Final Project Report

Jinfan Yang, Kesong Lin, Danqi Ding

1. Introduction

In the fast-paced world of financial markets, investors and traders are constantly seeking ways to make informed decisions and optimize their portfolios. One powerful tool in this quest is stock prediction. A stock predictor is a computational model designed to analyze historical stock data, identify patterns, and make predictions about future stock prices. It uses various factors such as historical stock prices, trading volumes, technical indicators, and sometimes external variables like economic indicators or news sentiment to generate forecasts. Stock predictors can employ different algorithms, including machine learning models like regression, decision trees, or more advanced methods like neural networks.
It is important to have a stock predictor because a stock prediction project is not just about making speculative guesses; it's about leveraging data-driven insights to make more informed and strategic decisions in the dynamic world of financial markets. By employing advanced technologies and analytics, stock predictors aim to provide investors with a valuable tool for navigating the complexities of the stock market.
This stock prediction project involves leveraging data, algorithms, and machine learning techniques to forecast the future movements of stock prices. We discussed how we selected and preprocessed data, how we built models using Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN), and what we found from our results. We also talked about challenges we met and summarized our work.

2. Data Preparing

2.1 Dataset

The dataset we used for this project is stock prices of Apple Inc. (AAPL) till now (December 1, 2023).

https://finance.yahoo.com/quote/AAPL/history/?guccounter=1&guce_referrer=aHR0cHM6Ly93d3cuZ29vZ2xlLmNvbS8&guce_referrer_sig=AQAAAH-OMQz9wk6JuDBzP1SDIkYT5gbP5QH6WreZerqaw-EvfoZnx_9XduGJ2r-iJh_m7-qiO4LdugWD1i_jCet8yQLTYYkVm8It0BaHPR9xYGRVaoA8GsboGXXQdqnXH_GJU8_OJJd9Jek3m5twZhsv73JwyJjJAGIKjdOqHU90PDaL

The dataset was sourced from Yahoo Finance, encompassing 20 years of Apple's stock history. The following variables were selected for further processing:

- *Date*
- *Open*: the first price at which the stock trades during regular market hours of the day.
- *High and Low*: the highest and lowest price at which the stock trades during the regular marketing hours of the day.
- *Close*: the final price at which the stock trades during regular market hours of the day, adjusted for splits.
- *Adj Close*: the adjusted close price and dividend and/or capital gain distributions.
- Volume: the number of shares traded during regular marketing hours of the day.

2.2 Preprocessing

- Missing Value Treatment: Utilized forward filling to maintain data continuity.
- Feature Engineering: A moving average feature was added to enhance trend analysis.
- Data Normalization: Applied MinMaxScaler for feature scaling, a crucial step for model accuracy.

3. Models

3.1 Architecture of systems

We have designed two models corresponding to two deep learning systems, CNN and RNN, for further performance comparison:

3.2.1 Convolutional Neural Network (CNN)

This model employed Conv1D layers for feature extraction from sequential data and MaxPooling1D for reducing dimensionality and was tailored to identify crucial patterns in stock price movements.

3.2.2 Recurrent Neural Network (RNN)

The model was designed with Bidirectional LSTM layers and adept at capturing both forward and backward time dependencies, which is essential for accurate stock price forecasting in the dynamic financial market.

3.2 Implementation

- The training process involved a strategic data split into training and validation sets to ensure robust model evaluation, which was also a part of data preparation. Both models were focused on minimizing Mean Squared Error, reflecting the precision in model training and evaluation for further comparison.
- For CNN model implementation, we defined a CNN-based neural network for time series prediction, with convolutional and pooling layers to capture local patterns in the input data. The architecture is designed to predict a continuous output (regression), and mean squared error is used as the loss function during training.
- For RNN model implementation, we imported the required classes and functions from the TensorFlow Keras library, which is a high-level neural networks API. The code implementation defines a deep Bidirectional LSTM-based neural network for time series prediction, with additional Dropout and Batch Normalization layers for regularization and normalization. The architecture is designed to capture complex temporal patterns in the input data for stock price prediction.

Then we provided each model onto the training data respectively. After training, we will get information about the training process, such as the loss and validation loss at each epoch. This information can be used for further analysis, such as plotting learning curves to visualize how well the model is learning from the training data.

4. Conclusion

4.1 Observations

The RNN model achieved a Mean Squared Error of 0.3923 and Mean Absolute Error of 0.6018. In comparison, the CNN model recorded a Mean Squared Error of 0.4035 and Mean Absolute Error of 0.6110. These results demonstrate the efficacy of both models in predicting stock prices, with a slight edge for the RNN model.

4.2 Discussion

In conclusion, the RNN model has a slightly better performance over the CNN model for predicting Apple stock price, but the advantage is not quite essential. Here are some thoughts over this result:

RNN is effective in capturing sequential dependencies and patterns in time series data, while CNN is better at capturing spatial patterns, which can be useful for identifying local patterns in time series data. This means that CNN may not capture long-range dependencies as effectively as RNNs. However, since it is just a slight edge of RNN over CNN, we don’t think the features of CNN and RNN made a huge difference on prediction performance here.

5. Challenge

There are several key challenges like overfitting were addressed through Dropout and BatchNormalization, ensuring model robustness. Hyperparameter tuning was meticulously conducted to optimize the models' performance.

6. Forward-Looking Statement

This extensive project underscores the potential of using sophisticated RNN and CNN models for stock price prediction. The slight advantage of the RNN model aligns with its inherent design for processing sequential data. Future explorations could include hybrid models or experimenting with newer architectures like Transformer models to further enhance prediction accuracy.