Realized Volatility Prediction of Stock Returns from 'Limit Order Book' using Machine Learning Tools

Creator: Pujan Maharjan

Supervisor: Haiyao Cao

Coordinator: Alfred Krzywicki

Date: 19/11/2023

Research Project

Masters of Artificial Intelligence and Machine Learning

The University of Adelaide

Table of Contents

1. Abstract

Predicting the value of realized volatility as accurately as possible is important because of its importance in the financial market. The realized volatility is used as the input to different risk management models, asset pricing models, and portfolio selection models. If the realized volatility is not predicted well, it might have a cascading effect. In this research, we explained the significance of different feature engineering sets and the stocks' correlation in the short-term forecasting. The total number of features created was 917. The RMSE value was lowest when the number of stocks in the input was 40 and the feature set used was volume-related nearest neighbor features. The stock correlation surpassed the univariate RMSE (0.001806) when the number of input stocks was greater than 13. In another set of experiments, we demonstrated the potential of using multi-dimensional input, automatic feature engineering with TCN, and using transformer-based models - Informer and Autoformer for long-term forecasting.

2. Introduction

Volatility represents the degree of price fluctuations in financial markets. Increased volatility is typically observed during periods of market turbulence, characterized by significant price swings. The value of volatility can be an input to different models to measure risk in the financial market, asset pricing, and portfolio selection (Lin et. al, 2022, p. 1). Accurately predicting the value of volatility has been a research topic for a long period both from the academic and industry. One of the recent competitions to predict realized volatility was hosted by Optiver on Kaggle.com (Kaggle.com, 2021a) which was launched in 2021. The objective of the competition was to predict realized volatility over 10-minute periods for multiple stocks (optiver.com, 2021). The prediction of volatility is not simple because of stylized facts of volatility (Engle and Patton, 2007; Dominique et. al, 1997; Philippe, 2011 in Ge et. al, 2023, p. 2), the efficient market hypothesis (Burton, 1989; Timmermann and Granger, 2004 in Ge et. al, 2023, p. 2), the ephemeral nature of financial relationships (Clarida, Gali and Gertler, 2000; Edwards, Biscarri, Gracia, 2003 in Ge et. al, 2023, p. 2) and others.

Over the period of many years, different models and their variants have been used to predict volatility. The traditional method that was generally used is Generalised AutoRegressive Conditional Heteroscedasticity and its variants. The modern family of models includes Machine Learning like Support Vector Regressor (SVR), Gradient Boosting methods, and Deep Learning models such as Recurrent Neural Networks and their variants like LSTM and GRU, Transformers, and Convolutional Neural Networks

and their variants like Temporal Convolutional Networks. The communities from finance, economy, and computer science are actively involved in the research of predicting volatility.

The research problem for this study is to understand the ways to improve the prediction of realized volatility using ‘Limit Order Book’ (LOB). With this study, we are trying to answer the research question, ‘How can we leverage different machine learning tools to enhance the prediction of the realized volatility using LOB data. In this research, the importance of feature engineering and the correlation of the stocks were investigated in both the short and long term forecasting in the dataset of optiver realized volatility kaggle challenge of 2021. The research question was to For the short term forecasting, different sets of the manual feature engineering were conducted along with a combination of different numbers of stocks in input. Multilayer perceptron (MLP), One dimensional Convolutional Neural Network (1-D CNN), Long Short Term Memory (LSTM), and Transformer were used for the one step prediction. And, for the long term forecasting, the input was changed to four dimensional input which was then transformed to two dimensional using Temporal Convolutional Network (TCN). The TCN acts as automatic feature engineering. This new input was the inputs to two transformer based models Informer and Autoformer for the long term forecasting.

3. Literature Review

For searching academic literature, the ‘library search’ - a service provided by ‘The University of Adelaide’ and ‘Google Scholar’ was used. With the search query “(financ*) AND (“machine learning” OR “deep learning” OR “neural network”) AND (predict* OR forecast*) AND ("realized volatility")” (Ge, et al, 2023, p. 7), as of June 30, 2023, the query returned 111 search results. The search was extended to find recent state of the art forecasting models, where Informer and Autoformer were selected for implementations in this research.

The search results returned papers published in different journals. A few notable ones are ‘Expert Systems with Applications’, ‘Applied Economics Letters’, ‘Journal of Banking and Finance’, ‘China Finance Review International’, ‘Economic Modelling’, ‘Energy’, ‘Journal of Risk and Financial Management’, ‘International Journal of Forecasting’, ‘Royal Statistical Society’ and others. This showed the interest of people of different disciplines - computer science, finance, economics, and statistics in the task of prediction/forecast of realized volatility. Mostly, the papers from the computer science discipline are related to the use of modern machine learning and deep learning algorithms for the task. The algorithms used are either standalone or hybrid models. The most commonly used one is Long short-term memory (LSTM) networks and their combination with others. A large community still prefers variants of traditional statistical models like Heterogeneous AutoRegressive (HAR) and its variants or Generalized AutoRegressive Conditional Heteroskedasticity (GARCH) and its variants such as

Exponential General AutoRegressive Conditional Heteroskedasticity (EGARCH), Glosten-Jagannathan-Runkle GARCH (GJR-GARCH) and so on.

The literature review consists of three categories of study. First, the prediction of realized volatility in general from any type of stock data be it intraday data or historical (OHLC) data. Second, the prediction of realized volatility specific to the Limit Order Book (LOB) data. Third, the prediction of realized volatility from the state of the art forecasting models.

The most common data used in the realized volatility prediction is the historical (Open, High, Low, Close) data. Zhang et al, 2022 used daily historical data to predict the stock volatility using Temporal Convolutional Network. Similarly, Lin, et al, 2022 used intraday OHLC data to predict realized volatility using Echo State Neural Network. Chen and Robert, 2022, used limit order book to predict the realized volatility using Graph Neural Network. The task of predicting realized volatility is neither limited to one source of data nor is limited to a single prediction model. Thus, this research project is different from other research in that the attempt was made to use the prediction of realized volatility for multiple stocks with four dimensional input data and using recent transformer based forecasting models.

4. Project Review

The source of inspiration for the proposed project is based on Kaggle's competition on predicting realized volatility (Kaggle.com, 2021a).

