Graduation Project:
Stock Price Prediction Using AI Models

2020095178, Yoonseon Choi
Department of Data Science, Hanyang University, Seoul, Korea

ABSTRACT

Time-series data is intricately intertwined with the flow of our lives. Particularly, stock price data is a prominent example within the realm of financial time-series data. By using the AI models for stock price prediction, we gain the ability to forecast the trends in the financial market and contribute to the formulation of effective financial strategies. Additionally, the surge in stock investors post-COVID-19 has instigated significant changes in our economy. Fueled by these issues, I have developed an interest in stock price data. Consequently, as a graduation project, to deal with long term time series data, I intend to utilize four AI models to predict S&P500 stock data and compare the predictive performance among these models: DLinear [1], Pyraformer [2], PatchTST [3], and ESTIMATE [4].

1 INTRODUCTION

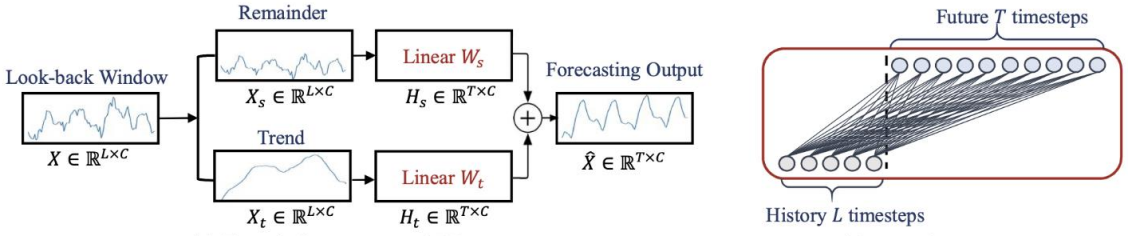
Background. Among various types of data, time series data, in particular, is deeply intertwined with our lives in flow of time. Data obtained from finance, weather, social media, electricity consumption, and more, are all forms of time series data closely associated with our daily existence. However, I have recognized a deficiency in my experience with time series data analysis.
Therefore, since last year, I have been gradually learning techniques for analyzing and predicting time series data. My aim is to strengthen my understanding by applying the theoretical models I've learned to practical situations. To achieve this goal, I decided to conduct experiments using one of the typical forms of time series data, stock market data. Through this experimentation, I intend to compare the prediction performance of the models I have studied.

Motivation. Since COVID-19, the number of individual investors has increased, which has had a great impact on the economy. I paid attention to this situation and tried to deal with stock price data, which is representative time series data. There are several reasons why stock price prediction is necessary. Predicting stock prices can assist in forecasting the flow of financial markets and in the development of financial strategies. Also, it can be of great help to individual investors in making investment decisions and evaluating the value of companies. For these reasons, I believe that undertaking a stock price prediction project is necessary for the better financial economy.
Furthermore, I think this project is an opportunity to think about what kind of research we should do to more accurately predict stock data with many variables and uncertain future. Therefore, I decided to analyze and predict stock data as the topic for my AI project.

This project is carried out using four AI models. I experiment with three representative models of Long-Term Time-Series Forecasting (LTSF) and one model for predicting financial data. I used Pyraformer and PatchTST which are based on transformer and DLinear which is a linear model, as the LTSF models. Also, I used ESTIMATE as the financial data model. To compare the performance of the four models, the S&P500 stock data was trained from each model and I confirmed the predictive performance. Existing transformer-based models (Dlinear, Pyraformer, PatchTST) have been tested with Electricity, ETT, Traffic, and Weather data, but have never

conducted experiments with stock price data. As results of the existing experiment, the order of PatchTST (SOTA), DLinear, and Pyraformer showed good performance. Therefore, I want to check whether similar performance results can be obtained even when the experiment is conducted with stock price data. In addition, I would like to compare the predictive performance of ESTIMATE to transformer-based models that handles general time series data.

