Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20)
Special Track on AI in FinTech

Hierarchical Multi-Scale Gaussian Transformer for Stock Movement Prediction

Qianggang Ding[1,2,*], **Sifan Wu**[2,*], **Hao Sun**[3], **Jiadong Guo**[1] and **Jian Guo**[1,†]
[1]Peng Cheng Laboratory
[2]Tsinghua University
[3]The Chinese University of Hong Kong
{dqg18, wusf18}@mails.tsinghua.edu.cn, sh018@ie.cuhk.edu.hk, {guojd, guoj}@pcl.ac.cn

Abstract

Predicting the price movement of finance securities like stocks is an important but challenging task, due to the uncertainty of financial markets. In this paper, we propose a novel approach based on the Transformer to tackle the stock movement prediction task. Furthermore, we present several enhancements for the proposed basic Transformer. Firstly, we propose a *Multi-Scale Gaussian Prior* to enhance the locality of Transformer. Secondly, we develop an *Orthogonal Regularization* to avoid learning redundant heads in the multi-head self-attention mechanism. Thirdly, we design a *Trading Gap Splitter* for Transformer to learn hierarchical features of high-frequency finance data. Compared with other popular recurrent neural networks such as LSTM, the proposed method has the advantage to mine extremely long-term dependencies from financial time series. Experimental results show our proposed models outperform several competitive methods in stock price prediction tasks for the NASDAQ exchange market and the China A-shares market.

1 Introduction

With the development of stock markets all around the world, the overall capitalization of stock markets worldwide has exceed 68 trillion U.S. dollar by 2018[1]. Recent years, more and more quantitative researchers get involved in predicting the future trends of stocks, and they help investors make profitable decisions using state-of-the-art trading strategies. However, the uncertainty of stock prices make it an extremely challenging problem in the field of data science.

Prediction of stock price movement belongs to the area of time series analysis which models rich contextual dependencies using statics or machine learning methods. Traditional approaches for stock price prediction are mainly based on fundamental factors technical indices or statistical time series models, which captures explicit or implicit patterns from historical financial data. However, the performance of those methods are limited by two aspects. Firstly, they usually require expertise in finance. Secondly, these methods only capture simple patterns and simple dependence structures of financial time series. With the rise of artificial intelligence technology, more and more researchers attempt to solve this problem using machine learning algorithms, such as SVM [Cortes and Vapnik, 1995], Nearest Neighbors [Altman, 1992], Random Forest [Breiman, 2001]. Recently, since deep neural networks empirically exhibited its powerful capabilities in solving highly uncertain and nonlinear problems, the stock prediction research based on deep learning technique has become more and more popular in recent years and show significant advantages over traditional approaches.

The stock prediction research based on deep learning technique can roughly be grouped to two categories: (1) *Fundamental analysis*, and (2) *Technical analysis*. Fundamental analysis constructs prediction signals using fundamental information such as news text, finance report and analyst report. For example, [Schumaker and Chen, 2009; Xu and Cohen, 2018; Chen *et al.*, 2019] use natural language processing approaches to predict stock price movement by extracting latent features from market-related texts information,such as news, reports, and even rumors. On the other hand, technical analysis predicts finance market using historical data of stocks. One natural choice is the RNN family, such as RNN [Rumelhart *et al.*, 1986], LSTM [Hochreiter and Schmidhuber, 1997], Conv-LSTM [Xingjian *et al.*, 2015], and ALSTM [Qin *et al.*, 2017]. However, the primary drawback of these methods is that RNN family struggles to capture extremely long-term dependencies [Li *et al.*, 2019], such as the dependencies across several months on financial time series.

Recently, a well-known sequence-to-sequence model called Transformer [Vaswani *et al.*, 2017] has achieved great success on natural machine translation tasks. Distinct from RNN-based models, Transformer employs a *multi-head self-attention mechanism* to learn the relationship among different positions globally, thereby the capacity of learning long-term dependencies is enhanced. Nevertheless, canonical Transformer is designed for natural language tasks, and therefore it has a number of limitations in tackling finance prediction: (1) *Locality imperception*: the global self-attention mecha-

[*]Equal contribution.
[†]Corresponding author.
[1]https://data.worldbank.org/indicator/CM.MKT.TRAD.CD?end=2018&start=2018&view=bar

nism in canonical Transformer is insensitive to local context, whose dependencies are much important in financial time series. (2) *Hierarchy poverty*: the point-wise dot-product self-attention mechanism lacks the capability of utilizing hierarchical structure of financial time series (*e.g.* learning intra-day, intra-week and intra-month features in financial time series independently). Intuitively, addressing those drawbacks will improve the robustness of the model and lead to better performance in the task of financial time series prediction.

