Accurate Multivariate Stock Movement Prediction via Data-Axis Transformer with Multi-Level Contexts

Jaemin Yoo
Seoul National University
Seoul, South Korea
jaeminyoo@snu.ac.kr

Yejun Soun
Seoul National University
DeepTrade Inc.
Seoul, South Korea
sony7819@snu.ac.kr

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

U Kang
Seoul National University
DeepTrade Inc.
Seoul, South Korea
ukang@snu.ac.kr

ABSTRACT

How can we efficiently correlate multiple stocks for accurate stock movement prediction? Stock movement prediction has received growing interest in data mining and machine learning communities due to its substantial impact on financial markets. One way to improve the prediction accuracy is to utilize the correlations between multiple stocks, getting a reliable evidence regardless of the random noises of individual prices. However, it has been challenging to acquire accurate correlations between stocks because of their asymmetric and dynamic nature which is also influenced by the global movement of a market. In this work, we propose DTML (Data-axis Transformer with Multi-Level contexts), a novel approach for stock movement prediction that learns the correlations between stocks in an end-to-end way. DTML makes asymmetric and dynamic correlations by a) learning temporal correlations within each stock, b) generating multi-level contexts based on a global market context, and c) utilizing a transformer encoder for learning inter-stock correlations. DTML achieves the state-of-the-art accuracy on six datasets collected from various stock markets from US, China, Japan, and UK, making up to 13.8%p higher profits than the best competitors and the annualized return of 44.4% on investment simulation.

CCS CONCEPTS

• **Mathematics of computing** → **Time series analysis**; • **Information systems** → *Data mining*; Clustering and classification.

KEYWORDS

stock movement prediction, transformers, attention mechanism

ACM Reference Format:
Jaemin Yoo, Yejun Soun, Yong-chan Park, and U Kang. 2021. Accurate Multivariate Stock Movement Prediction via Data-Axis Transformer with Multi-Level Contexts. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3447548.3467297

KDD '21, August 14–18, 2021, Virtual Event, Singapore

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467297

1 INTRODUCTION

How can we efficiently correlate multiple stocks for accurate stock movement prediction? Stock movement prediction is one of the core applications of financial data mining, which has received growing interest in data mining and machine learning communities due to its substantial impact on financial markets [9, 25, 31]. The problem is to predict the movement (rise or fall) of stock prices at a future moment. The potential of the problem is unquestionable, as accurate predictions can lead to the enormous profit of investment.

It is challenging to achieve high accuracy of stock movement prediction, since stock prices are inherently random; no clear patterns exist unlike in other time series such as temperature or traffic. On the other hand, most stocks can be clustered as sectors by the industries that they belong to [3]. Stocks in the same sector share a similar trend even though their prices are perturbed randomly in a short-term manner, and such correlations can be a reliable evidence for investors. For instance, one can buy or sell a stock considering the prices of other stocks in the same sector, based on the belief that their movements will coincide in the future.

Most previous works that utilize the correlations between stocks rely on pre-defined lists of sectors [15, 18]. However, using a fixed list of sectors makes the following limitations. First, one loses the dynamic property of stock correlations that naturally change over time, especially when training data span over a long period. Second, a prediction model cannot be applied to stocks that have no information of sectors or whose sectors are ambiguous. Third, the performance of predictions relies heavily on the quality of sector information rather than the ability of a prediction model.

In this work, we design an end-to-end framework that learns the correlations between stocks for accurate stock movement prediction; the quality of correlations is measured by how much it contributes to improving the prediction accuracy. Specifically, we aim to address the following challenges that arise from the properties of stock prices. First, multiple features at each time step should be used together to capture accurate correlations. For instance, the variability of stock prices in a day can be given to a model by including both the highest and lowest prices as features. Second, the global movement of a market should also be considered, since the relationship between stocks is determined not only by their local movements, but also by the global trend. Third, the learned correlations should be asymmetric, reflecting the different degrees of information diffusion in a market.

We propose DTML (Data-axis Transformer with Multi-Level contexts), an end-to-end framework that automatically correlates multiple stocks for accurate stock movement prediction. The main ideas of DTML are as follows. First, DTML generates each stock's