2 RELATED WORK



(a) The whole structure of *DLinear* (b) One Linear Layer

Figure 1: Illustration of the Decomposition Linear Model. (a) The overall structure of *DLinear* is shown in Figure 1. (b) The whole process is: $\hat{X} = H_s + H_t$, where $H_s = W_s X_s \in \mathbb{R}^{T \times C}$ are the decomposed trend and remainder features. $W_s \in \mathbb{R}^{T \times L}$ and $W_t \in \mathbb{R}^{T \times L}$ are two linear layers, as illustrated in Figure 1.

DLinear. Until the publication of the paper introducing this model, most LTSF models were based on the transformer architecture, which has been the prevailing trend. The authors raised doubts about the effectiveness of transformer-based models due to their high time and memory complexity. Consequently, they implemented for a simpler linear model to perform LTSF.

In summary of DLinear architecture, DLinear is a fusion of decomposition techniques from Autoformer [5] and FEDformer [6], incorporating linear layers. It initially breaks down raw input data into a trend component using a moving average kernel and a residual (seasonal) component. Following this, a pair of one-layer linear operations is applied to each component, and the final prediction is derived by summing these two features. By explicitly addressing the trend, DLinear boosts the performance compared to a basic linear model, especially when a distinct trend exists in the data.

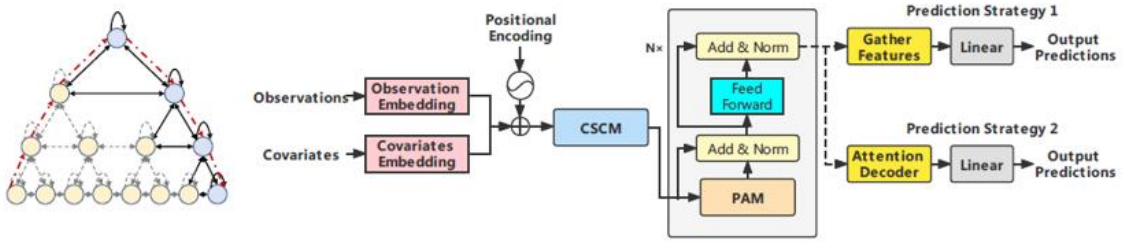


Figure 2: The architecture of Pyraformer: The CSCM summarizes the embedded sequence at different scales and builds a multi-resolution tree structure. Then the PAM is used to exchange information between nodes efficiently.

Pyraformer. This model introduces a novel approach called pyramidal attention to address the long-term dependency issue and alleviate the high time and space complexity, which previous models proposed for LTSF have not fully resolved. This model defines finer data such as Hour, Day, and Week, and coarser data like Year, Quarter, and Month. Nodes carrying time information are constructed in a pyramid shape, starting from finer data to coarser data. Each node exchanges information within both intra-scale and inter-scale.

