metrics, RMSE, MAE, MAPE, and TIC. A performance comparison of the proposed model and baseline models, including LSTM, RNN, and CNN-LSTM, revealed that the proposed model achieved accurate results for one-step-ahead forecasting and exhibited the most stable performance in multi-step-ahead forecasting for different forecast horizons. These findings suggested that the 1D-CapsNet-LSTM model is a more reliable and robust option for multistep-ahead forecasting tasks than other deep learning models.

Although the 1D-CapsNet-LSTM network has a complex architecture, it still shows great promise for large-scale and complex prediction tasks because its training time can be significantly reduced through a distributed training strategy using eight TPUs. In addition, with the advancement of computing hardware and software technologies, the 1D-CapsNet-LSTM architecture can be deployed on FPGAs, which are energy-saving and enable efficient model training through massive parallel computations.

Furthermore, it is important to note that the model evaluation was based solely on stock index data; therefore, additional studies are needed to determine the effectiveness of the 1D-CapsNet-LSTM network in other domains. Meanwhile, the influence of data volume on model performance should be investigated in the context of price forecasting, as training with suitably sized datasets can potentially reduce the computational costs while obtaining optimal model performance. Finally, the proper hyperparameter setting of the 1D-CapsNet-LSTM model is worthy of further investigation. The 1D CapsNet designed in this study is conceptualized based on the assumption that there is a “one-to-one correspondence” relationship between the original data points from the input sequence and high-level capsules. This assumption leaves space for future work to explore the optimal setting of this relationship. Whether there should be a “many-to-one” relationship and how we should select the corresponding temporal slices for generating one high-level capsule are challenging research questions that should be answered in the future.

CRediT authorship contribution statement

Cheng Zhang: Conceptualization, Data curation, Methodology, Resources, Software, Visualization, Writing – original draft, Writing – review & editing. **Nilam Nur Amir Sjarif:** Validation, Writing – review &