The key activities of the first place winner of the competition are:

- Reverse engineering of time-id order to perform time-series cross-validation
- Features aggregation by using the Nearest neighbor method
- Ensemble of LightGBM, MLP, and 1D-CNN.

Feature Engineering:

The author used the nearest neighbor algorithm to group similar prices, volatility, and trading volume of the stock to predict realized volatility at a certain time-id. Another computed feature was to use the realized volatility of the immediate past time interval as an input to predict the next value of realized volatility. After identifying features, the author detected features that are changing over time by performing adversarial validation, for which he transformed those features into ranks within the same time-id.

Modeling:

The author used an ensemble model of LightGBM, 1D CNN, and MLP (Kaggle.com, 2021b). The pipeline of the model used is displayed in Fig. 1:

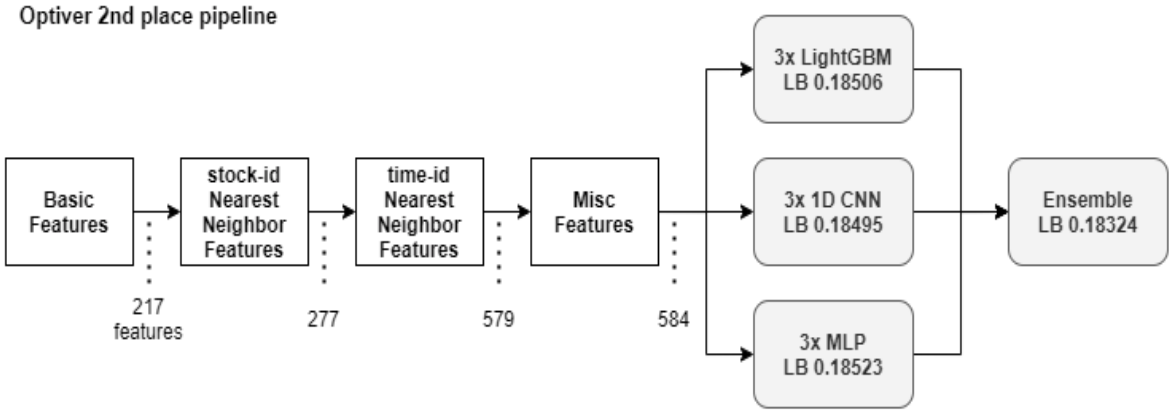

Fig 1: Model pipeline with an ensemble of LightGBM, 1D CNN, and MLP

This research project extends this methodology in two perspectives. First, in the scenario of the existing project, a combination of various feature engineering and various numbers of stocks were tested to see its impact on the prediction of RV in one stock. Second, the input was converted to four dimensional input and new transformer based models were used for long term forecasting, while the objective of the existing project was short term forecasting.

5. Methodology

5.1. Dataset

The dataset used for the study was the “Optiver Realized Volatility Prediction” dataset from Kaggle (Kaggle.com, 2021 a). This dataset comprises stock market information essential for effectively implementing trades in financial markets. Specifically, it encompasses snapshots of order books and completed trades. With a resolution of one second, it offers a remarkably detailed examination of the microstructure of contemporary financial markets (Kaggle.com, 2021b). There are 3 different files for each of the 112 stocks in the dataset.

Book_train.parquet:

The term "order book" describes a digital compilation of buy and sell orders for a particular security or financial instrument, structured by price level. Within an order book, the quantity of shares being bid on or offered at each price point is detailed (Kaggle.com, 2021a).

The book data contains order book details regarding the most competitive buy and sell orders placed in the market. One file consists of data for one stock which has the following columns:

- *stock_id*: ID for the stock.
- *time_id*: ID for time bucket. These IDs are not sequential. However, they are consistent across all stocks.
- *seconds_in_bucket*: Number of seconds from the start of the bucket.

- *bid_price[1 / 2]*: Normalized prices of the first/second most competitive buy level
- *ask_price[1 / 2]*: Normalized prices of the first/second most competitive sell level
- *bid_size [1 / 2]*: The number of shares on the first/second most competitive buy level
- *ask_size [1 / 2]*: The number of shares on the first/second most competitive sell level.

Trade_train.parquet:

The trade data contains trades executed. One file consists of data for one stock which has the following columns:

- *stock_id*: ID for the stock.
- *time_id*: ID for time bucket. These IDs are not sequential. However, they are consistent across all stocks.
- *price*: The normalized average price of trades in one second.
- *size*: The total number of shares traded in one second.
- *order_count*: The count of unique trade orders

Train.csv: This file contains the ground truth values and has the following columns:

- *stock_id*: ID for the stock.
- *time_id*: ID for time bucket. These IDs are not sequential. However, they are consistent across all stocks.
- *target*: The realized volatility over a specific time_id

5.2. Data Processing

The Realized Volatility (RV) was calculated based on the weighted average price of the order book data. First, the weighted average price was calculated from the price and size of bid and ask levels.

Weighted Average Price:

The weighted average price (WAP) was calculated with the following equation:

$$WAP = \frac{BidPrice_1 * AskSize_1 + AskPrice_1 * BidSize_1}{BidSize_1 + AskSize_1}$$

Equation 1: Weighted Average Price

Log Returns:

The returns (stock return) are defined as the percentage change in the price. The log returns were calculated using the following equation 2:

$$r_{t_1,t_2} = \log\left(\frac{S_{t_2}}{S_{t_1}}\right)$$

Equation 2: Log returns

In the formula above, St is the price of the Stock (S) as time (t), and r is the log return over the time periods t1 and t2.

The log returns are often used in financial data analysis because of their important characteristics - stationarity, symmetry and normality, additivity and mathematical convenience.

Realized Volatility

The most common volatility measurements are realized volatility, historical volatility, and implied volatility. The realized volatility (RV) is defined as the square root of the sum of squared log returns within a time window t1 to t2.

$$RV = \sqrt{\sum_{t=\tau_1}^{\tau_2} r_t^2},$$

Equation 3: Realized volatility

In equation 1, RV = Realized volatility, rt = log (Pt / Pt-1) is the log return at time t, Pt is the price at time t, t1 is the initial time of consideration and t2 is the final time of consideration. The realized volatility is generally used with high-frequency data. Some previous works suggest an interval of around 5 to 15 minutes to provide good results (Andersen, et al, 2021; Hsieh, 1991; Schwert, 1991 in Ge et. al 2023, p. 3).

