Accurate Stock Movement Prediction via Multi-Scale and Multi-Domain Modeling

JinGee Kim
Seoul National University
Seoul, Republic of Korea
jingeekim9@snu.ac.kr

Yong-chan Park
Seoul National University
Seoul, Republic of Korea
wjdakf3948@snu.ac.kr

Jaemin Hong
Seoul National University
Seoul, Republic of Korea
jmhong0120@snu.ac.kr

U Kang
Seoul National University
Seoul, Republic of Korea
ukang@snu.ac.kr

***Abstract*—How can we utilize historical stock prices for accurate stock movement prediction? There have been several attempts to predict the movement of a stock using historical stock prices. However, due to the erratic nature of the stock market, it is difficult to accurately predict the future trajectory of stocks. Existing works are limited in capturing complex dependencies that exist within the stock data.**

In this paper, we propose ZoomStock, an accurate method for stock movement prediction. ZoomStock captures complex patterns in stock price data with multi-scale and multi-domain modeling. First, ZoomStock captures multi-scale patterns by examining short-term, mid-term, and long-term dependencies from data. By feeding multi-scale data into a 1D convolution layer and applying a circular convolution on data, we learn patterns in various scales. Second, ZoomStock leverages multi-domain features to extract diverse patterns from stock data. In addition to finding patterns in time domain, ZoomStock exploits patterns in the frequency domain via Fourier Transform. We conduct extensive experiments on real-world datasets from diverse stock markets to evaluate the effectiveness of ZoomStock. Our results demonstrate that ZoomStock outperforms previous state-of-the-art models by up to 15.7%p in accuracy and 31.7%p in MCC, showcasing its superior predictive capabilities.

***Index Terms*—stock price movement prediction, time series forecasting, discrete Fourier Transform, convolutions**

I. Introduction

How can we accurately predict stocks' price movement? Stock movement prediction continues to gather attention in the data mining and machine learning community because of its large impact on the financial domain [1]–[5]. Stock movement prediction is formulated as a binary classification problem where we predict whether a stock's price will either rise or fall [1], [2], [4], [6]. By accurately predicting a stock's future price movement, investors can potentially obtain large profits.

However, predicting the future trajectory of stocks accurately is challenging due to the erratic and nonstationary nature of the stock market [7]–[10]. To solve this, it is necessary to determine complex dependencies within stock data for precise price forecasting. There have been many previous attempts to try and find these complex dependencies for accurate stock movement prediction. Among these, attention-based models have demonstrated strong potential in stock market prediction [1], [4], [11]–[14]. These methods use the attention mechanism to capture both long and short-term dependencies within the stock price and to find relationships between features such as sectors. However, attention-based methods fall short when capturing extremely long dependencies that exist across several months of financial data. In addition, they fail to utilize both multi-scale and multi-domain patterns available in the price data. Other methods, such as using Convolutional Neural Network (CNN) models [15]–[19] or finding patterns in the frequency domain [20]–[24], have also shown promising results. However, they either cannot adequately find both long and short-term dependencies within the data or consider dependencies only within a limited range.

In this paper, we propose ZoomStock, an accurate method for stock movement prediction. ZoomStock solves the stock movement prediction problem by multi-scale and multi-domain modeling. First, ZoomStock captures short-term, mid-term, and long-term dependencies from the stock data using convolutions. Short- and mid-term dependencies are extracted using a 1D CNN with varying sampling periods of stock data, while global patterns are found using a circular convolution. Second, ZoomStock uses features from the frequency domain to capture characteristics of the stock signal that are not available in the time domain. We use a Fast Fourier Transform to convert the stock signal in the time domain to that in the frequency domain, and concatenate the coefficients with the other hidden features to generate rich features helpful for prediction. This multi-scale and multi-domain approach allows ZoomStock to effectively capture the complex dynamics of the stock market. Empirical evaluations demonstrate that ZoomStock consistently outperforms existing state-of-the-art methods in stock movement forecasting.

We summarize our main contributions as follows:

- **Multi-scale and multi-domain modeling.** We propose ZoomStock, an accurate method that extracts multi-scale and multi-domain features from the stock data.
- **Theory.** We theoretically analyze the computational complexity and the number of parameters of ZoomStock.
- **Performance.** ZoomStock outperforms the state-of-the-art baseline models on six real-world datasets collected from various stock markets, improving the accuracy and Matthews correlation coefficient by up to 15.7%p and 31.7%p, respectively.

The rest of the paper is organized as follows. First, we provide an overview of the preliminaries and related works on stock movement prediction in Section II. Next, we describe

TABLE I
TABLE OF SYMBOLS

Symbol	Description
$\mathcal{S}$	Set of target stocks
$\mathcal{T}$	Set of available training days
X^s	Feature matrix for stock s
$\tilde{X}^s$	Daily-sampled feature matrix for stock s
$\hat{X}^s$	Weekly-sampled feature matrix for stock s
$\mathbf{x}_t^s$	Input features for stock s at time t
$e^{-i2\pi k(\cdot)/N}$	k-th Fourier basis of length N
D	Dimension of features
w	Window size
k	Size of convolution kernels
d	Latent dimension of the attention layer
c	Channel
$\tilde{h}^s$	Output of 1D Convolution on daily-sampled data
$\hat{h}^s$	Output of 1D Convolution on weekly-sampled data
$\hat{g}^s$	Output of circular convolution
z^s	Concatenated feature matrix for stock s
$\tilde{z}^s$	Feature vector for stock s after average pooling
$\hat{z}^s$	Hidden features after feature transformation

the motivation behind the research and present the main ideas of ZoomStock in Section III. Then, we show the experimental results of ZoomStock in Section IV and conclude the paper in Section V. Table I summarizes the symbols used in the paper.

II. PRELIMINARIES AND RELATED WORKS

We describe the problem definition, preliminaries, and related works on stock movement prediction.