In this paper, we propose a new Transformer-based method for stock movement prediction. The primary highlight of the proposed model is the capability of capturing long-term, short-term as well as hierarchical dependencies of financial time series. For these aims, we propose several enhancements for the Transformer-based model: (1) *Multi-Scale Gaussian Prior* enhances the locality of Transformer. (2) *Orthogonal Regularization* avoids learning redundant heads in the multi-mead self-attention mechanism. (3) *Trading Gap Splitter* enables Transformer to learn intra-day features and intra-week features independently. Numerical results comparing with other competitive methods for time series show the advantages of the proposed method.

In summary, the main contributions of our paper include:

- We propose a Transformer-based method for stock movement prediction. To the best of our knowledge, this is the first work using Transformer model to tackle financial time series forecasting problems.
- We propose several enhancements for Transformer model include *Multi-Scale Gaussian Prior*, *Orthogonal Regularization*, and *Trading Gap Splitter*.
- In experiments, the proposed Transformer-based method significantly outperform several state-of-the-art baselines, such as CNN, LSTM and ALSTM, on two real-world exchange market.

2 Related Work

Fundamental Analysis Machine learning for fundamental analysis developed with the explosion of finance alternative data, such as news, location, e-commerce. [Schumaker and Chen, 2009] proposes a predictive machine learning approach for financial news articles analysis using several different textual representations. [Weng *et al.*, 2017] outlines a novel methodology to predict the future movements in the value of securities after tapping data from disparate sources. [Xu and Cohen, 2018] uses a stochastic recurrent model (SRM) with an extra discriminator and an attention mechanism to address the adaptability of stock markets. [Chen *et al.*, 2019] proposes to learn event extraction and stock prediction jointly.

Technical Analysis On the other hand, technical analysis methods extract price-volume information from historical trading data and use machine learning algorithms for prediction. For instances, [Lin *et al.*, 2013] proposes an SVM-based approach for stock market trend prediction. Meanwhile, LSTM neural network [Hochreiter and Schmidhuber, 1997] has been employed to model stock price movement. [Nelson *et al.*, 2017] proposes an LSTM model to predict stock movement based on the technical analysis indicators. [Zhang *et al.*, 2017] proposes an LSTM model on historical data to discover multi-frequency trading. [Wang *et al.*, 2019] proposes a ConvLSTM-based Seq2Seq framework for stock movement prediction. [Qin *et al.*, 2017] proposes an Attentive-LSTM model with an attention mechanism to predict stock price movement and [Feng *et al.*, 2019] further introduces an data augmentation approach with the idea of adversarial training. However, [Li *et al.*, 2019] points out that LSTM can only distinguish 50 positions nearby with an effective context size of about 200. That means that LSTM-based models suffer from the difficulty in capturing extremely long-term dependencies in time series. To tackle this issue, we propose a Transformer-based method to better mine the intrinsic long-term and complex structures in financial time series.

3 Problem Formulation

Since the exact price of a stock is extremely hard to be predicted accurately, we follow the setup of [Walczak, 2001] and predict the stock price movement instead. Usually the stock movement prediction is treated as a binary classification problem — *e.g.*, discretizing the stock movement into two classes (Rise or Fall). Formally, given the stock features $\mathbf{X} = [\mathbf{x}_{T-\Delta t+1}, \mathbf{x}_{T-\Delta t+2}, ..., \mathbf{x}_T] \in \mathbb{R}^{\Delta t \times F}$ (also represented as $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N] \in \mathbb{R}^{N \times F}$ in the rest of the paper for simplicity) in the latest Δt time-steps, the prediction model $f_\theta(\mathbf{X})$ with parameters θ can output the predicted movement label $y = \mathbb{I}(p_{T+k} > p_T)$, where T denotes the target trading time, F denotes the dimension of stock features and p_t denotes the close price at time-step t. Briefly, the proposed model utilizes the historical data of a stock s in the lag $[T - \Delta t + 1, T]$ (where Δt is a fixed lag size) to predict the movement class y (0 for Fall, 1 for Rise) of the future k time-steps.

4 Proposed Method

In this section, we first describe the basic Transformer model we designed. Then we introduce the proposed enhancements of Transformer for financial time series.

4.1 Basic Transformer for Stock Movement Prediction

In our work, we instantiate $f_\theta(\cdot)$ with Transformer-based model. To adapt the stock movement prediction task which takes time series as inputs, we design a variant of Transformer with encoder-only structure which consists of $\mathbf{L}$ blocks of multi-head self-attention layers and position-wise feed forward layers (see Figure 1). Given the input time series $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N] \in \mathbb{R}^{N \times F}$, we first add the position encoding and adopt an linear layer with $tanh$ activation function as follows:

$$\bar{\mathbf{X}} = \sigma_{tanh}\mathbf{W}_h^{(I)}[\text{PositionEncoding}(\mathbf{X})]. \quad (1)$$