5.3. Data Analysis

Fig 2 shows the line charts of the realized volatility for 4 stocks. Among them, stock 0 seems to have the lowest variation, and stock 9 seems to have the highest variation.

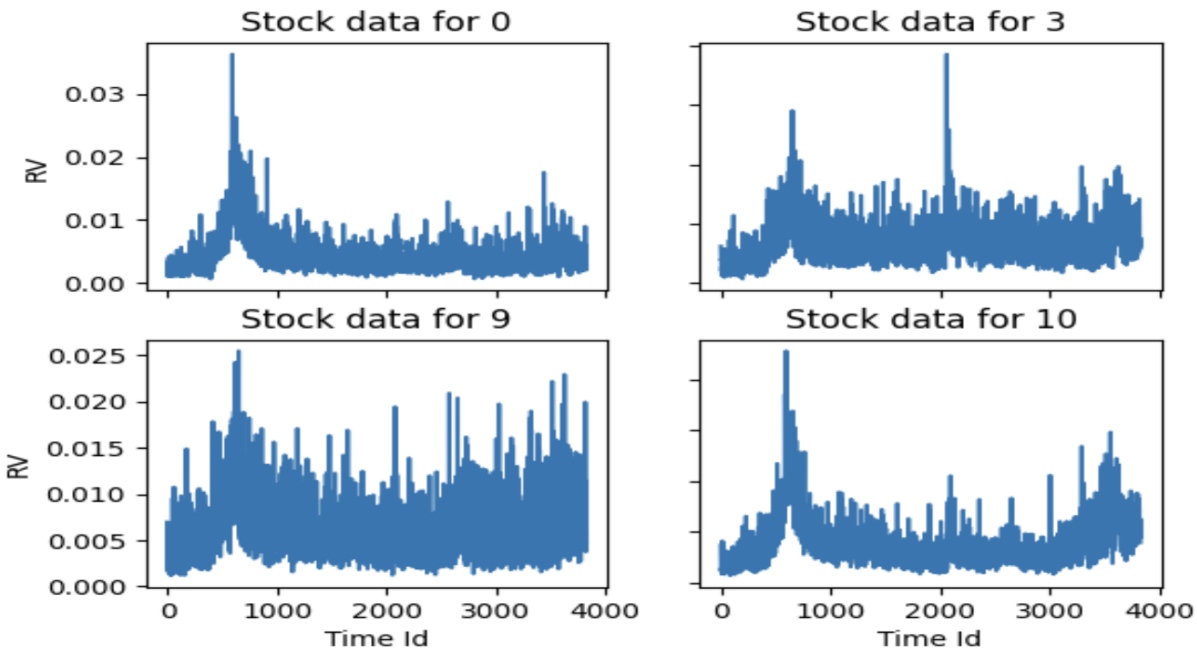

Fig 2: Line plot of realized volatility for 4 stocks

Residual Plot:

The residual plot in Fig 3 shows the Autocorrelation function (ACF) value and distribution of the realized volatility. The ACF plot shows the clear correlation with the lags. The distribution chart does not show any abnormal data, and all data fairly lies together. At around 600 time id, the variation is the highest, and the variation is almost consistent over other time periods.

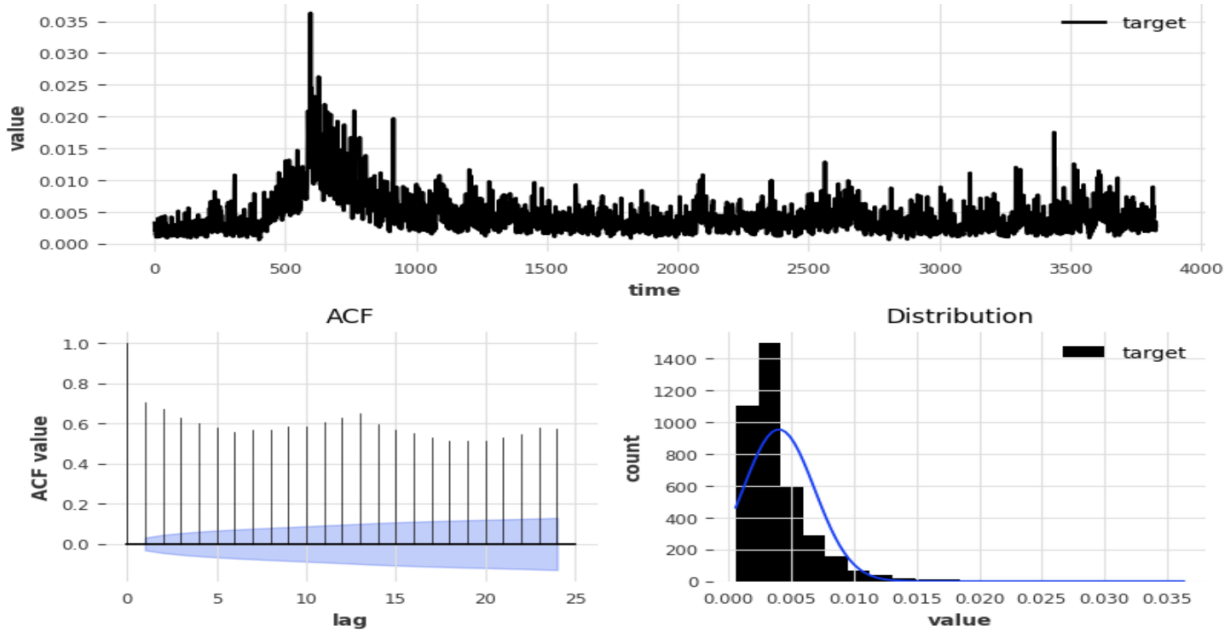

Fig 3: Residual Plot for stock id = 0

5.4. Methods

Different architectures were followed based on the forecasting objective (short-term vs long-term). Architecture 1 was used for one-step ahead (short-term) forecasting and architecture 2 for long-term forecasting.

5.4.1 Architecture 1

A schematic diagram of Architecture 1 is shown in Fig 4.

