Expert Systems With Applications 208 (2022) 118128

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Stock market index prediction using deep Transformer model

Chaojie Wang [a,*], Yuanyuan Chen [b], Shuqi Zhang [b], Qiuhui Zhang [b]

[a] The Fourth Affiliated Hospital of Jiangsu University, School of Mathematical Science, Jiangsu University, Zhenjiang, 212013, China
[b] School of Mathematical Science, Jiangsu University, Zhenjiang, 212013, China

ARTICLE INFO

Keywords:
Deep learning
Transformer
Stock index prediction

ABSTRACT

Applications of deep learning in financial market prediction have attracted widespread attention from investors and scholars. From convolutional neural networks to recurrent neural networks, deep learning methods exhibit superior ability to capture the non-linear characteristics of stock markets and, accordingly, achieve a high performance on stock market index prediction. In this paper, we utilize the latest deep learning framework, Transformer, to predict the stock market index. Transformer was initially developed for the natural language processing problem, and has recently been applied to time series forecasting. Through the encoder–decoder architecture and the multi-head attention mechanism, Transformer can better characterize the underlying rules of stock market dynamics. We implement several back-testing experiments on the main stock market indices worldwide, including CSI 300, S&P 500, Hang Seng Index, and Nikkei 225. All the experiments demonstrate that Transformer outperforms other classic methods significantly and can gain excess earnings for investors.

1. Introduction

Stock market prediction is an external topic in the financial markets. The fluctuation at every moment is related to the flow of enormous wealth. Extreme cases can even affect the stability of financial systems, such as the financial crisis in 2008 (Anagnostidis, Varsakelis, & Emmanouilides, 2016). In the long history of capital markets, many prediction methods have been proposed, including technical analysis, fundamental analysis, and time series analysis (Murphy, 1999). At present, with the rapid development of computer science, especially the progress in the field of deep learning, researchers hope that big data may help to understand the dynamics of stock prices (Tsai & Hsiao, 2010). More and more financial institutions are trying to make investment decisions based on deep learning algorithms instead of human subjectivity. Due to the high complexity of financial markets, the combination of deep learning techniques and financial time series forecasting is considered to be one of the most attractive topics (Cavalcante, Brasileiro, Souza, Nobrega, & Oliveira, 2016).

Some existing literature has illustrated the performances of deep learning frameworks on financial market prediction. The typical network structures, such as Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), and Long Short-term Memory Neural Network (LSTM), are shown to have better prediction ability than traditional machine learning algorithms (Di Persio & Honchar, 2016; Fischer & Krauss, 2018). (Selvin, Vinayakumar, Gopalakrishnan, Menon, & Soman, 2017) compared three stock prediction models based on CNN, RNN, and LSTM, and demonstrated that LSTM has the best prediction performance due to its long-term memory for stock sequences. Bao, Yue, and Rao (2017) proposed to combine Autoencoder and LSTM for stock price prediction. The results show that prediction accuracy and profitability are improved compared to the single structure. Besides, more and more cutting-edge deep learning frameworks are also being tried on the stock prediction problem, including generative adversarial networks (Zhang, Zhong, Dong, Wang, & Wang, 2019), reinforcement learning (Li, Ni, & Chang, 2020), and so on. Jiang (2021) summarized the recent progress of applications of deep learning in stock market prediction.

Although deep learning has been widely used in stock market prediction, traditional architectures, like CNN and RNN, still have some limitations. For example, the pooling layers in CNN cause the loss of valuable information by ignoring the part-whole relationships (Xi, Bing, & Jin, 2017); RNN has been prone to result in gradient disappearance and gradient explosion in the back-propagation process (Huang, Shen, & Liu, 2019). To get rid of these drawbacks, Vaswani et al. (2017) proposed an innovative deep learning architecture called Transformer, which uses the attention mechanism instead of traditional CNN and RNN frameworks and achieves great success in natural language processing (NLP) problems. Compared with the sequential structure of RNN and LSTM, the self-attention mechanism in Transformer can be trained in parallel and it is easier to obtain global information. Recently, Transformer has been widely applied in various fields, including

* Corresponding author.
E-mail addresses: cjwang@ujs.edu.cn (C. Wang), 3180702005@stmail.ujs.edu.cn (Y. Chen), 3190114009@stmail.ujs.edu.cn (S. Zhang), 3190102008@stmail.ujs.edu.cn (Q. Zhang).

https://doi.org/10.1016/j.eswa.2022.118128
Received 29 October 2021; Received in revised form 15 June 2022; Accepted 8 July 2022
Available online 14 July 2022
0957-4174/© 2022 Elsevier Ltd. All rights reserved.

computer vision (Parmar et al., 2018), audio processing (Li, Liu, Liu, Zhao, & Liu, 2019a), chemical synthesis (Tetko, Karpov, Van Deursen, & Godin, 2020), and so on.

Inspired by the success of Transformer in modeling the sequential data in NLP, it is a straightforward idea to apply it to predict stock markets. However, to the best of our knowledge, few researches evaluate the performance of Transformer on stock market prediction. The existing results mainly focus on the applications of Transformer in sentiment analysis (Köksal & Özgür, 2021; Liu et al., 2019). They predicted the direction of price movements by analyzing textual information like financial news and comments from social media. In this paper, we predict the stock market index based on the Transformer framework. Different from previous studies, we model the daily closing price data directly rather than unstructured textual data. We are concerned about the specific values of the stock index in the future, not only the direction of price movements. The main stock market indices worldwide are considered, including the CSI 300 Index, the S&P 500 Index, the Hang Seng Index, and the Nikkei 225 Index. Each dataset is split into training and testing sets. The training set is used to train the model parameters and learn the patterns hidden the data. The testing set is used to evaluate the performance of trained models and make model comparisons. Specifically, we preprocess the dataset first to make the model converge better. Then the daily closing prices of the last few days are fed into Transformer as input data. Through the encoder–decoder architecture, Transformer outputs the predicted closing price of the next day, which is compared with the actual value for evaluation. The results are evaluated by the accuracy of prediction and the performance of trading strategies. Compared with the traditional deep learning models, such as CNN, RNN, and LSTM, Transformer exhibits higher prediction accuracy and better net worth curves in all experiments. These results demonstrate that Transformer outperforms other existing models in stock market prediction.

This paper is organized as follows: Section 2 reviews the past research relevant to our work. Section 3 introduces the background knowledge of traditional deep learning models used in the experiments. Section 4 details the architecture of Transformer used for stock prediction. Section 5 describes the process of back-testing experiments, including data processing, parameter settings, evaluation criteria, error analysis, and net value analysis. Section 6 concludes this paper.

2. Related work

Stock market prediction methods are divided into two main categories: fundamental and technical analysis (Alzazah & Cheng, 2020). Fundamental analysis relies on analyzing unstructured textual information like financial news, earnings reports, and macroeconomic factors. Technical analysis mainly focuses on analyzing historical stock prices and trading information to predict future values. From different perspectives, many deep learning methods are proposed to predict the stock market.