A. Problem Definition

We formally define the problem of stock movement prediction. The input consists of historical stock prices denoted as $\{\mathbf{x}_{st}\}_{s\in\mathcal{S},t\in\mathcal{T}}$, where $\mathcal{S}$ represents the set of target stocks, $\mathcal{T}$ represents the set of available training days, and $\mathbf{x}_{st}$ denotes the input features (*e.g.*, open and close prices) for stock s at time t. The task is to predict the binary movement of each stock price at time $T+1$ given the historical price data up to time T [2], [25].

B. Related Works

We introduce related works on stock movement prediction which attempt to effectively find the complex dependencies within the stock data.

Attention-based methods. Numerous models utilize the attention mechanism [26] to address the stock movement prediction problem. Attention-based models find both short-term and long-term dependencies by attending to every single data point. Qin et al. [4] proposed a dual-stage attention-based recurrent neural network for stock movement prediction. The dual-stage attention is divided into an input attention mechanism that extracts relevant driving series and a temporal attention mechanism to select relevant encoder hidden states to increase the prediction accuracy. Feng et al. [2] includes an adversarial training method that stabilizes the training of the attention-based LSTM model and prevents overfitting. Li et al. [27] proposed a multi-input attention-based LSTM model that makes correlations between distant time steps to prevent the features of the last time step from dominating the prediction.

Ding et al. [12] took the original Transformer model [28] and introduced a multi-scale Gaussian prior to enhance the locality of the model and a trading gap splitter to address the challenges posed by both intra-day and intra-week financial data features. Yoo et al. [1] proposed the DTML model which consists of a time-axis attention module and a data-axis attention module. The time-axis attention module finds temporal correlations for each stock and the data-axis attention module uses a Transformer encoder to learn inter-stock correlations. However, these attention-based methods often struggle to capture the full spectrum of dependencies, neglecting either short-term fluctuations or long-term trends, and typically focus on correlations within a restricted range. Moreover, their reliance solely on the time domain restricts the diversity of features that can be extracted for comprehensive analysis.

CNN and multi-frequency methods. Convolutional Neural Network (CNN) models have also been used in combination with LSTM models to increase the accuracy of the model. Zhang et al. [15] proposed a CNN-BiLSTM-Attention model where the CNN identifies local features within the data, the BiLSTM learns bidirectional sequential features, and the attention mechanism assigns greater weights to the more important feature components. Cui et al. [29] proposed the Multi-scale Convolutional Neural Network that applies convolutions of different sampling frequencies of time series data to extract features from different types of time scales for stock trend prediction. There are also other approaches to the stock movement prediction problem that include analyzing the data in the frequency domain. Zhang et al. [30] and Liu et al. [20] proposed a method of using multi-frequency patterns within the stock data. They leverage Discrete Fourier Transform (DFT) [31] to find patterns in the stock data across multiple frequencies.

However, convolutional neural networks struggle to capture global dependencies, and relying on data solely from the frequency domain overlooks crucial local dependencies inherent in financial time series data. Given the stochastic nature of stock data, it is imperative to account for dependencies across all ranges. In this study, we address short-term, mid-term, and long-term dependencies by incorporating both daily and weekly sampled data. Additionally, we integrate information from the time and frequency domains to obtain a comprehensive set of features.

C. Fast Fourier Transform

Fast Fourier Transform (FFT) [31]–[35] is a method of converting a signal in the time-space to the frequency space. The time domain represents the samples of the signal, while the frequency domain decomposes the signal into sinusoids that make up the original signal. Given a sequence x of length

N, FFT is defined as:

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-i2\pi kn/N} \tag{1}$$

where x_n is the nth input, i is the imaginary unit, and X_k is the kth output. The output of FFT is an array that contains information about the frequencies, amplitudes, and phases constituting the original input signal. Viewing the signal in the frequency domain enables us to identify characteristics that may be difficult to discern in the time domain. For instance, the frequency domain reveals cyclic behaviors inherent in the signal [14].

Similarly, there is also a method of transforming a signal from the frequency domain to the time domain. Given the data X in the frequency domain, the Inverse Fast Fourier Transform (IFFT) transforms the signal back to the time domain. The formula for IFFT is as follows:

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k \cdot e^{i2\pi kn/N} \tag{2}$$

FFT is a key component of ZoomStock because it allows us to extract characteristics of the stock signal in the frequency domain.

III. Proposed Method

We propose ZoomStock, an accurate method for stock movement prediction with multi-scale and multi-domain modeling.

A. Overview

ZoomStock addresses the following challenges to solve the stock movement prediction problem:

1) **Capturing multi-scale dependencies.** Finding complex dependencies within the data is important when analyzing historical stock prices. How can we extract these patterns from the stock data?
2) **Considering multi-domain features.** The frequency domain is used to gain new insights on global patterns that exist within the time series data. How can we use features from the frequency domain to improve the performance of the model?
3) **Increasing efficiency.** Many models, such as attention-based LSTM models, attempt to identify correlations within stock data but often suffer from heavy computational complexity. How can we efficiently find these correlations?

We address the challenges with the following main ideas:

1) **Multi-scale modeling (Section III-B).** We find short-term, mid-term, and long-term dependencies by passing data with different sampling periods through various convolutional layers.
2) **Fast Fourier Transform (Section III-C).** We exploit Fast Fourier Transform (FFT) on the stock data to efficiently transform the data to the frequency domain and use the output as additional hidden features.
3) **Efficient circular convolution (Section III-D).** We exploit the Convolution Theorem to perform a circular convolution on the stock data through a point-wise product in the frequency domain, significantly enhancing the efficiency compared to calculating circular convolution in the time domain.

B. Multi-Scale Modeling

We first find short-term, mid-term, and long-term dependencies in the time domain using various convolutional layers.