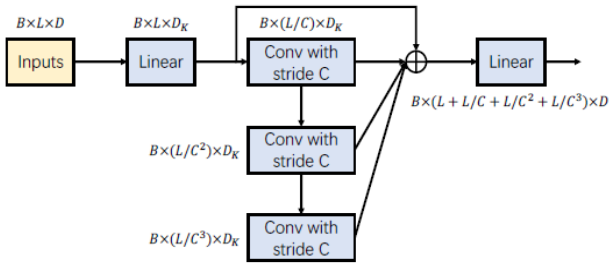
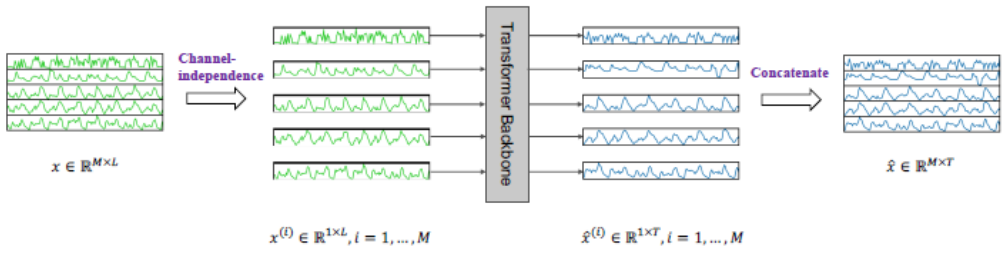
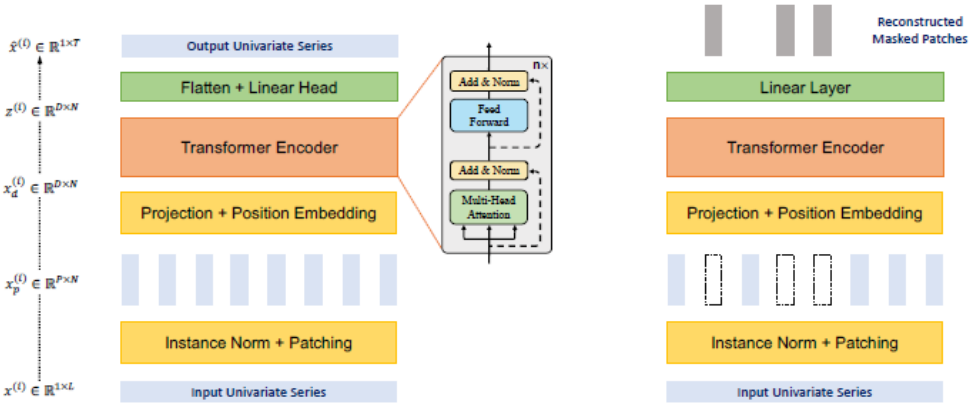


Figure 3. Coarser-scale construction module: B is the batch size and D is the dimension of a node.

The core structure of this model consists of the Coarser-Scale Construction Module (CSCM) and the Pyramidal Attention Module (PAM). CSCM is a module that continuously combines low-scale nodes to complete coarser nodes, forming a pyramidal structure of nodes. As shown in Figure 3, for the efficiency of parameters and computations, dimensionality is reduced before convolution through a fully connected layer, and after convolution, it goes through the process of dimensionality restoration. PAM, on the other hand, is a module that exchanges information between the nodes of the tree passed on by CSCM. It is included in the Transformer encoder. This information exchange is crucial because the upper nodes (coarser data) encapsulate information over a longer period, enabling a better understanding of long-range correlations in time series data. Meanwhile, the lower nodes (finer data) hold detailed information over shorter periods, allowing for a better understanding of short-range correlations.



(a) PatchTST Model Overview



(b) Transformer Backbone (Supervised) (c) Transformer Backbone (Self-supervised)

Figure 4: PatchTST architecture. (a) Multivariate time series data is divided into different channels. They share the same Transformer backbone, but the forward processes are independent. (b) Each channel univariate series is passed through instance normalization operator and segmented into patches. These patches are used as Transformer input tokens. (c) Masked self-supervised representation learning with PatchTST where patches are randomly selected and set to zero. The model will reconstruct the masked patches.

PatchTST. In the paper "Are Transformers Effective for Time Series Forecasting?", doubts were raised about the effectiveness of transformers in LTSF, as a simple linear model outperformed various transformer-based LTSF models. In this paper, the authors propose the channel-independence patch time series transformer to demonstrate that transformers can indeed be effective in LTSF, contrary to previous research findings. Specifically, PatchTST utilizes sub-series level patches to incorporate locality in transformers, reducing memory usage. Moreover, it employs a channel-independence setting, allowing each variable to be independently learned and thus reducing the computational workload of the transformer.

One of the core modules of this model, patching, captures comprehensive semantic information not available at the point-wise level by aggregating time steps into sub-series patches, thereby enhancing locality. In other words, the point-wise approach used in conventional methods suffered from the loss of local information. By cutting the data into sub-series and creating a single token, it is possible to partially extract information. Additionally, this model performed LTSF using a vanilla transformer encoder.