2.1. Fundamental analysis

The fundamental information includes financial reports, news, and comments from social media like Twitter. Based on the textual information, financial analysts analyze market sentiment and make investment recommendations. Nguyen, Shirai, and Velcin (2015) examined the relationship between sentiments in social media and stock market movements, and demonstrated that sentiment analysis indeed improves the performance of stock prediction. Sun, Lachanski, and Fabozzi (2016) utilized the latent space model to investigate the potential use of textual information from Twitter in predicting the stock market. Malandri, Xing, Orsenigo, Vercellis, and Cambria (2018) demonstrated that public mood is correlated with financial markets and affects the optimal asset allocation.

Sentiment analysis is one of the key topics in NLP. Recently, researchers performed sentiment analysis using text mining and computational techniques to automatically extract sentiments from text (Agarwal, Mittal, Bansal, & Garg, 2015). They aimed to classify the given text into a positive, negative, or neutral view, and then predict the direction of price movements (Rajput & Bobde, 2016). Sohangir, Wang, Pomeranets, and Khoshgoftaar (2018) considered several typical neural network models, including LSTM and CNN, to improve the performance of sentiment analysis for StockTwits. Xing, Cambria, and Zhang (2019) proposed a novel model termed "sentiment-aware volatility forecasting", which incorporates market sentiment for stock return fluctuation prediction. Besides, Picasso, Merello, Ma, Oneto, and Cambria (2019) considered the combination of technical analysis and sentiment analysis through deep learning techniques, which provides a novel way to predict market trends.

Inspired by the success of attention mechanisms in computer vision, Jin, Yang, and Liu (2020) combined LSTM with attention mechanisms and proposed a stock market prediction model that can focus on critical information. For the emerging stock markets, Köksal and Özgür (2021) evaluated the performance of Transformers on a Twitter dataset for Turkish sentiment analysis. de Oliveira Carosia, Coelho, and da Silva (2021) identified the most suitable artificial neural network architecture to perform sentiment analysis on financial news in the Brazilian stock market.

In general, fundamental analysis and sentiment analysis mainly focus on predicting the direction of price movements based on unstructured textual information. Deep learning classification algorithms improve the accuracy of prediction, but may ignore the amplitude of fluctuation. A 1% rise in the stock price is in the same direction as a 10% rise. But they have quite different impacts on the net values. Thus, in this paper, we consider modeling the stock price data directly.

2.2. Technical analysis

Technical analysis believes that all the information on the stock market is reflected in the price movement. It is sufficient to model and predict the stock prices directly. Deep learning techniques can efficiently predict future values by learning the underlying patterns from historical data (Long, Lu, & Cui, 2019).

In early studies, RNN and LSTM were typical frameworks used to predict financial time series data. Selvin et al. (2017) compared three stock prediction models based on CNN, RNN, and LSTM, and demonstrated that LSTM has the best prediction performance due to its long-term memory for stock sequences. Inspired by the applications of attention mechanisms in neural machine translation, Cheng, Huang, and Wu (2018) proposed an attention-based LSTM model to predict stock price movement and make trading strategies. Chen and Ge (2019) explored the attention mechanism in the LSTM network to predict stock price movement in Hong Kong.

Vaswani et al. (2017) proposed the seminal architecture Transformer and achieved great success in NLP problems. After that, Li et al. (2019b) applied Transformer to the time series prediction and broke the memory bottleneck problem. Mohammdi Farsani and Pazouki (2021) showed that Transformer based on self-attention has better performance and lower computational complexity to predict time series problems through the electricity consumption dataset in a power grid and traffic data. Zhou et al. (2021) proposed an improved Transformer model called Informer for long-sequence time series forecasting. Based on similar ideas, this paper considers Transformer to predict the stock market index. As far as we know, it is an innovative work to evaluate the performance of Transformer on the stock market prediction.

3. Background

This section introduces several classic deep learning models for stock market prediction, including CNN, RNN, and LSTM, which are used in the following model comparisons.

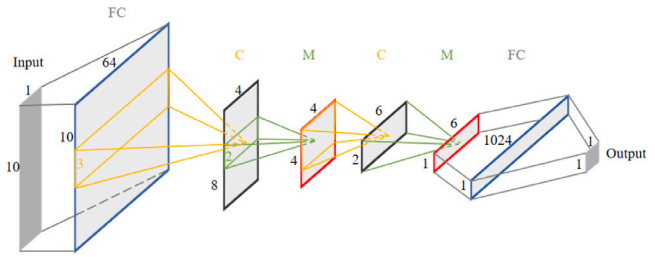


Fig. 1. An example of CNN architecture for stock market prediction. Here FC represents the fully connected layer, C represents the one-dimensional convolutional layer (with kernel size = 3 and stride = 1), and M represents the max-pooling layer (with kernel size = 2 and stride = 2).

3.1. CNN

CNN is one of the most important neural networks in deep learning. Through the combination of convolutional layers and pooling layers, CNN achieves great success in image problems. It motivates researchers to apply it to sequential data. Unlike two-dimensional image processing problems, CNN uses the one-dimensional convolutional operator for stock market prediction.

Assume that the observed sequence is $\boldsymbol{X} = \{x_t : t = 1, \dots, T\}$, where $x_t \in \mathbb{R}$ denotes the stock price at time t. First, we map the sequential data into a higher dimensional space through a fully connected network, which outputs a new dataset $\tilde{\boldsymbol{X}} = \{\tilde{\boldsymbol{x}}_t : t = 1, \dots, T\}$, where $\tilde{\boldsymbol{x}}_t \in \mathbb{R}^d$ denotes the d-dimensional data. Then, the high dimensional dataset is fed into an one-dimensional convolutional layer with K filters, where the weights $\boldsymbol{W}_k = (\boldsymbol{w}_1, \dots, \boldsymbol{w}_j, \dots, \boldsymbol{w}_m) \in \mathbb{R}^{d \times m}$ and biases $b_k \in \mathbb{R}$, $k = 1, \dots, K$. The output of the convolutional layer is written as $\boldsymbol{Z} = \{\boldsymbol{z}_k : k = 1, \dots, K\}$, where $\boldsymbol{z}_k \in \mathbb{R}^{T-m+1}$ and

$$z_{k,i} = \phi(b_k + (\boldsymbol{W}_k * \tilde{\boldsymbol{X}})) = \phi(b_k + \sum_{j=1}^{m} \langle \boldsymbol{w}_j, \tilde{\boldsymbol{x}}_{m+i-j} \rangle), \quad 1 \le i \le T - m + 1. \tag{1}$$

Here $(\cdot * \cdot)$ represents the convolution operation, $\langle \cdot, \cdot \rangle$ represents the inner product in $\mathbb{R}^d$, and $\phi : \mathbb{R} \longrightarrow \mathbb{R}$ is an activation function.

Through the one-dimensional convolutional operator, the original sequence is transformed into a new sequence with extracted features. Then the sequence may be subsampled through the pooling layer and output the final result through a fully connected layer. The final result should be a real value as the predicted stock price at time $T + 1$. Fig. 1 presents an example of the CNN architecture for stock market prediction.