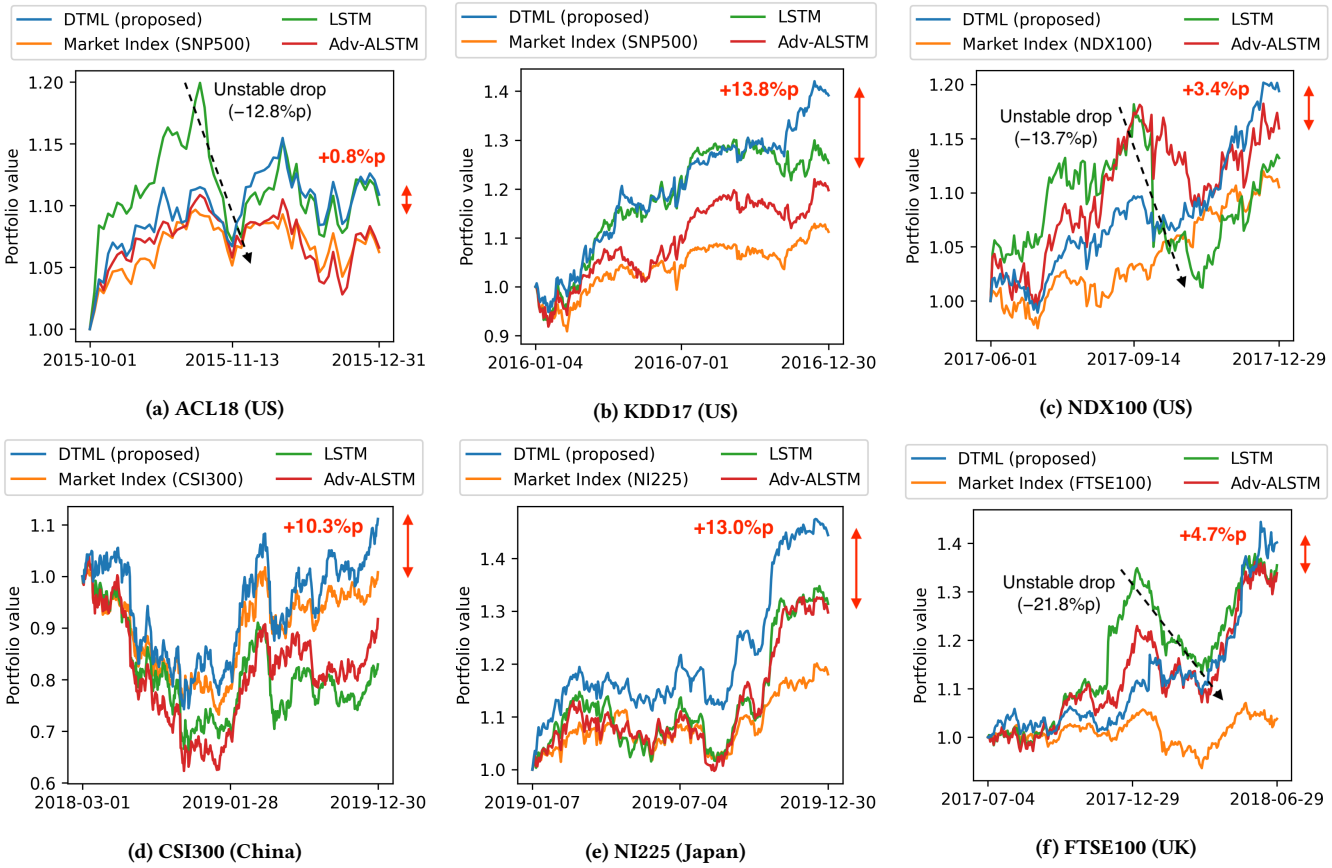


Figure 1: Investment simulation on real world datasets of diverse countries including US, China, Japan, and UK. We manage a portfolio during the test period of each dataset by choosing the top three stocks by the equal weight at each day based on predicted probabilities. DTML makes up to 13.8 % points higher profits than the best competitors, without having the sudden drops of portfolio values, which are frequently observed from the baseline models that predict individual stock prices.

comprehensive context vector that summarizes multivariate historical prices by temporal attention. Second, DTML extends the generated contexts into multi-level by combining it with the global movement of the market. Third, DTML learns asymmetric and dynamic attention scores from the multi-level contexts using the transformer encoder, which calculates different query and key vectors for multi-head attention between stocks.

Our contributions are summarized as follows:

- **Algorithm.** We propose DTML, a novel framework for stock movement prediction that utilizes asymmetric and dynamic stock correlations in an end-to-end way.
- **Experiments.** We run extensive experiments on six datasets for stock movement prediction, collected from various stock markets of US, China, Japan, and UK.
- **Accuracy.** DTML achieves state-of-the-art accuracy on six datasets for stock prediction, improving the accuracy and the Matthews correlation coefficients of the best competitors by up to 3.6 and 10.8 points, respectively.
- **Simulation.** We run invest simulation on six datasets and show that DTML makes up to 13.8%p higher profits than the previous approaches, resulting in the annualized return of up to 44.4% (Figure 1).
- **Case studies.** We show in case studies that the attention scores learned by DTML give valuable insights of the stock market even without any prior knowledge.

The rest of this paper is organized as follows. In Section 2, we introduce related works for stock movement prediction, which are categorized as individual and correlated stock prediction models. In Section 3, we present the main ideas and individual components of DTML. In Section 4, we present experimental results on real-world datasets of four countries, including the evaluation of accuracy and investment simulation. We conclude at Section 5.

2 RELATED WORKS

We introduce related works on stock movement prediction, which are categorized as individual and correlated stock prediction. Individual prediction models use the information of only stock i when predicting its price after the training, independently from the other stocks, while correlated prediction models use the information of multiple stocks other than the target stock i.

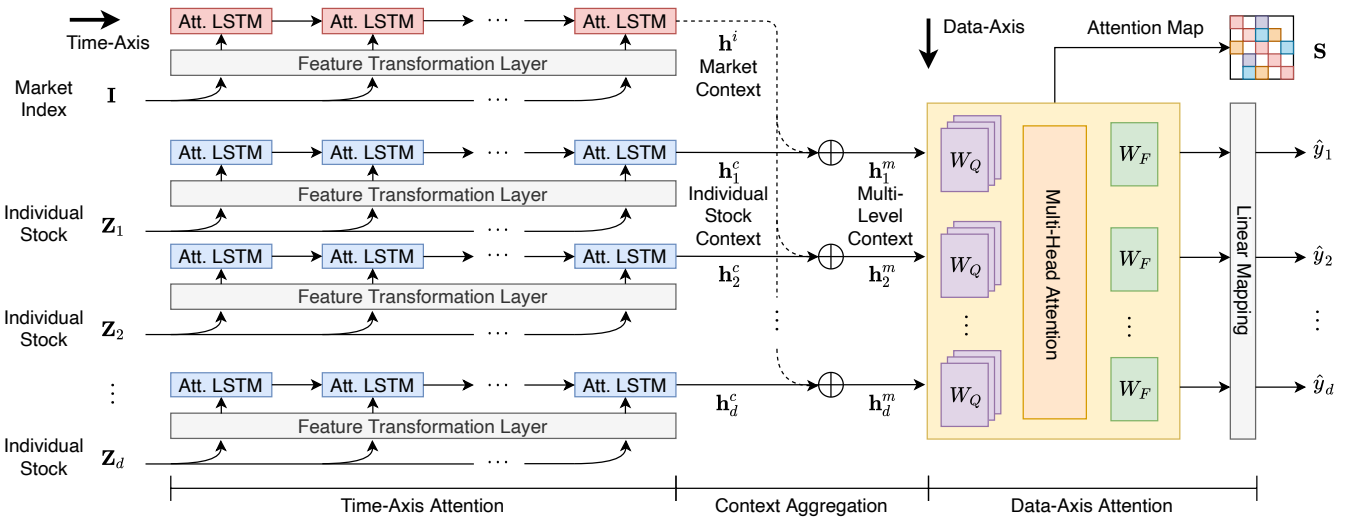


Figure 2: The structure of DTML, which consists of three main modules. First, attention LSTM computes the context vector $\mathbf{h}_u^c$ of each stock u by temporal attention, extracting repetitive patterns of stock prices. Second, the context aggregation module makes the multi-level context $\mathbf{h}_u^m$ by combining the individual contexts and the global market context $\mathbf{h}^i$, which is generated from historical index data. Lastly, the transformer encoder correlates the multi-level contexts of different stocks by multi-head attention and produces the final predictions for all individual stocks. The attention map S is returned also as an output.

2.1 Individual Stock Prediction

Most approaches for stock movement prediction focus on predicting individual prices. They capture complex repetitive patterns from the historical prices of each individual stock, instead of finding the correlations between multiple stocks. We categorize the models into a) those based on long short-term memory units (LSTM), b) those not based on LSTM, and c) those using additional information such as news articles other than historical prices.

