Feature is all you need

Yuelong Zhang
University of California, Los Angeles
Department of Statistics and Data Science

Abstract

In this research, I investigate the performance of Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM) networks, and a CNN-LSTM hybrid in forecasting the forthcoming minute's stock prices for the S&P 500 ETF, utilizing data from preceding 1000-minute intervals. My methodology encompasses the use of three progressively enriched datasets, starting with basic stock market features, then incorporating technical indicators, and finally integrating news sentiment and economic factors. This gradual data enrichment allows me to methodically examine each model's predictive efficiency and the influence of augmenting data complexity on their forecasting ability.

1. Introduction

This study investigates the efficacy of three deep learning models—Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM) networks, and a combined CNN+LSTM approach—in predicting the next minute stock price of S&P 500 ETF SPY prices based on the previous 1000 minute data. By employing three distinct datasets, this research not only examines the models' predictive power but also assesses the impact of progressively integrating diverse data sources. Dataset 1 (D1) offers a foundational perspective with basic stock market features derived from these 1000-minute intervals, Dataset 2 (D2) enhances this with technical indicators, and Dataset 3 (D3) further enriches the analysis by incorporating news sentiment and economic factors, providing a comprehensive view of market dynamics. This layered approach allows for an in-depth evaluation of how additional data dimensions influence the predictive capabilities of each model, offering valuable insights into the intersection of machine learning and financial analysis.

2. Dataset

Three datasets (D1, D2, D3) contain data spanning from September 1, 2023, to March 15, 2024, with recordings at one-minute intervals. The total number of data is around 50000 in each dataset.

The input data is structured in the format (N, T, D), where N denotes the total number of data points, T is set to 1000, representing the lookback length in minutes, and D varies, corresponding to the number of features in each dataset. The output is formatted as $(N, 1)$, representing the stock price for the next minute.

MinMax scaler was applied to each numerical attribute, and categorical variables were transformed via one-hot encoding.

2.1. Dataset 1(D1)

The dataset contains 5 numerical features related to ETF trading: price, volume, high, low, and transaction count. Feature engineering includes calculating a 10-point simple moving average and standard deviation for each numerical feature, which expands the feature set to a total of 15.

2.2. Dataset 2(D2)

In addition to D1, this dataset contains technical indicators, so there are 11 numerical features: price, volume, high, low, transaction count, simple moving average, exponential moving average, moving average convergence divergence(value, signal, histogram), and relative strength index. Feature engineering includes calculation of 10-point average on volume and transaction count, and 10-point standard deviation on price, volume, high, low, transaction count. Therefore, the total number of features in this dataset is 18.

2.3. Dataset 3(D3)

In addition to D2, D3 contains two additional features: news sentiment and us dollar index ETF price(UUP). I used the pretrained BERT model and fine-tuned the model on the manually labelled news data, and used this model to generate sentiment labels of Negative, Neutral and positive. Since I used one-hot encoder to transform the categorical variable, the total number of features reach 22.

3. Models

3.1. Convolutional Neural Network (CNN)

The input data is formatted as (N, T, D), where it can be interpreted as a 2D image, with time representing the width and features representing the height. This analogy aligns with the use of Convolutional Neural Networks (CNNs) in computer vision for processing image data. Consequently, I employed a CNN architecture comprising three layers, incorporating max pooling and batch normalization, followed by a fully connected layer.

3.2. Long Short-Term Memory Networks (LSTM)

LSTM, a specific form of Recurrent Neural Network, is inherently adept at identifying and leveraging temporal dependencies due to its design, which facilitates the retention of information from previous time intervals. Moreover, LSTM addresses a common issue found in traditional RNNs – the vanishing gradient problem – by incorporating gates that manage information flow. This capability enables it to preserve data across extensive time sequences. Therefore, I used a single-layer LSTM followed by a densely connected layer.

3.3. CNN-LSTM (C-LSTM)

A notable challenge in stock prediction is the insufficiency of features. To address this issue, the input data is reshaped to $(N, D, T, 1)$, and by employing a kernel size of (1,1) along with zero padding, convolutional operations are applied to the input data's features to generate new features. Following a single convolutional layer, the data has the shape (N, T, C), where C represents the number of filters in the convolutional layer. The model then follows with an LSTM layer and a fully connected layer.

4. Training

Train, validation, and test data are split with proportion 0.8, 0.1, 0.1. In order to keep continuous temporal relation, the split follows the time order and training batch in dataset is not shuffled.

The loss function is mean squared loss and optimizer is the Adam with learning rate 10^{-3}.