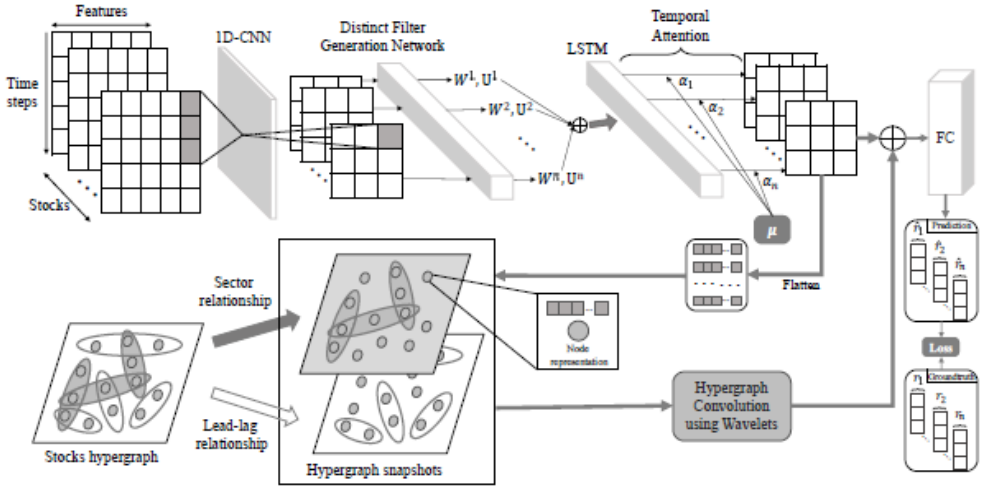


Figure 5: Overview of our framework for stock movement prediction.

ESTIMATE. This model is designed suitable for financial data, and it is predicted by using CNN, LSTM, and Hypergraph modules. The authors considered the following four components when designing this model.

- R1: Multi-dimensional data integration: Consider multivariate factors of stock price data with open-high-low-close-volume (OHLCV) columns.
- R2: Non-stationary awareness: Since stock price data is affected by various factors, it presents a solution to respond against to the unpredictable market behavior.
- R3: Analysis of multi-order dynamics: It analyzes dynamic factors by understanding the complex relationships between stocks.
- R4: Analysis of internal dynamics: Each stock shows individual behaviors, so it should be able to analyze it.

ESTIMATE uses 1D CNN and temporal attention LSTM as a solution to R1. This mitigates the influence (R2) of the noisy market behaviors and serves as a memory for storing the independent information of each stock (R4). Finally, an industry-based hypergraph module is used to deal with multi-order dynamics (R3).

Local trends are identified by using CNN, and temporal LSTM is used to capture temporal dependencies. In addition, wavelet hypergraph convolution is introduced to aggregate the temporal information derived from the industrial hypergraph.

3 EXPERIMENTS

Dataset. The daily S&P 500 dataset of 2526 days per corporation is used from 2012-05-14 to 2022-05-25. There are 474 corporations in this dataset. The dataset includes timestamp, corporation, close, open, high, low, volume, and adjclose columns.

Baseline. The experiment involves dividing the dataset into training, validation, and testing sets with a ratio of 80:10:10. The input data comprises [close, open, high, low, volume, adjclose], and the goal is to predict the data of all columns.

Two types of predictions are conducted for each model: short-term prediction and long-term prediction. For short-term prediction, a lookback window (input sequence length) of 20 and a lookahead window (prediction sequence length) of 1 are set. This means the model utilizes the past 20 days of data to predict the next 1 day's data. On the other hand, long-term prediction involves a lookback window of 96 and a lookahead window of 5. In this case, the model utilizes the past 96 days of data to predict the next 5 days' data.