Various models for individual stock movement prediction are based on LSTM, which is a representative neural network designed for sequential data. Nelson et al. [24] have shown that LSTM outperforms previous machine learning models on stock movement prediction. Li et al. [16] and Wang et al. [28] have improved LSTM mainly using the attention mechanism for correlating distant time steps, addressing the limitation of LSTM that the features of the last time steps dominate the prediction. Feng et al. [9] have applied adversarial training to improve the robustness of LSTM.

Some approaches make stock predictions without using LSTM. Zhang et al. [35] and Liu et al. [20] have utilized multi-frequency patterns of stock prices. Ding et al. [7] have used extreme events in the historical prices as the main evidence of predictions. Li et al. [19] have computed the importance of technical indicators for each stock from a fund-stock network. Ding et al. [8] have proposed hierarchical Gaussian Transformers for modeling stock prices, instead of relying on recurrent neural networks. Wang et al. [29] have used reinforcement learning for efficient portfolio management using the stock price prediction as an intermediate component.

Lastly, there are approaches that use external data, mostly textual data such as news articles or public comments [10]. Xu and Cohen [31] have extracted the latent information of prices and tweets at each day using variational autoencoders. Kalyani et al. [13] and Mohan et al. [22] have performed sentimental analysis to extract meaningful evidence from textual data, while Nam and Seong [23] have utilized causal analysis for the same purpose. Chen et al. [5] have utilized the trading patterns of professional fund managers as a reliable source of information. Extracting deep features from textual data is an effective approach to combine the information from multiple sources [12, 21, 34].

Such methods for individual stock prediction lose the essential information of stock correlations, focusing on the temporal patterns of historical prices of each stock. The performance of these models is limited, since the stock prices are inherently random and thus have no clear patterns to detect. Instead, we consider the movement of other stocks as essential evidence of prediction by learning the inter-stock correlations dynamically from historical prices.

2.2 Correlated Stock Prediction

Several approaches use the correlations between multiple stocks to make robust and consistent predictions. The main challenge is to get accurate correlations, because the performance of such models is mainly determined by the quality of correlations, and they can perform worse than the methods for individual prediction if the correlations are inaccurate. We categorize previous works based on whether the correlations are learned or given as an input.

Dual-stage attention-based recurrent neural network (DA-RNN) [25] is an LSTM-based model that learns the correlations between stocks from historical prices. DA-RNN is an early method that has a limited ability of learning correlations, as it uses only the closing prices of days as the input of stock correlations, even when other observations such as the opening or the highest prices are available.

DA-RNN is also designed to predict only a single market index at a time, and requires huge computation to predict individual stocks. There are other approaches [4, 30, 33] that learn the correlations between target variables in a multivariate time series, optionally with graph neural networks [4, 32], but none of them focuses on the financial domain, which has unique characteristics.

Li et al. [15] have proposed a novel framework that captures the patterns of individual prices and then correlates them by treating them as nodes of a Markov random field. The graph between stocks is given as prior knowledge. Li et al. [18] have also modeled the correlations between stocks as a graph, which is learned separately from the prediction model and given as an input. The main limitation of such approaches is that they rely on the pre-defined correlations between stocks, which are fixed for all training and test time, while the true correlations keep changing. The accuracy of these models is mainly affected by the quality of given correlations, rather than the ability of prediction. Furthermore, they cannot be applied to stocks whose correlation information is not given.

In this work, we aim at learning accurate correlations between stocks without any prior knowledge, addressing the limitations of previous works, as an end-to-end framework that combines the abilities of learning correlations and making actual predictions. We learn dynamic and asymmetric stock correlations that incorporate all available features along with the global movement.

2.3 Transformer Models

The transformer model [27] has drawn enormous attention from the research community, due to its great performance on the NLP domain [6]. Recent works have applied the transformer encoder to time series data of different domains such as stock prices [8], electricity consumption [17], and influenza prevalence [30] to effectively correlate sequential elements, replacing recurrent neural networks (RNN). The main advantage of transformer encoder is that they do not suffer from the limitations of RNNs such as the gradient vanishing because all time steps participate in each layer by the self-attention. Instead of the time-axis attention of previous works, we use the transformer encoder to correlate multiple stocks by the data-axis. The self-attention effectively combines multiple stocks to maximize the accuracy of stock movement prediction.

3 PROPOSED APPROACH

We introduce DTML (Data-axis Transformer with Multi-Level contexts), a novel method for improving the accuracy of stock movement prediction by learning the correlations between stocks.

3.1 Overview

Our objective is to make accurate stock movement predictions by correlating multiple stocks. The following challenges arise from the properties of stock prices:

(1) **Considering multivariate features.** The price of a stock is naturally represented as multiple features of a day. However, previous approaches for computing stock correlations use only univariate features such as the closing prices. *How can we utilize multivariate features for accurate correlations?*
(2) **Capturing global movements.** The correlation between stocks is determined by the global movement of the market, not only by their local movements. For instance, stocks are highly correlated in a bull market, where their prices keep rising. *How can we incorporate the global market trend with the local movements of stocks in an end-to-end way?*
(3) **Modeling asymmetric and dynamic relationships.** The true correlations between stocks are asymmetric, as the stock prices change in an asynchronous manner due to the different speed of information diffusion. They also change dynamically over time. *How can we consider the asymmetric and dynamic nature of stock correlations in our framework?*

We address the aforementioned challenges by building an end-to-end framework that considers both the multivariate features and global market trend for computing the correlations between stocks in an asymmetric and dynamic way. Figure 2 depicts the resulting framework, DTML, which consists of the three main modules each of which is designed carefully to address each challenge:

(1) **Attentive context generation (Section 3.2).** We capture the temporal correlations within the historical prices of each stock by attention LSTM that utilizes multivariate features of each day. We use the generated context vectors, instead of raw prices, as the key of finding stock correlations.
(2) **Multi-level context aggregation (Section 3.3).** We generate multi-level context vectors by combining a) local contexts generated from individual stocks and b) a global market context generated from historical index data. As a result, the global movement is naturally incorporated in the individual contexts, affecting the inter-stock correlations.
(3) **Data-axis self-attention (Section 3.4).** We use the transformer encoder to correlate the generated multi-level contexts by asymmetric attention scores. The attention maps are naturally incorporated in the final predictions, while giving a novel insight on the market as interpretable correlations that change dynamically over time.

3.2 Attentive Context Generation

The first idea is to summarize the multivariate historical prices of each stock into a single context vector for the subsequent modules. Given feature vectors $\{\mathbf{z}_{ut}\}_{t \le T}$ of length l, where u and t are stock and time indices, respectively, we aim to learn a comprehensive context $\mathbf{h}^c_u$ that summarizes its local movements until the current step T. We adopt the attention LSTM [25] for the purpose, because it effectively aggregates the state vectors of all RNN cells based on the temporal attention with the last state vector at step T.