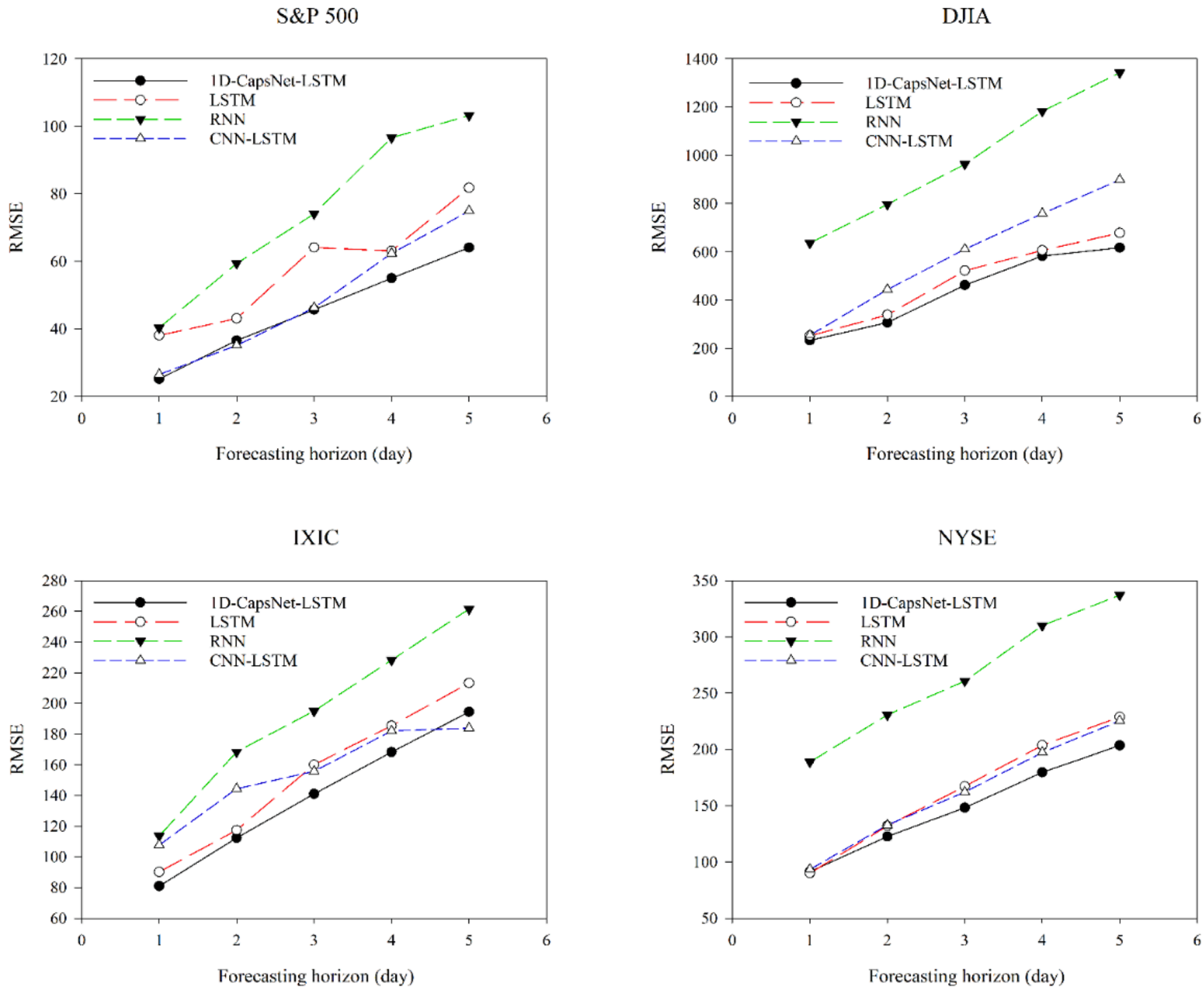

Fig. 10. Comparison of the RMSEs for five-step forecasting.

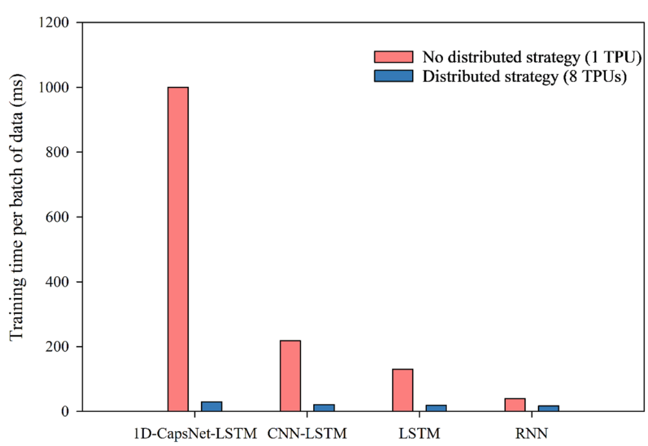

Fig. 11. Training speed comparison.

editing, Supervision. **Roslina Ibrahim:** Writing – review & editing, Supervision.

Acknowledgment

This study did not receive any specific grants from funding agencies in the public, commercial, or not-for-profit sectors.

References

Afshar, P., Mohammadi, A., Plataniotis, K.N., 2018. Brain Tumor Type Classification via Capsule Networks. Paper presented at the 2018 25th IEEE International Conference on Image Processing (ICIP), Athens, Greece.

Aldhyani, T.H.H., Alzahrani, A., 2022. Framework for predicting and modeling stock market prices based on deep learning algorithms. Electronics 11 (19). https://doi.org/10.3390/electronics11193149.

Altan, A., Karasu, S., Zio, E., 2021. A new hybrid model for wind speed forecasting combining long short-term memory neural network, decomposition methods and grey wolf optimizer. Appl. Soft Comput. 100, 106996 https://doi.org/10.1016/j.asoc.2020.106996.

Aryal, S., Nadarajah, D., Rupasinghe, P.L., Jayawardena, C., Kasthurirathna, D., 2020. Comparative analysis of deep learning models for multi-step prediction of financial time series. J. Comput. Sci. 16 (10) https://doi.org/10.3844/jcssp.2020.1401.1416.

Berman, D.S., 2019. DGA CapsNet: 1D application of capsule networks to DGA detection. Information 10 (5). https://doi.org/10.3390/info10050157.

Bontempi, G., 2008. Long term time series prediction with multi-input multi-output local learning. Proc. 2nd ESTSP 145–154.

Bontempi, G., Taieb, S.B., 2011. Conditionally dependent strategies for multiple-step-ahead prediction in local learning. Int. J. Forecast. 27 (3), 689–699.

Butun, E., Yildirim, O., Talo, M., Tan, R.S., Acharya, U.R., 2020. 1D-CADCapsNet: One dimensional deep capsule networks for coronary artery disease detection using ECG signals. Physica Medica-Eur. J. Med. Phys. 70, 39–48. https://doi.org/10.1016/j.ejmp.2020.01.007.