Then multi-head self-attention layers take $\bar{\mathbf{X}}$ as input, and are computed by

$$\mathbf{Q}_h = \mathbf{W}_h^{(Q)}\bar{\mathbf{X}}, \quad \mathbf{K}_h = \mathbf{W}_h^{(K)}\bar{\mathbf{X}}, \quad \mathbf{V}_h = \mathbf{W}_h^{(V)}\bar{\mathbf{X}}, \quad (2)$$

where $h = 1, ..., H$ and $\mathbf{W}_h^{(Q)}, \mathbf{W}_h^{(K)}$ and $\mathbf{W}_h^{(V)}$ are learnable weight matrices for Query, Key and Value, respectively

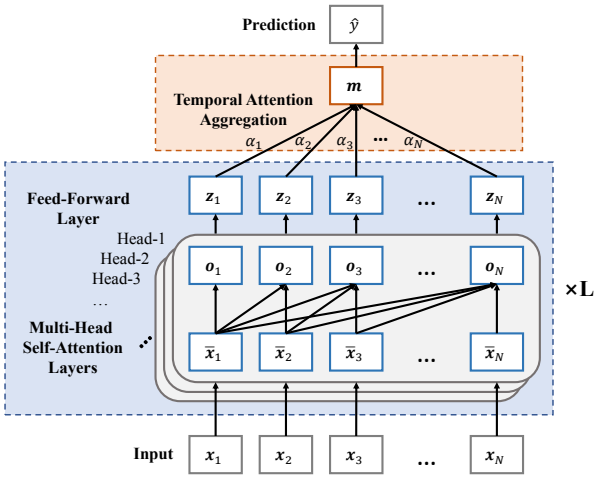


Figure 1: The proposed Basic Transformer overview.

(refers to [Vaswani *et al.*, 2017] for more details). Then the attention score matrix $\mathbf{a}_h \in \mathbb{R}^{N\times N}$ of the h^{th} head is computed by

$$\mathbf{a}_h = \text{softmax}(\frac{\mathbf{Q}_h\mathbf{K}_h^T}{\sqrt{d_k}} \cdot \mathbf{M}), \tag{3}$$

where $\mathbf{M}$ is a position-wise mask matrix to filter out temporal attention, so as to avoid future information leakage. Afterwards, the output of the h^{th} head is a weighted sum defined as follows:

$$[\mathbf{O}_h]_i = \sum_{j=1}^{N}(\mathbf{a}_h)_{i,j} \cdot [\mathbf{V}_h]_j. \tag{4}$$

The final output of multi-head attention layers is the concatenation of all heads by $\mathbf{O} = [\mathbf{O}_1, \mathbf{O}_2, ..., \mathbf{O}_H]$. Afterward, the position-wise feed forward layer takes $\mathbf{O}$ as input and transforms it to $\mathbf{Z}$ by two fully-connected layers and a ReLU activation layer. Upon the output $\mathbf{z}_i$ of the last self-attention layer, a temporal attention layer [Qin *et al.*, 2017] is deployed to aggregate the latent features from each position as $\mathbf{m} = \sum_{i=1}^{N}\alpha_i\mathbf{z}_i$. Then the scalar prediction score $\hat{y}$ is computed by a fully-connected layer and a sigmoid transformation:

$$\hat{y} = \text{sigmoid}(\mathbf{W}_{fc}\mathbf{m}). \tag{5}$$

Our ultimate goal is to maximize the log-likelihood between $\hat{y}$ and y via the following loss function:

$$\mathcal{L}_{CE} = (1-y)\log(1-\hat{y}) + y\log(\hat{y}) \tag{6}$$

4.2 Enhancing Locality with Multi-Scale Gaussian Prior

Recently, Transformer exhibits its powerful capability of extracting global patterns in natural language processing fields. However, the self-attention mechanism in Transformer considers the global dependencies with very weak position information. Note that, the position information serves as the temporal variant patterns in time series, which is much important. To address it, we incorporate *Multi-Scale Gaussian Prior* into the canonical multi-head self-attention mechanism

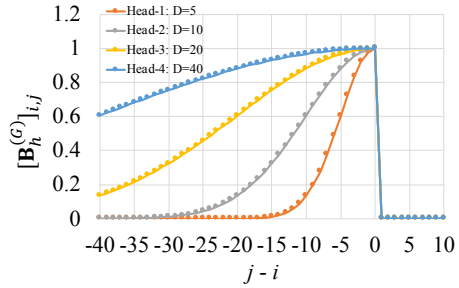


Figure 2: Visualization of $[\mathbf{B}_h^{(G)}]_{i,j}$ in Eq. 8 with window-size set $\mathbf{D} = \{5, 10, 20, 40\}$.

with the intuition that the relevance of data in two positions is directly proportional to the temporal distance between them.