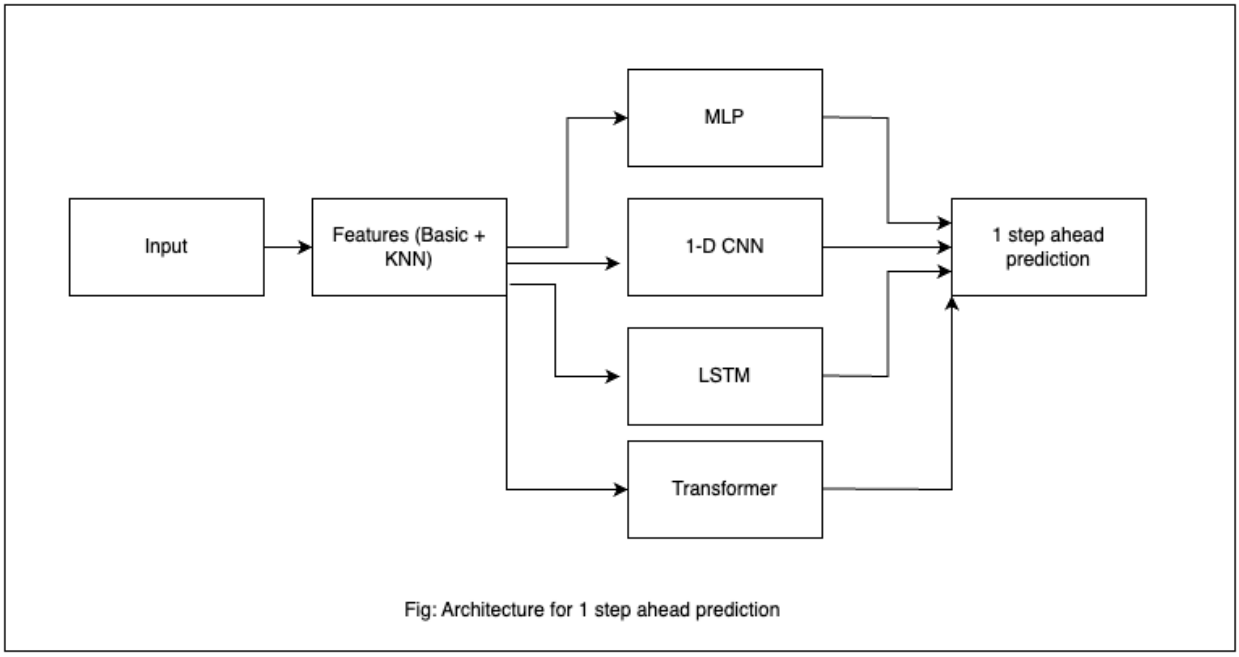

Fig 4: Architecture for one step ahead forecasting

Input:

The input was the two-dimensional data where the columns consist of features and rows consist of data for all stocks. The combination of time_id and stock_id uniquely identifies each row.

Feature Engineering:

Feature engineering consists of creating basic and nearest-neighbor features. The basic features included creating lags, spreads of price, and size of both the bid and ask, and some basic statistical features like sum, standard deviation and mean. In order to extract the relation between the related time frames and related stocks, the nearest neighbor features were calculated based on price, volume, and size.

Models:

Four models were used in the implementation of the Architecture 1. They were Multilayer perceptron (MLP), One-dimensional Convolutional Neural Networks (1-D CNN), Long Short Term Memory (LSTM), and Transformer.

Output:

The output of the models is the one-step-ahead prediction (short-term prediction), which means the prediction was the realized value for next time_id.

5.4.2 Architecture 2

A schematic diagram of Architecture 2 is shown in Fig 5.

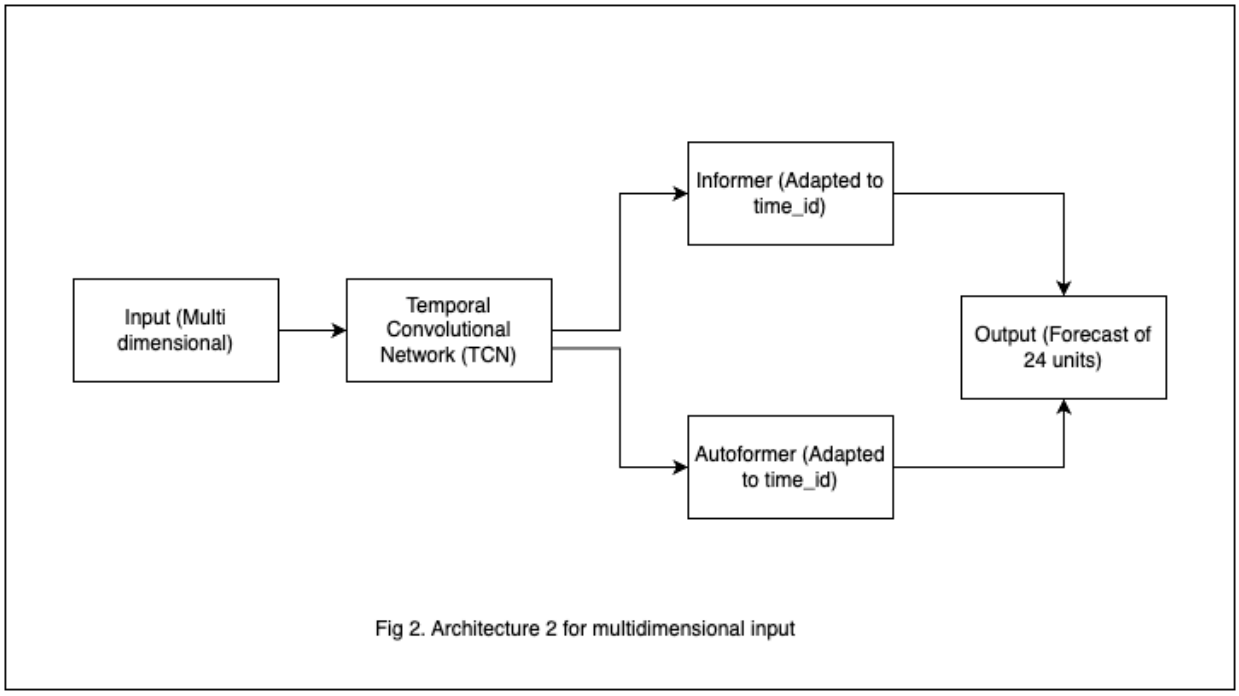

Fig 5: Architecture for long-term forecasting with multidimensional input

Input:

The input was the multidimensional input. It consisted of four dimensions - time_ids (batch), window length, stocks, and features. Fig 6 shows the comparison between the input in Architecture 1 and Architecture 2. In architecture 1, stock_id is considered as the feature input to the models. In the new input structure, we eliminated stock_id from the feature and considered stock data as a new dimension to the input.

