PREDICTING THE VOLATILITY OF STOCK PRICE MOVEMENTS

Xiangmin Shen, Michael Zhu

Northwestern University

1. ABSTRACT

We developed 3 neural network models to forecast the price volatility of the S&P 500 Value Index at market close. The 3 models have the following structures: (1) Fully Connected Network (baseline model) , (2) Convolutional Network, and (3) Gated Recurrent Unit Network. The GRU model performed the best, followed by the baseline model, and the CNN model performed the worst. All three models could accurately make forecasts.

2. INTRODUCTION

Market making at market-close is a popular trading strategy. A market maker is someone who quotes both a buy (bid) and a sell (ask) price in a financial instrument, such as stocks, hoping to make a profit on the bid-ask spread. To quote a price is to post a price, and the bid-ask spread is the difference between the prices quoted for an immediate sale and an immediate purchase of a financial instrument. A market maker tends to be most active during market-close because there is a spike in liquidity during market-close. Market liquidity of a financial instrument refers to how rapidly the instrument can be bought or sold, which is usually measured by its trading volume. A liquid market would actively move back-and-forth and thus provide more opportunities for a market maker to capitalize.

A key decision for a market maker is how far they would like to quote above and below the market. Suppose that the current price of a financial instrument is x. To quote a above the market is to quote a sell (ask) price of $x + a$. Similarly, to quote b below the market is to quote a buy (bid) price of $x - b$. In other words, by quoting a above the market and b below the market, we hope to buy low at $x - b$ and sell high at $x + a$. For each successful trade, we would make a profit of $(x+a)-(x-b) = a+b$. The further we quote above and below the market (the larger the values of a and b), the more profit we would capitalize on each successful trade. However, the likelihood of success decreases as we quote further away from the market.

When we make markets, it would be ideal if we could set our quoting range according to the then-current market condition. We use price volatility as a measure to help us make the quoting decision. If we believe that the price volatility is going to be high, then we could quote far away from the market to maximize our gains per trade and still achieve a decent success rate. On the contrary, if we believe that price volatility is going to be low, then we could quote near the market to make sure that we could still achieve a decent success rate. Therefore, it is very helpful for a market maker to be able to forecast the price volatility.

3. PRIOR WORK

Predicting the stock market is a popular topic that many researchers have investigated. However, our problem is different from most of the prior works for two reasons. Firstly, researchers in this area tend to study classification problems, such as Yakup Kara[1], Can Yang[2] and Mustafa[3], whereas we are looking for an accurate forecast. For example, there are many prior works that used neural networks to predict whether the stock market is going to go up or down in some time interval, which is a classification problem with two outcomes (up and down). In our final project, we would like to accurately forecast the volatility, which is a number and not a classification problem. Secondly, our problem statement is focused on the real-life implications of market-making only, which is not commonly studied by prior works. Though there are prior works that try to predict the volatility of the stock market, such as Shaikh[4], our definition of volatility is unique to us and is only useful for market-making during market-close.

4. PROBLEM STATEMENT

We would like to predict the volatility of the S&P 500 Value Index during market-close. The ticker for the S&P 500 Value Index is IVE, so we shall refer to it as IVE from this point onward. A ticker is an abbreviation of the name of a financial instrument. We chose to study IVE because it is the only free second-by-second data that we could found online, and it has a decent amount of trading volume. We focus market-close because in real life many market makers participate during market-close.

5. DATA SOURCE

We obtained IVE second-by-second trading data from 09/28/2009 9:30:00 to 11/20/2020 20:00:00. We found the data from Kibot, a historical data vendor. The data is stored in a .txt file, which includes the last traded price, bid price, ask price, and trading volume for each second. The dataset is comprehensive because it covers volatile periods such as 2009, 2015, 2020 as well as quiet periods such as 2012, 2013. All kinds of market conditions could be found in our dataset, and this makes sure that our models would not be biased towards any particular kind of market conditions, such as bullish market or a bearish market.

The closing price for IVE is released at 15:00:00 CST every trading day. The closing price is used as the baseline for market making. For example, to quote \$5 above the market is to quote \$5 above the closing price. The closing price is calculated based on the volume-weighted average price of all transactions during the interval 14:59:00 CST and 15:00:00 CST. Consequently, the market price for IVE at 15:00:00 CST is almost always different from the closing price. For our final project, the volatility of IVE during market-close shall be defined as the normalized absolute difference between the closing price and the market price at 15:00:00 CST. For example, if the closing price is \$100 but IVE trades at \$105 at market close, then the volatility shall be $\frac{|100-105|}{100} \cdot 100\% = 5\%$. It is essential to normalize the volatility because our data ranges across more than ten years. During this time, the IVE has grown from the initial \$57 per share to \$120 per share. We must normalize the volatility term in order to sensibly compare them across different time periods.