Each model is trained on 100 epochs with batch size 1000, and the final model is selected as the one that has least validation loss.

5. Results

In this section, I present the results of experiments. The performance of the models CNN, LSTM, and CLSTM is evaluated across three different datasets: D1, D2, and D3. I summarize the test loss for each model on these datasets in Table 1 and illustrate the comparisons graphically in Figures 1, 2, and 3.

Model	D1	D2	D3
CNN	12.6737	7.1158	2.8437
LSTM	0.4197	0.0589	0.0567
CLSTM	0.2261	0.0801	0.0202

Table 1. Test loss of models on Datasets D1, D2, and D3.

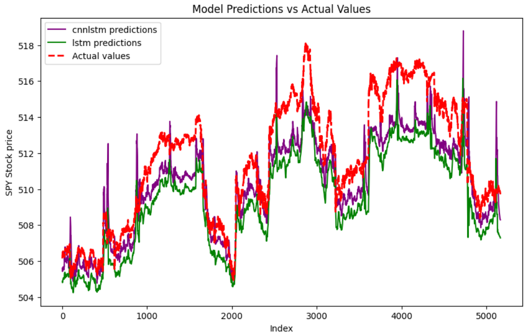

Figure 1. Models trained on D1

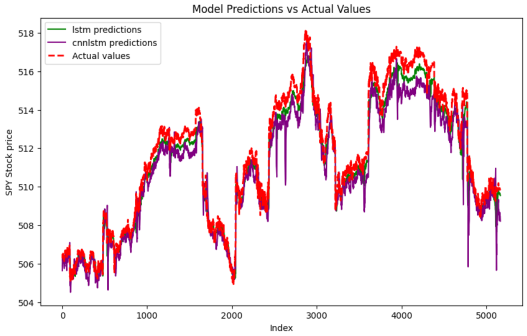

Figure 2. Models trained on D2

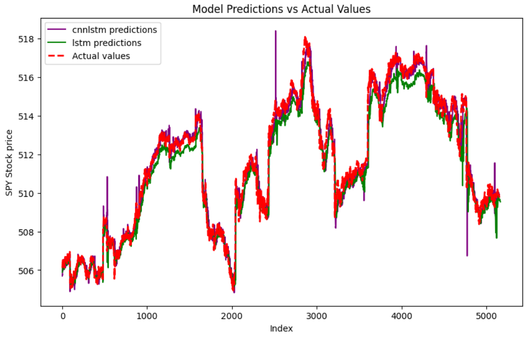

Figure 3. Models trained on D3

6. Discussion

6.1. Dataset comparison

The loss table illustrates that the inclusion of technical indicators in D2 results in enhanced performance across all three models. Moreover, the addition of news sentiment and the US dollar index in D3 contributes to further performance gains.

The most significant enhancement is observed when technical indicators are introduced in D2 compared to D1, underscoring the value of these indicators in analyzing stock market trends. Specifically, the technical indicators utilized include SMA(20), EMA(20), MACD(12, 26, 9), and RSI(14).

6.2. Model discussion

6.2.1 CNN

The loss table illustrates that the CNN architecture is not suitable for handling time series data with long dependencies. The inherent limitations of the convolution operation, particularly its sparse interactions across the receptive fields, hinder the architecture's ability to grasp long-term temporal relationships within the input data.

6.2.2 LSTM

LSTM inherently is good at managing long-term temporal relationships, marking a significant advancement over the CNN architecture. As evidenced in the loss table, LSTM demonstrates a notable improvement compared to the CNN framework.

6.2.3 C-LSTM

While the CNN architecture may not good in capturing long-term temporal relationships, it is useful at identifying local feature interactions at individual timestamps. Through its convolution operations applied to the features at each timestamp, the CNN architecture generate new features, potentially offering a better representation of the original data.

The observed enhancement in performance within D3 can be attributed, in part, to the CNN architecture's improvement in this dataset, where it achieved a 60.16% reduction in loss. This improvement also contributes to the C-LSTM model surpassing the LSTM in performance, making it the superior model across all datasets.

To mitigate the risk of the convolutional layer failing to produce valuable features or generating irrelevant information, the filter size is set to double the number of original features. This approach ensures that, in the worst-case scenario, the original features can at least be identity-mapped to the convolutional output, thereby minimizing potential information loss in the convolutional layer.

7. Code Availability

The code, models, and supplementary materials associated with this study are available in our GitHub repository. Interested parties can access these resources to replicate my results and further explore the methodologies employed. The repository can be found at Github repo.