Input features. As input, we use the feature matrix $\boldsymbol{X}^s = [\boldsymbol{x}_1^s, \cdots, \boldsymbol{x}_T^s] \in \mathbb{R}^{D \times T}$ which holds the price features for each stock $1 \leq s \leq S$ for T time-steps, where D is the dimension of features. We divide the input features into daily-sampled and weekly-sampled data to get multiple perspectives of the input data. The daily-sampled matrix $\tilde{\boldsymbol{X}}^s = [\tilde{\boldsymbol{x}}_1^s, \cdots, \tilde{\boldsymbol{x}}_w^s] \in \mathbb{R}^{D \times w}$ consists of input features collected from day 1 through day w, where w represents the window size. The weekly-sampled matrix, $\hat{\boldsymbol{X}}^s = [\hat{\boldsymbol{x}}_1^s, \cdot \cdot, \hat{\boldsymbol{x}}_w^s] \in \mathbb{R}^{D \times w}$, is constructed by sampling input features from day 1 to day T, with a sampling interval of 5 business days. We set T such that the weekly-sampled matrix also has a column length of w.

1D Convolution on Daily-Sampled Data. We employ a 1D Convolution module with daily-sampled data to extract short-term dependencies from the stock data. The output of the 1D Convolution module is given as:

$$\tilde{h}_c^s[i] = \sum_{j=1}^{k} \tilde{x}_c^s[i+j-1] \cdot f_c[j] + b_c \tag{3}$$

where for the channel $1 \leq c \leq D$, $\tilde{h}_c^s[i]$ is the output at position i, $f_c[j]$ denotes the kernel weight associated with the input at position j, and k is the size of the filter. We set the padding size to be $\frac{k-1}{2}$ and stride as 1 so that the output $\tilde{h}^s$ retains the shape $\mathbb{R}^{D \times w}$. As the 1D CNN module aggregates information among neighboring data points, we effectively capture short-term dependencies at each data point.

1D Convolution on Weekly-Sampled data. We also perform 1D Convolution with Equation (3), but on weekly-sampled data to create the matrix $\hat{h}^s \in \mathbb{R}^{D \times w}$. The use of weekly-sampled data means we aggregate information from neighboring data points that are farther away from each other, compared to the case for the daily-sampled data. Therefore, the 1D Convolution module extracts mid-term dependencies from the stock signal.

Circular Convolution on Weekly-Sampled data. Another type of convolution that we use is circular convolution [36], [37]. Our circular convolution module consists of applying a convolution on weekly-sampled data where the kernel size of the convolution is the same as that of the input. Fig. 2 shows an illustration of how circular convolution works. While 1D convolution on weekly-sampled data finds dependencies only among neighboring data points, circular convolution on weekly-sampled data finds dependencies throughout the entire data sequence.

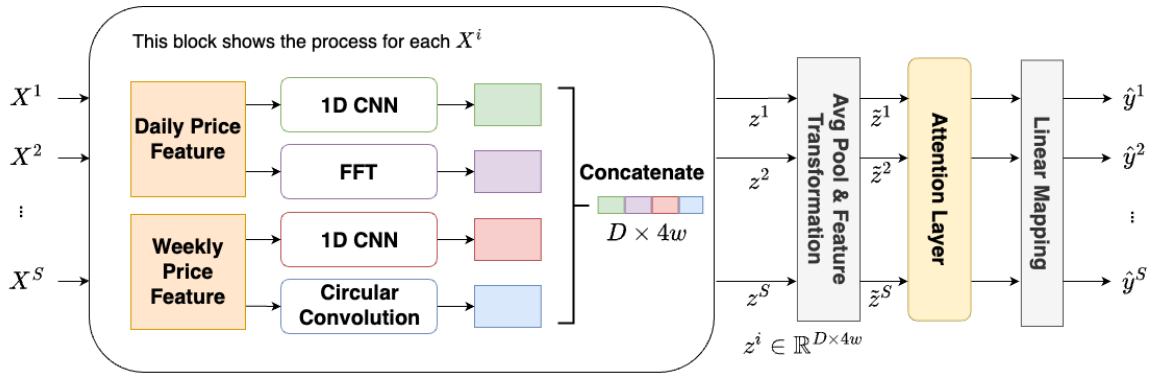

Fig. 1. The overall structure of ZoomStock for making a prediction $\hat{y}_t^s$ for stock s on day t. ZoomStock learns stock embeddings by concatenating the outputs of various convolutional layers with features derived from the Fast Fourier Transform (FFT). An average pooling layer reduces the dimensionality of the concatenated hidden features, followed by a feature transformation layer that maps the hidden features to a latent space. These transformed features are then passed through an attention layer, which identifies correlations between different stocks by applying attention across each stock's hidden features. Finally, a linear layer is used to predict stock price movements.

When applying a convolutional layer to a sequence with a kernel size equal to the sequence length, traditional zero-padding can limit the kernel's ability to fully capture the features. Specifically, as the kernel moves along the sequence, zero-padding causes it to only access portions of the sequence at a time, making it difficult to extract comprehensive features. In contrast, circular convolution ensures that the kernel always has access to the entire sequence, regardless of its position, by wrapping around the data. This allows for more effective feature extraction and a more holistic view of the sequence. We empirically observe that the circular convolution approach consistently outperforms the zero-padding method, demonstrating superior performance in capturing the underlying patterns in the sequence (see Section IV-D). This suggests that circular convolution provides a more robust mechanism for sequence-based tasks, especially when the kernel size is large relative to the sequence length.

Explicitly, the circular convolution is defined as

$$\hat{g}_c^s[i] = \sum_{j=0}^{N-1} \hat{x}_c^s[j] \cdot F_c[(i-j) \mod N] \tag{4}$$

where $1 \leq c \leq D$ is the channel, N is the length of the sequence, and $F_c[\cdot]$ is the kernel weight of circular convolution. Each output element of $\hat{g}_c^s[i]$ is computed by taking a weighted sum of elements from $\hat{x}_c^s[j]$ and $F_c[(i-j) \mod N]$, with the indices wrapped around circularly.

