21366 term project

lingruop

April 2025

1 Introduction

The crux of this project is to train a model with a large dataset through a self-improving method. Specifically, data from the previous Kaggle Competition (Optiver, Trading at close) was chosen for its size and level of detail at every 10 second interval. In this project, we want to test if combining pre-trained Large Time Series Models and residual correction will give better performance with our chosen dataset.

2 Data & Feature Engineer

Our data comes from a high frequency dataset for auction book. It consists 198 stocks and their information at each 10 seconds 1 hour before the stock market closes. In our paper, we focus on training the data from the stock which id is 1.

There are 26455 rows of data. The raw features are date id, seconds in bucket, imbalance size, imbalance buy sellflag, reference price, matched size, far price, near price, bid price, bid size, ask price, ask size, wap, time id, row id where they are mostly finance terms and indexes. Inside of them, far price and near price has less than a half of the data, so we are not counting it inside the input features. Our target is the 60 second future move in the wap of the stock, less the 60 second future move of the synthetic index.

In this project, we designed features to help the model better understand market behavior and price movements using high-frequency trading data. We created features based on the order book, such as mid-price, bid-ask spread, and liquidity imbalance to reflect buying and selling pressure. We also added rolling averages, volatility, momentum, and mean reversion features to capture short-term trends and price changes over time. Time-related features like the day of the week, minutes in the trading session, and time to market close were included to account for patterns that happen at specific times. Additionally, we used technical indicators like the Relative Volatility Index and Fourier Transform to detect hidden market signals. Finally, we cleaned the data by handling missing and extreme values to ensure stable model training. These features together provide a detailed view of market dynamics to improve prediction performance.

For our target value, we first decided to do difference based on previous knowledge. However, because the predicted value is cumulative, each data point would have cost a non negligible bias that cause greatly decrease in our accuracy. We decided not to apply this transformation to our target and remain its own value.

3 Method

We specifically utilized the TimeMoE-50M from Hugging Face, as it is designed for high-frequency or complex time series and comprises a family of decoder-only time series foundation models with a mixture-of-experts architecture, designed to operate in an auto-regressive manner (Shi et al., 2024). Then aspired by boosting method, we want to test if combining different models will give better results. We choose Xgboost as the other model because it not only prevents overfitting circumstances, but also is an application of boosting method.

Time-MoE is a pre-trained large time-series model that utilized Input Token Embedding, MoE Transformer Block and Multi-resolution Forecasting. In this model, they first use point-wise tokenization for time-series embedding and then employ SwiGLU (Shazeer, 2020) to embed each time series point. The MoE Transformer Block is built upon a decoder-only transformer and optimized it using recent advancement from large language model.(Vaswani, 2017). They also introduce a novel multi-resolution forecasting head, which allows for forecasting at multiple scales at the same time (Shi et al., 2024). The model was then pre-trained using a diversified public datasets with high quality with the following loss function:

$$\mathcal{L}_{\text{ar}}(x_t, \hat{x}_t) = \begin{cases} \frac{1}{2}(x_t - \hat{x}_t)^2, & \text{if } |x_t - \hat{x}_t| \leq \delta, \\ \delta \times \left(|x_t - \hat{x}_t| - \frac{1}{2} \times \delta\right), & \text{otherwise,} \end{cases}$$

Boosting is a machine learning technique that combines many weak learners to build a strong,

accurate model. The procedure starts by training a simple model (weak learner) on the data. After that, it focuses on the errors (residuals) made by this model. Each subsequent model is trained to correct these errors. This process is repeated, with each new model improving upon the mistakes of the previous ones. Finally, all the models are combined (usually by weighted sum) to make the final prediction. The process continues until a set number of models is reached or performance stops improving.