To pay more attention to the closer time-steps, we add biases of Gaussian prior to the attention score matrices based on the assumption that such scores would obey Gaussian distributions. Note that this operation is equivalent to multiplying the original attention distribution with a Gaussian distribution mask (See [Guo *et al.*, 2019] for the proof). In details, we transform Eq.3 to the following form by adding Gaussian biases:

$$\mathbf{a}_h = \text{softmax}[(\frac{\mathbf{Q}_h\mathbf{K}_h^T}{\sqrt{d_k}} + \mathbf{B}_h^{(G)}) \cdot \mathbf{M}], \tag{7}$$

where $\mathbf{B}_h^{(G)} \in \mathbb{R}^{N\times N}$ is a matrix computed by

$$[\mathbf{B}_h^{(G)}]_{i,j} = \begin{cases} \exp(-\frac{(j-i)^2}{2\sigma_h^2}) & j \le i; \\ 0 & j > i. \end{cases} \tag{8}$$

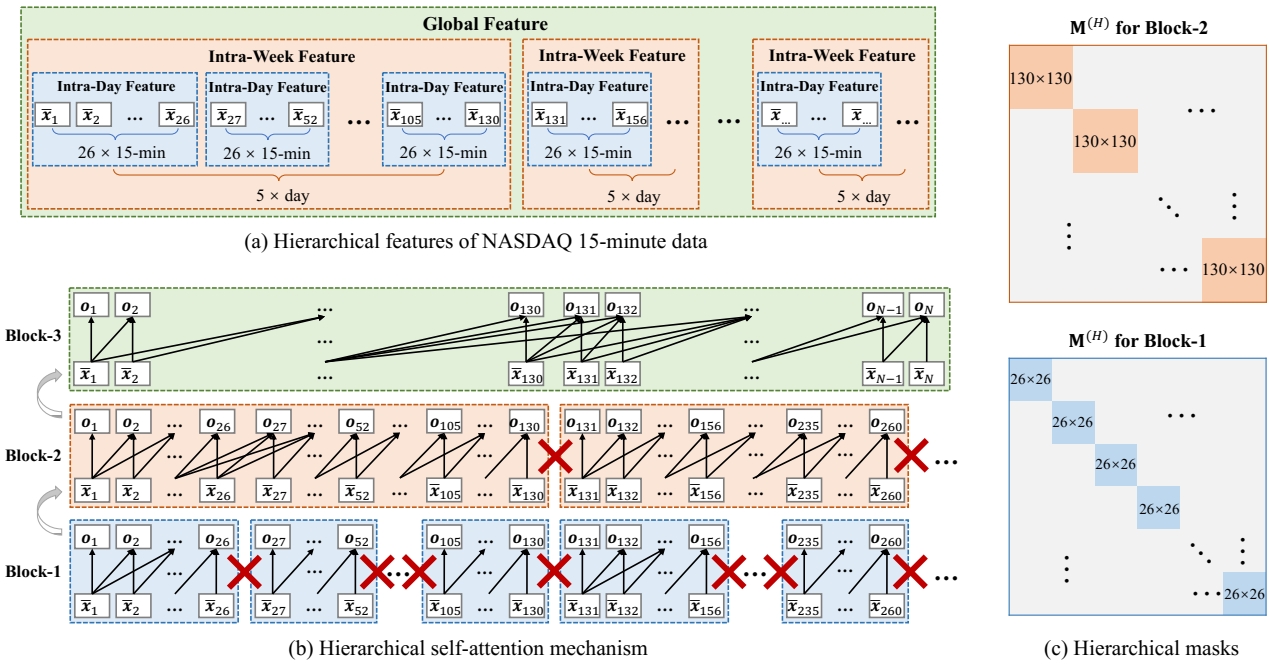
Note that we allow σ_h in $\mathbf{B}_h^{(G)}$ are different for different heads in the multi-head self-attention layer.

Besides, we also give an empirical approximation for σ_h. Suppose we want to pay more attention to the D_h closest time-steps, the variance can be empirically set as $\sigma_h = D_h$. By this way, we allow different D_h in different attention heads in order to provide *Multi-Scale Gaussian Prior*.

In finance, the temporal features from last 5-day, 10-day, 20-day or 40-day are usually considered in trading strategies. That means, for a 4-head self-attention layer, we can empirically assign the window-size set $\mathbf{D} = \{5, 10, 20, 40\}$ to σ_h with $h = 1, ..., 4$, respectively as is shown in Figure 2. In conclusion, the proposed Multi-Scale Gaussian Prior enables Transformer to learn multi-scale localities from financial time series.