3.2. RNN–LSTM

For the time series prediction problem, the greatest challenge is how to model the interdependent information in the context. An early attempt is RNN, which tackles this problem by introducing an internal state called a memory cell to store the past information. RNN succeeds in characterizing the connection of sequential data in the context, but this connection decays with the increase of gap distance. The long-term dependence in RNN may cause the vanishing gradient and gradient explosion problems in the back-propagation process (Huang et al., 2019).

As a variant of RNN, LSTM improves models with a specific gate structure. The interactive information are passed through three gates, i.e., input gate i_t, forget gate f_t and output gate o_t. The useful information will be stored in the memory cells and passed to the next neuron, while the useless part can be forgotten to save memory space. Fig. 2 presents the details of the LSTM architecture.

Given the observed sequence is $\boldsymbol{X} = \{x_t : t = 1, \dots, T\}$, where $x_t \in \mathbb{R}$ denotes the stock price at time t, the update steps of LSTM can be expressed as follows:

$$\begin{aligned}
f_t &= \sigma\left(W_f \cdot \left[h_{t-1}, x_t\right] + b_f\right), \\
i_t &= \sigma\left(W_i \cdot \left[h_{t-1}, x_t\right] + b_i\right), \\
\tilde{c}_t &= \tanh\left(W_c \cdot \left[h_{t-1}, x_t\right] + b_c\right), \\
c_t &= f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t, \\
o_t &= \sigma\left(W_o \cdot \left[h_{t-1}, x_t\right] + b_o\right), \\
h_t &= o_t \cdot \tanh\left(c_t\right),
\end{aligned} \tag{2}$$

where W and b are weights and bias for the corresponding connection, σ and tanh represent the sigmoid function and the tanh function, and $[\cdot, \cdot]$ denotes the concat operation which merges the two vectors together.

4. Transformer

Different from traditional deep learning models, we consider the Transformer architecture for stock market prediction in this paper. This section introduces the Transformer architecture in detail. Fig. 3 presents the whole framework of our Transformer model briefly. The code for this paper is available upon request.

4.1. Embeddings and positional encoding

Embedding is a commonly used technique in NLP problems, which maps the sparse and high-dimensional word vectors into a low-dimensional space (Mikolov, Chen, Corrado, & Dean, 2013). Similarly, here we feed the sequential data into the embedding layer. Given the input data $X = \{x_t : 1, \dots, T\} \in \mathbb{R}^T$, the d-dimensional embedding layer outputs a matrix $A \in \mathbb{R}^{T \times d}$ through a fully connected network.

To characterize the sequential information in the time series, we add the positional encoding to the embedded input. The sine and cosine functions of different frequencies are used to encode the positional information:

$$\begin{aligned}
\mathrm{PE}_{(t,2s)} &= \sin\left(t/10000^{2s/d}\right), \\
\mathrm{PE}_{(t,2s+1)} &= \cos\left(t/10000^{2s/d}\right),
\end{aligned} \tag{3}$$

where $1 \le 2s \le d$. Thus, the positional encoding information is $\mathrm{PE} \in \mathbb{R}^{T \times d}$. Then, the embedded input and positional encoding are concatenated together and fed into the encoder layers.

4.2. Encoder–decoder

Transformer takes the encoder–decoder architecture, which is widely used in machine translation problems (Cho, Merriënboer, Bahdanau, & Bengio, 2014). The encoder compresses the key information of the input sequence into a fixed-length vector, and then the decoder converts it into an output (Sutskever, Vinyals, & Le, 2014). The encoder–decoder architecture provides an approach to deal with long sequential data (Bahdanau, Cho, & Bengio, 2014).

The encoder in Fig. 3 (left block) is composed of a stack of M layers with identical structures. Each layer includes two sub-layers: a multi-head self-attention layer and a fully connected neural network. The residual connection and normalization are used in each sub-layer to improve the performance. The decoder in Fig. 3 (right block) is similar to the encoder. The only difference is that it includes two multi-head self-attention layers. Different from the original decoder in Vaswani et al. (2017), here we do not use the mask attention mechanism since all the inputs in the decoder are observed historical data without future information.

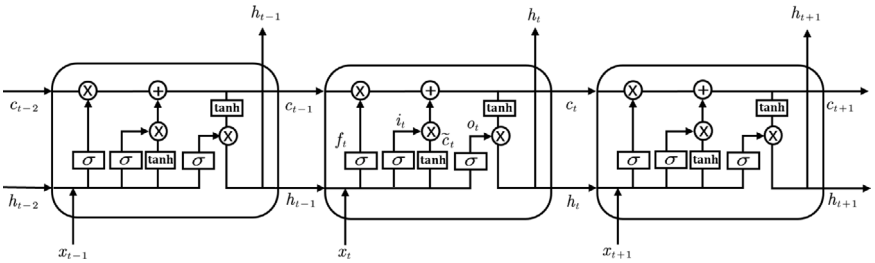


Fig. 2. Long Short-Term Memory network.

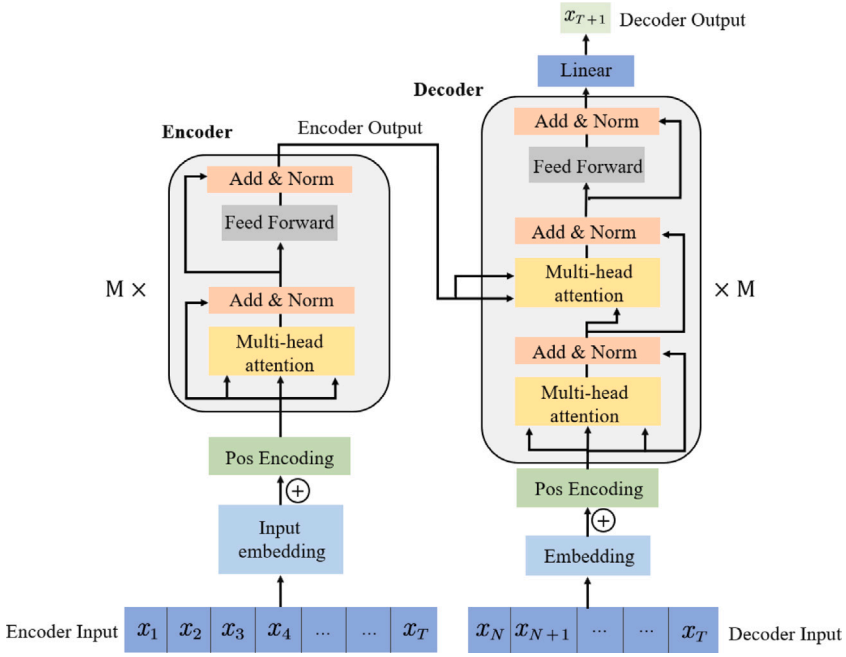


Fig. 3. The architecture of Transformer for stock price prediction. Here $x_1, x_2, \ldots, x_T$ represent the historical stock prices as the input data. Finally, the model outputs a real value x_{T+1} as the predicted stock price at time $T+1$.