For circular convolution, we set the stride to 1, but padding is not required because when the filter reaches the edge of the input, the filter wraps around and continues to compute using the elements from the beginning of the input sequence. This ensures that the output size matches the input size, resulting in the matrix $\hat{g}^s \in \mathbb{R}^{D \times w}$.

C. Multi-Domain Features

Time series data in finance often exhibits complex patterns, encompassing both gradual trends and periodic patterns. While gradual trends captured in the time domain provide insights into the historical evolution of financial data, periodic or seasonal patterns are more apparent in the frequency domain [38]. To achieve a comprehensive analysis, we leverage both time and frequency domain information. By incorporating features extracted from both domains, we effectively capture the intricate dynamics of financial time series data, encompassing both gradual changes and recurring cycles. This approach enables a more robust and deep understanding of the underlying patterns, leading to more accurate and reliable predictions.

Specifically, we perform Fast Fourier Transform (FFT) on daily-sampled data to convert it to the frequency domain. The features from the frequency domain represent global dependencies within the stock signal. Note that it is unnecessary to use weekly-sampled data in the frequency domain as daily-sampled data suffice in detecting both high and low frequencies within the data. The FFT outputs a sequence of complex numbers, with each number representing a specific frequency component of the original stock data. To derive a real-valued feature, we take the magnitude of each complex number by computing the absolute value. This magnitude serves as the multi-domain feature used in ZoomStock. Moreover, performing FFT on daily-sampled data does not change the output size. Therefore, the output of the FFT will retain the input dimensions of $\mathbb{R}^{D \times w}$.

Concatenation, Transformation, and Normalization. For each stock s, the hidden features generated by each module are concatenated across all channels, resulting in a matrix z^s with dimensions $\mathbb{R}^{D \times 4w}$, as depicted in Fig. 1. Subsequently, an average pooling layer with a filter size of D is applied. This means the average pooling layer is applied so that the values across all rows are averaged, effectively compressing the information across the D rows. This operation reduces the dimensionality of the matrix to $\mathbb{R}^{1 \times 4w}$, resulting in the vector $\hat{z}^s$. The inclusion of the average pooling layer reduces the dimensionality of the hidden features and mitigates overfitting by emphasizing the most significant features. Then, we use a fully connected layer with a tanh activation function to map

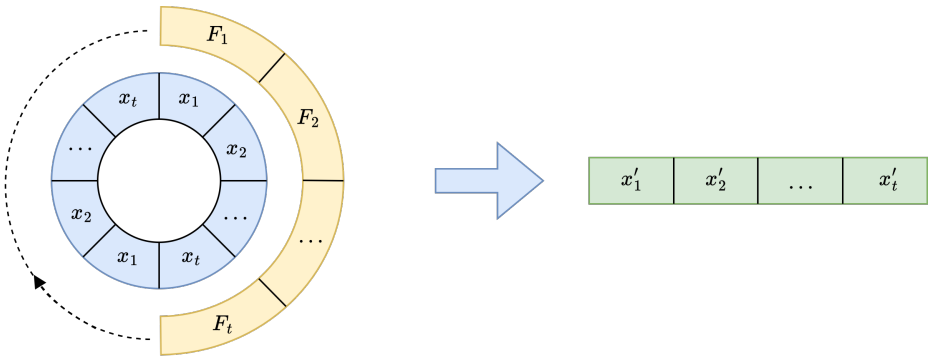

Fig. 2. Illustration of the process of applying circular convolution on weekly-sampled data. Given the input x of length t, the kernel size of the convolution is also t. The figure shows how two copies of the input x can be stacked together to create the wrap-around effect needed for circular convolution. Then, the kernel is shifted one step at a time to complete the circular convolution and create the hidden features x'.

the concatenated features to a latent space of size d:

$$\tilde{z}^s = \tanh(\hat{z}^s W^s + b^s) \tag{5}$$

where $W^s \in \mathbb{R}^{4w \times d}$ and $b^s \in \mathbb{R}^d$ are parameters. Additionally, to stabilize training, we introduce layer normalization, allowing each stock to exhibit a similar range of features and patterns [39].

D. Efficient Circular Convolution

In (4), calculating the weighted sum for each output element is a computationally heavy task. Our idea is to use the Convolution Theorem to get the same result by calculating the element-wise multiplication in the frequency domain instead. The Convolution Theorem states that the convolution of two sequences in the real space corresponds to the product of their individual Fourier transforms [40], [41].

Therefore, we obtain the result of our circular convolution module by

$$\hat{g}_c^s = \text{IFFT}(\text{FFT}(\hat{x}_c^s) \cdot \text{FFT}(W_c)) \tag{6}$$

where FFT denotes the Fast Fourier Transform, IFFT denotes the Inverse Fast Fourier Transform, and W_c is the weight vector for channel c. Based on the Convolution Theorem, we transform the stock signal and kernel weights to the frequency domain, and perform element-wise multiplication to get the circular convolution. Equation (6) reduces the computational complexity of circular convolution from $O(N^2)$ to $O(N \log N)$ (see Theorem 1). This approach allows us to efficiently extract long-term dependencies from the stock signal.

Theorem 1. *Equation* (6) *reduces the time complexity of* (4) *from* $O(N^2)$ *to* $O(N \log N)$. □

Proof. The direct computation of circular convolution shown in (4) involves a double summation over m and n, with N elements in each summation. The modulo operation is a constant-time operation and does not affect the overall complexity. Therefore, the total number of operations is $O(N^2)$.