4.3 Orthogonal Regularization for Multi-Head Self-Attention Mechanism

With the proposed *Multi-Scale Gaussian Prior*, we let different heads learn different temporal patterns in the multi-head attention layer. However, some previous research [Tao *et al.*, 2018][Li *et al.*, 2018][Lee *et al.*, 2019] claims that canonical multi-head self-attention mechanism tend to learn redundant heads. To enhance the diversity between each head, we induce an orthogonal regularization with regard to the weight tensor $\mathbf{W}_h^{(V)}$ in Eq.2. Specifically, we first calculate the



(a) Hierarchical features of NASDAQ 15-minute data

(b) Hierarchical self-attention mechanism

(c) Hierarchical masks

Figure 3: Trading Gap Splitter overview.

tensor $\mathbf{W}^{(V)} = [\mathbf{W}_1^{(V)}, \mathbf{W}_2^{(V)}, ..., \mathbf{W}_H^{(V)}]$ by concatenating $\mathbf{W}_h^{(V)}$ of all heads. Note that the size of $\mathbf{W}^{(V)}$ is $H \times F \times d_v$ where d_v denotes the last dimension of $\mathbf{V}_h$. Then we flatten the tensor $\mathbf{W}^{(V)}$ to a matrix $\mathbf{A}$ with size of $H \times (F * d_v)$, and further normalize it as $\widetilde{\mathbf{A}} = \mathbf{A}/||\mathbf{A}||_2$. Finally, the penalty loss is computed by

$$\mathcal{L}_p = ||\widetilde{\mathbf{A}}\widetilde{\mathbf{A}}^T - I||_F, \tag{9}$$

where $|| \circ ||_F$ denotes the Frobenius norm of a matrix and I stands for an identity matrix. We briefly add the penalty loss to the original loss with a trade-off hyper-parameter γ as follows:

$$\mathcal{L} = \mathcal{L}_{CE} + \gamma \mathcal{L}_p. \tag{10}$$

For simplicity in expression, here we omit the number of multi-head self-attention layers in the model. In our experiment, we sum up the penalty losses from each multi-head self-attention layer as the final penalty loss as follows:

$$\mathcal{L}_p = \mathcal{L}_p^{(1)} + \mathcal{L}_p^{(2)} + ... + \mathcal{L}_p^{(L)}. \tag{11}$$

4.4 Trading Gap Splitter

As is known that the input of the model is a continuous time series. However, due to the trading gaps, the input time series is essentially NOT continuous. Takes the 15-minute data from NASDAQ stock market[2] as an example, of which one trading day contains 26 15-minute time-steps and one trading week contains 5 trading days. This means there are inter-day and inter-week trading gaps. However, when the basic Transformer model is applied to this data, the self-attention layer inside treats all time-steps equally and omit the implicit inter-day and inter-week trading gaps. To solve this problem, we design a new hierarchical self-attention mechanism for the Transformer model to learn the hierarchical features of stock data (see Figure 3 (a)).

Takes a 3-block Transformer model as an example, we aim to learn the hierarchical features of stock data by the order "intra-day→intra-week→global". In order to do so, we set two extra position-wise masks to the first and second self-attention blocks respectively in order to limit the attention scopes. Formally, we modify Eq.7 to the following form:

$$\mathbf{a}_h = \text{softmax}[(\frac{\mathbf{Q}_h\mathbf{K}_h^T}{\sqrt{d_k}} + \mathbf{B}_h^{(G)}) \cdot \mathbf{M}^{(H)} \cdot \mathbf{M}], \tag{12}$$

where $\mathbf{M}^{(H)}$ is an $N \times N$ matrix filled with $-\inf$ whose diagonal is composed of continuous sub-matrices filled with 0. The $\mathbf{M}^{(H)}$ for the first and second self-attention blocks are shown in Figure 3 (c). Specifically, the size of sub-matrices in $\mathbf{M}^{(H)}$ for the first block is 26×26 since one trading day contains 26 15-minute time-steps, and the size of sub-matrices for the second block changes to 130×130 $(26 * 5)$ since one trading week contains 5 trading days. By this way, the first and second self-attention blocks will learn the intra-day and intra-week features of stock data, respectively. Moreover, for the last self-attention block, we keep the original attention mechanism without $\mathbf{M}^{(H)}$ to learn global features of stock data. As a result, the Transformer model with the proposed hierarchical attention mechanism avoids suffering from the trading gaps. Note that, all attention heads in the same multi-head self-attention layer share the same $\mathbf{M}^{(H)}$.

[2]the NASDAQ Stock Exchange Market is open 5 days per week for 6.5 hours per day

Property	Data	
	Daily	15-min
Market	NASDAQ	China A-shares
Start Date	2010/07/01	2015/12/01
End Date	2019/07/01	2019/12/01
Time Frequency	1 day	15 minutes
Total Stocks	3243	500
Total Records	9749098	7928000
Rising Threshold β_{rise}	0.55%	-0.5%
Falling Threshold β_{fall}	-0.1%	0.105%

Table 1: Data description.

5 Experiments

To evaluate the proposed methods, we use two stock data: one from NASDAQ market and the other from China A-shares market. The details of the two data are listed in Table 1. In the following subsections, we will introduce the data collection process and show our empirical results from numerical experiments. We also conduct an incremental analysis to explore the effectiveness of each proposed enhancements for Transformer.

