Project Report: Stock Market Prediction Using a Transformer Model

Ayush Rawat *https://github.com/ayushrawat6367*

Introduction

This project focuses on creating a machine learning model that can predict whether to buy, sell, or hold a stock based on past market data. The model uses a Transformer, a type of machine learning architecture known for handling time-based data efficiently. To improve the accuracy of predictions, various technical indicators were included as features.

Data Preprocessing

The raw data includes stock prices, bid prices, ask prices, and trading volumes. To prepare this data for the model, the following steps were taken:

1. *Normalization:*
   Prices and other monetary values were scaled down by dividing them by a billion to make the data easier for the model to process.
2. *Feature Engineering*:
   New columns were created for Open, Close, High, Low, and Volume based on the existing data. This helps capture essential information about stock movements.
   Several technical indicators were added to the data. These indicators are common tools used in finance to analyze market trends and make predictions.
3. *Handling Missing Data:*
   Any rows with missing values were removed to ensure the data was clean and ready for analysis.
4. *Feature Scaling:*
   All features were standardized, meaning they were adjusted to have similar scales, which helps the model learn better.

Technical Indicators

Various technical indicators were calculated and added to the dataset. These indicators provide insights into the stock market's momentum, volume, volatility, and trends. Here are the types of indicators used:

1. *Momentum Indicators*: RSI (Relative Strength Index), MACD (Moving Average Convergence Divergence), Stochastic Oscillator
2. *Volume Indicators*: OBV (On-Balance Volume)
3. *Volatility Indicators*: Bollinger Bands, ATR (Average True Range)

4. *Trend Indicators*: ADX (Average Directional Index), CCI (Commodity Channel Index)
5. *Other Indicators*: DLR (Daily Log Return), TWAP (Time Weighted Average Price), VWAP (Volume Weighted Average Price)

Model Architecture

The model is based on a Transformer, which is typically used in natural language processing but works well for time-series data too. The main components of the model are:

1. *Input Embedding*: A layer that converts the input data into a format that the Transformer can understand.
2. *Transformer Layers*: These layers are responsible for analyzing the relationships between different data points over time.
3. *Output Layer*: A final layer that produces predictions, telling whether to buy, sell, or hold a stock.

Output Generated:

Epochs	Loss
1	1.7505
2	1.4749
3	1.4328
4	1.9508
5	2.8072
6	1.9793
7	1.5202
8	1.6588
9	1.6807
10	1.5016

Training Process

The model was trained for 10 cycles (epochs) using a loss function that measures how well the model's predictions match the actual outcomes. An optimizer, called Adam, was used to adjust the model's settings to minimize errors. The training showed that the model was learning, although the results varied, suggesting that some adjustments might be needed.

Loss Trend Analysis:

Epoch 1: The initial loss was 1.7505. This is expected to be higher since the model is starting from random initialization.

Epochs 2-3: The loss decreased to 1.4749 and 1.4328 respectively. This indicates that the model is learning and improving its predictions during the early epochs.

Epoch 4: The loss increased to 1.9508. Such an increase could be due to several factors, such as the model overfitting to the training data or encountering a more complex batch that it couldn't generalize well to.

Epoch 5: The loss spiked to 2.8072. This is a significant increase and may indicate that the model is struggling to learn or that there might be an issue with the data or model configuration (such as learning rate, batch size, or model complexity).

Epochs 6-10: The loss fluctuated but generally decreased, with the final loss being 1.5016. The fluctuations could suggest that the model is encountering varying difficulty levels in different batches, but it generally trends towards improvement.

Final Loss Value:

By the end of training, the loss stabilized around 1.5016. While this is better than the initial loss, it's relatively high in the context of classification problems (especially with CrossEntropyLoss).

Evaluation

After training, the model was tested using the same type of data. It generated recommendations for each point in time, advising whether to buy, sell, or hold the stock:

0: Hold

1: Buy

2: Sell

The recommendations were based on the patterns the model learned during training.

Conclusion

The combination of a Transformer model and a variety of technical indicators provides a strong approach to predicting stock market movements. The model is good at capturing complex patterns in the data, which is essential for financial analysis.