The time complexity of calculating the Fast Fourier Transform and the Inverse Fast Fourier Transform of a sequence length N has a time complexity of $O(N \log N)$. The element-wise multiplication in (6) has a time complexity of $O(N)$. Therefore, the computation of circular convolution in the frequency domain is dominated by the FFT operations and the total time complexity is $O(N \log N)$. □

E. Main Predictor

For the main predictor layer, we use the data-axis self-attention module similar to the one used in DTML [1]. This module uses a Transformer encoder to apply attention across each stock's hidden features, enabling the identification of correlations between different stocks. We also include layer normalization [39], a dropout layer [42], and a tanh nonlinear transformation after the attention layer. We lastly apply a single linear layer as the output layer to produce the final predictions. The single linear layer is defined as:

$$\hat{y} = \sigma(H_p W_p + b_p) \tag{7}$$

where $H_p \in \mathbb{R}^{S \times d}$ is the output from the data-axis self-attention module, $W_p \in \mathbb{R}^{d \times 1}$ is the weight, and $b_p \in \mathbb{R}^S$ is the bias. We apply a sigmoid function σ to convert each element as a probability and use it directly as the output for ZoomStock for stock movement prediction. The overall architecture of ZoomStock can be seen in Fig. 1.

F. Theoretical Analysis

We present the computational complexity and the number of parameters of our model. Let us use the following notations: D is the dimension of features, S is the number of stocks, w is the window size, k is the size of convolution kernels, and d is the latent dimension of attention layer. Theorem 2 describes the computational complexity of ZoomStock.

Theorem 2. *The computational complexity of ZoomStock is given by* $O(SDw(k + \log w) + Sdw + S^2 d + Sd^2)$. □

Proof. For the daily feature extraction, each stock is fed into a 1D CNN and an FFT, requiring $O(Dwk)$ and $O(Dw \log w)$ operations, respectively. For the weekly feature extraction, each stock is processed using a 1D CNN and a circular convolution, which take $O(Dwk)$ and $O(Dw \log w)$ costs due to

TABLE II
SUMMARY OF DATASETS. THE DAYS COLUMN REPRESENTS THE NUMBER OF AVAILABLE DAYS IN EACH DATASET.

Data	Country	Stocks	Days	Dates
ACL18[1]	US	87	652	2013-06-03 to 2015-12-31
ACL23[2]	US	87	504	2018-01-02 to 2023-04-27
KOSPI[2]	South Korea	200	1528	2018-01-02 to 2023-11-09
HK23[2]	Hong Kong	26	1528	2018-01-02 to 2023-11-09
TWSE23[2]	Taiwan	37	1528	2018-01-02 to 2023-11-09
DE23[2]	Germany	23	1528	2018-01-02 to 2023-11-09

[1]https://github.com/fulifeng/Adv-ALSTM

[2]https://github.com/anonymous231129/ZoomStock

Theorem 1. Also, the feature transformation layer has $O(dw)$ computation. These sum up to $O(SDw(k + \log w) + Sdw)$ computational cost for S stocks. The multi-head attention layer for S stocks with latent dimension d requires $O(S^2d + Sd^2)$ time cost. Finally, the prediction layer takes $O(Sd)$ time. These results lead to a total of $O(SDw(k + \log w) + Sdw + S^2d + Sd^2)$ computational complexity of ZoomStock. □

We provide the number of parameters of our proposed method in Theorem 3.

Theorem 3. *The number of parameters of ZoomStock is given by* $\Theta(D(k + w) + dw + d^2 + S)$. □

Proof. Note that our 1D CNN works in a depth-wise manner, requiring $\Theta(Dk)$ parameters, and a circular convolution module contains $\Theta(Dw)$ parameters, which sum up to $\Theta(D(k + w))$. The feature transformation layer has $\Theta(dw)$ number of parameters, the multi-head attention layer has $\Theta(d^2)$ parameters, and the prediction layer requires $\Theta(d + S)$ learnable weights, thus we obtain the total of $\Theta(D(k+w)+dw+d^2+S)$ parameters. □

IV. EXPERIMENTS

We conduct experiments to answer the following questions about the performance of ZoomStock:

Q1. **Univariate Accuracy (Section IV-B).** Does ZoomStock outperform previous approaches in stock movement prediction using univariate time series data?

Q2. **Multivariate Accuracy (Section IV-C).** Does ZoomStock outperform previous approaches in stock movement prediction using multivariate time series data?

Q3. **Ablation Study (Section IV-D).** Does each idea of ZoomStock help improve the performance?

A. Experimental Setup

We present our experimental setup including datasets, baseline approaches, hyperparameters, and evaluation metrics. All of our experiments were done in a workstation with RTX 3070 Ti.

Datasets. We use 6 benchmark datasets of stock movement prediction: ACL18, ACL23, KOSPI, HK23, TWSE23, and DE23 shown in Table II. The ACL18 dataset has been used in previous works for stock movement prediction. ACL23, KOSPI, HK23, TWSE23, and DE23 are new benchmark datasets we collected from the US, South Korea, Hong Kong, Taiwan, and German stock markets, respectively. We label the instances according to the increase rate of adjusted closing prices. The increase rate is given as $r_s^d = p_s^d / p_s^{d-1} - 1$ where p_s^d is the adjusted closing price of stock s at day d. Instances are divided into $rise$ or $fall$ where instances whose increase rate is greater than 0 are labeled as $rise$, and the rest are labeled as $fall$ for binary classification. We split each dataset into training, validation, and test data chronologically in 70:15:15 ratio.

Feature Vectors. We normalize the input feature vectors following previous works [1], [2]. When dealing with multivariate data, we utilize the normalized open, high, low, close, and adjusted close features, along with six additional features representing long-term trends in stock prices. Conversely, for univariate data, we rely solely on the normalized closing price feature.

Competitors. We compare ZoomStock with the following competitors for stock movement prediction.

- **LSTM** [43] is a representative model for sequential data.
- **Attention LSTM (ALSTM)** [4] combines the hidden states of multiple LSTM cells with the attention technique, preserving the information of distant inputs.
- **Adv-ALSTM** [2] applies adversarial training to ALSTM to improve its generalization performance. Adv-ALSTM generates artificial features that the current model fails to predict during training, making the model more robust.
- **DTML** [1] is the state-of-the-art model which combines both time-axis attention and data-axis attention for multivariate time series forecasting.