5.1 Data Collection

We collect the daily quote data of all 3243 stocks from NASDAQ stock market from July 1st, 2010 to July 1st, 2019 and the 15-min quote data of 500 CSI-500 component stocks from China A-shares market from December 1st, 2015 to December 1st, 2019. We move a lag window with size of N time-steps along these time series to construct candidate examples. For the NASDAQ data, we construct 5 datasets with window sizes $N = 5, 10, 20, 40$ (denoting 5-, 10-, 20-, 40-day, respectively), and the lag strides are all fixed to 1. For the China A-shares data, we also construct 5 datasets with window sizes $N = 5 * 16, 10 * 16, 20 * 16, 40 * 16$ (denoting 5-, 10-, 20-, 40-day since one trading day contains 16 15-minute in China A-shares market), and the lag strides are all fixed to 16 (*i.e.* 1 day). The features used in all datasets consist of $\mathrm{open}, \mathrm{high}, \mathrm{low}, \mathrm{close}$ and volume, which are all adjusted and normalized following [Feng *et al.*, 2019; Xu and Cohen, 2018]. We initially label both datasets by the strategy mentioned in Section 3.1 (*i.e* $y = \mathbb{I}(p_{T+k} > p_T)$), where k is set to 1 and 16 (both denote 1 day) for the NASDAQ data and the China A-shares data, respectively. Furthermore, we set two threshold parameter $\beta_{rise}, \beta_{fall}$ to the labels in each dataset in order to balance the number of positive and negative samples to roughly $1:1$ as follows:

$$y = \begin{cases} 1 & \frac{p_{T+k}-p_T}{p_T} > \beta_{rise}; \\ -1 & \frac{p_{T+k}-p_T}{p_T} < \beta_{fall}; \\ \text{abandon} & \text{otherwise.} \end{cases} \tag{13}$$

To avoid the data leakage problem, we strictly follow the sequential order to split training/validation/test sets. For instances, we split the NASDAQ data and the China A-shares data into training/validation/test sets by 8-year/1-year/1-year and 3-year/6-month/6-month, respectively.

Method	Accuracy (%)/MCC ($\times 10^{-2}$) with window size of K-day			
	$K = 5$	$K = 10$	$K = 20$	$K = 40$
NASDAQ Daily Data				
CNN	52.33/3.16	52.02/2.68	51.84/2.28	52.60/2.52
LSTM	53.86/7.73	53.89/7.72	53.59/7.15	53.81/7.48
ALSTM	54.06/8.35	53.94/7.92	54.05/8.11	54.19/8.56
B-TF†	54.78/8.48	54.84/8.89	54.90/9.13	56.01/9.45
MG-TF†	**55.10/8.98**	**56.18/9.74**	**56.77/10.39**	**57.30/11.46**
China A-shares 15-min Data				
CNN	53.53/2.62	52.25/1.80	52.03/1.81	51.61/1.77
LSTM	56.59/6.42	56.70/6.19	56.18/3.74	54.93/2.98
ALSTM	57.03/8.23	57.42/9.16	55.69/6.65	55.68/6.65
B-TF†	57.14/9.68	57.42/9.52	57.32/9.14	57.55/10.41
HMG-TF†	**57.36/10.52**	**57.79/9.98**	**57.90/10.33**	**58.70/14.87**

Table 2: The results of comparison experiments. All values are average results from 5 repeated experiments. The best results are in bold and † denotes our methods.

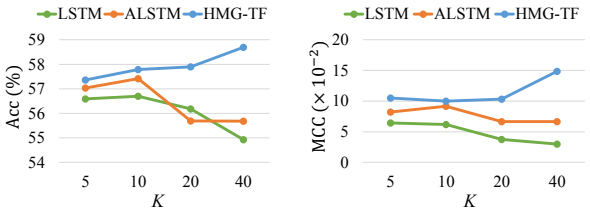


Figure 4: The accuracy and MCC trends along with the window size K-day on the China A-shares data.

5.2 Evaluation Metrics

Following previous research [Feng *et al.*, 2019; Xu and Cohen, 2018], we evaluate the prediction performance with two metrics: Accuracy (Acc) and Matthews Correlation Coefficient (MCC), which is defined as below:

$$\mathrm{MCC} = \frac{\mathrm{tp} \times \mathrm{tn} - \mathrm{fp} \times \mathrm{fn}}{\sqrt{(\mathrm{tp}+\mathrm{fp})(\mathrm{tp}+\mathrm{fn})(\mathrm{tn}+\mathrm{fp})(\mathrm{tn}+\mathrm{fn})}} \tag{14}$$

where $\mathrm{tp}, \mathrm{tn}, \mathrm{fp}, \mathrm{fn}$ denote the number of samples classified as true positive, true negative, false positive and false negative, respectively.