Cavalcante, R.C., Brasileiro, R.C., Souza, V.L.F., Nobrega, J.P., Oliveira, A.L.I., 2016. Computational intelligence and financial markets: a survey and future directions. Expert Syst. Appl. 55, 194–211. https://doi.org/10.1016/j.eswa.2016.02.006.

Chen, Y., Fang, R., Liang, T., Sha, Z., Li, S., Yi, Y., Song, H., 2021. Stock price forecast based on CNN-BiLSTM-ECA model. Sci. Program. 2021 https://doi.org/10.1155/2021/2446543.

Cheng, H., Tan, P.-N., Gao, J., Scripps, J., 2006. Multistep-ahead time series prediction. Paper Presented at the Advances in Knowledge Discovery and Data Mining: 10th Pacific-Asia Conference, PAKDD 2006.

Choudhary, S., Saurav, S., Saini, R., Singh, S., 2023. Capsule networks for computer vision applications: a comprehensive review. Appl. Intell. https://doi.org/10.1007/s10489-023-04620-6.
Deng, C., Huang, Y., Hasan, N., Bao, Y., 2022. Multi-step-ahead stock price index forecasting using long short-term memory model with multivariate empirical mode decomposition. Inf. Sci. 607, 297–321. https://doi.org/10.1016/j.ins.2022.05.088.
Duan, J.D., Kashima, H., 2021. Learning to rank for multi-step ahead time-series forecasting. IEEE Access 9, 49372–49386. https://doi.org/10.1109/access.2021.3068895.
Durairaj, M., Mohan, B.K., 2019. A review of two decades of deep learning hybrids for financial time series prediction. Int. J. Emerg. Technol. 10 (3), 324–331.
Fama, E.F., 1970. Efficient capital markets: a review of theory and empirical work. J. Financ. 25 (2), 383–417.
Fama, E.F., 1995. Random walks in stock market prices. Financ. Anal. J. 51 (1), 75–80.
Gilliland, M., 2010. The business forecasting deal: Exposing myths, eliminating bad practices, providing practical solutions. John Wiley & Sons (Chapter, p. 9)..
Hewamalage, H., Ackermann, K., Bergmeir, C., 2023. Forecast evaluation for data scientists: common pitfalls and best practices. Data Min. Knowl. Disc. 37 (2), 788–832. https://doi.org/10.1007/s10618-022-00894-5.
Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780.
Hu, Z., Zhao, Y., Khushi, M., 2021. A survey of forex and stock price prediction using deep learning. Appl. Syst. Innovat. 4 (1), 9.
Jaiswal, R., Singh, B., 2022. A Hybrid Convolutional Recurrent (CNN-GRU) Model for Stock Price Prediction. Paper presented at the Proceedings - 2022 IEEE 11th International Conference on Communication Systems and Network Technologies, CSNT 2022.
Jayasekara, H., Jayasundara, V., Athif, M., Rajasegaran, J., Jayasekara, S., Seneviratne, S., Rodrigo, R., 2019. Timecaps: Capturing time series data with capsule networks. arXiv preprint arXiv:1911.11800.
Kang, C.Y., Lee, C.P., Lim, K.M., 2022. Cryptocurrency price prediction with convolutional neural network and stacked gated recurrent unit. Data 7 (11). https://doi.org/10.3390/data7110149.
Ke, L.Z., Liu, Y., Yang, Y., 2022. Compound fault diagnosis method of modular multilevel converter based on improved capsule network. IEEE Access 10, 41201–41214. https://doi.org/10.1109/access.2022.3166948.
Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
Kumar, R., Kumar, P., Kumar, Y., 2021. Analysis of financial time series forecasting using deep learning model. Paper Presented at the 2021 11th International Conference on Cloud Computing, Data Science & Engineering (confluence).
LaLonde, R., Bagci, U., 2018. Capsules for object segmentation. arXiv preprint. doi: 10.48550/arXiv.1804.04241.
Lara-Benítez, P., Carranza-García, M., Riquelme, J.C., 2021. An experimental review on deep learning architectures for time series forecasting. Int. J. Neural Syst. 31 (03), 2130001.
Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., Talwalkar, A., 2017. Hyperband: a novel bandit-based approach to hyperparameter optimization. J. Mach. Learn. Res. 18 (1), 6765–6816.