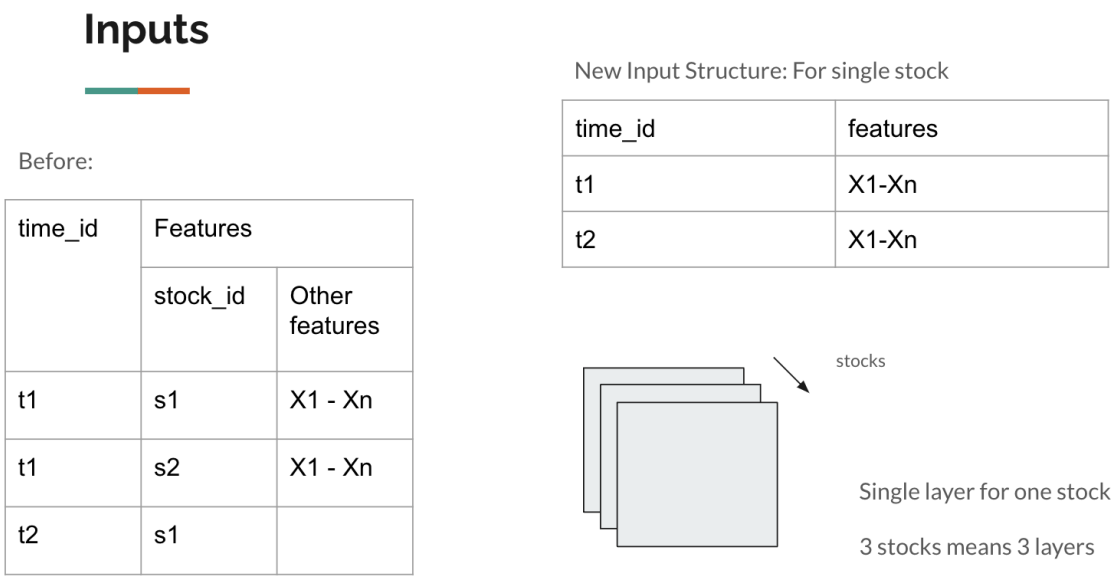

Fig 6: Multidimensional input

Feature Engineering:

The temporal convolutional network (TCN) that was implemented in the DeepTrader paper (Wang et al, 2021) was utilized here to convert the four-dimensional input into two-dimensional input. A limited number of transformed features were created from the original data of the limit order book and trade data.

Models:

The output of the TCN was the input to two forecasting models - informer and autoformer.

Informer:

The Informer (Zhou, et. al, 2021) is a transformer-based model adapted for Long Short-Term Forecasting (LSTF), that features three key attributes:

- Incorporation of a ProbSparse self-attention mechanism, demonstrating a time complexity and memory usage of O(LlogL). This mechanism yields comparable performance in aligning sequences' dependencies.
- Introduction of a self-attention distillation method that emphasizes dominant attention by reducing cascading layer input by half. This approach efficiently manages extremely long input sequences.
- Implementation of a generative-style decoder, which, despite its conceptual simplicity, predicts long time-series sequences in a single forward operation rather than a step-by-step manner. This leads to a significant enhancement in the inference speed for predictions involving long sequences.

Autoformer:

Autoformer (Wu et al, 2022) features Auto-Correlation mechanism. Departing from the traditional approach to series decomposition preprocessing, they transformed it into a fundamental inner block within deep models. This reimagined design endows Autoformer with advanced decomposition capabilities tailored for intricate time series. Additionally, drawing inspiration from stochastic process theory, they crafted the

discovery of dependencies and the aggregation of representations at the sub-series level.

6. Ethics

Data ethics encompasses the assessment of methods employed in producing, gathering, examining, and sharing data that has the potential to negatively impact individuals and society. The dataset that was used in this research has been prepared by following ethical considerations. The data has been anonymised. Both the limit order book data and the trade data does not contain any personal or company information about the buyer and the seller, thus protecting the user's privacy. Furthermore, the stocks identifiers are encoded to stock_id - an integer value, thus giving no information on the exact stock data being dealt with. As far as we understand,the data cannot be reverse engineered to back track any personal information of the traders from this dataset.

7. Results

The experiment results are listed below:

7.1 Feature Engineering

Table 1 consists of the number of columns generated after each type of feature engineering.

Table 1: Features with number of feature columns

	Features	Number of Columns
Basic	Realized volatility lags	254
	Log return	266
	Log return + Spread	277
	Log return + Spread + Statistics	**307**
Basic + Nearest Neighbor	Price Nearest Neighbor	621
	Volume Nearest Neighbor	727
	Size Nearest Neighbor	709
	Random Nearest Neighbor	695
	ALL Nearest Neighbor	**917**

7.2 Predictions with one stock (univariate and multivariate)

The univariate prediction was a baseline to compare for multivariate prediction models.

Table 2: Univariate and Multivariate results for stock id = 0

	Model	RMSE
Univariate (Realized Volatility)	ARCH (AutoRegressive Conditional Heteroskedasticity)	0.001806
	GARCH (General - ARCH)	0.002606
Multivariate (Features from LOB)	Multi-Layer Perceptron (MLP)	0.012859

7.3 Short term forecasting (Architecture 1) Results

The short term forecasting was conducted with different combinations of feature types and different numbers of stocks as input.