5.3 Numerical Experiments

Approaches in comparison. We compare our approaches B-TF, MG-TF and HMG-TF with the baselines CNN, LSTM and ALSTM:

- CNN [Selvin *et al.*, 2017]: Here we use 1D-CNN with the kernel size of 1×3.
- LSTM [Nelson *et al.*, 2017]: A variant of recurrent neural network with feedback connections.
- Attentive LSTM (ALSTM) [Feng *et al.*, 2019]: A variant of LSTM model with a temporal attentive aggregation layer.
- Basic Transformer (B-TF): The proposed Basic Transformer model introduced in Section 4.1.

Method	Settings			Metrics	
	MG	OR	TS	Acc. (%)	MCC ($\times 10^{-2}$)
B-TF	✗	✗	✗	57.55	10.41
MG-TF*	✓	✗	✗	58.03	12.11
MG-TF	✓	✓	✗	58.25	12.75
HMG-TF	✓	✓	✓	**58.70**	**14.87**

Table 3: Experimental results of incremental analysis. **MG**, **OR** and **TS** denote *Multi-Scale Gaussian Prior*, *Orthogonal Regularization* and *Trading Gap Splitter*, respectively. * denotes wo/ **OR**.

- Multi-Scale Gaussian Transformer (MG-TF): The B-TF model with enhancements of *Multi-Gaussian Prior* and *Orthogonal Regularization* introduced in Section 4.2 and Section 4.3, respectively.
- Hierarchical Multi-Scale Gaussian Transformer (HMG-TF): The MG-TF model with the enhancement of *Trading Gap Splitter* introduced in Section 4.4.

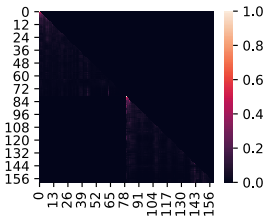
Settings. We implement B-TF, MG-TF and HMG-TF with PyTorch framework on Nvidia Tesla V100 GPU. We use Adam optimizer with an initial learning-rate of $1e-4$. The size of mini-batch is set to 256. The trade-off hyper-parameter γ is set to 0.05. All TF-based models have 3 multi-head self-attention blocks each with 4 heads. We train the model in an end-to-end manner from raw quote data without any data augmentation. Since *Trading Gap Splitter* is less appropriate for low-frequency data like daily quote data, we only perform MG-TF on our NASDAQ dataset.

Results. The performance of comparison experiments on both data are shown in Table 2, from which we have the following observations:

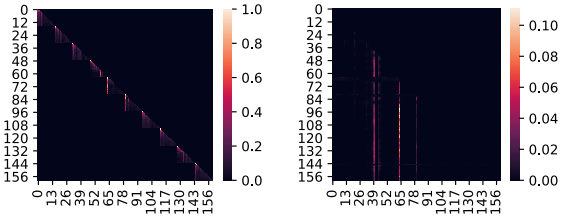
- The proposed approaches B-TF, MG-TF and HMG-TF show significant gains on both metrics Acc and MCC compared with baselines in all cases. It exhibits that Transformer-based approaches have significant performance advantages over RNN- and CNN-based approaches.
- Transformer-based approaches have better capabilities of learning long-term dependencies. Especially on China A-shares data, in which the window size of 40-day comtains 640 (40*16) time-steps, and it is hard for RNN-based approaches to learn the dependencies across so many time-steps (see Figure 4). While the advantages of the self-attention mechanism enables Transformer-based approaches achieve consistently better performance as the window size becomes larger.
- The modified Transformer model MG-TF and HMG-TF have better performance than the basic Transformer model. More detailed analysis will be shown in the next section.

5.4 Incremental Analysis

To explore the effectiveness of the proposed components *Multi-Scale Guassian Prior*, *Orthogonal Regularization* and *Trading Gap Splitter*, we further conduct an incremental analysis on different settings of the Transformer-based models.



(a) Daily attention scores



(b) Weekly attention scores (c) Global attention scores

Figure 5: Hierarchical attention scores. (a)(b)(c) represents the attention score matrices learned by Block-1, Block-2 and Block-3, respectively.

As is shown in Table 3, these components all contribute to the performance of Transformer-based method. Moreover, we can further observe that the performance improvement mainly benefits from *Multi-Scale Guassian Prior* and *Trading Gap Splitter*, while the gain from *Orthogonal Regularization* is relatively insignificant.

Besides, we illustrate the effectiveness of *Trading Gap Splitter* by visualizing the attention score matrix learned by HMG-TF model with $N = 160$ (is equivalent to $K = 10$-day or 2-week). As is shown in Figure 5, the attention scope becomes larger gradually (16→80→160) from (a) to (c). That means, with the proposed hierarchical self-attentiom mechanism, the time-steps only attend to those belonging to the same **day** in the first self-attention block, the time-steps only attend to those belonging to the same **week** in the second block, and the attention scopes of time-steps are no limit in the last block.