Evaluation metrics. We use two metrics to evaluate the performance of stock movement prediction in different perspectives: accuracy (ACC) and the Matthews correlation coefficient (MCC) [44]. ACC is a popular metric, which has been used widely in various classification problems. MCC improves the fairness of evaluation especially when the numbers of positive and negative samples are different. Let TP, TN, FP, and FN denote true positives, true negatives, false positives, and false negatives, respectively. Then, the MCC is defined as follows:

$$\mathrm{MCC} = \frac{\mathrm{TP} \times \mathrm{TN} - \mathrm{FP} \times \mathrm{FN}}{\sqrt{(\mathrm{TP} + \mathrm{FP})(\mathrm{TP} + \mathrm{FN})(\mathrm{TN} + \mathrm{FP})(\mathrm{TN} + \mathrm{FN})}}$$

Hyperparameters. We train our ZoomStock and all competitors based on deep neural networks with the Adam optimizer [45]. We use the following hyperparameters: learning rate 0.001, lag window size 10, epochs 150, batch size 256, and learning rate decay 0.9. We run each model five times with different random seeds and report the average performance in all cases.

B. Univariate Accuracy (Q1)

We compare the accuracy of ZoomStock and baseline stock prediction methods in Table III using univariate time series data, including the previous state-of-the-art model DTML. Our

TABLE III
CLASSIFICATION PERFORMANCE OF ZOOMSTOCK AND COMPETITORS IN UNIVARIATE SETTING. THE BEST IS IN BOLD. NOTE THAT OUR ZOOMSTOCK SHOWS THE BEST PERFORMANCE IN ALMOST ALL CASES.

Method	ACL18		ACL23		KOSPI		HK23		TWSE23		DE23	
	ACC	MCC	ACC	MCC	ACC	MCC	ACC	MCC	ACC	MCC	ACC	MCC
LSTM [43]	0.504	0.009	0.504	0.007	0.535	0.015	0.626	0.248	0.587	0.174	0.545	0.089
ALSTM [4]	0.508	0.019	0.504	0.008	**0.536**	0.012	0.627	0.251	0.584	0.168	0.549	0.010
Adv-ALSTM [2]	0.502	0.005	0.503	0.005	0.535	0.009	0.627	0.251	0.582	0.164	0.544	0.086
DTML [1]	0.521	**0.055**	0.535	0.069	0.532	0.011	**0.657**	**0.312**	0.633	0.267	0.689	0.377
ZoomStock	**0.524**	0.050	**0.542**	**0.084**	0.534	**0.017**	0.647	0.296	**0.708**	**0.417**	**0.702**	**0.404**

TABLE IV
CLASSIFICATION PERFORMANCE OF ZOOMSTOCK AND COMPETITORS IN MULTIVARIATE SETTING. THE BEST IS IN BOLD. NOTE THAT OUR ZOOMSTOCK SHOWS THE BEST PERFORMANCE IN ALMOST ALL CASES.

Method	ACL18		ACL23		KOSPI		HK23		TWSE23		DE23	
	ACC	MCC	ACC	MCC	ACC	MCC	ACC	MCC	ACC	MCC	ACC	MCC
LSTM [43]	0.516	0.034	0.502	0.002	0.533	**0.028**	0.669	0.340	0.622	0.245	0.577	0.151
ALSTM [4]	0.510	0.023	0.502	0.000	0.532	0.023	0.663	0.326	0.619	0.239	0.578	0.154
Adv-ALSTM [2]	0.514	0.031	0.503	0.004	0.532	0.025	0.669	0.341	0.619	0.239	0.575	0.148
DTML [1]	0.533	**0.073**	0.531	0.061	0.527	0.019	0.681	0.368	0.647	0.296	0.665	0.339
ZoomStock	**0.534**	0.070	**0.537**	**0.075**	**0.534**	0.014	**0.683**	**0.374**	**0.729**	**0.460**	**0.733**	**0.468**

ZoomStock consistently performs better than all other methods in almost all datasets in terms of accuracy and MCC, and comparably well in the HK23 dataset. ZoomStock shows up to 15.7%p higher accuracy and up to 31.7%p higher MCC than other competitors.

The high performances of ALSTM and Adv-ALSTM are achieved because the attention network within the models captures long-term dependencies within the data. Combining this advantage with an LSTM network allows the model to capture both short and long-term patterns. However, using attention and LSTM networks cannot capture cyclic or global patterns within the dataset. ZoomStock combines CNN and FFT to capture short, long, and cyclic patterns within the stock data, allowing the model to effectively find useful features.

C. Multivariate Accuracy (Q2)

We also evaluate the accuracy of ZoomStock compared to previous stock prediction methods in Table IV using multivariate time series data. Note that ZoomStock beats previous methods in accuracy for all datasets, and almost all datasets in MCC. ZoomStock gives up to 8.2%p higher accuracy and up to 16.3%p higher MCC than the previous state-of-the-art model DTML. Although the time-axis attention module in DTML attempts to find both local and global dependencies in historical stock data, there is still room for improvement in terms of extracting both types of dependencies effectively. By replacing the time-axis attention in DTML with our ZoomStock architecture, ZoomStock extracts complex dependencies more effectively.

TABLE V
AN ABLATION STUDY OF ZOOMSTOCK WHERE THE BEST RESULT IS IN BOLD. NOTE THAT ALL THE IDEAS OF ZOOMSTOCK HELP IMPROVE THE PERFORMANCE.