4.3. Self-attention

The idea of attention mechanisms may be the most exciting innovation in deep learning in recent years (Mnih, Heess, Graves, et al., 2014). It focuses limited attention on the important local areas so as to save computational resources and obtain the most useful information quickly. In early works, attention mechanisms are often used with CNN and RNN frameworks together (Bahdanau et al., 2014). Vaswani et al. (2017) first demonstrated that deep learning models could get rid of traditional CNN and RNN frameworks and proposed the Transformer architecture, which uses self-attention mechanisms only. Just as they said, "attention is all you need".

The self-attention mechanism in Vaswani et al. (2017) is defined as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V, \tag{4}$$

where $Q \in \mathbb{R}^{T\times d}$, $K \in \mathbb{R}^{T\times d}$ and $V \in \mathbb{R}^{T\times d}$ are query, key and value matrices respectively, which are outputs of three different linear layers with the same input. Fig. 4 presents the structure of the self-attention mechanism.

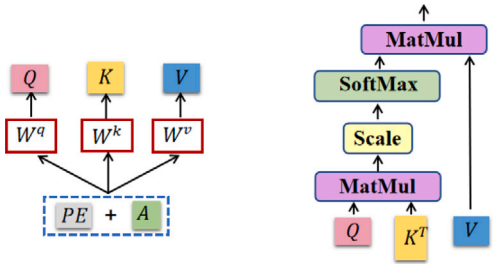


Fig. 4. The structure of the self-attention mechanism.

4.4. Multi-head attention

The self-attention mechanism provides a new perspective to focus on the important local information. Vaswani et al. (2017) also mentioned that multiple self-attention, called multi-head attention, can achieve better performance. In the multi-head attention mechanism, each attention function is executed in parallel with the respective projected version of the query, key, and value matrices. Then the outputs of all attention functions are concatenated together to produce the final result through a linear layer. The formula of multi-head attention is expressed as follows:

$$\begin{aligned}\text{MultiHead}(Q, K, V) &= \text{Concat}\left(\text{head}_1, \text{head}_2, \ldots, \text{head}_h\right) W^O, \\ \text{head}_i &= \text{Attention}\left(QW_i^Q, KW_i^K, VW_i^V\right),\end{aligned} \tag{5}$$

where $i = 1, \ldots, h$ and W_i^Q, W_i^K, W_i^V, W^O are weights of corresponding networks.

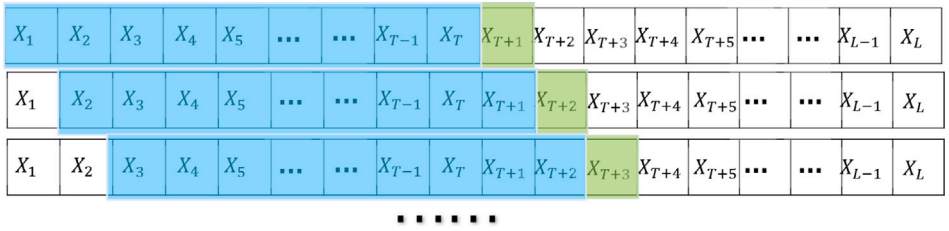


Fig. 5. A moving window approach is utilized to construct features and labels from the observed time series. In each prediction, T data marked with blue are used as the input features of the model, and the next data marked with green is used as the output label.

Table 1
The main hardware and software configurations used for experiments.

	Configuration
CPU	AMD Ryzen 7 5800H
GPU	NVIDIA QUADRO GV100
Anaconda version	Anaconda 4.5.11
Python version	Python 3.7
Pytorch version	Pytorch 1.7.0

The multi-head attention mechanism allows the model to jointly attend to information from different representation subspaces at different positions. It has been prone to be a better choice than single self-attention.

5. Experiments

In this paper, we implement several back-testing experiments for main stock market indices worldwide to demonstrate the performance of Transformer in stock market prediction. Since the United States, China and Japan are the top three economies in the world and their GDP made up about 50% of the world economy in 2020, we consider four stock market indices from these countries to represent the global market: the Shanghai and Shenzhen 300 Index (CSI 300) in China, the Standard & Poor's 500 Index (S&P 500) in the US, the Nikkei 225 Index (N225) in Japan, and the Hang Seng Index (HSI) in Hong Kong. We model the daily closing prices of these indices over the period from Jan 1, 2010 to Dec 31, 2020. The hardware and software configurations used for experiments are shown in Table 1.

5.1. Data processing

The observed data for each index is a one-dimensional time series of daily closing prices. First, we split the whole dataset into the training and testing sets. The training set includes the first 80% data, which is used to train model parameters. The last 20% data as the testing set is used to evaluate the performance of models.

To reduce the volatility of datasets and obtain a robust model, we normalize the original data (including the training and testing sets) as follows:

$$\hat{x}_t = \frac{x_t - \mu}{\sigma}, \tag{6}$$

where $\hat{x}_t$ is the normalized price at time t, μ and σ are the sample mean and sample standard deviation of the training set.

In each prediction, we use the previous T closing prices to predict the closing price on the next trading day $T+1$. A moving window approach is utilized to construct features and labels from the observed time series. Fig. 5 presents the procedure in detail.

5.2. Hyper-parameters setting

We implement a large number of experiments on the training set to determine the optimal hyper-parameters in advance. The batch size for the mini-batch training is set as 16. We use the mean squared error as the loss function to compare predicted values with actual values. The Adam optimizer with a learning rate of 0.0001 is used for training models. We set the number of epochs as 1000 to guarantee the convergence of the training process. Fig. 6 presents the loss functions varying with epochs for each dataset. They exhibit significant downward trends and converge within 1000 epochs eventually.

To avoid the overfitting problem, we apply the dropout technique for each sub-layer, where the dropout rate is set as 0.1. For the input data, we take the input length for Encoder as $T = 9$ and Decoder as $N = 2$. The embedding dimension for input data is $d = 32$. Besides, the initial values of parameters waiting for training, such as weights and biases in neural networks, are sampled from the uniform distribution $[-1, 1]$ independently.