6 Discussion & Future Work

In this paper, we propose to apply Transformer model for stock movement prediction in which the attention mechanism can help to capture extremely long-term dependencies of finance time series. Furthermore, equipped with the proposed enhancements *Multi-Scale Gaussian Prior*, *Orthogonal Regularization* and *Trading Gap Splitter*, our Transformer-based model achieves significant gains over several state-of-the-art baselines on two real-world market dataset. In the future, except for the model itself, the following aspects can be investigated for further improvements: (1) cross-sectional features of financial data can be engaged to improve the model, (2) regularization methods can be explored to avoid suffering from overfitting on financial data, (3) data augmentation such as adversarial and stochastic perturbations can be adopted to improve the robustness of the model. Also, it is significant to investigate the theoretical guarantee for our method.

References

[Altman, 1992] Naomi S Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[Breiman, 2001] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[Chen *et al.*, 2019] Deli Chen, Yanyan Zou, Keiko Harimoto, Ruihan Bao, Xuancheng Ren, and Xu Sun. Incorporating fine-grained events in stock movement prediction. *arXiv preprint arXiv:1910.05078*, 2019.

[Cortes and Vapnik, 1995] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[Feng *et al.*, 2019] Fuli Feng, Huimin Chen, Xiangnan He, Ji Ding, Maosong Sun, and Tat-Seng Chua. Enhancing stock movement prediction with adversarial training. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 5843–5849. AAAI Press, 2019.

[Guo *et al.*, 2019] Maosheng Guo, Yu Zhang, and Ting Liu. Gaussian transformer: a lightweight approach for natural language inference. In *The Thirty-Third AAAI Conference on Artificial Intelligence*, 2019.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Lee *et al.*, 2019] Mingu Lee, Jinkyu Lee, Hye Jin Jang, Byeonggeun Kim, Wonil Chang, and Kyuwoong Hwang. Orthogonality constrained multi-head attention for keyword spotting. *arXiv preprint arXiv:1910.04500*, 2019.

[Li *et al.*, 2018] Jian Li, Zhaopeng Tu, Baosong Yang, Michael R Lyu, and Tong Zhang. Multi-head attention with disagreement regularization. *arXiv preprint arXiv:1810.10183*, 2018.

[Li *et al.*, 2019] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In *Advances in Neural Information Processing Systems*, pages 5244–5254, 2019.

[Lin *et al.*, 2013] Yuling Lin, Haixiang Guo, and Jinglu Hu. An svm-based approach for stock market trend prediction. In *The 2013 international joint conference on neural networks (IJCNN)*, pages 1–7. IEEE, 2013.

[Nelson *et al.*, 2017] David MQ Nelson, Adriano CM Pereira, and Renato A de Oliveira. Stock market's price movement prediction with lstm neural networks. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 1419–1426. IEEE, 2017.

[Qin *et al.*, 2017] Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. *arXiv preprint arXiv:1704.02971*, 2017.

[Rumelhart *et al.*, 1986] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[Schumaker and Chen, 2009] Robert P Schumaker and Hsinchun Chen. Textual analysis of stock market prediction using breaking financial news: The azfin text system. *ACM Transactions on Information Systems (TOIS)*, 27(2):12, 2009.

[Selvin *et al.*, 2017] Sreelekshmy Selvin, R Vinayakumar, EA Gopalakrishnan, Vijay Krishna Menon, and KP Soman. Stock price prediction using lstm, rnn and cnn-sliding window model. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1643–1647. IEEE, 2017.

[Tao *et al.*, 2018] Chongyang Tao, Shen Gao, Mingyue Shang, Wei Wu, Dongyan Zhao, and Rui Yan. Get the point of my utterance! learning towards effective responses with multi-head attention mechanism. In *IJCAI*, pages 4418–4424, 2018.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[Walczak, 2001] Steven Walczak. An empirical analysis of data requirements for financial forecasting with neural networks. *Journal of management information systems*, 17(4):203–222, 2001.

[Wang *et al.*, 2019] Jia Wang, Tong Sun, Benyuan Liu, Yu Cao, and Hongwei Zhu. Clvsa: A convolutional lstm based variational sequence-to-sequence model with attention for predicting trends of financial markets. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 3705–3711. AAAI Press, 2019.

[Weng *et al.*, 2017] Bin Weng, Mohamed A Ahmed, and Fadel M Megahed. Stock market one-day ahead movement prediction using disparate data sources. *Expert Systems with Applications*, 79:153–163, 2017.

[Xingjian *et al.*, 2015] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015.

[Xu and Cohen, 2018] Yumo Xu and Shay B Cohen. Stock movement prediction from tweets and historical prices. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1970–1979, 2018.

[Zhang *et al.*, 2017] Liheng Zhang, Charu Aggarwal, and Guo-Jun Qi. Stock price prediction via discovering multi-frequency trading patterns. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 2141–2149. ACM, 2017.