Method	ACL23		HK23		DE23	
	ACC	MCC	ACC	MCC	ACC	MCC
ZoomStock-G	0.534	0.068	0.660	0.327	0.732	0.468
ZoomStock-F	0.529	0.058	0.653	0.308	0.728	0.456
ZoomStock-D	0.528	0.056	0.516	0.023	0.639	0.285
ZoomStock-W	0.532	0.063	0.654	0.323	0.731	0.464
ZoomStock-C	0.526	0.054	0.634	0.274	0.711	0.422
ZoomStock	**0.537**	**0.075**	**0.683**	**0.374**	**0.733**	**0.468**

D. Ablation Study (Q3)

Module Comparison. We conduct an ablation study for ZoomStock and summarize the results in Table V. ZoomStock-G is the variant of ZoomStock that does not use any features provided by the circular convolution performed on weekly sampled data. ZoomStock-F is the variant of ZoomStock that excludes the features from the frequency domain, created by FFT. ZoomStock-D is the variant of ZoomStock that omits the 1D CNN module with daily-sampled data, while ZoomStock-W excludes the 1D CNN module with weekly-sampled data. We compare the performance of ZoomStock with its variants

to see how each module of ZoomStock contributes toward its performance. Note that having all four modules in ZoomStock produces the highest ACC and MCC scores in all cases. This shows that each module of ZoomStock contributes to the superior performance of ZoomStock.

Effect of Circular Convolution. We further analyze the effect of using a standard 1D CNN layer with a filter size equal to the input length, in comparison to our proposed circular convolution module. The results for the standard 1D CNN layer, denoted as ZoomStock-C, are presented in Table V. Notably, the standard 1D CNN consistently demonstrates reduced performance in both accuracy and MCC across all datasets. This performance gap is due to the inherent differences in how each method handles the convolution operation. In the standard 1D CNN, zero-padding is applied to accommodate the large filter size, causing the kernel to lose access to parts of the sequence as it shifts across the input. This leads to a truncated view of the sequence, preventing the network from fully capturing long-range dependencies. In contrast, the circular convolution technique effectively eliminates this limitation by wrapping the input sequence when shifting the filter. This ensures that the kernel always has access to the entire sequence, enabling the model to capture information from distant time points more comprehensively. Consequently, the circular convolution module is able to extract richer and more complete feature representations, leading to improved performance. These findings highlight the benefits of circular convolution, particularly in scenarios where capturing long-term dependencies is critical for accurate prediction.

Efficiency of Circular Convolution. To optimize the computational efficiency of ZoomStock, we investigate two distinct approaches for calculating circular convolution: direct computation and an implementation leveraging the Convolution Theorem. A comparative analysis of training time efficiency, illustrated in Fig. 3, reveals a substantial advantage when employing the Convolution Theorem. Our results demonstrate an average reduction in training time of 30% compared to the direct calculation method. This significant improvement highlights the crucial role of the Convolution Theorem in enhancing the computational efficiency of our model, enabling faster training without compromising prediction accuracy. This efficiency gain is particularly valuable in the context of financial time series analysis, where timely model training is crucial for adapting to rapidly changing market dynamics.

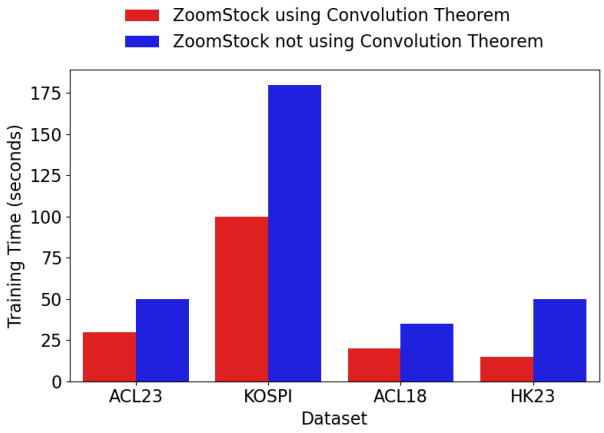

Fig. 3. Time comparison of training ZoomStock using (red bar) and not using (blue bar) the Convolution Theorem. Note that utilizing the Convolution Theorem significantly reduces the training time of ZoomStock.

V. Conclusion

We propose ZoomStock, an accurate method for predicting stock movements by exploiting multi-scale and multi-domain modeling. ZoomStock effectively captures local and global patterns within the stock data, allowing us to make an accurate prediction on the direction the stock price movement. ZoomStock finds short-term dependencies by feeding daily historical stock prices through a 1D CNN. ZoomStock also extracts long-term dependencies by performing circular convolution on weekly-sampled stock data. In addition to the features from the time domain, ZoomStock uses those from the frequency domain to further capture global trends. Extensive experiments on real-world datasets show that ZoomStock provides the state-of-the-art performance.

Acknowledgment

This work was supported by the National Research Foundation of Korea(NRF) funded by MSIT(2022R1A2C3007921). This work was also supported by Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) [No.RS-2021-II211343, Artificial Intelligence Graduate School Program (Seoul National University)], and [NO.RS-2021-II212068, Artificial Intelligence Innovation Hub (Artificial Intelligence Institute, Seoul National University)]. The Institute of Engineering Research at Seoul National University provided research facilities for this work. The ICT at Seoul National University provides research facilities for this study. U Kang is the corresponding author.

References

[1] J. Yoo, Y. Soun, Y. Park, and U. Kang, "Accurate multivariate stock movement prediction via data-axis transformer with multi-level contexts," in *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, F. Zhu, B. C. Ooi, and C. Miao, Eds.

[2] F. Feng, H. Chen, X. He, J. Ding, M. Sun, and T.-S. Chua, "Enhancing stock movement prediction with adversarial training," 2019.

[3] Y. Chen, Z. Wei, and X. Huang, "Incorporating corporation relationship via graph convolutional neural networks for stock price prediction," ser. CIKM '18, New York, NY, USA, 2018.

[4] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," 2017.

[5] J. Kalyani, P. Bharathi, P. Jyothi *et al.*, "Stock trend prediction using news sentiment analysis," *arXiv preprint arXiv:1607.01958*, 2016.

[6] Y. Soun, J. Yoo, M. Cho, J. Jeon, and U. Kang, "Accurate stock movement prediction with self-supervised learning from sparse noisy tweets," in *IEEE International Conference on Big Data, Big Data 2022, Osaka, Japan, December 17-20, 2022*. IEEE, 2022, pp. 1691–1700.

[7] A. W. Lo and A. C. MacKinlay, "Stock market prices do not follow random walks: Evidence from a simple specification test," *The review of financial studies*, vol. 1, no. 1, pp. 41–66, 1988.