Table 3: RMSE with number of stocks and type of feature engineering

Stocks count	Features	RMSE
1	Log Return + Spread + Statistics	0.012859
2	Size Nearest Neighbor (NN)	0.005027
10	Random NN	0.001863
20	Volume NN	0.001456
30	All NN	0.001524
40	**Volume NN**	**0.001274**
50	Volume NN	0.001293
60	Size NN	0.001468
70	All NN	0.001470
80	Size NN	0.001728
90	Volume NN	0.001440

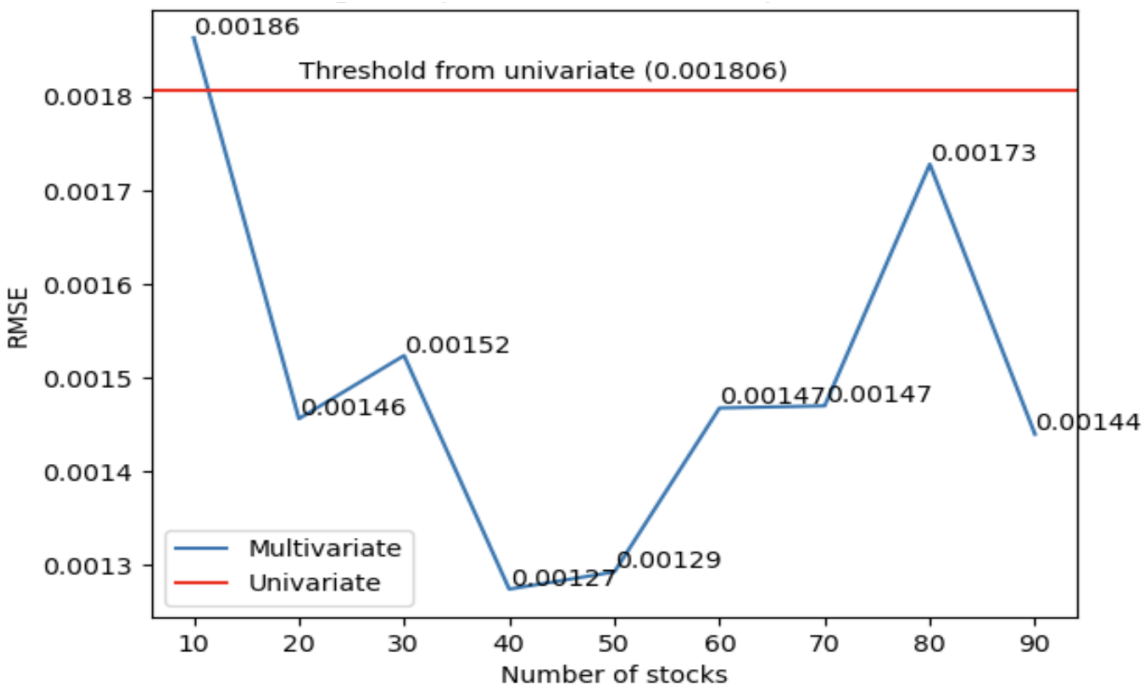

Fig 7: Line chart of experiments with multiple stocks and different features

7.4 Long-term forecasting (Architecture 2) Results

The long term forecasting experiments were conducted with the group of four similar and dissimilar stocks. The output types considered were two types - Multivariate Input and Single Output, and Multivariate input and multivariate output.

Table 4: Experiment results with multi-dimensional input with architecture 2
(MS: Multivariate input Single output, M: Multivariate input multivariate output)

Algorithms		**Informer**			**Autoformer**		
Method	**o/p type**	**RMSE**	**MAPE**	**MSPE**	**RMSE**	**MAPE**	**MSPE**
Similar Stocks	**MS**	**0.8641**	2.0561	775.35	**0.8699**	2.3013	1032.02
	M	0.9610	1.3733	192.55	0.9668	1.5627	109.74
Dissimilar stocks	**MS**	0.8720	2.6109	1450.23	0.8798	2.2511	667.76
	M	1.0091	1.8516	539.13	1.0023	2.4943	997.18
Stock 0	MS	0.6341	1.7121	46.13	0.8807	2.9856	2388.46

7.5 Prediction Graph

The prediction graph for stock 0 for first 50 time_ids shows only a decent prediction success.

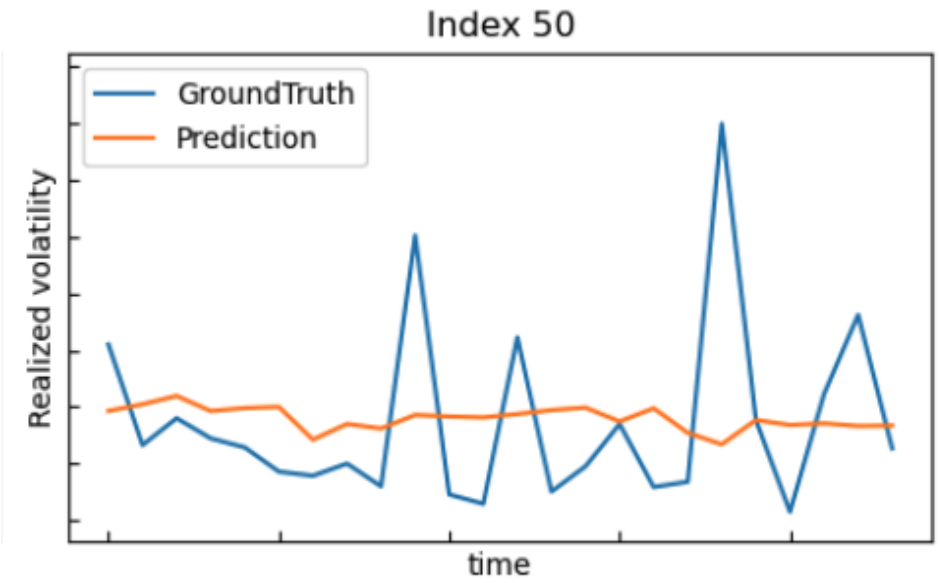

Fig 8: Prediction graph for stock 0

7.6 Loss curves

The train and test loss curves were descending in nature, while validation loss was fluctuating more.