Feature Transformation. We first transform every feature vector $\mathbf{z}_{ut}$ by a single layer with the tanh activation:

$$\tilde{\mathbf{z}}_{ut} = \tanh(\mathbf{W}_s \mathbf{z}_{ut} + \mathbf{b}_s), \tag{1}$$

where the parameters $\mathbf{W}_s \in \mathbb{R}^{h \times l}$ and $\mathbf{b}_s \in \mathbb{R}^h$ are shared for all u and t. This makes a new representation of features before applying LSTM over the time series, improving the learning capacity without increasing the complexity of aggregating over multiple time steps [9]. From now, we omit the stock symbol u for simplicity.

LSTM. Long short-term memory (LSTM) [11] learns a representation of a time series by updating two state vectors through the time series. Given a feature vector $\tilde{\mathbf{z}}_{ut}$ and the state vectors $\mathbf{h}_{t-1}$ and $\mathbf{c}_{t-1}$ of the previous time step, an LSTM cell computes the new

state vectors $\mathbf{h}_t$ and $\mathbf{c}_t$, which are fed into the next cell. The state vector $\mathbf{h}_T$ at the last step becomes the final representation of the time series that summarizes all historical information.

Attention LSTM. Instead of using the last hidden state $\mathbf{h}_T$ as an output, we adopt the attention mechanism to combine the hidden states of all LSTM cells. Given the hidden states $\mathbf{h}_1, ..., \mathbf{h}_T$, attention LSTM computes the attention score α_i of each step i using the last hidden state $\mathbf{h}_T$ as the query vector:

$$\alpha_i = \frac{\exp(\mathbf{h}_i^\top \mathbf{h}_T)}{\sum_{j=1}^{T} \exp(\mathbf{h}_j^\top \mathbf{h}_T)}. \tag{2}$$

The context vector $\tilde{\mathbf{h}}^{\mathrm{c}}$ is computed as $\tilde{\mathbf{h}}^{\mathrm{c}} = \sum_i \alpha_i \mathbf{h}_i$. The attention score α_i of step i measures the importance of step i with regard to the current step T. The attention effectively resolves the limitation that LSTM forgets the information of previous steps [2].

Context Normalization. The context vectors generated by the attention LSTM have values of diverse ranges, because each stock has its own range of features and pattern of historical prices. Such diversity makes subsequent modules unstable, such as the multi-level context aggregation (Section 3.3) or data-axis self-attention (Section 3.4). We thus introduce a context normalization, which is a variant of the layer normalization [1]:

$$h^{\mathrm{c}}_{ui} = \gamma_{ui} \frac{\tilde{h}^{\mathrm{c}}_{ui} - \mathrm{mean}(\tilde{h}^{\mathrm{c}}_{ui})}{\mathrm{std}(\tilde{h}^{\mathrm{c}}_{ui})} + \beta_{ui}, \tag{3}$$

where i is the index of an element in a context vector, $\mathrm{mean}(\cdot)$ and $\mathrm{std}(\cdot)$ are computed for all stocks and elements, and γ_{ui} and β_{ui} are learnable parameters for each pair (u, i).

3.3 Multi-Level Context Aggregation

Stocks comprise a market, and the global movement of a market is a fundamental factor that affects the correlations between stocks. Such global movements are typically represented as market index, such as NDX100 in the US market or CSI300 in the China market. In a market with strong movements, the amount that a stock correlates with the global movement determines its basic influence to the other stocks, since most stocks eventually follow the market movement in a long-term perspective regardless of the short-term fluctuation or the properties of individual stocks.

Thus, we propose to utilize additional index data for capturing the global movement and then to use it as base knowledge of stock correlations. We are given a historical market index such as SNP500 on the same time range as the individual stocks, which reflects the global movement of the market. For instance, we use CSI300 for the China stock market. We apply the attention LSTM to the historical index to generate a market context $\mathbf{h}^i$, which is considered as an effective summarization of all individual contexts.

Multi-Level Contexts. Then, we generate a multi-level context $\mathbf{h}^m_u$ for each stock u by using the global market context $\mathbf{h}^i$ as base knowledge of all correlations:

$$\mathbf{h}^m_u = \mathbf{h}^c_u + \beta \mathbf{h}^i, \tag{4}$$

where β is a hyperparameter that determines the weight of $\mathbf{h}^i$. As a result, the correlation between two stocks is determined not only by their local movements, but also by the relationship to the global market context $\mathbf{h}^i$.

The Effect of Global Contexts. Assume that we use a simple attention by the dot product to calculate the correlation between two stocks u and v. The result of dot product is given as

$$\mathbf{h}^{m\top}_u \mathbf{h}^m_v = \mathbf{h}^{c\top}_u \mathbf{h}^c_v + \beta \mathbf{h}^{i\top}(\mathbf{h}^c_u + \mathbf{h}^c_v) + \beta^2 \mathbf{h}^{i\top}\mathbf{h}^i. \tag{5}$$

The first term in the right hand side is the same as the dot product between individual stock contexts. The difference comes from the second and third terms, where the global context $\mathbf{h}^i$ participates in increasing the amount of correlation. The second term gives larger weights to the stocks whose movements correlate with the global movement, reflecting our motivation for multi-level contexts. The third term becomes the background value of correlations between all stocks, considering the strength of market movements.

3.4 Data-Axis Self-Attention

The last step is to correlate the computed contexts and feed them to the final predictor. We use the transformer encoder [6, 27] for the correlations due to the following advantages. First, it computes *asymmetric* attention scores by learning different query and key weight matrices, reflecting the information diffusion in stock markets. Second, it imposes no locality of the target stocks. This works in the opposite way when computing the temporal stock contexts by attention LSTM, because we intentionally focus on the hidden states of recent steps as they contain the most information.

Self-Attention. To apply the self-attention with respect to the data-axis, we first build a multi-level context matrix $\mathbf{H} \in \mathbb{R}^{d \times h}$ by stacking $\{\mathbf{h}^m_u\}_u$ for $u \in [1, d]$, where d is the number of stocks and h is the length of context vectors. We then generate query, key, and value matrices by learnable weights of size $h \times h$:

$$\mathbf{Q} = \mathbf{H}\mathbf{W}_{\mathrm{q}} \qquad \mathbf{K} = \mathbf{H}\mathbf{W}_{\mathrm{k}} \qquad \mathbf{V} = \mathbf{H}\mathbf{W}_{\mathrm{v}}. \tag{6}$$

Then, we compute the attention scores from the query and key vectors, and aggregate the value vectors as follows:

$$\tilde{\mathbf{H}} = \mathbf{S}\mathbf{V} \quad \text{where} \quad \mathbf{S} = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{h}}\right). \tag{7}$$

The softmax function is applied along the rows of $\mathbf{Q}\mathbf{K}^\top$ to apply an attention to the row vectors in $\mathbf{V}$. The generated attention matrix $\mathbf{S}$ tells the relationships between stocks in an asymmetric way, as a form of influence rather than symmetric correlations. Specifically, S_{ji} represents the amount of influence that stock i makes to the prediction of stock j at the current time step. $\mathbf{S}$ is also returned as the output of DTML as valuable information about the market.