Xgboost(Extreme Gradient Boosting) focuses on fine-tuning the predictions by correcting errors (residuals), thus improving the model's accuracy. This is a boosting method that greatly improves the accuracy of the initial model through focusing on mistake correction rather than raw prediction. XGBoost is especially strong at handling non-linear relationships, which seems to be the case with our residuals given the visualization of the residual plot showing not a very clear linear relationship.At each step, XGBoost adds a new tree to improve predictions by minimizing the following objective function where $\hat{y}_i^{(t-1)}$ = prediction from previous $t-1$ trees, $f_t(x)$ = the new tree added at step t, $\Omega(f_t)$ = regularization term to control complexity :

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t)$$

By choosing the best new tree $f_t(x)$, it can minimizing $\mathcal{L}^{(t)}$ the loss function. The model continues with second order Taylor expansion to approximate the loss. Finally, it computes the score for each leaf and determines the optimal weight for each leaf node.

Our metrics for comparing the model performance before and after training residual are MAE (Mean Absolute Error) and MSE (Mean Squared Error) where MAE $= \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$, MSE $= \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$. MAE focuses on average size of the errors and MSE penalized large errors.

4 Result

4.1 Pre-trained time series model

Though mentioned above that we are not going to use differenced target, we tested the result and gave us a very high MAE in the range of 500 (significantly worse than our below results).

4.1.1 Application on model and visualization

We first train the whole data with Time-MoE to gain its residual as our target. Because in high-frequency trading, a 14-period moving average can smooth out short-term fluctuations while retaining enough sensitivity to detect medium-term trends. We chose a time period of 14 as our moving window to predict the next value. That is, input = x_1.....x_{14}, output =x_{15}. After training, we selected a random time interval and formed Figure 1. For clarity, all plots against time are based on picked time interval.

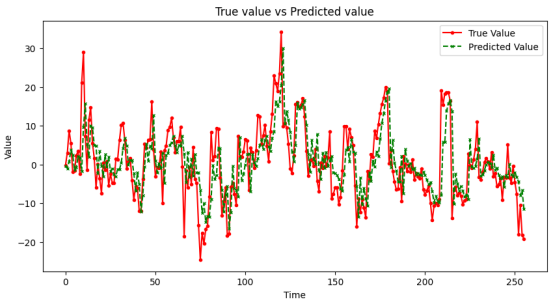

Figure 1: True vs Predict for random interval

In Figure 1, the red solid line with circular markers represents the True Values, and the green dashed line with star markers represents the model's Predicted Values. Both lines follow a similar pattern, indicating that the model captures the overall trend and fluctuations of the true data reasonably well. Although there are slight deviations at certain points—especially around sharp peaks and troughs—the predicted values generally align closely with the true values, suggesting that the model has good predictive performance on this dataset. However, there seems to be a lag between true and predicted value. So we are going to test if the residual fits assumption.

4.1.2 Residual Analysis

Since we use complex models, we don't have a standard assumption for our residuals. But we can still look at its mean, trend and correlation. We first look at the residual in Figure 2.

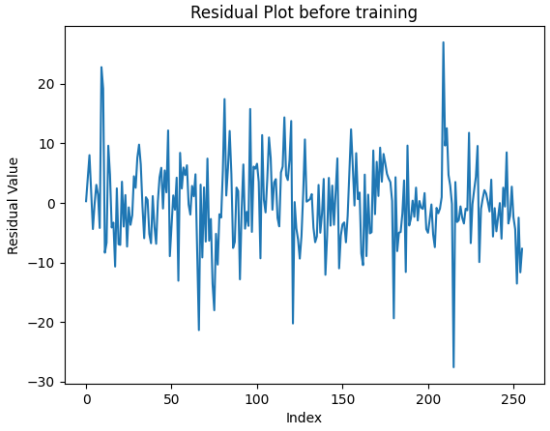

Figure 2: Residual Plot before training

From Figure 2 we can see that the residual is

not very stable. The mean is around 0 but there seems to be a general downward trend. These observations indicates that there should be patterns that the previous model hasn't captured.

We then plot the ACF/PACF plots in Figure 3. These plots Visualizes correlation of residuals with lagged versions of themselves with or without specific effect.