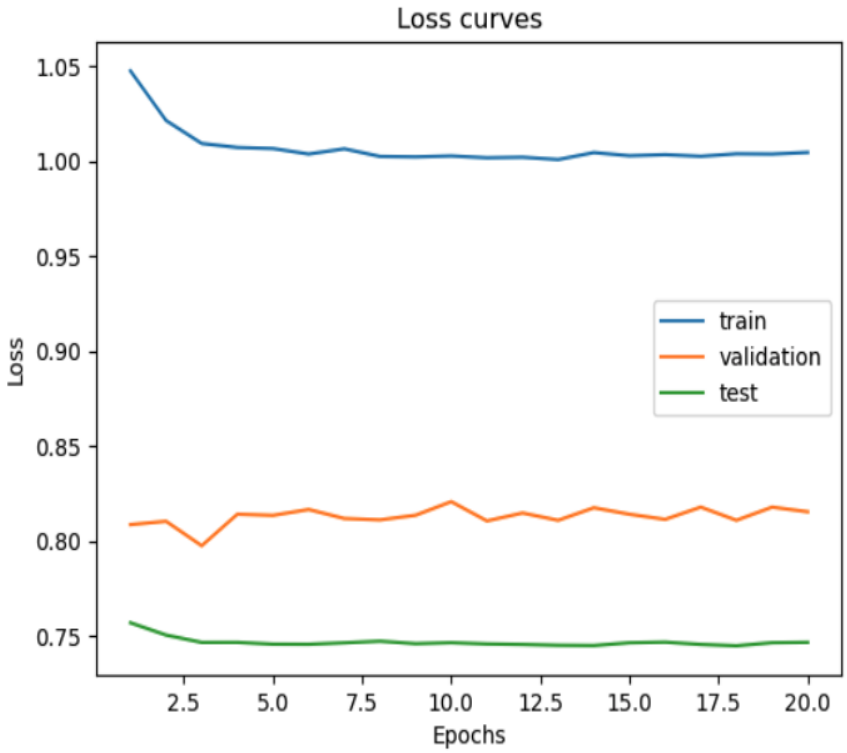

Fig 9: Loss curve for stock 0

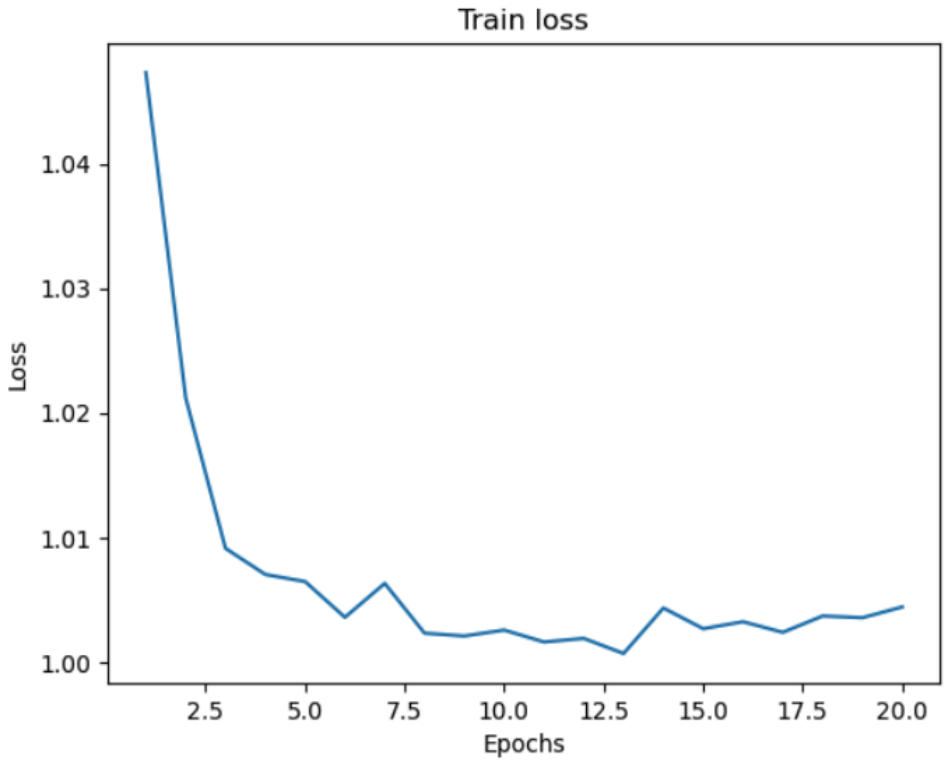

Fig 10: Train loss curve for stock 0

Fig 11: Validation loss curve

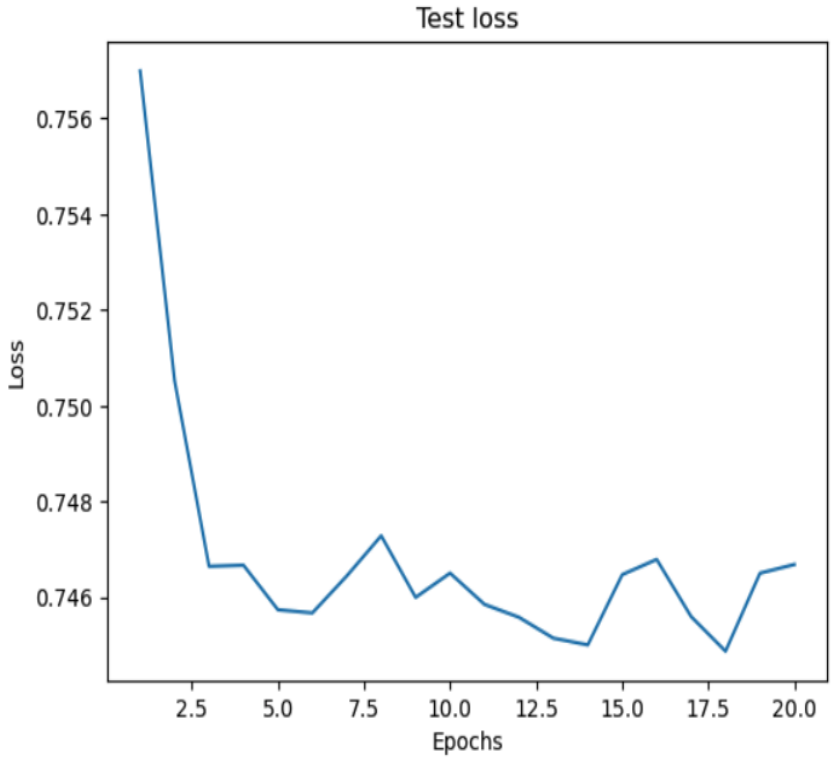

Fig 12: Test loss curve

8. Discussion

In the short-term forecasting (one step ahead forecasting), the features were generated manually. The basic features included lags of realized volatility as the previous value of the realized volatility is a great influencer of future values, and the spread of price, and size of ask and bid determined the variation in the prices and sizes. The sum, mean and standard deviation normalized the values over the window length of considerations. The nearest neighbor features helped find out the correlation between various stocks and time periods. In the real market, the value of stock price, thus its realized volatility tends to have a correlation with other related stocks. If the value of one stock changes, then the value of other stock generally tends to follow a similar pattern within the same industry. Additionally, in the intraday data, the time of the day also affects the trading behavior. Generally, the number of transactions is larger during the opening and closing of the stock market or in particular hours of the day. The time-related features try to find a correlation among various time periods. Table 1 displays the number of features generated for each type of feature method implemented. The count of basic features was 307 and with all nearest neighbor features, the count was 917.

