Stock Movement Prediction

1. Introduction

The goal of this project is to predict the movement of stocks within five days. The training data includes the historical data from 2011/04/28 to 2021/04/30 of 306 stocks. We need to use the training data to build a model to predict the movements of 15 stocks within 5 business days (from 2021/05/03 to 2021/05/07). There are five movement classes: “<= -5%”, “<= -2%”, "-2% < r < 2%", “2% <=”, and “5% <=”, which are indicated by -2, -1, 0, 1, and 2. In addition, the 15 stocks are VZ, T, WMT, MGM, GPS, GT, BBY, AFG, ERJ, MYE, ECPG, GCO, MPC, TRI, UFI.

2. Methodology

LSTM vs Transformer

In traditional machine learning, recurrent neural networks are usually used to model sequential data such as text data or time series. The benefit of recurrent neural networks is that the gradient can be back-propagated through the dimension of time therefore learn the relative order of the data and recognize the pattern in the data. Compared to vanilla RNN, both the gated recurrent unit, and the long-short term memory unit offer advantages over vanilla RNN since they have dedicated "gates" in the recurrent unit to enable the machine learning model to remember a relatively long sequential input. Therefore, LSTMs are frequently used in text generation and other natural language processing tasks because the meaning of texts is very context-dependent.

In 2017, Google published a paper named "Attention is All You Need". This paper describes transformer architecture which is a sequence-to-sequence learning structure. For the particular problem that the researchers were dealing with, translation, the model needs to take into account both the relative position and the context-based meaning of the words to generate correct translation. The new architecture allows the prediction to be based on the entire input sequence simultaneously which means there is no back-propagation through time. This

characteristic of transformer architecture makes the training of a system much faster because much work can be parallelized compared to LSTMs. The greatness of parallelized computing came at the cost of having a limited input length; LSTMs have no such limit.

Long term dependency

For our stock prediction task, we were given the previous day's closing price of the stocks and we want to predict the closing price range for the next 5 days. We choose to use transformer architecture because it has not been widely used in time series prediction and the transformer is more capable of taking advantage of the long input sequence. For a historical pricing-based trading system, we do not have any external information, which means the only source of information is the people's reactive trading pattern. We believe the pattern can be as long as a year or as short as a week; in order to capture this pattern, the transformer can have an input window size as big as 400. Although in theory LSTM does not have a limited input window size, in practice, Goldberg pointed out that BERT(a transformer-based model) is more powerful than LSTM when dealing with long sequence syntactic test cases(*Assessing BERT's Syntactic Abilities*). We assume this tendency is also present in time series prediction.

Prediction Tasks & Trading Strategy

In practice, the prediction task is highly related to the trading strategy. In our case, instead of giving an exact number for the stock price for some future date, we were asked to provide a range of labels for the next five trading days. [+5%, +2%, 0, -2%, -5%] These five labels mean if the closing price of the stock over the next 5 days is over for example 5%, then the +5% and +2% labels are positive. This type of stock prediction is in some ways easier than regular regression problems because point prediction can be more difficult to evaluate the correctness. Corresponding to this trading strategy, we can have limit-sell and limit-buy strategies as well as put/call options over the next 5 days.

Stock Prediction Specificity

Excluding External Factors

To make the daily price sequence easier to learn for the system, we want to eliminate the externalities that cannot be predicted because they can destabilize the training process. We think

the COVID-19 stock crash/recovery is a great example of such externality. No one can predict the stock crash and the v-shaped recovery based only on daily closing prices. Therefore, for the majority of the training process, we choose to exclude the daily closing price after 2100 days(around March 2020).

Preprocessing Data

Black-Scholes Model Assumption

The Black-Scholes-Merton model is a pricing model for financial instruments and is often used for the valuation of stock options. One key assumption for Black-Scholes Model is that the return of a particular investment tends to the risk-free rate when the risk tends to zero. The risk-free rate is the interest rate in this case. If we take into account the continuously compounded risk-free rate of return over the years, the price of a certain investment will become an exponential curve. We can also observe this assumption in the stock market because the S&P 500 index in some ways looks like an exponential curve. From the risk-free rate assumption of BSM, we can derive that for a particular time point, the stock price follows a log-normal distribution. Therefore, we can take the log of the price and then assume the price is no more affected by the interest rate.