Li, Z., Liu, F., Yang, W., Peng, S., Zhou, J., 2022. A survey of convolutional neural networks: analysis, applications, and prospects. IEEE Trans. Neural Networks Learn. Syst. 33 (12), 6999–7019. https://doi.org/10.1109/TNNLS.2021.3084827.
Liang, T., Chai, C., Sun, H., Tan, J., 2022. Wind speed prediction based on multi-variable Capsnet-BILSTM-MOHHO for WPCCC. Energy 250, 123761. https://doi.org/10.1016/j.energy.2022.123761.
Lin, Y., Liao, Q.D., Lin, Z.X., Tan, B., Yu, Y.Y., 2022. A novel hybrid model integrating modified ensemble empirical mode decomposition and LSTM neural network for multi-step precious metal prices prediction. Resour. Pol. 78 https://doi.org/10.1016/j.resourpol.2022.102884.
Livieris, I.E., Pintelas, E., Kiriakidou, N., Stavroyiannis, S., 2020a. An Advanced Deep Learning Model for Short-Term Forecasting U.S. Natural Gas Price and Movement. In: In*: Vol. 585 IFIP. IFIP Advances in Information and Communication Technology*, pp. 165–176.
Livieris, I.E., Pintelas, E., Pintelas, P., 2020b. A CNN–LSTM model for gold price time-series forecasting. Neural Comput. Appl. 32 (23), 17351–17360. https://doi.org/10.1007/s00521-020-04867-x.
Lu, W.J., Li, J.Z., Li, Y.F., Sun, A.J., Wang, J.Y., 2020. A CNN-LSTM-based model to forecast stock prices. Complexity 2020. https://doi.org/10.1155/2020/6622927.
Ma, X.L., Zhong, H.Y., Li, Y., Ma, J.Y., Cui, Z.Y., Wang, Y.H., 2021. Forecasting transportation network speed using deep capsule networks with nested LSTM models. IEEE Trans. Intell. Transp. Syst. 22 (8), 4813–4824. https://doi.org/10.1109/tits.2020.2984813.
Makridakis, S., Spiliotis, E., Assimakopoulos, V., 2018. The M4 Competition: results, findings, conclusion and way forward. Int. J. Forecast. 34 (4), 802–808. https://doi.org/10.1016/j.ijforecast.2018.06.001.
Nguyen, H.-P., Baraldi, P., Zio, E., 2021. Ensemble empirical mode decomposition and long short-term memory neural network for multi-step predictions of time series signals in nuclear power plants. Appl. Energy 283, 116346. https://doi.org/10.1016/j.apenergy.2020.116346.
Nosratabadi, S., Mosavi, A., Duan, P., Ghamisi, P., Filip, F., Band, S.S., Gandomi, A.H., 2020. Data science in economics: comprehensive review of advanced machine learning and deep learning methods. Mathematics 8 (10), 25. https://doi.org/10.3390/math8101799.
Paquet, E., Soleymani, F., 2022. QuantumLeap: hybrid quantum neural network for financial predictions. Expert Syst. Appl. 195 https://doi.org/10.1016/j.eswa.2022.116583.
Pawan, S.J., Rajan, J., 2022. Capsule networks for image classification: a review. Neurocomputing 509, 102–120. https://doi.org/10.1016/j.neucom.2022.08.073.
Qin, Y., Yuen, C., Shao, Y.M., Qin, B., Li, X.L., 2022. Slow-varying dynamics-assisted temporal capsule network for machinery remaining useful life estimation. IEEE Trans. Cybern. 1–15 https://doi.org/10.1109/tcyb.2022.3164683.
Ragab, M., Alshehri, S., Alhakamy, N.A., Mansour, R.F., Koundal, D., 2022. Multiclass classification of chest X-Ray images for the prediction of COVID-19 using capsule network. Comput. Intell. Neurosci. 2022 https://doi.org/10.1155/2022/6185013.
Sabour, S., Frosst, N., Hinton, G.E., 2017. Dynamic routing between capsules. Paper presented at the Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA.
Sezer, O.B., Gudelek, M.U., Ozbayoglu, A.M., 2020. Financial time series forecasting with deep learning: a systematic literature review: 2005–2019. Appl. Soft Comput. 90, 106181.
Shahin, I., Hindawi, N., Nassif, A.B., Alhudhaif, A., Polat, K., 2022. Novel dual-channel long short-term memory compressed capsule networks for emotion recognition. Expert Syst. Appl. 188 https://doi.org/10.1016/j.eswa.2021.116080.