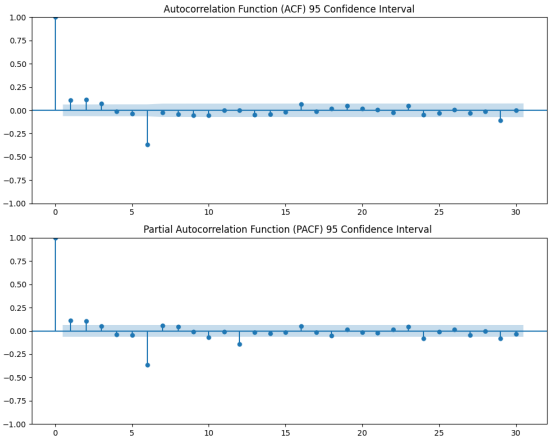

Figure 3: ACF/PACF plots

In Figure 2, the ACF plot is at above and PACF plot is at the bottom. In the ACF plot, most points stay close to zero after the first few lags, which suggests that there isn't strong repeated correlation over time. We utilized 95 % confidence interval such that spikes outside of this range is statistically significant to have correlations. There are few spikes outside of the confidence interval at the beginning show that some small patterns might still exist in the residuals. What's more important is the larger spike at lag 6 indicating there is strong correlation at this point. The PACF plot also shows small spikes in the first few lags, but like the ACF, most values quickly drop near zero and with a strong spike at lag 6. This indicates that while there are minor correlations at short lags, overall, the residuals don't show strong patterns. But the strong spikes at lag 6 still indicates that the model needs more advancements.

Based on previous residual analysis, combining with our proposed idea of training residual, our next step is to apply Xgboost model on our residual column.

4.2 Training on Residual

4.2.1 Training Procedure

To train for residual correction, the dataset is split into training and testing sets by an 80:20 split. Inside the training dataset, we utilized Time-series split to do cross-validation to avoid overfitting. X_{train} and X_{test} are the feature sets, while y_{train} and y_{test} are the residuals from the Transformer model predictions. By time-series split and 5-Cross Validation, we train several XGBoost models and apply GridSearch to find the optimized and generalized model. The test set fits the best model, capturing further adjustments to the Transformer's predictions. The plot below is the procedure of cross-validation with time series split.

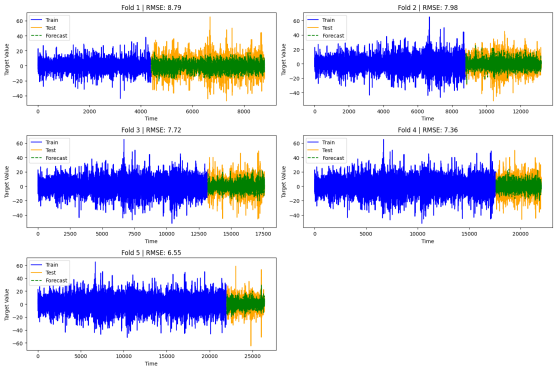

Figure 4: Cross validation- Time series split

The visualization of cross validation shows clearly how the process goes through time. The final prediction is an addition of the Transformer model's predictions and the residual predictions. The following comparing figures were produced to see if there is difference between before and after training:

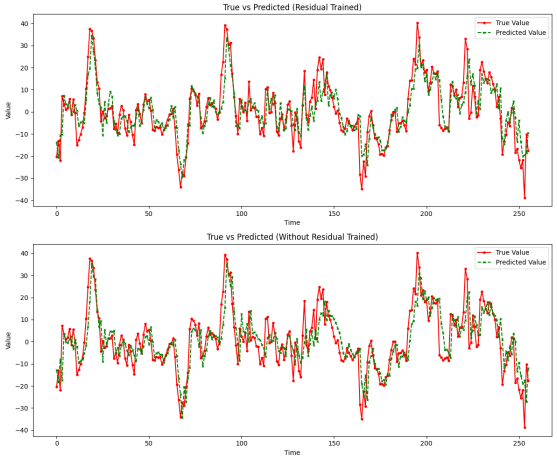

Figure 5: Value Before and After Training

In Figure 5, the top plot is the visualization of predicted value and true value after training residual, the bottom plot is without training. We can see slight improvement in Predicted value verses True value. At the start and middle of the graph, the upper plot fits better than the bottom plot. However, the bottom plot seems to fit better at the end of the graph. We would want metric comparison and testing to identify the differences.