The attention scores are divided by $\sqrt{h}$, since high-dimensional contexts are prone to generate sharp scores that are close to one-hot vectors, which select only a few values. We also use the multi-head attention that applies Equation (7) m times with different sets of $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$, and concatenates the results. The output of each head is of size h/m. The attention matrix $\mathbf{S}$ that we give as an output of DTML is calculated as the average for all attention heads.

Nonlinear Transformation. We update the aggregated contexts with residual connections as follows:

$$\mathbf{H}_{\mathrm{p}} = \tanh(\mathbf{H} + \tilde{\mathbf{H}} + \mathrm{MLP}(\mathbf{H} + \tilde{\mathbf{H}})), \tag{8}$$

where the MLP transforms each context vector with changing its size as $h \to 4h \to h$ by one hidden layer of size $4h$ with the ReLU activation. This is done for refining the representations of contexts,

Table 1: Classification accuracy (ACC) and the Matthews correlation coefficients (MCC) of our DTML and the baselines. DTML gives the state-of-the-art accuracy in all six datasets with up to 3.6 points higher ACC and 10.8 points higher MCC over the best competitors, which are significant amounts considering the difficulty of the problem.

Model	ACL18 (US)		KDD17 (US)		NDX100 (US)	
	ACC	MCC	ACC	MCC	ACC	MCC
LSTM [24]	0.4987 ± 0.0127	0.0337 ± 0.0398	0.5118 ± 0.0066	0.0187 ± 0.0110	0.5263 ± 0.0003	0.0037 ± 0.0049
ALSTM [31]	0.4919 ± 0.0142	0.0142 ± 0.0275	0.5166 ± 0.0041	0.0316 ± 0.0119	0.5260 ± 0.0007	0.0028 ± 0.0084
StockNet [31]	0.5285 ± 0.0020	0.0187 ± 0.0011	0.5193 ± 0.0001	0.0335 ± 0.0050	0.5392 ± 0.0016	0.0253 ± 0.0102
Adv-ALSTM [9]	0.5380 ± 0.0177	0.0830 ± 0.0353	0.5169 ± 0.0058	0.0333 ± 0.0137	0.5404 ± 0.0003	0.0046 ± 0.0090
DTML (proposed)	**0.5744 ± 0.0194**	**0.1910 ± 0.0315**	**0.5353 ± 0.0075**	**0.0733 ± 0.0195**	**0.5406 ± 0.0037**	**0.0310 ± 0.0193**

Model	CSI300 (China)		NI225 (Japan)		FTSE100 (UK)	
	ACC	MCC	ACC	MCC	ACC	MCC
LSTM [24]	0.5367 ± 0.0038	0.0722 ± 0.0050	0.5079 ± 0.0079	0.0148 ± 0.0162	0.5096 ± 0.0065	0.0187 ± 0.0129
ALSTM [31]	0.5315 ± 0.0036	0.0625 ± 0.0076	0.5060 ± 0.0066	0.0125 ± 0.0139	0.5106 ± 0.0038	0.0231 ± 0.0077
StockNet [31]	0.5254 ± 0.0029	0.0445 ± 0.0117	0.5015 ± 0.0054	0.0050 ± 0.0118	0.5036 ± 0.0095	0.0134 ± 0.0135
Adv-ALSTM [9]	0.5337 ± 0.0050	0.0668 ± 0.0084	0.5160 ± 0.0103	0.0340 ± 0.0201	0.5066 ± 0.0067	0.0155 ± 0.0140
DTML (proposed)	**0.5442 ± 0.0035**	**0.0826 ± 0.0074**	**0.5276 ± 0.0103**	**0.0626 ± 0.0230**	**0.5208 ± 0.0121**	**0.0502 ± 0.0214**

since the self-attention does not impose additional nonlinearity. It also contains two residual connections to learn the identity function if needed: one for the self-attention and another for the MLP. We also apply the dropout [26] and layer normalization [1] after the attention and nonlinear transformation, respectively, which are not shown in Equation (8) for simplicity.

Final Prediction. We lastly apply a single linear layer to the transformed contexts to produce the final predictions as

$$\hat{\mathbf{y}} = \sigma(\mathbf{H}_{\mathrm{p}}\mathbf{W}_{\mathrm{p}} + \mathbf{b}_{\mathrm{p}}). \tag{9}$$

We apply the logistic sigmoid function σ to interpret each element $\hat{y}_u$ for stock u as a probability and use it directly as the output of DTML for stock movement prediction.

3.5 Training with Selective Regularization

The training of DTML is done to minimize the cross entropy loss between its predictions and true stock movements:

$$\mathcal{L}(\mathcal{X}, \mathbf{y}) = -\frac{1}{d}\sum_{u}\big(y_u \log \hat{y}_u + (1-y_u)\log(1-\hat{y}_u)\big), \tag{10}$$

where $\mathcal{X} \in \mathbb{R}^{w\times d\times l}$ is an input tensor of the current time step, and $\mathbf{y}$ is the true stock movements. w is the length of observations, d is the number of stocks, and l is the number of features.

Selective Regularization. The L2 regularization of model parameters is a popular technique to avoid the overfitting of deep neural networks, which is to add to the objective function the L2 norm of all learnable parameters multiplied with a coefficient λ. The main limitation is the difficulty of tuning the optimal value of λ, which can disturb the proper training of a network.

As a solution, we adopt the approach of [9] which is to penalize the parameters of only the last predictor, which is in Equation (9) in our case. The resulting objective function is given as

$$\mathcal{L}_{\mathrm{reg}}(\mathcal{X}, \mathbf{y}) = \mathcal{L}(\mathcal{X}, \mathbf{y}) + \lambda(\|\mathbf{W}_{\mathrm{p}}\|_{\mathrm{F}}^2 + \|\mathbf{b}_{\mathrm{p}}\|_2^2). \tag{11}$$

As a result, the regularizer restricts the output space but preserves the representation power of core modules including the attention LSTM and transformer encoder. This makes it possible to adopt large λ without harming the accuracy, while improving its robustness for stock markets with noisy prices.