Closing Price vs Relative Movement

We also thought about using the relative stock price movement from the previous day for prediction. The core idea behind this is that we often assume stock price movement is a stochastic process. Under the Brownian model of the financial market, the assets have continuous prices evolving continuously in time and are driven by the Brownian motion process. Another supporting idea for movement prediction is the Markov chain. We want to assume the stochastic process in the stock market is a Markov chain and we can train a machine learning model to recognize the market state from the previous daily movements and predict the following movement in the near future.

Single-Name vs Index-Based

Since single-name stocks are extremely random and volatile, the daily stock price is heavily affected by the randomness of the market. One idea to eliminate the randomness is to use

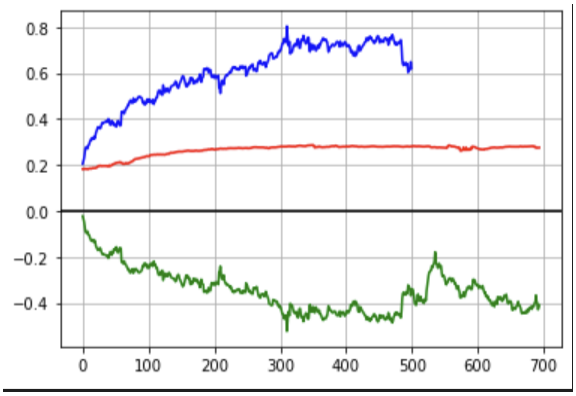
stock market indices for prediction. The index is a measurement for the stock market or a subset of the stock market that helps investors to see the overall trend of the market. An index often consists of many stocks and this nature of the index can greatly eliminate the randomness of single-name stocks. For the 15 stocks that we want to predict, they belong to 6 big market sectors, we can also use the sector-specific index for the forecasting. For the public service sector(VZ, T), we can use S&P 500 Utilities Index for the prediction. For the consumer service sector(WMT, MGM, GPS, TRI, BBY, GCO), we can use the Dow Jones U.S. Consumer Services Index for the prediction. For the Consumer durable and non-durables sector(MYE, UFI, GT), we can use the Dow Jones U.S. Consumer Goods Index. For the finance sector(AFG), we can use S&P 500 Financials Index for the prediction. For the energy sector(MPC), we can use S&P 500 Energy Index for the prediction. For the capital goods sector(ERG), we can use S&P BSE CAPITAL GOODS Index for prediction.

We borrowed the main architecture for our transformer code from GitHub since there is already a repository setup for transformer-based time series prediction.

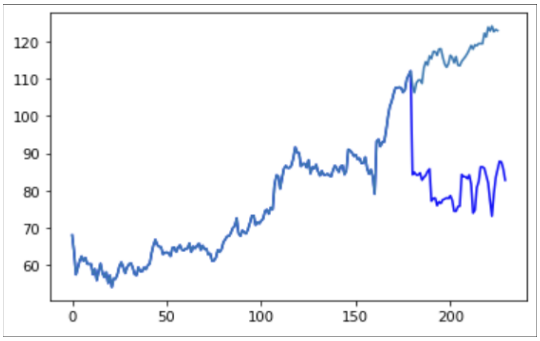
3. Results

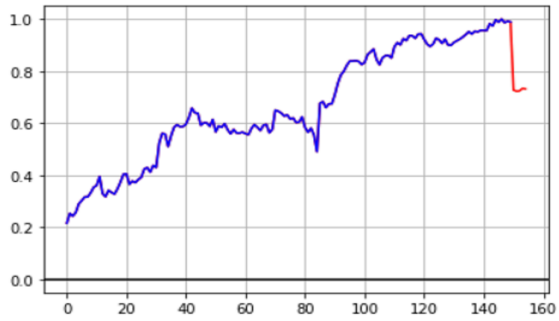
We first took the sample code from latte and ran it without any modifications, the result was reasonably good with the highest CR score close to 0.9. But run to run variance was significant and in some cases the CR score was only 0.01. Also, when we lowered the threshold from 5 to 2, the CR score was much worse as predictions on many stocks only got no more than 0.02 CR score. So we decided to try a different architecture.