6. METHODS

For our final project, we developed three neural network models using Pytorch: (1) Baseline model (2) CNN model (3) GRU model. Before we delve into more details. The following features are common across all three models:

(1) All three models use the mean squared error loss function. We chose the MSE loss function because it is popular, easy to implement, easy to interpret, and serves our purpose.

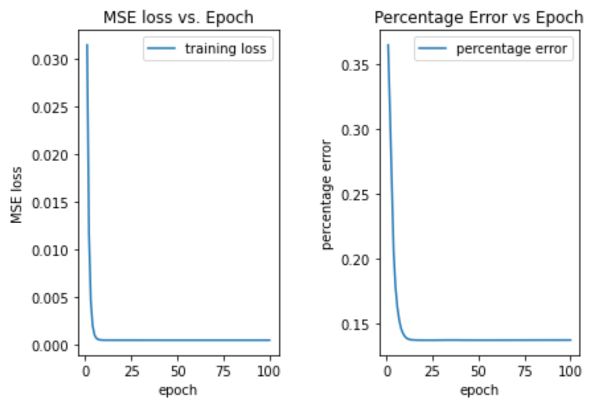
(2) All three models use the ReLU activation function at each hidden layer. We chose the ReLU activation function because it reduces the likelihood of vanishing gradient.

(3) The dataset is randomly split into a training set (10%) and a testing set (90%) using Pytorch data loader. As mentioned above, this hinders the problem of biased models.

(4) For all three models, the inputs are volatility calculated at the end of each minute for the 30 minutes before the market close. In other words, the inputs are volatility at 14:30:00 CST, 14:31:00, ..., 14:59:00 CST. The method for calculating the volatility at the end of each minute is the same as calculating the volatility for the end of day. Consequently, the input has a size of 29. For each trading day we have in the dataset, the model shall predict the volatility at market close. Then we, will compare the predicted volatility to the actual observed volatility.

(5) To measure the performance of our models, we decided to use the symmetric mean absolute percentage error (sMAPE) metric. The formula for calculating sMAPE is $\frac{1}{n}\sum_{t=1}^{n}\frac{|F_t-A_t|}{(A_t+F_t)/2}$, where F_t is the forecast value and A_t is the actual value. We chose the sMAPE method over the classic MAPE method because sMAPE is resistant to outliers.

In the next section, we shall present each model in mode details and show the corresponding results.



```
lowest MSE loss: 0.0004441871424205601 at epoch 99
lowest percentage error: 0.1373279833285786 at epoch 24
```

Fig. 1. Baseline model

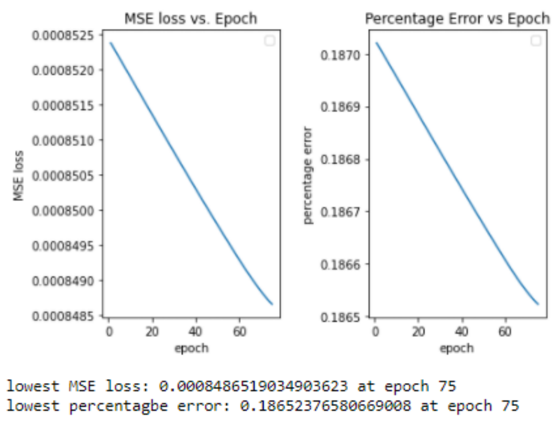
7. EVALUATIONS

7.1. Baseline Model

Our baseline model is a fully connected neural network with 1 hidden layer that contains 10 nodes. Thus, the structure of the baseline model is 29 $\rightarrow$ 10 $\rightarrow$ 1, which gives a total of 311 trainable parameters. We trained the baseline model for 100 epochs with the learning rate of 0.001. Fig.1 shows the results from the baseline model. The training loss quickly falls down in a few epochs and stays low. The percentage errors over the epochs follow a similar pattern as the training loss. The lowest percentage error is 13.7% reached at epoch 24. We will compare our more sophisticated models to this baseline model in the following subsections.

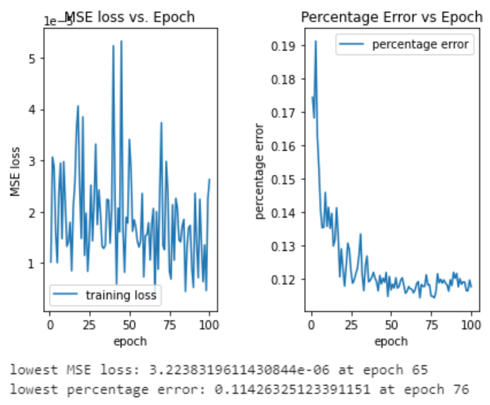
7.2. CNN Model

The CNN model is a convolutional neural network with 1 one-dimensional-convolution layer, 1 max-pool layer, 1 fully connected hidden layer of 10 nodes, and an output layer of 1 node. The convolution layer has 1 input channel, and 2 output channels, with a kernel size of 3 and stride size of 1. The max-pool layer has a kernel size of 3 and stride size of 1.



```
lowest MSE loss: 0.0008486519034903623 at epoch 75
lowest percentagbe error: 0.18652376580669008 at epoch 75
```