4 EXPERIMENTS

We present experimental results to answer the following research questions about our DTML:

- **Q1**. **Prediction Accuracy (Section 4.2):** Does DTML produce higher accuracy of stock movement prediction compared to previous approaches?
- **Q2**. **Investment Simulation (Section 4.3):** Is DTML effective for making profits in investments? Is the result consistent in different stock markets from various countries?
- **Q3**. **Interpreting Attention Maps (Section 4.4):** Does DTML perform meaningful attention correlations between stocks and between times? Do they give novel insights?
- **Q4**. **Ablation Study (Section 4.5):** Does each module of DTML contribute to improving the prediction accuracy?

4.1 Experimental Setup

We present datasets, competitors, and evaluation metrics for our experiments on stock movement prediction. All of our experiments were done in a workstation with GTX 1080 Ti.

Datasets. We use six datasets in experiments, which are summarized in Table 2. ACL18 [31] and KDD17 are public datasets that were used in [9]. We take the preprocessed versions from their code repository, and use the same training, validation, and test splits (see Table 2 for URL). NDX100, CSI300, NI225, and FTSE100 are new benchmark datasets that we collected from the US, China, Japan, and UK stock markets, respectively. Given the stocks prices until

Table 2: Summary of datasets.

Dataset	Country	Stocks	Days	From	To
ACL18[1]	US	87	652	2014-01-01	2016-01-01
KDD17[1]	US	50	2,518	2007-01-01	2017-01-01
NDX100	US	95	1,258	2013-01-02	2017-12-29
CSI300	China	219	1,120	2015-06-01	2019-12-30
NI225	Japan	51	856	2016-07-01	2019-12-30
FTSE100	UK	24	1,134	2014-01-06	2018-07-01

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

Table 3: Generating a feature vector $\mathbf{z}_{ut}$ of stock u at day t. adj_close_t represents an adjusted closing price preserving the continuity of prices regardless of stock splits.

Features	Calculation
z_{open}	$z_{\text{open}} = \text{open}_t / \text{close}_t - 1$
z_{high}	$z_{\text{high}} = \text{high}_t / \text{close}_t - 1$
z_{low}	$z_{\text{low}} = \text{low}_t / \text{close}_t - 1$
z_{close}	$z_{\text{close}} = \text{close}_t / \text{close}_{t-1} - 1$
$z_{\text{adj\_close}}$	$z_{\text{adj\_close}} = \text{adj\_close}_t / \text{adj\_close}_{t-1} - 1$
z_{d5}, z_{d10} z_{d15}, z_{d20} z_{d25}, z_{d30}	e.g., $z_{dk} = \frac{\sum_{i=0}^{k} \text{adj\_close}_{t-i}}{k \cdot \text{adj\_close}_t} - 1$

day t, we aim to predict the price movement at the next day as either rise ($y_i = 1$) or fall ($y_i = 0$), as in [9, 31].

Feature Vectors. We generate a feature vector $\mathbf{z}_{ut}$ that describes the trend of stock u at day t as in [9]. The features that we use are shown in Table 3. z_{open}, z_{high}, and z_{low} represent the relative values of the opening, the highest, and the lowest prices compared to the closing price of the day, respectively. z_{close} and $z_{\text{adj\_close}}$ represent the relative values of the closing and the adjusted closing prices compared with day $t-1$, respectively. z_{dk} represents a long-term trend of the adjusted closing prices during the previous k days.

Competitors. We compare DTML with the following baselines for stock movement prediction.

- **LSTM** [24] is the simplest baseline for stock movement prediction, which uses the simple LSTM for prediction.
- **ALSTM** [25] represents attention LSTM that correlates multiple time steps using a temporal attention mechanism.
- **StockNet** [31] uses variational autoencoders to encode stock movements as latent probabilistic vectors.
- **Adv-ALSTM** [9] is the previous state-of-the-art model that trains ALSTM with adversarial data to improve the robustness of stock movement prediction.

Evaluation. We run 10 experiments with different random seeds in $[0, 9]$, and report the average and standard deviation. We evaluate the result of stock movement prediction by two metrics: accuracy (ACC) and the Matthews correlation coefficient (MCC) [31]. ACC is the ratio of correct predictions over all test examples. MCC is a balanced metric that can be used even if the two classes have

(a) 2015-10-01 **(b) 2015-10-30**

Figure 3: The attention scores between stocks, generated by DTML for the ACL18 dataset. The stocks are ordered by the influence score $f(u) = \sum_i S_{iu}$. Each matrix S is divided into four regions; the region C shows that a few stocks have large influence to the other stocks, while the region B shows that most stocks rely on the predictions for the other stocks.

different sizes, which is defined as follows:

$$\frac{tp \times tn - fp \times fn}{\sqrt{(tp + fp)(tp + fn)(tn + fp)(tn + fn)}}, \tag{12}$$

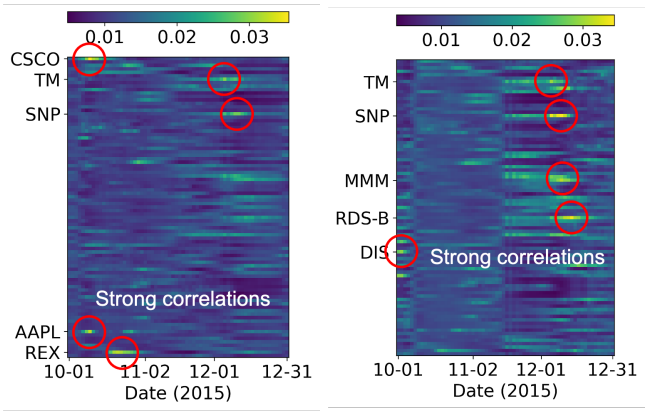
where tp, tn, fp, and fn represent the true positives, true negatives, false positives, and false negatives on test examples, respectively.

Hyperparameters. We search the hyperparameters of DTML as follows: the window size w in $\{10, 15\}$, the market context weight β in $\{0.01, 0.1, 1\}$, the hidden layer size h in $\{64, 128\}$, the number of epochs in $\{100, 200\}$, and the learning rate in $\{0.001, 0.0001\}$. We set the strength λ of selective regularization to 1 and the dropout rate to 0.15. We use the Adam optimizer [14] for the training with the early stopping by the validation accuracy. For competitors, we use the default settings in their public implementations.

4.2 Prediction Accuracy (Q1)

Table 1 compares the accuracy of our DTML and baselines for stock movement prediction in the six datasets. DTML achieves the state-of-the-art accuracy in all datasets with consistent improvements in both ACC and MCC over the previous approaches. The improvement is more significant by MCC, which evaluates the prediction in a more balanced manner than ACC.

