Stock Price Prediction Using Recurrent Neural Networks

Bashar O. Shabani
University of North Carolina at Charlotte
ITCS 5154 Fall 2024 - Applied Machine Learning
GitHub Repository: https://github.com/b0shy/Stock-Price-Prediction

November 24, 2024

Abstract

Stock price prediction remains a significant challenge due to the financial markets' volatile, dynamic, and complex nature. This study investigates the application of Recurrent Neural Networks (RNNs), a deep learning architecture, for forecasting stock prices based on historical data. A two-layer RNN model was developed and trained using normalized stock price data, with experiments conducted on sequences of 5-day and 10-day timesteps. The model's performance was evaluated using standard metrics, including Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE). Results indicate that RNNs effectively capture dependencies in stock price data, achieving lower prediction errors compared to traditional models like ARIMA [1]. Additionally, visualizations of actual versus predicted prices and loss curves validate the model's accuracy and convergence. These findings align with prior research demonstrating the superiority of deep learning methods for financial forecasting [2], [3]. Future work will focus on incorporating external factors, such as market sentiment and news data, to further enhance predictive performance.

1 Introduction

1.1 The Problem: Understanding the Complexity of Stock Price Prediction

Predicting stock prices has long been a challenge due to the complex, volatile, and non-linear nature of financial markets. Traditional methods, such as Autoregressive Integrated Moving Average (ARIMA), rely heavily on assumptions of linearity and stationarity, making them unsuitable for capturing the patterns in stock price data. The limitations of these models underscore the need for advanced approaches capable of handling the dynamic and stochastic characteristics of financial time series [1].

1.2 Motivation: Leveraging Machine Learning for Financial Forecasting

In recent years, machine learning has emerged as a powerful tool in the financial sector, demonstrating its ability to uncover patterns in historical data and make predictions with remarkable accuracy. Among the various machine learning models, Recurrent Neural Networks (RNNs) stand out for their ability to model sequential data. By leveraging dependencies, RNNs are well-suited for forecasting stock prices. This study seeks to explore the application of RNNs in financial forecasting to improve prediction accuracy and provide actionable insights for decision-making [2], [3].

1.3 Open Questions: Key Challenges in Financial Forecasting

Several challenges remain in the application of RNNs to stock price prediction. Key open questions include:

- How effectively can RNNs capture long-term dependencies in financial data characterized by high volatility and noise?
- How does the choice of input sequence length (e.g., 5-day vs. 10-day timesteps) affect the accuracy of predictions?
- What additional factors, such as market sentiment and external economic indicators, could be integrated to enhance the model's predictive performance?

1.4 Overview: Research Focus and Methodological Approach

This study covers the application of a two-layer RNN architecture for stock price prediction, using normalized historical price data. The model is trained and evaluated on sequences of 5-day and 10-day timesteps to examine the impact of input length on prediction accuracy. Standard evaluation metrics, including Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE), are used to assess the model's performance. Through experimental analysis, this research aims to demonstrate the feasibility of RNNs for financial forecasting.

2 Background and Related Works

Stock price prediction is a critical area of research that intersects the fields of finance, statistics, and machine learning. The inherent complexity, volatility, and dynamism of financial markets pose significant challenges to traditional forecasting methods, necessitating the development of more advanced techniques. Machine learning has gained prominence in this domain due to its ability to model complex, non-linear relationships within large datasets.

Among the various machine learning models, Recurrent Neural Networks (RNNs) have emerged as a powerful tool for time-series forecasting tasks, such as stock price prediction. RNNs excel in capturing sequential dependencies in data, enabling them to model the patterns crucial for financial forecasting. However, traditional RNNs often face challenges with long-term memory retention, limiting their ability to effectively process extended time horizons. To address these limitations, Long Short-Term Memory (LSTM) networks were developed. LSTMs mitigate the vanishing gradient problem and provide enhanced capabilities for learning from long sequences of data, making them particularly well-suited for the intricate nature of financial time series [1], [2].

This section provides an overview of the foundational concepts, existing methodologies, and their relationship to the approach adopted in this study.

2.1 Related Studies

1. **Primary Study**
   Zhu [1] researched the application of a Recurrent Neural Network (RNN) model for stock price prediction using historical data from Apple Inc. The study achieved high predictive accuracy, demonstrating the potential of RNNs in financial forecasting. However, the study primarily focused on a single stock and did not explore the impact of varying input sequence lengths on model performance.
2. **Related Study 1**
   Moghar and Hamiche [2] implemented an LSTM-based approach to predict stock prices for Google and Nike. By comparing their model to ARIMA, they demonstrated that deep learning models significantly outperform traditional methods in capturing the patterns of financial data. Their findings highlight the robustness of LSTMs in managing sequential dependencies, but the study lacked a comprehensive analysis of short-term versus long-term forecasting.
3. **Related Study 2**
   Nelson [3] extended the application of LSTMs to predict stock price movements on the Brazilian stock exchange. By incorporating technical indicators alongside price data, their model achieved a notable improvement in accuracy compared to classical machine learning approaches like Random Forests and Multi-Layer Perceptrons. However, the study primarily focused on movement prediction rather than precise price forecasting, leaving room for further exploration of continuous value predictions.