In all the experiments, the predictions were made for one stock (stock id = 0) for the simplicity of the comparisons. The impact of considering different numbers of stocks and different types of features are tabulated in Table 3. In table 3, the Root Mean Squared Error (RMSE) was found to be least i.e 0.001274 when the number of stocks was 40 and the features were basic features and volume related nearest neighbor features. Overall, there are four volume related features and three size related nearest neighbor

features in the table, which explains that the nearest neighbor features were useful in predicting the realized volatility.

In Fig 7, as the number of stocks increases the RMSE surpasses the threshold value of univariate. This result explains that the correlation between stocks exists and is helpful in predicting the value.

After the short term forecasting, the long term forecasting was attempted. From their papers, Informer and Autoformer seem to work well in the long term forecasting task. Thus, we attempted these two models in our task. The major difference is that these two models used time related features extracted from the date. However, the optiver data does not have a date column, but has a time_id column instead which is an integer value. Thus, the informer and autoformer codes were updated to remove all the time related features.

In the previous short term forecasting, the input was two dimensional with features as columns and rows as data for stock on time period. The major drawback here is that the stock_id is listed as a feature. It's an integer value. Thus, the attempt was made to enhance this. The enhancement made was the four dimensional input, which eliminated the stock_id as the feature. The four dimensional input was transformed into two dimensional input with the use of Temporal convolutional network. This TCN was adopted from the DeepTrader paper. Then, these transformed features were the input for the Informer and Autoformer. The values of RMSE in Table 4 are almost similar for

each experiment conducted. The value of RMSE was 0.8641 when the output was a single column forecast. For the same, single column forecast, the value for autoformer was 0.8699 which is only 0.67 % change. Similarly, the multiple combinations of stocks were analyzed. One group of input was similar stocks and dissimilar stocks. The similarity in a sense, that the realized volatility showed a similar nature of line chart. Between these two, the similar stocks seem to have less RMSE.

The prediction was not that good. However, the predictions slightly attempted to match the ground truth. The validation and test losses were less than the train loss, which means the model is not overfitting. And, the individual loss curves tend to be decreasing. This is a good starting point and possibly can lead to a good direction for further study. In the experiments of Architecture 1, the lowest value of RMSE was when the number of stocks was 40. And the If we could increase the number of stocks than currently used stock, i.e 4, then there is a good probability that the forecasting might be improved which ultimately improved the performance of the models that are dependent on the value of realized volatility, hence improving the overall profit of the stakeholders.

9. Conclusion

The importance of stock correlation and feature engineering were evident from the experiment results of the Architecture 1 (short term forecasting). The experiment result demonstrated that the RMSE value was lowest when the number of stocks was 40 and the input being volume related nearest neighbors. In Fig 7, as the number of stocks increased from around 13, the RMSE value was less than that of the univariate model. Additionally, the results from Architecture 2 showed the potential of these forecasting models. The improvement could be possibly achieved by two methods - by increasing the number of stocks into consideration and by increasing the number of features generated from the TCN. Additionally, the study can be done with the ensemble of the informer and autoformer.

10. References

Chen, Q. and Robert, C. (2022). Multivariate Realized Volatility Forecasting with Graph Neural Network. arXiv (Cornell University). doi:https://doi.org/10.1145/3533271.3561663.

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H. and Zhang, W. (2021). Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. arXiv:2012.07436 [cs]. [online] Available at: https://arxiv.org/abs/2012.07436.

Ge, W, Lalbakhsh, P, Isai, L, Lenskiy, A & Suominen, H 2023, 'Neural Network–Based Financial Volatility Forecasting: A Systematic Review', ACM Computing Surveys, vol. 55, no. 1, pp. 1–30.

Jansen, S 2020, Machine Learning for Algorithmic Trading: Predictive models to extract signals from market and alternative data for systematic trading strategies with Python, 2nd Edition, Packt Publishing, Birmingham.

kaggle.com. (n.d.). *Optiver Realized Volatility Prediction*. [online] Available at: https://www.kaggle.com/competitions/optiver-realized-volatility-prediction.

kaggle.com. (n.d.). *Optiver Realized Volatility Prediction*. [online] Available at: https://www.kaggle.com/competitions/optiver-realized-volatility-prediction/data.

Lin, Y, Lin, Z, Liao, Y, Li, Y, Xu, J & Yan, Y 2022, 'Forecasting the realized volatility of stock price index: A hybrid model integrating CEEMDAN and LSTM', Expert Systems with Applications, vol. 206, p. 117736–.

Wang, Z., Huang, B., Tu, S., Zhang, K. and Xu, L. (2021). DeepTrader: A Deep Reinforcement Learning Approach for Risk-Return Balanced Portfolio Management with Market Conditions Embedding. Proceedings of the AAAI Conference on Artificial Intelligence, 35(1), pp.643–650. doi:https://doi.org/10.1609/aaai.v35i1.16144.

Wu, H, Xu, J, Wang, J & Long, M 2022, 'Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting', ArXiv (Cornell University).

Zhang, C-X, Li, J, Huang, X-F, Zhang, J-S & Huang, H-C 2022, 'Forecasting stock volatility and value-at-risk based on temporal convolutional networks', Expert Systems with Applications, vol. 207, pp. 117951-.

11. Appendices

Github code:

Github repo	https://github.com/poo5zan/realized-volatility
Data Analysis:	https://github.com/poo5zan/realized-volatility/blob/main/data-analysis.ipynb
Optiver code (original):	https://github.com/poo5zan/realized-volatility/blob/main/1st-place-public-2nd-place-solution.ipynb
Optiver code (updated)	https://github.com/poo5zan/realized-volatility/blob/main/from-kaggle-optiver-challenge.ipynb
Data preparation	https://github.com/poo5zan/Informer/blob/main/data_preparation.ipynb
DeepTrader - TCN:	https://github.com/poo5zan/DeepTrader/blob/pujan_changes/src/run2.ipynb
Informer	https://github.com/poo5zan/Informer/blob/main/run.ipynb
Autoformer:	https://github.com/poo5zan/Autoformer/blob/main/Autoformer_Research_project_Pujan.ipynb