5.3. Evaluation criteria

The performance of models is evaluated from two perspectives: prediction accuracy and net value analysis. For prediction accuracy, we compare the predicted values with the true data in the testing set and calculate the prediction errors. Three common indicators for prediction errors are used to evaluate the performance:

- Mean Absolute Error (MAE):

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i|, \tag{7}$$

  where $\hat{y}_i$ is the predicted value, y_i is the true value, and N is the sample size. MAE is an average of the absolute difference between the predicted and actual values, which can avoid the mutual cancellation of errors. It reflects the absolute errors in forecasting (Willmott & Matsuura, 2005).
- Mean Squared Error (MSE):

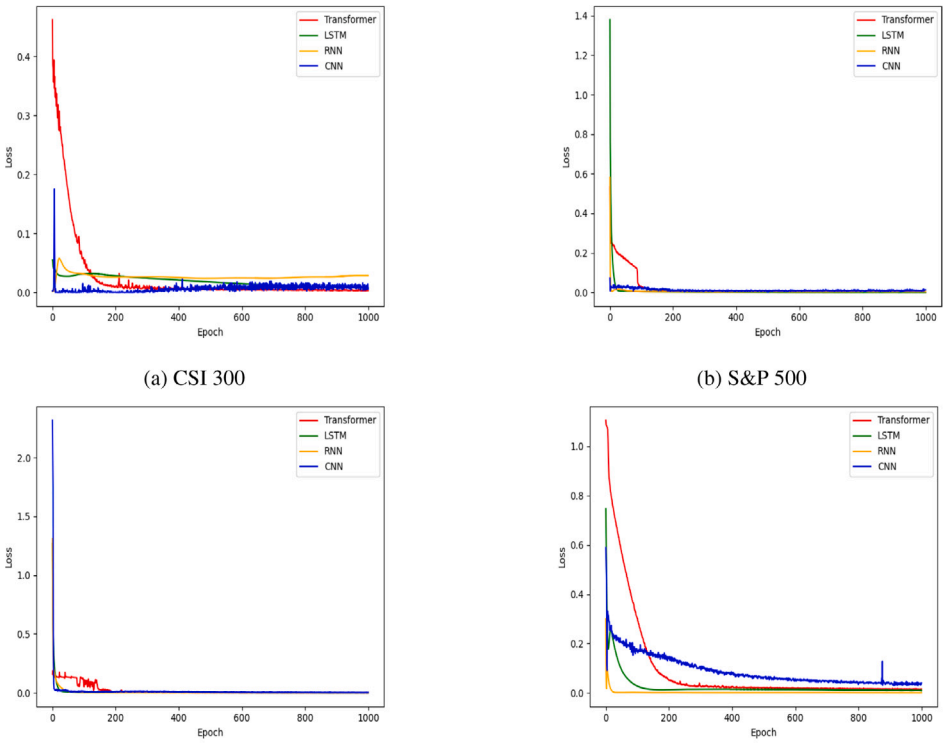
$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2, \tag{8}$$

  where $\hat{y}_i$ is the predicted value, y_i is the true value, and N is the sample size. MSE is a commonly used indicator to measure the performance of time series forecasting. Similarly with MAE, it also measures the absolute errors in forecasting (Lehmann & Casella, 2006). Here MSE is used as the loss function for model training.
- Mean Absolute Percentage Error (MAPE):

$$\text{MAPE} = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{\hat{y}_i - y_i}{y_i}\right| \times 100\%, \tag{9}$$

  where $\hat{y}_i$ is the predicted value, y_i is the true value, and N is the sample size. MAPE also measures the prediction accuracy. Different from MAE and MSE, MAPE is a relative indicator to measure the percentage of errors (Myttenaere, Golden, Le Grand, & Rossi, 2016).

The three indicators measure the prediction accuracy of models from different perspectives. MAE and MSE measure the absolute errors



(a) CSI 300 (b) S&P 500

(c) N225 (d) HSI

Fig. 6. The loss functions of four datasets in the training process.

of prediction, and MAPE reports the relative errors. The smaller values of these indicators mean better performance.

We also evaluate the performance of models from net values by constructing a simple trading strategy according to the predicted values. The trading strategy is designed as follows: if the predicted value $\hat{y}_{t+1}$ is larger than the latest observed value y_t, we long one position index; if $\hat{y}_{t+1}$ is smaller than y_t, we short one position index; otherwise, we hold no position. So, the return at time $t+1$ can be expressed as follows:

$$R_{t+1} = \ln \frac{y_{t+1}}{y_t} \times \text{sign}(\hat{y}_{t+1} - y_t), \tag{10}$$

where sign$(\cdot)$ denotes the sign function. Then we can calculate the net value (NV) of the strategy as follows:

$$\text{NV}_t = 1 + \sum_{i=2}^{t} R_i, \tag{11}$$

for $t \geq 2$ and $\text{NV}_1 = 1$. The net value represents the total return of the strategy. Here we assume that the transaction cost is 1‰ by referring to the rates in the global main markets.

Besides the total return, we are also concerned on the risks of strategies, which is measured by volatility, max drawdown, and Sharpe ratio:

- Volatility: a classic measure of risk, which is defined as the standard deviation of returns:

  $$\text{Volatility} = \sigma(R_i). \tag{12}$$

  So the model with lower volatility has less risk.
- Max Drawdown: an important risk indicator used to describe the worst situation in the whole investment period, which is defined as follows:

  $$\text{Max Drawdown} = \max_{i<j} \frac{\text{NV}_j - \text{NV}_i}{\text{NV}_i}. \tag{13}$$

  So the model with smaller max drawdown is better.
- Sharpe ratio: the most commonly used indicator in portfolio management, proposed by William F. Sharpe (Sharpe, 1994). It takes both returns and risks into consideration and is defined as follows:

  $$\text{Sharpe ratio} = \frac{E(R_t) - R_f}{\sigma(R_t)}, \tag{14}$$

  where $E(R_t)$ denotes the expected daily return, $\sigma(R_t)$ denotes the standard deviation of daily return and R_f denotes the risk-free interest rate. A higher Sharpe ratio means a better model.

5.4. Error analysis

Here we report the outputs of back-testing experiments to demonstrate the performance of our Transformer model. Considering the uncertainty of deep learning methods, we implement 10 independent runs of learning and validation for each dataset. The means and standard errors of the three accuracy indicators are presented in Table 2. We also perform the Mann–Whitney U test to show the advantages of Transformer are significant (Mann & Whitney, 1947). Table 3 reports the P-values of the Mann–Whitney U test on the difference in MSE between Transformer and compared models.

From Table 2, all three indicators demonstrate that our Transformer model outperforms other classic methods in prediction accuracy. Whether from the perspective of absolute errors (MAE and MSE) or relative errors (MAPE), Transformer obtains the smallest mean prediction errors among the compared models in all datasets. Table 3 demonstrates that the advantages of Transformer over compared models are significant. Besides, our Transformer model reports a minor standard error in each setting. It means that our model is robust for model training.