The high accuracy of Adv-ALSTM, the previous state-of-the-art model on our datasets, is made by improving the robustness via adversarial training. DTML achieves the same objective by correlating different stocks; the stock correlations act as a regularizer that restrict predictions from being random and noisy. Our context aggregation strengthens this regularization by considering the market movement, which is more stable than individual stock prices. As a result, DTML outperforms Adv-ALSTM with better generalization, showing the highest accuracy in all datasets.



(a) Amazon (AMZN)

(b) Google (GOOG)

Figure 4: The attention scores for AMZN and GOOG learned by DTML in the ACL18 dataset. The scores change smoothly over time, adapting to the price movement at each moment. We annotate the top five correlations at each matrix.

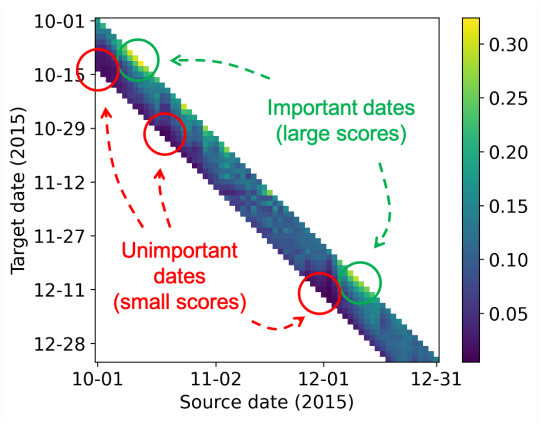


Figure 5: The temporal attention scores generated by DTML for Google (GOOG) in the ACL18 dataset. The green and the red circles denote important and unimportant dates, respectively Important dates are used actively in the future predictions as shown by the large scores.

4.3 Investment Simulation (Q2)

We simulate actual investment using the predictions of models in Figure 1. At the end of each day, we rebalance our portfolio using top three stocks with the highest rise probability. We also visualize the market indices as baselines: SNP500 is used for the ACL18 and KDD17 datasets, where the original data have no index information. DTML makes up to 13.8%p higher profits than the best competitors, resulting in the annualized return of up to 44.4% (in NI225).

LSTM and Adv-ALSTM experience the sudden drops of portfolio values in ACL18, NDX100, and FTSE100. This makes the predictions of such models unreliable, even though a high profit is observed at a few moments during the investment; one needs to manually stop the investment if the portfolio reaches a certain value to avoid such drops. This is because they ignore the correlations between stocks and are vulnerable to the randomness of stock prices. DTML makes consistent and reliable predictions through the multi-level contexts and stock correlations that utilize the overall movement of market, resulting in superior profits. The difference is shown well in CSI300, where the baselines end up with the portfolio values smaller than 1 which means a negative profit, while DTML makes 11.1% profit in the same period, which is 10.3%p higher than the index.

4.4 Interpreting Attention Maps (Q3)

Figure 3 visualizes the attention map $\mathbf{S}$ of Equation (7), learned by DTML for the ACL18 dataset. Each entry S_{ji} represents the amount of influence that stock i gives to stock j. We reorder the stocks by the influence score $f(u) = \sum_j S_{ju}$ to compare the overall influence of stocks. The figure shows that a few stocks have large influence to the market, leading the prediction at each prediction moment; this supports our motivation that capturing dynamic correlations is essential for modeling accurate relationships between stocks. The list of most influential stocks keeps changing as follows:

- **2015-10-01:** DHR (1.91), DIS (1.83), and WFC (1.71)
- **2015-10-30:** PG (1.61), JPM (1.50), and CHTR (1.33)

where the values represent the influence scores.

We also visualize the attention scores between stocks in Figure 4 with respect to Amazon (AMZN) and Google (GOOG). The scores represent how much AMZN and GOOG consider the other stocks at each moment as a weight vector that sums to one. We have two observations from the figures. First, the attention scores change smoothly over time, showing a temporal locality between adjacent dates. This is natural, because the property of a stock for calculating its correlation does not change instantly at a single moment. Second, strong correlations with other stocks are observed at certain moments. The predictions at such moments use the movements of other stocks as the main information, which is not available by the previous models for individual predictions.

Figure 5 shows the temporal attention scores generated by the attention LSTM of DTML for Google (GOOG). Each entry with the source date i and the target date j represents how much influence date i makes to the prediction at date j; this is the temporal version of the stock attention matrix $\mathbf{S}$. Green circles represent large attention scores that are preserved for future steps. On the other hand, the red circles represent unimportant dates; they are not selected as the evidence of future predictions. This demonstrates the ability of DTML to attend to important observations for each prediction, not just for different stocks, but for previous time steps.

4.5 Ablation Study (Q4)

We compare the accuracy of DTML and its variants where each of the three main modules is removed in Table 4:

- **DTML-TA:** DTML without the temporal attention (TA)
- **DTML-SA:** DTML without the stock attention (SA)
- **DTML-MC:** DTML without the market contexts (MC)
- **DTML-TA-SA-MC:** DTML without TA, SA, and MC

Each module improves the prediction accuracy, and DTML having all three modules produces the best accuracy. We observe that the attention modules TA and SA are more important than MC, as

Table 4: An ablation study of DTML on ACL18. TA, SA, and MC represent the temporal attention, stock attention, and multi-level context, respectively. Each module improves the accuracy, and DTML performs the best with all modules.

Model	ACC	MCC
DTML-TA-SA-MC	0.5349 ± 0.0140	0.0828 ± 0.0246
DTML-TA	0.5574 ± 0.0163	0.1387 ± 0.0334
DTML-SA	0.5622 ± 0.0153	0.1453 ± 0.0205
DTML-MC	0.5724 ± 0.0177	0.1856 ± 0.0313
DTML	**0.5744 ± 0.0194**	**0.1910 ± 0.0315**

they determine the amount of information that DTML can utilize at each prediction; TA provides the information of previous time steps, and SA provides the information of other stocks. MC works as a supportive module on top of the other modules, improving the consistency of stock attention by incorporating the movement of market in the form of multi-level context vectors.

5 CONCLUSION

We propose DTML, a novel framework for stock movement prediction, which efficiently correlates multiple stocks without any prior knowledge. DTML consists of three main steps: a) extracting stock contexts from multivariate features by the temporal attention, b) generating multi-level contexts using a global market context, and c) learning dynamic stock correlations using the transformer encoder. DTML achieves the state-of-the-art accuracy on six datasets for stock movement prediction, which are collected from various stock markets of US, China, Japan, and UK, improving the accuracy and the Matthews correlation coefficients of the best competitors up to 3.6 and 10.8 points, respectively. DTML makes up to 13.8% higher profits than the previous state-of-the-art models, resulting in annualized returns of up to 44.4%. In addition, the learned attention maps give novel insights on stock markets about the relationships between stocks. Future works include extending DTML to consider diverse global features along with local market features.