The architecture we tried was transformer, which was known to be good at handling long sequence data. The first attempt ended up getting a model that failed to converge. After discussion and trials, we found out that the transformer is heavily influenced by batch size, which is very counterintuitive, in theory batch size shouldn't affect training at all. Training using an inadequate batch size will not converge unless trained with thousands of epochs.

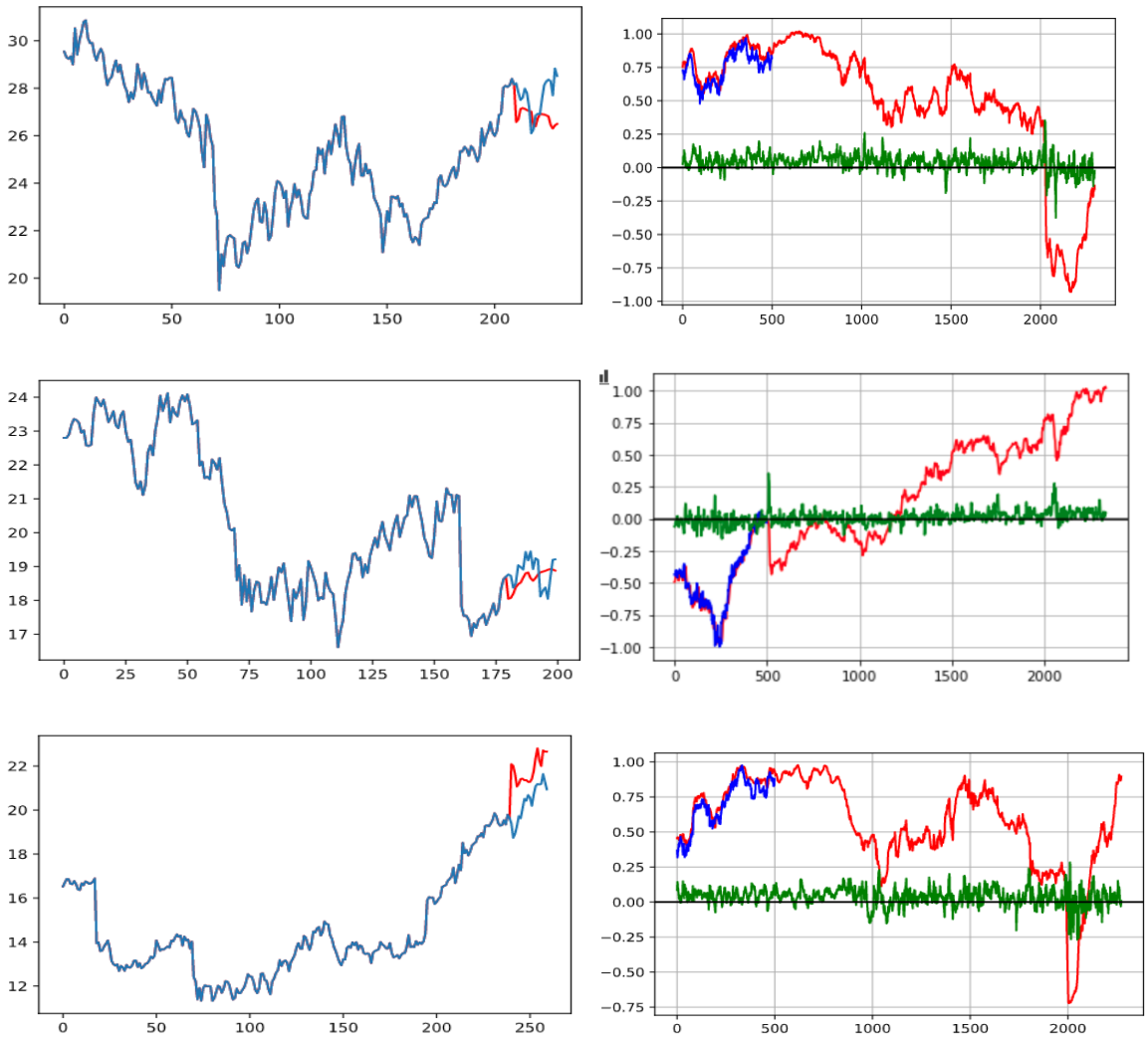


After increasing the batch size we find that some models can successfully converge but others failed, some stocks require up to 3 times larger batch size to converge. So finally we used 240 as batch size so that the transformer can converge on all stocks. After the model can successfully converge we found that the predictions made are strongly inclined to plunge out of no reason and stays that way no matter how we set the window size.