Fig. 7 presents one of the fitted curves generated by Transformer for four main stock market indices. It is observed that the predicted values are quite close to the real data in both training and testing sets. Fig. 8 compares the fitted curves generated by CNN, RNN, LSTM and

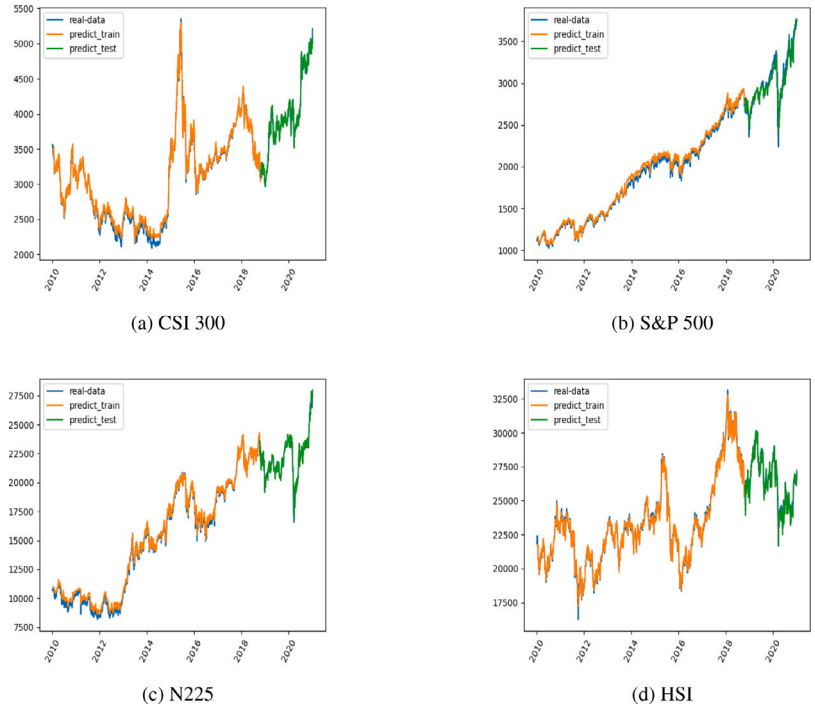


Fig. 7. The predicted curves of the Transformer model. Here the blue lines mean the real stock index prices data. The orange lines represent the predicted values in the training set and the green lines represent the predicted values in the testing set.

Table 2
The means and standard errors of three indicators across 10 independent experiments for four datasets.

Index	Models	MAE	MSE	MAPE
CSI 300	RNN	0.0825(0.0035)	0.0127(0.0010)	1.2986(0.0503)
	CNN	0.0948(0.0020)	0.0170(0.0006)	1.4816(0.0306)
	LSTM	0.0693(0.0091)	0.0091(0.0002)	1.1216(0.0144)
	Transformer	**0.0641(0.0025)**	**0.0079(0.0004)**	**0.9549(0.0317)**
Nikkei 225	RNN	0.0680(0.0046)	0.0086(0.0004)	1.4477(0.1044)
	CNN	0.0851(0.0036)	0.0158(0.0009)	1.7881(0.0870)
	LSTM	0.0585(0.0025)	0.0066(0.0003)	1.2438(0.0572)
	Transformer	**0.0471(0.0017)**	**0.0043(0.0002)**	**1.0072(0.0343)**
Hang Seng	RNN	0.1719(0.0073)	0.0446(0.0040)	1.9048(0.2489)
	CNN	0.1506(0.0122)	0.0353(0.0059)	1.7989(0.1514)
	LSTM	0.0985(0.0047)	0.0164(0.0013)	1.1623(0.0584)
	Transformer	**0.0881(0.0025)**	**0.0138(0.0005)**	**1.0403(0.0271)**
S&P 500	RNN	0.1359(0.0274)	0.0321(0.0087)	2.2567(0.4682)
	CNN	0.1533(0.0188)	0.0414(0.0101)	2.5158(0.2709)
	LSTM	0.1092(0.0256)	0.0236(0.0099)	1.7768(0.4001)
	Transformer	**0.0814(0.0131)**	**0.0145(0.0037)**	**1.3800(0.2163)**

Table 3
P-values of the Mann–Whitney *U* test on the difference in MSE between transformer and compared models.

	CSI 300	Nikkei 225	Hang Seng	S&P 500
RNN	0.0001	0.0001	0.0001	0.0001
CNN	0.0001	0.0001	0.0001	0.0001
LSTM	0.0001	0.0001	0.0001	0.0001

Transformer. It also demonstrates that our Transformer model has the best prediction accuracy among the compared methods in all datasets.

5.5. Net value analysis

Besides the prediction accuracy, we also evaluate the performance of models by net values, which are generated by the given trading strategy in Section 5.3. Table 4 reports the total return, volatility, max drawdown, and Sharpe ratio of different models for four datasets. We also consider the passive strategy, which means "buy & hold" (B&H), as a benchmark. Our Transformer model reports the highest total return and Sharpe ratio among the compared models in all four datasets. It also has a competitive performance on the volatility and max drawdown. Fig. 9 presents the net value curves of different models in each dataset. They demonstrate that investors can gain higher excess earnings with the prediction by Transformer.

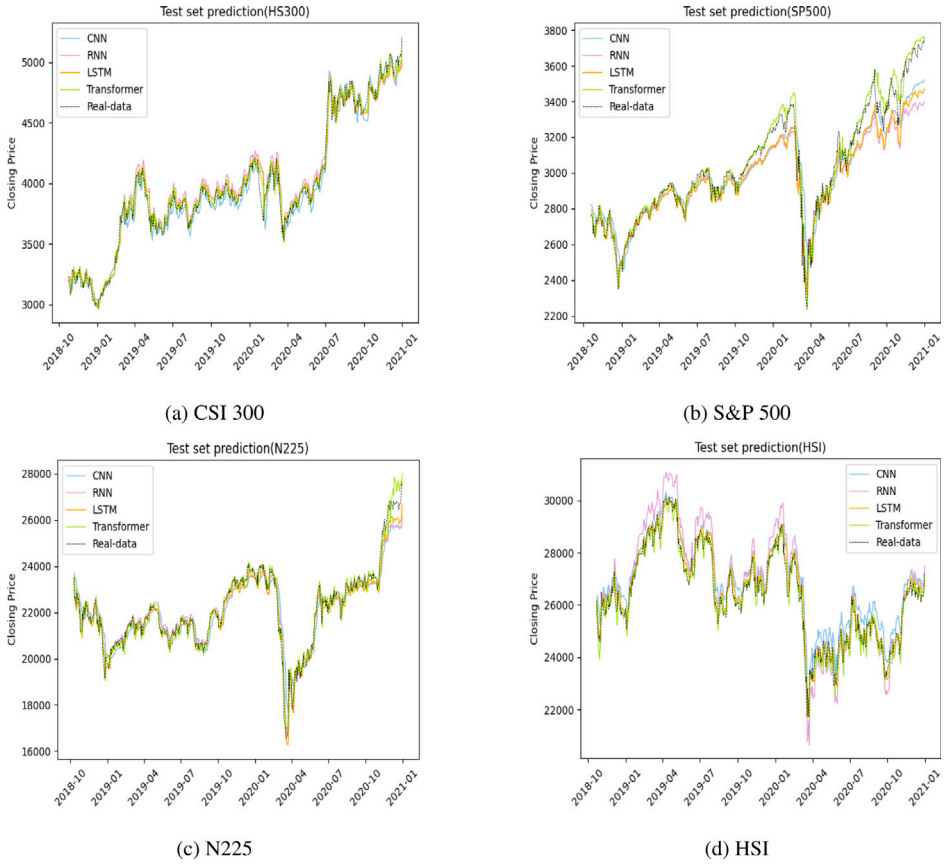
Again, we perform the Mann–Whitney *U* test to show the excess earnings of Transformer are significant. Table 5 reports the *P*-values of the Mann–Whitney *U* test on the difference in total returns between Transformer and compared models. It demonstrates that the total return of Transformer is superior to other strategies significantly.