[8] D. A. Hsieh, "Chaos and nonlinear dynamics: application to financial markets," *The journal of finance*, vol. 46, no. 5, pp. 1839–1877, 1991.

[9] A. W. Lo and A. C. MacKinlay, *A non-random walk down Wall Street*. Princeton University Press, 2011.

[10] K. Adam, A. Marcet, and J. P. Nicolini, "Stock market volatility and learning," *The Journal of finance*, vol. 71, no. 1, pp. 33–82, 2016.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] Q. Ding, S. Wu, H. Sun, J. Guo, and J. Guo, "Hierarchical multi-scale gaussian transformer for stock movement prediction." in *IJCAI*, 2020, pp. 4640–4646.

[13] B. Bini and T. Mathew, "Clustering and regression techniques for stock prediction," *Procedia Technology*, vol. 24, pp. 1248–1255, 2016.

[14] J. Jeon, J. Park, C. Park, and U. Kang, "Frequant: A reinforcement-learning based adaptive portfolio optimization with multi-frequency decomposition," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 1211–1221.

[15] J. Zhang, L. Ye, and Y. Lai, "Stock price prediction using cnn-bilstm-attention model," *Mathematics*, vol. 11, no. 9, p. 1985, 2023.

[16] S. Mehtab and J. Sen, "Stock price prediction using convolutional neural networks on a multivariate timeseries," *arXiv preprint arXiv:2001.09769*, 2020.

[17] W. Lu, J. Li, Y. Li, A. Sun, and J. Wang, "A cnn-lstm-based model to forecast stock prices," *Complexity*, vol. 2020, no. 1, p. 6622927, 2020.

[18] J. Eapen, D. Bein, and A. Verma, "Novel deep learning model with cnn and bi-directional lstm for improved stock market index prediction," in *2019 IEEE 9th annual computing and communication workshop and conference (CCWC)*. IEEE, 2019, pp. 0264–0270.

[19] S. Mukherjee, B. Sadhukhan, N. Sarkar, D. Roy, and S. De, "Stock market prediction using deep learning algorithms," *CAAI Transactions on Intelligence Technology*, vol. 8, no. 1, pp. 82–94, 2023.

[20] G. Liu, Y. Mao, Q. Sun, H. Huang, W. Gao, X. Li, J. Shen, R. Li, and X. Wang, "Multi-scale two-way deep neural network for stock trend prediction." in *IJCAI*, 2020, pp. 4555–4561.

[21] M. Mallikarjuna and R. P. Rao, "Evaluation of forecasting methods from selected stock market returns," *Financial Innovation*, vol. 5, no. 1, p. 40, 2019.

[22] S. Lahmiri, "Wavelet low-and high-frequency components as features for predicting stock prices with backpropagation neural networks," *Journal of King Saud University-Computer and Information Sciences*, vol. 26, no. 2, pp. 218–227, 2014.

[23] W. Shu and Q. Gao, "Forecasting stock price based on frequency components by emd and neural networks," *Ieee Access*, vol. 8, pp. 206 388–206 395, 2020.

[24] D. Song, A. M. C. Baek, and N. Kim, "Forecasting stock market indices using padding-based fourier transform denoising and time series deep learning models," *IEEE Access*, vol. 9, pp. 83 786–83 796, 2021.

[25] Y. Xu and S. 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)*, 2018, pp. 1970–1979.

[26] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[27] H. Li, Y. Shen, and Y. Zhu, "Stock price prediction using attention-based multi-input lstm," in *Asian conference on machine learning*. PMLR, 2018, pp. 454–469.

[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[29] Z. Cui, W. Chen, and Y. Chen, "Multi-scale convolutional neural networks for time series classification," *arXiv preprint arXiv:1603.06995*, 2016.

[30] L. Zhang, C. Aggarwal, and G.-J. 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*, 2017, pp. 2141–2149.

[31] E. O. Brigham, *The fast Fourier transform and its applications*. Prentice-Hall, Inc., 1988.

[32] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex fourier series," *Mathematics of computation*, vol. 19, no. 90, 1965.

[33] A. V. Oppenheim, *Discrete-time signal processing*. Pearson Education India, 1999.

[34] Y. chan Park, J.-G. Jang, and U. Kang, "Fast and accurate partial fourier transform for time series data," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2021.

[35] Y. chan Park, J. Kim, and U. Kang, "Fast multidimensional partial fourier transform with automatic hyperparameter selection," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2024.

[36] R. A. Roberts and C. T. Mullis, *Digital signal processing*. Addison-Wesley Longman Publishing Co., Inc., 1987.

[37] J. W. Cooley, P. A. Lewis, and P. D. Welch, "The fast fourier transform and its applications," *IEEE Transactions on Education*, vol. 12, no. 1, pp. 27–34, 1969.

[38] K. Yi, Q. Zhang, W. Fan, S. Wang, P. Wang, H. He, N. An, D. Lian, L. Cao, and Z. Niu, "Frequency-domain mlps are more effective learners in time series forecasting," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[39] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[40] Y. Katznelson, *An Introduction to Harmonic Analysis*, ser. Cambridge Mathematical Library. Cambridge University Press, 2004.

[41] Y. Rao, W. Zhao, Z. Zhu, J. Lu, and J. Zhou, "Global filter networks for image classification," *Advances in neural information processing systems*, vol. 34, pp. 980–993, 2021.

[42] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[43] A. Moghar and M. Hamiche, "Stock market prediction using lstm recurrent neural network," *Procedia Computer Science*, vol. 170, pp. 1168–1173, 2020.

[44] Y. Xu and S. 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)*, I. Gurevych and Y. Miyao, Eds., Melbourne, Australia, 2018.

[45] D. P. Kingma, J. A. Ba, and J. Adam, "A method for stochastic optimization. arxiv 2014," *arXiv preprint arXiv:1412.6980*, vol. 106, 2020.