After discussion we concluded that we cut the training data too early so that the model couldn't get the sense of how recent stock data moves. Originally we cut the training data months before the current date, so what the model learned would be a lower stock price from several months ago, when given with recent data to predict, it will conclude that the stock should fall to a lower price.



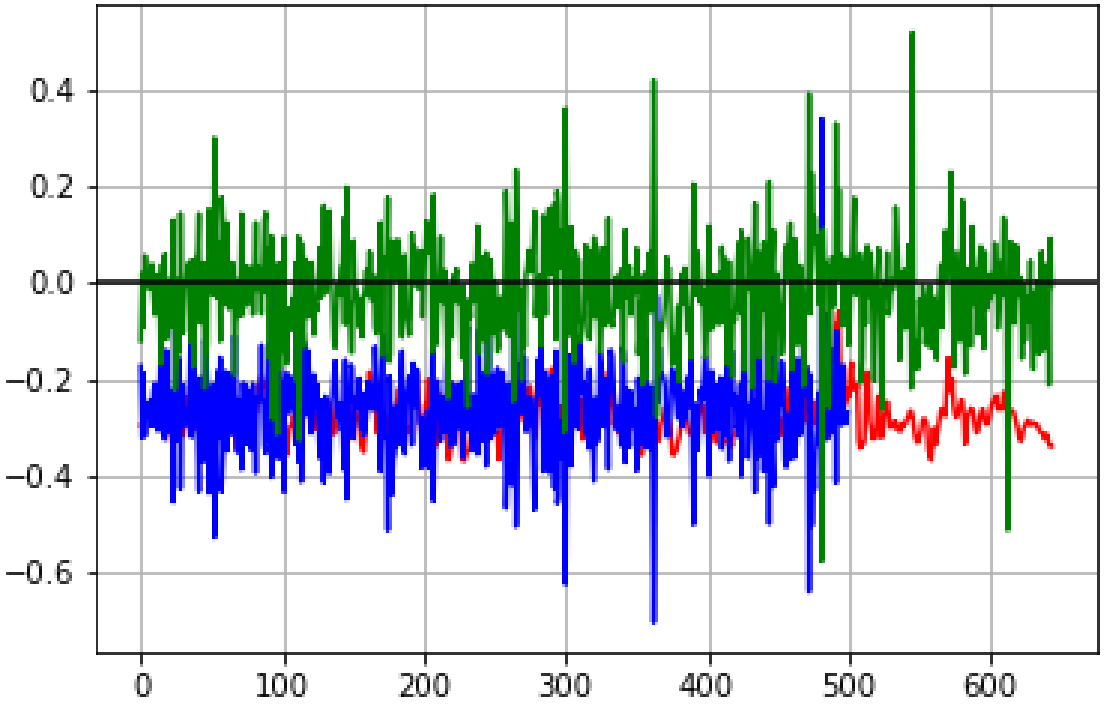
The above three sets of figures are EJR, ECPG and GPS respectively. The figures on the left side are predictions(represented in red line) versus ground truth and the green lines in figures on the right side are deviations from predictions and ground truth. The final predictions, although not perfect, can largely mimic the trend of how the stocks will move in the future.