Metrics. The evaluation metrics for experimental results are MSE (Mean Squared Error), MAE (Mean Absolute Error), RMSE (Root Mean Squared Error), RankIR (Rank Information Ratio), and RankIC (Rank Information Coefficient). Each model's predictive performance is compared using these metrics. Here, RankIR refers to the Pearson correlation coefficient, while RankIC refers to the Spearman correlation coefficient.

(seq, pred)	(20,1)					(96,5)				
Metrics / Models	MSE	MAE	RMSE	RankIC	RankIR	MSE	MAE	RMSE	RankIC	RankIR
DLinear	0.070	0.147	0.264	0.968	**0.974**	0.485	0.250	0.696	0.589	0.656
Pyraformer	0.072	0.123	0.268	0.946	0.876	0.087	0.138	0.295	0.922	0.846
PatchTST	**0.025**	**0.053**	**0.237**	**0.972**	0.971	**0.030**	**0.065**	**0.257**	**0.966**	**0.966**

Table 1: This table show results according to models (DLinear, Pyraformer, PatchTST), lookback window, lookahead window, and evaluation metrics.

Metrics / Models	MAE	RMSE	RankIC
ESTIMATE	0.053	0.572	0.034

Table 2: This table show the result of ESTIMATE experiment.

Results. The papers originated from the Transformer have indicated that PatchTST, DLinear, and Pyraformer exhibit superior performance in that order. Conducting experiments with S&P500 stock price data, as shown in Table 1, the performance of PatchTST stands out as the best across most metrics when compared on a model basis. Consistent with the mentioned papers, at least, with seq_len=20 and pred_len=1 settings, PatchTST outperforms DLinear and Pyraformer. Otherwise, with seq_len=96 and pred_len=5 settings, the performance ranking shifts to PatchTST, Pyraformer, and DLinear.

Furthermore, when compared based on the settings of seq_len and pred_len, it is evident that they exhibit better predictive performance in short-term predictions.

I encountered a significant challenge in training the ESTIMATE model. I have trouble with adjusting the model structure of ESTIMATE, so I could not conduct the comparison of ESTIMATE performance under the same conditions. Even with these limitations, a rough comparison of MAE and RMSE suggests that the performance of ESTIMATE appears to be inferior to the state-of-the-art PatchTST model.

4 FUTURE WORK

- As mentioned earlier, there were difficulties in modifying the ESTIMATE model to match the experimental conditions. To address this, a more detailed understanding of the model structure will be gained, and I will make effort to compare its prediction performance with the other models under the same conditions.
- For a more intuitive assessment, it is necessary to visualize the results. I will conduct visualization to understand the differences between actual data and predicted data.
- I plan to explore the feasibility of training the aforementioned models with general time-series data, in addition to the time-series data used in the paper and the stock price data I've employed. This will be part of the continuation of the research in Professor Kyungsik Han's lab. I will investigate the applicability of VR data to PatchTST to assess its performance.

REFERENCES

[1] Ailing Zeng, Muxi Chen, Lei Zhang, Qiang Xu. Are transformers effective for time series forecasting? *arXiv preprint arXiv:2205.13504, 2022.*

[2] Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X Liu, and Schahram Dustdar. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International Conference on Learning Representations, 2022.*

[3] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, Jayant Kalagnanam. A time series is worth 64 words: long-term forecasting with transformers. In *International Conference on Learning Representations, 2023.*

[4] Thanh Trung Huynh, Minh Hieu Nguyen, Thanh Tam Nguyen, Phi Le Nguyen, Matthias Weidlich, Quoc Viet Hung Nguyen, Karl Aberer. Efficient integration of multi-order dynamics and internal dynamics in stock movement prediction. *arXiv preprint arXiv:2211.07400, 2022.*

[5] Haixu Wu, Jiehui Xu, Jianmin Wang, Mingsheng Long. Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting. Advances in Neural Information Processing Systems, 34, *2021*.

[6] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In International Conference on Machine Learning, *2022*.