6. Conclusion

This paper evaluates the performance of Transformer in stock market prediction. Through several experiments on four main stock market indices, we demonstrate that our Transformer model outperforms other traditional deep learning models and the buy & hold strategy significantly from the perspectives of both prediction accuracy and net value analysis. It implies that financial time series forecasting is a promising application area for the Transformer architecture. Investors can gain higher excess earnings with the prediction by Transformer in practice.

The superior performance of Transformer may mainly owe to the multi-head attention mechanism. Note that financial time series, especially the movements of stock indices, are highly noisy. The multi-head attention mechanism is an efficient way to capture important information while filtering out irrelevant noise. Compared with the typical CNN and RNN architectures, Transformer has a stronger ability to extract key features, and thus achieves better prediction performance. This paper provides empirical evidence on Transformer's performance in stock market prediction. More theoretical results will be explored in the future.

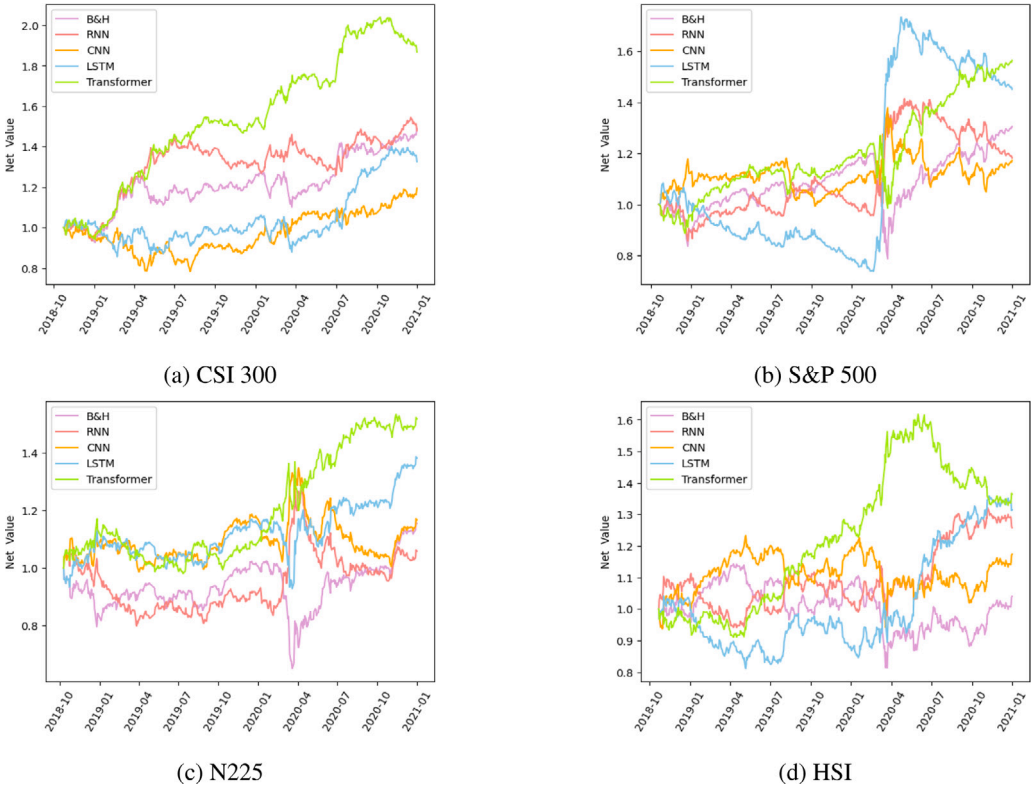
There are also some limitations in this study. This paper only considers the prediction of a single stock market index separately, which is a one-dimensional financial time series data. Actually, the



(a) CSI 300 (b) S&P 500

(c) N225 (d) HSI

Fig. 8. The predicted curves of different models in the testing set.



(a) CSI 300 (b) S&P 500

(c) N225 (d) HSI

Fig. 9. Net value curves of different models for four datasets.

Table 4
The performance of the trading strategy for four datasets.

Index	Models	Return(%)	Volatility(%)	MaxDrawdown(%)	Sharpe ratio
CSI 300	B&H	49.03	1.34	−17.52	1.0868
	RNN	48.03	1.34	−19.94	1.0636
	CNN	19.50	1.34	−24.32	0.4323
	LSTM	30.06	1.34	−18.69	0.6657
	Transformer	**86.88**	**1.33**	**−17.14**	**1.9384**
Nikkei 225	B&H	15.48	1.35	−37.49	0.3370
	RNN	5.82	1.35	−34.35	0.1267
	CNN	16.71	1.34	−34.34	0.3643
	LSTM	38.28	1.34	−24.15	0.8342
	Transformer	**51.86**	**1.34**	**−19.08**	**1.1304**
Hang Seng	B&H	4.03	1.26	−32.92	0.0932
	RNN	25.74	1.26	**−17.07**	0.5955
	CNN	17.36	1.26	−29.10	0.4008
	LSTM	31.50	1.26	−23.02	0.7280
	Transformer	**36.41**	**1.26**	−28.87	**0.8412**
S&P 500	B&H	30.53	1.63	−41.43	0.5353
	RNN	18.22	1.63	**−23.13**	0.3193
	CNN	17.23	1.63	−32.83	0.3023
	LSTM	45.02	1.63	−34.55	0.7900
	Transformer	**56.35**	**1.62**	−28.50	**0.9892**

Table 5
P-values of the Mann–Whitney *U* test on the difference in total returns between transformer and compared models.

	CSI 300	Nikkei 225	Hang Seng	S&P 500
B&H	0.0001	0.0001	0.0001	0.0001
RNN	0.0001	0.0001	0.0001	0.0001
CNN	0.0001	0.0001	0.0001	0.0001
LSTM	0.0001	0.0001	0.0001	0.0001

global financial markets are highly correlated with each other. How to utilize the mutual information and model the high-dimensional time series data jointly is an interesting topic that will be discussed in future work. Besides, the applications of Transformer in many other financial markets, such as the commodities futures market and bond market, are valuable things to explore.

CRediT authorship contribution statement

Chaojie Wang: Conceptualization, Methodology, Writing – review & editing. **Yuanyuan Chen:** Software, Writing – original draft. **Shuqi Zhang:** Software, Visualization. **Qiuhui Zhang:** Investigation, Data curation.

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.

Data availability

Data will be made available on request.

Acknowledgments

This research was funded by a scientific grant by Jiangsu Commission of Health [ZD2021009] and one UJS direct grant [5501190012].

References

Agarwal, B., Mittal, N., Bansal, P., & Garg, S. (2015). Sentiment analysis using common-sense and context information. *Computational Intelligence and Neuroscience*, *2015*.

Alzazah, F. S., & Cheng, X. (2020). Recent advances in stock market prediction using text mining: A survey. *IntechOpen*.