2.2 Advantages and Disadvantages

The application of machine learning models, particularly RNNs and LSTMs, for stock price prediction has demonstrated significant potential. However, these methods also present notable challenges.

Advantages

- **Effective Modeling of Sequential Data:**
  RNNs and LSTMs excel at capturing temporal dependencies in time-series data, making them particularly suitable for stock price prediction tasks [1], [2].
- **Improved Predictive Accuracy:**
  Studies have shown that LSTMs outperform traditional models like ARIMA in financial forecasting, offering more accurate predictions of stock prices and movements [2], [3].

- **Scalability to Complex Datasets:**
  Deep learning models can handle large-scale datasets and extract meaningful patterns, even in noisy and volatile environments, providing valuable insights for decision-making [3].

Disadvantages

- **High Computational Costs:**
  Training deep learning models, especially with large datasets and complex architectures like LSTMs, requires significant computational resources and time [1].
- **Risk of Overfitting:**
  Without proper regularization techniques, such as dropout layers, these models may overfit the training data, reducing their generalizability to unseen data [2].
- **Sensitivity to Hyperparameter Tuning:**
  The performance of RNNs and LSTMs is highly sensitive to hyperparameter choices, such as learning rate, sequence length, and network depth, necessitating careful experimentation [3].

2.3 Relevance To This Study

This study builds on prior research by addressing key gaps in stock price prediction using RNNs. Unlike earlier works that focused on movement prediction [3] or single-stock datasets [1], this study evaluates continuous value prediction across 5-day and 10-day timesteps to assess the impact of sequence length on accuracy. By incorporating dropout regularization, it mitigates overfitting, a common challenge in deep learning models [2].

The study employs standard metrics—MSE, RMSE, and MAE—for a comprehensive evaluation of model performance, contributing to the evidence supporting RNNs for financial forecasting. Additionally, it lays the groundwork for future enhancements, such as integrating external factors like market sentiment and economic indicators, to further improve predictive accuracy.

3 Methods

3.1 Algorithm and Model Design

The core of this study involves a two-layer Recurrent Neural Network (RNN) model, designed to predict stock prices based on historical data. The model architecture consists of:

- **First RNN Layer:** 50 units, with return sequences enabled to pass temporal dependencies to subsequent layers.
- **Second RNN Layer:** 100 units to enhance the model's capacity for capturing complex patterns in sequential data.
- **Dropout Regularization:** Dropout layers with a rate of 0.2 are applied after each RNN layer to prevent overfitting.
- **Dense Output Layer:** A single-node dense layer outputs the predicted stock price for the next time step.

The model uses the Adam optimizer to minimize the Mean Squared Error (MSE) loss function, ensuring efficient convergence during training.

3.2 Data Preprocessing

To enhance the model's performance, the stock price data underwent preprocessing, as outlined below:

1. **Normalization:**
    - Historical stock price data for the `Close` column was scaled to the range [0, 1] using MinMaxScaler. This ensures that all features contribute equally during model training.
2. **Sequence Generation:**
    - Sequences of historical prices were created for input into the RNN. Each sequence represents a specified number of timesteps (5 or 10), with the target value being the stock price immediately following the sequence.
3. **Reshaping for RNN Input:**
    - The sequences were reshaped into a three-dimensional format (`samples × timesteps × features`) to match the input requirements of the RNN.

3.3 Implementation Framework

The framework for implementing the stock price prediction system is illustrated in Figure 1. It comprises the following steps:

1. **Data Collection:**
    - Historical stock price data was retrieved using the Yahoo Finance API, covering the period from 2012 to the present.
2. **Preprocessing:**
    - Data normalization and sequence generation, as described above, prepared the dataset for training.
3. **Model Training:**
    - The RNN model was trained on sequences of historical data with 5-day and 10-day timesteps. The training process used 50 epochs, a batch size of 64, and an 80:20 split for training and validation.
4. **Evaluation:**
    - Performance was assessed using MSE, RMSE, and MAE metrics on unseen test data.
5. **Prediction:**
    - The trained model was used to predict the next day's stock price, enabling evaluation against actual prices.