Model & metric	MAE	MSE
After	4.86	46.45
Before	5.89	64.40

Table 1: Metric comparison

From Table 1, we see a huge improvement after training the residual. However, by two sample t-test for identifying the difference between before and after training residual, we found the result not statistically significant with a p-value of 0.84. Though the metric score has better performance, the non-significant difference between the values causes concern. This suggests that while training on the residuals helped reduce error metrics like MAE and MSE, the change is not strong enough to confidently conclude that the improvement is meaningful in a statistical sense. We will still check the residual of the new model and see if it can give us more insights.

4.2.2 Residual Analysis

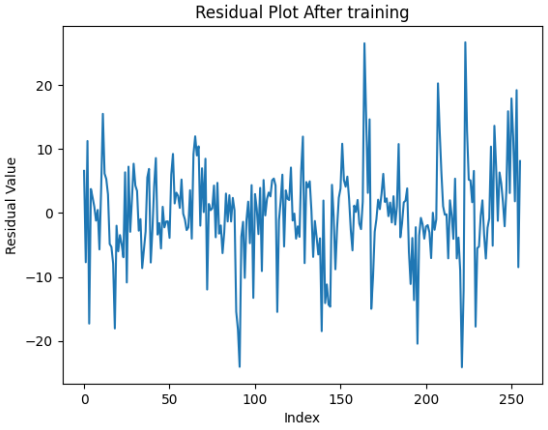

Figure 6: Residual Plot after Training

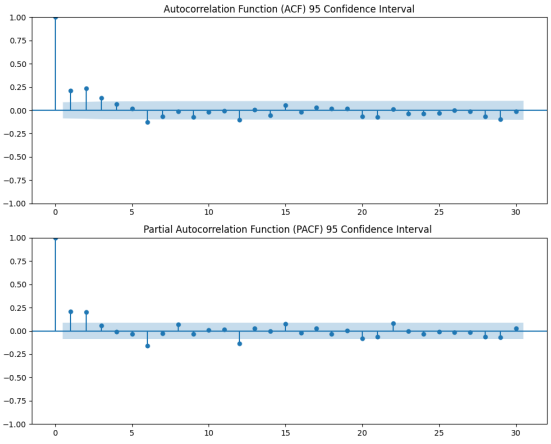

Figure 7: ACF/PACF plot after Training

The Residual plot is approximately similar to those before training residual. ACF and PACF plots still show minor correlation in these values but the large spikes appearing at lag 6 has become smaller. The observation through ACF and PACF indicates that there is some improvement for the residual after it has been trained. This finding enhances the non-significant difference in the previous section but also indicates an improvement of the model fitting.

5 Conclusion

In the results, we notice that although there is a clear improvement in metric scores after training the residual, the difference between the predicted values before and after training is not statistically significant. This suggests that, while the model appears better based on metrics like MAE and MSE, the actual predictive change may not be reliable. Furthermore, the residual plots and ACF/PACF for the model after training indicate that there are still patterns not captured, but the model has better fit based on the observations.

This issue may be attributed to the inherent fluctuations and noise commonly found in financial data, which can make it difficult for models to fully capture underlying patterns. Additionally, since boosting techniques typically require multiple rounds of training to refine predictions, the lack of significant improvement could be a result of overfitting or because the residual patterns have not yet been fully learned. Further tuning or additional training iterations might be needed to achieve more stable and meaningful improvements.

6 Future Work

Given the fitting performance of this method, we can try different models for both parts. For our next step, we will explore more models such as LSTM, RNN and some more complex models and train on the residuals to see if it can have a better fit. Further beyond this, we would like to combine traditional statistical methods and machine learning models for improving overall performance and address the limitations observed in the current method.

7 Reference

1. Shi, X., Wang, S., Nie, Y., Li, D., Ye, Z., Wen, Q., & Jin, M. (2024). *Time-MoE: Billion-scale time series foundation models with mixture of experts*. arXiv. https://arxiv.org/abs/2409.16040

2. Noam Shazeer. *Glu variants improve transformer*. arXiv preprint arXiv:2002.05202, 2020

3. Ashish Vaswani. *Attention is all you need*. arXiv preprint arXiv:1706.03762, 2017