ACKNOWLEDGMENTS

This work is supported by DeepTrade Inc. U Kang is the corresponding author.

REFERENCES

[1] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer Normalization. *CoRR* abs/1607.06450 (2016). arXiv:1607.06450
[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*.
[3] BS Bini and Tessy Mathew. 2016. Clustering and regression techniques for stock prediction. *Procedia Technology* 24 (2016), 1248–1255.
[4] Defu Cao, Yujing Wang, Juanyong Duan, Ce Zhang, Xia Zhu, Congrui Huang, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, and Qi Zhang. 2020. Spectral Temporal Graph Neural Network for Multivariate Time-series Forecasting. In *NeurIPS*.
[5] Chi Chen, Li Zhao, Jiang Bian, Chunxiao Xing, and Tie-Yan Liu. 2019. Investment Behaviors Can Tell What Inside: Exploring Stock Intrinsic Properties for Stock Trend Prediction. In *KDD*.
[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*.
[7] Daizong Ding, Mi Zhang, Xudong Pan, Min Yang, and Xiangnan He. 2019. Modeling Extreme Events in Time Series Prediction. In *KDD*.
[8] Qianggang Ding, Sifan Wu, Hao Sun, Jiadong Guo, and Jian Guo. 2020. Hierarchical Multi-Scale Gaussian Transformer for Stock Movement Prediction. In *IJCAI*.
[9] Fuli Feng, Huimin Chen, Xiangnan He, Ji Ding, Maosong Sun, and Tat-Seng Chua. 2019. Enhancing Stock Movement Prediction with Adversarial Training. In *IJCAI*.
[10] Michael Hagenau, Michael Liebmann, and Dirk Neumann. 2013. Automated news reading: Stock price prediction based on financial news using context-capturing features. *Decis. Support Syst.* 55, 3 (2013), 685–697.
[11] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.
[12] Ziniu Hu, Weiqing Liu, Jiang Bian, Xuanzhe Liu, and Tie-Yan Liu. 2018. Listening to Chaotic Whispers: A Deep Learning Framework for News-oriented Stock Trend Prediction. In *WSDM*.
[13] Joshi Kalyani, H. N. Bharathi, and Rao Jyothi. 2016. Stock trend prediction using news sentiment analysis. *CoRR* abs/1607.01958 (2016). arXiv:1607.01958
[14] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.
[15] Chang Li, Dongjin Song, and Dacheng Tao. 2019. Multi-task Recurrent Neural Networks and Higher-order Markov Random Fields for Stock Price Movement Prediction. In *KDD*.
[16] Hao Li, Yanyan Shen, and Yanmin Zhu. 2018. Stock Price Prediction Using Attention-based Multi-Input LSTM. In *ACML*.
[17] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. 2019. Enhancing the Locality and Breaking the Memory Bottleneck of Transformer on Time Series Forecasting. In *NeurIPS*. 5244–5254.
[18] Wei Li, Ruihan Bao, Keiko Harimoto, Deli Chen, Jingjing Xu, and Qi Su. 2020. Modeling the Stock Relation with Graph Network for Overnight Stock Movement Prediction. In *IJCAI*.
[19] Zhige Li, Derek Yang, Li Zhao, Jiang Bian, Tao Qin, and Tie-Yan Liu. 2019. Individualized Indicator for All: Stock-wise Technical Indicator Optimization with Stock Embedding. In *KDD*.
[20] Guang Liu, Yuzhao Mao, Qi Sun, Hailong Huang, Weiguo Gao, Xuan Li, Jianping Shen, Ruifan Li, and Xiaojie Wang. 2020. Multi-scale Two-way Deep Neural Network for Stock Trend Prediction. In *IJCAI*.
[21] Qikai Liu, Xiang Cheng, Sen Su, and Shuguang Zhu. 2018. Hierarchical Complementary Attention Network for Predicting Stock Price Movements with News. In *CIKM*.
[22] Saloni Mohan, Sahitya Mullapudi, Sudheer Sammeta, Parag Vijayvergia, and David C. Anastasiu. 2019. Stock Price Prediction Using News Sentiment Analysis. In *BigDataService*.
[23] Kihwan Nam and NohYoon Seong. 2019. Financial news-based stock movement prediction using causality analysis of influence in the Korean stock market. *Decis. Support Syst.* 117 (2019), 100–112.
[24] David M. Q. Nelson, Adriano C. M. Pereira, and Renato A. de Oliveira. 2017. Stock market's price movement prediction with LSTM neural networks. In *IJCNN*.
[25] Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison W. Cottrell. 2017. A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction. In *IJCAI*.
[26] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* 15, 1 (2014), 1929–1958.
[27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NIPS*.
[28] Jia Wang, Tong Sun, Benyuan Liu, Yu Cao, and Hongwei Zhu. 2019. CLVSA: A Convolutional LSTM Based Variational Sequence-to-Sequence Model with Attention for Predicting Trends of Financial Markets. In *IJCAI*.
[29] Jingyuan Wang, Yang Zhang, Ke Tang, Junjie Wu, and Zhang Xiong. 2019. AlphaStock: A Buying-Winners-and-Selling-Losers Investment Strategy using Interpretable Deep Reinforcement Attention Networks. In *KDD*.
[30] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. 2020. Connecting the Dots: Multivariate Time Series Forecasting with Graph Neural Networks. In *KDD*. ACM, 753–763.
[31] Yumo Xu and Shay B. Cohen. 2018. Stock Movement Prediction from Tweets and Historical Prices. In *ACL*. Association for Computational Linguistics, 1970–1979.
[32] Jaemin Yoo, Hyunsik Jeon, and U Kang. 2019. Belief Propagation Network for Hard Inductive Semi-Supervised Learning. In *IJCAI*. ijcai.org, 4178–4184.
[33] Jaemin Yoo and U Kang. 2021. Attention-Based Autoregression for Accurate and Efficient Multivariate Time Series Forecasting. In *SDM*. SIAM, 531–539.
[34] Chen Zhang, Yijun Wang, Can Chen, Changying Du, Hongzhi Yin, and Hao Wang. 2018. StockAssIstant: A Stock AI Assistant for Reliability Modeling of Stock Comments. In *KDD*.
[35] Liheng Zhang, Charu C. Aggarwal, and Guo-Jun Qi. 2017. Stock Price Prediction via Discovering Multi-Frequency Trading Patterns. In *KDD*.