Shi, B., Bai, X., Yao, C., 2017. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Trans. Pattern Anal. Mach. Intell. 39 (11), 2298–2304. https://doi.org/10.1109/TPAMI.2016.2646371.
Sridhar, S., Sanagavarapu, S., Ieee, 2021. Fake News Detection and Analysis using Multitask Learning with BiLSTM CapsNet model. Paper presented at the 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India.
Staffini, A., 2022. Stock price forecasting by a deep convolutional generative adversarial network. Front. Artif. Intell. 5 https://doi.org/10.3389/frai.2022.837596.
Taieb, S.B., Hyndman, R.J., 2012. Recursive and direct multi-step forecasting: the best of both worlds (Vol. 19): Department of Econometrics and Business Statistics, Monash Univ.
Taieb, S.B., Bontempi, G., Sorjamaa, A., Lendasse, A., 2009. Long-term prediction of time series by combining direct and mimo strategies. Paper Presented at the 2009 International Joint Conference on Neural Networks.
Taieb, S.B., Bontempi, G., Atiya, A.F., Sorjamaa, A., 2012. A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition. Expert Syst. Appl. 39 (8), 7067–7083.
Tampubolon, H., Yang, C.L., Chan, A.S., Sutrisno, H., Hua, K.L., 2019. Optimized CapsNet for traffic jam speed prediction using mobile sensor data under urban swarming transportation. Sensors 19 (23). https://doi.org/10.3390/s19235277.
Tang, Y., Song, Z., Zhu, Y., Yuan, H., Hou, M., Ji, J., Li, J., 2022. A survey on machine learning models for financial time series forecasting. Neurocomputing 512, 363–380.
Tripathi, B., Sharma, R.K., 2022. Modeling bitcoin prices using signal processing methods, bayesian optimization, and deep neural networks. Comput. Econ. https://doi.org/10.1007/s10614-022-10325-8.
Tsantekidis, A., Passalis, N., Tefas, A., Kanniainen, J., Gabbouj, M., Iosifidis, A., 2020. Using deep learning for price prediction by exploiting stationary limit order book features. Appl. Soft Comput. 93, 106401 https://doi.org/10.1016/j.asoc.2020.106401.
Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., Lang, K.J., 1989. Phoneme recognition using time-delay neural networks. IEEE Trans. Acoust. Speech Signal Process. 37 (3), 328–339.
Wang, B., Wang, J., 2020. Energy futures and spots prices forecasting by hybrid SW-GRU with EMD and error evaluation. Energy Econ. 90, 104827 https://doi.org/10.1016/j.eneco.2020.104827.
Wang, H., Wang, J., Cao, L., Li, Y., Sun, Q., Wang, J., 2021. A stock closing price prediction model based on CNN-BiSLSTM. Complexity 2021. https://doi.org/10.1155/2021/5360828.
Wu, C., Wang, J., Hao, Y., 2022. Deterministic and uncertainty crude oil price forecasting based on outlier detection and modified multi-objective optimization algorithm. Resour. Policy 77, 102780. https://doi.org/10.1016/j.resourpol.2022.102780.
Xiang, H.L., Huang, Y.S., Lee, C.H., Chien, T.Y.C., Lee, C.K., Liu, L.X., Chang, R.F., 2021. 3-D Res-CapsNet convolutional neural network on automated breast ultrasound tumor diagnosis. Eur. J. Radiol. 138 https://doi.org/10.1016/j.ejrad.2021.109608.
Zhan, X., Li, Y., Li, R., Gu, X., Habimana, O., Wang, H., 2018. 2018//). Stock Price Prediction Using Time Convolution Long Short-Term Memory Network, Paper presented at the Knowledge Science, Engineering and Management, Cham.
Zhang, C., Sjarif, N.N.A., Ibrahim, R., 2023a. Deep learning models for price forecasting of financial time series: A review of recent advancements: 2020–2022. WIREs Data Mining and Knowledge Discovery, n/a(n/a), e1519. doi:10.1002/widm.1519.
Zhang, J.-L., Zhang, Y.-J., Zhang, L., 2015. A novel hybrid method for crude oil price forecasting. Energy Econ. 49, 649–659. https://doi.org/10.1016/j.eneco.2015.02.018.
Zheng, W. (2021). *Exchange-Traded Fund Price Prediction Based on the Deep Learning Model.* Paper presented at the Proceeding - 2021 China Automation Congress, CAC 2021.