Fig. 2. CNN model



```
lowest MSE loss: 3.2238319611430844e-06 at epoch 65
lowest percentage error: 0.11426325123391151 at epoch 76
```

Fig. 3. GRU model

We chose a single convolution layer and a single pooling layer with very small kernel and stride sizes because unlike an image data with 3 channels of colors each with thousands of pixels, our input only has a single channel of size 29. We tried to add more convolution layers and increase the kernel and stride size, but that would drastically shrink the size of our data to something around 3-5 by the time they reach the fully connected layer.

We believed that a CNN model would be suitable for our problem because CNN works well with data that has spatial relationship. We believe that the trading activities before the market-close is correlated with the trading activities at the market-close, and each minute shortly before the market close is also correlated with one another.

The performance of the CNN model is shown in Fig.2. The learning rate we used was 0.025. We stopped training the model after the test error stopped decreasing, which occurred after epoch 75. Compared to the baseline model, the CNN model actually performed worse. It achieved a sMAPE of 0.1865, which is higher than that of the baseline model, which achieved a sMAPE of 0.1373. That being said, the performance is still very good, and, like the baseline model, the CNN model could also accurately forecast the price volatility.

7.3. GRU Model

The GRU model is a recurrent neural network that takes inputs of size 29 and gives outputs of size 1. The GRU model is a natural model to use for this problem because we are dealing with time-sequence data. There are several parameters to set when we construct a GRU neural network, including the hidden layer dimension and the number of layers. Besides, the learning rate would also affect the model training results. After trying a various set of those parameters, our best GRU model has the hidden layer dimension of 256, 2 layers and the learning rate of 0.0001. Fig.3 shows the results from this GRU model. It is able to produce a lower percentage error than the baseline model. A noticeable difference of this model from the baseline model and the CNN model is the training loss and percentage error keep oscillating over epochs, although following a downward trend. The oscillation exists even when I lowered the learning rate and applied gradient clipping. We guess this is a common feature of recurrent neural networks.

8. DISCUSSION

We developed 3 neural network models to forecast the price volatility of IVE. All 3 models could forecast the price volatility with error percentages between 10% and 20%, which is not bad. The GRU model is the most accurate model among all three, followed by the baseline model, then followed by the CNN model. We expected the GRU model to be the best performing model because this model is designed for processing time-sequence data. We did not expect the CNN model to perform the poorest. One explanation could be that our input has a size of 29, which might be too small to use for a CNN model. We might have over generalized the features during convolution and pooling. As mentioned above, it was not even possible to add a second convolution layer because doing so would shrink the data size to almost nothing.

9. FUTURE WORK

We initially wanted to choose the SNP 500 index (SPY) as our instrument of study. SPY is the most popular stock index in the world. However, because SPY is a very popular product, data on SPY is costly, especially second-by-second data, so we chose to study IVE instead, which has free data. If we could obtain data on SPY, we would love to see how our models perform on SPY. Since SPY is more popular, its price fluctuations are also more volatile, which makes it a more difficult instrument to predict than IVE. Our models were able to accurately predict the volatility at market close for IVE, but the same result might not be true for SPY. Testing our models on SPY is something that we could explore in the future.

10. REFERENCES

[1] Yakup Kara, Melek Acar Boyacioglu, and Ömer Kaan Baykan, "Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the istanbul stock exchange," *Expert Systems with Applications*, vol. 38, no. 5, pp. 5311 – 5319, 2011, Available at https://doi.org/10.1016/j.eswa.2010.10.027.

[2] Can Yang, Junjie Zhai, and Guihua Tao, "Deep learning for price movement prediction using convolutional neural network and long short-term memory," *Mathematical Problems in Engineering*, vol. 2020, pp. 1–13, 07 2020, Available at https://doi.org/10.1155/2020/2746845.

[3] Mustafa Göçken, Mehmet Özçalıcı, Aslı Boru, and Ayşe Tuğba Dosdoğru, "Integrating metaheuristics and artificial neural networks for improved stock price prediction," *Expert Systems with Applications*, vol. 44, pp. 320 – 331, 2016, Available at https://doi.org/10.1016/j.eswa.2015.09.029.

[4] Zahib Iqbal Shaikh A. Hamid, "Using neural networks for forecasting volatility of sp 500 index futures prices," *Journal of Business Research*, vol. 57, pp. 1116–1125, 2004, Available at https://doi.org/10.1016/S0148-2963(03)00043-2.