4. Discussion

Since the predicted results are movement of stocks, we decide to use the historical movements of the stocks to train the model first. In the data loading part, add a new row called Percentage (Close/Open -1) at the end of each stock data. Then use a transformer to train the data

of Date and Percentage. We set the input window as 200, the output window as 5, the batch size as 60, the learning rate as 0.005 and the epochs as 200.

However, the result is not great. We can see the picture below. The blue line is the ground truth. The red line is the prediction. The green line is the difference between the prediction and the ground truth. The green line is not closing to 0 and jumping up and down violently. We don't think the prediction will be accurate. Using stock price becomes a better choice.



5. Conclusion

After many trial-and-error, we used some of the ideas proposed in the methodology. We choose not to use stock movement prediction because after excluding the price itself, the movement becomes very random and the model simply doesn't learn anything from the historical movement. Another key takeaway is that we cannot simply disregard the price of the COVID-19

crash and recovery. The transformer can predict the value around the mean of the previous input. If we simply shift the prediction period and have a big time gap between training and testing, the transformer predicts the future price will drop to around the previous mean because the stock price has moved upward a lot since the crash in March-April 2020.

After taking the logarithm of the price, the price becomes easier to train and predict because the extremely high and low prices of the stock are closer to the mean. This result proves that our assumption of the BSM model is valid.

We also tried using the index for training and prediction. However, the results are not very ideal. If we only use the index for training and use transfer learning and directly apply the model for the single-name prediction. We do not get any plausible result because the index price is usually far from single stock in terms of price.

6. Future work

There are several strategies we can try if we have more available time. One of the most feasible ones is the “stacking” strategy.

The “stacking” strategy here not only refers to combining the prediction from the multiple machine learning models on the same dataset but also means taking predictions from different stocks into consideration. Looking at some existing open-source examples of stock-price prediction using machine learning, we found that although it is hard to get a good performance through a single model, people improve their prediction accuracy after combining multiple models and get a fair result at last.

Furthermore, we also consider utilizing the prediction results of different stocks instead of one stock. When humans try to predict the trend of stocks, even just one stock, they won’t just keep their eyes on a single stock. Instead, they look at a range of stocks that are relative to the target stock. This strategy makes sense because the relative stocks may provide more information that will affect the trend of the target stock. For example, A is the raw material supplier of B. If the stock price of B increases, it may give a sign that the stock price of A will also increase. We can try to use LSTM or a transformer as a model, and the prediction of target stock and relative stocks as input to get our final prediction.

Obviously, we need to train lots of models if we take the “stacking” strategy, which will cost lots of time and memory. Therefore, we have to find a method to reduce consumption. Informer may be a good choice because it uses a “ProbSparse” self-attention mechanism, which achieves O(LlogL) in time complexity and memory usage, and its self-attention distilling highlights dominating attention by halving cascading layer input, and efficiently handles extreme long input sequences.

7. Contribution

Hao Hu:

Code: Modified code for the transformer and implemented data input and output method. Fine tuned the model to facilitate model convergence and make predictions more reasonable.

Paper: Wrote results section. Adjusted the format of the paper.

Kexu Qian:

Code: Modified the data loading for the transformer and used the historical movements to train and predict.

Paper: Wrote the Introduction and Discussion. Edited the rest of the paper.

Zian Yang:

Code: Modified the code to make it runnable on local machine and stock data. Implemented and tested all ideas that are presented in the Methodology.

Paper: Wrote Methodology and Conclusion. Edited the rest of the paper.

Wei Liu:

Code: Search open source example code. Modify TA’s sample code and Transformer code. Fine tune, run several result of the 15 stocks

Paper: Write Future work and polish up report