Anagnostidis, P., Varsakelis, C., & Emmanouilides, C. J. (2016). Has the 2008 financial crisis affected stock market efficiency? The case of eurozone. *Physica A: Statistical Mechanics and its Applications*, *447*, 116–128.

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.

Bao, W., Yue, J., & Rao, Y. (2017). A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PLoS One*, *12*, Article e0180944.

Cavalcante, R. C., Brasileiro, R. C., Souza, V. L., Nobrega, J. P., & Oliveira, A. L. (2016). Computational intelligence and financial markets: A survey and future directions. *Expert Systems with Applications*, *55*, 194–211.

Chen, S., & Ge, L. (2019). Exploring the attention mechanism in LSTM-based Hong Kong stock price movement prediction. *Quantitative Finance*, *19*, 1507–1515.

Cheng, L.-C., Huang, Y.-H., & Wu, M.-E. (2018). Applied attention-based LSTM neural networks in stock prediction. In *2018 IEEE international conference on big data* (pp. 4716–4718). IEEE.

Cho, K., Merriënboer, B. van., Bahdanau, D., & Bengio, Y. (2014). On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, eighth workshop on syntax, semantics and structure in statistical translation* (pp. 103–111).

de Oliveira Carosia, A. E., Coelho, G. P., & da Silva, A. E. A. (2021). Investment strategies applied to the Brazilian stock market: A methodology based on sentiment analysis with deep learning. *Expert Systems with Applications*, *184*, Article 115470.

Di Persio, L., & Honchar, O. (2016). Artificial neural networks architectures for stock price prediction: Comparisons and applications. *International Journal of Circuits, Systems and Signal Processing*, *10*, 403–413.

Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, *270*, 654–669.

Huang, Y., Shen, L., & Liu, H. (2019). Grey relational analysis, principal component analysis and forecasting of carbon emissions based on long short-term memory in China. *Journal of Cleaner Production*, *209*, 415–423.

Jiang, W. (2021). Applications of deep learning in stock market prediction: recent progress. *Expert Systems with Applications*, Article 115537.

Jin, Z., Yang, Y., & Liu, Y. (2020). Stock closing price prediction based on sentiment analysis and LSTM. *Neural Computing and Applications*, *32*, 9713–9729.

Köksal, A., & Özgür, A. (2021). Twitter dataset and evaluation of transformers for turkish sentiment analysis. In *2021 29th Signal processing and communications applications conference* (pp. 1–4). IEEE.

Lehmann, E. L., & Casella, G. (2006). *Theory of point estimation*. Springer Science & Business Media.

Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X., et al. (2019b). Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in Neural Information Processing Systems*, *32*, 5243–5253.

Li, N., Liu, S., Liu, Y., Zhao, S., & Liu, M. (2019a). Neural speech synthesis with transformer network. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 6706–6713).

Li, Y., Ni, P., & Chang, V. (2020). Application of deep reinforcement learning in stock trading strategies and stock forecasting. *Computing*, *102*, 1305–1322.

Liu, J., Lin, H., Liu, X., Xu, B., Ren, Y., Diao, Y., et al. (2019). Transformer-based capsule network for stock movement prediction. In *Proceedings of the first workshop on financial technology and natural language processing* (pp. 66–73).

Long, W., Lu, Z., & Cui, L. (2019). Deep learning-based feature engineering for stock price movement prediction. *Knowledge-Based Systems*, *164*, 163–173.

Malandri, L., Xing, F. Z., Orsenigo, C., Vercellis, C., & Cambria, E. (2018). Public mood–driven asset allocation: The importance of financial sentiment in portfolio management. *Cognitive Computation*, *10*, 1167–1176.

Mann, H. B., & Whitney, D. R. (1947). On a test of whether one of two random variables is stochastically larger than the other. *The Annals of Mathematical Statistics*, *135*, 50–60.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

Mnih, V., Heess, N., Graves, A., et al. (2014). Recurrent models of visual attention. In *Advances in neural information processing systems* (pp. 2204–2212).

Mohammdi Farsani, R., & Pazouki, E. (2021). A transformer self-attention model for time series forecasting. *Journal of Electrical and Computer Engineering Innovations*, *9*, 1–10.

Murphy, J. J. (1999). *Technical analysis of the financial markets: A comprehensive guide to trading methods and applications*. Penguin.

Myttenaere, A. De., Golden, B., Le Grand, B., & Rossi, F. (2016). Mean absolute percentage error for regression models. *Neurocomputing*, *192*, 38–48.

Nguyen, T. H., Shirai, K., & Velcin, J. (2015). Sentiment analysis on social media for stock movement prediction. *Expert Systems with Applications*, *42*, 9603–9611.

Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, L., Shazeer, N., Ku, A., et al. (2018). Image transformer. In *International conference on machine learning* (pp. 4055–4064). PMLR.

Picasso, A., Merello, S., Ma, Y., Oneto, L., & Cambria, E. (2019). Technical analysis and sentiment embeddings for market trend prediction. *Expert Systems with Applications*, *135*, 60–70.

Rajput, V., & Bobde, S. (2016). Stock market forecasting techniques: literature survey. *International Journal of Computer Science and Mobile Computing*, *5*, 500–506.

Selvin, S., Vinayakumar, R., Gopalakrishnan, E., Menon, V. K., & Soman, K. (2017). Stock price prediction using LSTM, RNN and CNN-sliding window model. In *2017 International conference on advances in computing, communications and informatics* (pp. 1643–1647). IEEE.

Sharpe, W. F. (1994). The sharpe ratio. *Journal of Portfolio Management*, *21*, 49–58.

Sohangir, S., Wang, D., Pomeranets, A., & Khoshgoftaar, T. M. (2018). Big data: Deep learning for financial sentiment analysis. *Journal of Big Data*, *5*, 1–25.

Sun, A., Lachanski, M., & Fabozzi, F. J. (2016). Trade the tweet: Social media text mining and sparse matrix factorization for stock market prediction. *International Review of Financial Analysis*, *48*, 272–281.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems* (pp. 3104–3112).

Tetko, I. V., Karpov, P., Van Deursen, R., & Godin, G. (2020). State-of-the-art augmented NLP transformer models for direct and single-step retrosynthesis. *Nature Communications*, *11*, 1–11.

Tsai, C.-F., & Hsiao, Y.-C. (2010). Combining multiple feature selection methods for stock prediction: Union, intersection, and multi-intersection approaches. *Decision Support Systems*, *50*, 258–269.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998–6008).

Willmott, C. J., & Matsuura, K. (2005). Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance. *Climate Research*, *30*, 79–82.

Xi, E., Bing, S., & Jin, Y. (2017). Capsule network performance on complex data. arXiv preprint arXiv:1712.03480.

Xing, F. Z., Cambria, E., & Zhang, Y. (2019). Sentiment-aware volatility forecasting. *Knowledge-Based Systems*, *176*, 68–76.

Zhang, K., Zhong, G., Dong, J., Wang, S., & Wang, Y. (2019). Stock market prediction based on generative adversarial network. *Procedia Computer Science*, *147*, 400–406.

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., et al. (2021). Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of AAAI*.