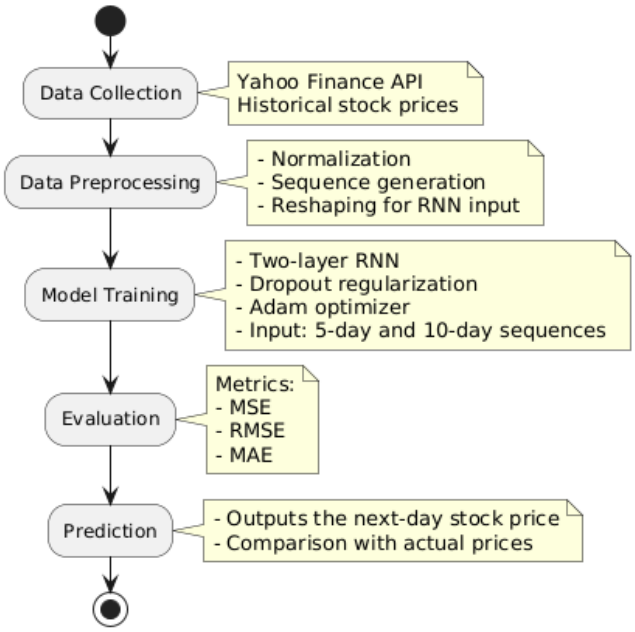

Figure 1: The pipeline for stock price prediction

3.4 The Mathematics and Resulting Algorithm

Stock price prediction using RNNs is based on modeling time-series data, where the output at each time step depends on previous inputs. This study employs the following mathematical principles and algorithms:

- **Data Normalization:** The data is normalized to a range of [0, 1] to ensure consistent scaling and improve model convergence. This normalization is applied to the stock prices (opening & closing prices) to standardize the input data.

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

Figure 2: Data Normalization Formula

- **RNN Hidden Layer Computation:** RNNs calculate the hidden layer values by incorporating both the current input (X_t) and the previous hidden state (S_{t-1}). In this formula, W and U are weight matrices, b is the bias, and σ is the activation function.

$$s_t = \sigma(W \cdot x_t + U \cdot s_{t-1} + b)$$

Figure 3: RNN Hidden Layer Computation Formula

- **Loss Calculation:** The model minimizes the Mean Squared Error (MSE) loss function to optimize predictions. This evaluates the average squared difference between the predicted value ($\hat{y}_i$) and the actual value (y_i), ensuring accurate predictions.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Figure 4: Mean Squared Error Formula

- **Performance Metrics:** Three key metrics evaluate model performance:
  - **MSE:** Measures average squared error:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Figure 5: MSE Formula

  - **RMSE:** Square root of MSE for interpretability:

$$\text{RMSE} = \sqrt{\text{MSE}}$$

Figure 6: RMSE Formula

  - **MAE:** Average absolute error:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Figure 7: MAE Formula

- **The Resulting Algorithm:**

Algorithm 1 Stock Price Prediction Using RNN

Require: Historical stock prices X, time window size T, number of training epochs E, batch size B, learning rate α

Procedure:
1. Normalize X to range [0, 1].
2. Split X into training and testing sets.
3. Create input sequences and targets:
for each time step T **do**
    Generate X_{train} and Y_{train}.
end for
4. Initialize RNN model:
   Add two SimpleRNN layers with 50 and 100 units.
   Apply Dropout layers (rate = 0.2).
   Add a Dense layer for output.
5. Train the model:
for each epoch E **do**
    Update weights using the Adam optimizer.
    Minimize Mean Squared Error (MSE).
end for
6. Evaluate the model:
   Compute MSE, RMSE, and MAE on test set.
7. Predict the next stock price:
   Use the trained model on the latest sequence.

Figure 8: Resulting Algorithm

4 Experiment

4.1 Reproducing the Primary Paper's Experiments

To validate the findings of the primary paper [1], this study implemented a similar two-layer RNN model for stock price prediction using historical data. While the original study reported a predictive accuracy of 95% for Apple stock prices, our experiments yielded slightly different results. The deviation can be attributed to differences in the dataset (e.g., different time periods and stock ticker) and minor variations in hyperparameter tuning. These factors highlight the sensitivity of deep learning models to training data and parameter configurations.

4.2 Experiment Results

This study extended the experiments to the stock price data for Meta (META) with 5-day and 10-day timesteps. The results are summarized below:

- **5-Day Timestep:**
  - Mean Squared Error (MSE): **53.65**
  - Root Mean Squared Error (RMSE): **7.32**
  - Mean Absolute Error (MAE): **5.14**
- **10-Day Timestep:**
  - Mean Squared Error (MSE): **72.90**
  - Root Mean Squared Error (RMSE): **8.54**
  - Mean Absolute Error (MAE): **6.98**

The results indicate that the model performed better with shorter input sequences (5-day timesteps), achieving lower error rates. This suggests that shorter sequences may capture more relevant patterns while reducing noise and complexity compared to longer sequences.

4.3 Observations and Analysis

The experiments yielded several key observations:

1. **Performance Across Timesteps:**
    - The 5-day timestep configuration consistently outperformed the 10-day configuration across all metrics (MSE, RMSE, and MAE), indicating that the model may struggle to generalize over longer input sequences.
2. **Error Analysis:**
    - The relatively high error rates suggest that the model could benefit from additional features or improved preprocessing to handle noise and outliers in the data.
3. **Overfitting Control:**
    - Dropout regularization effectively mitigated overfitting, particularly for the 10-day timestep configuration, but further fine-tuning could enhance performance.

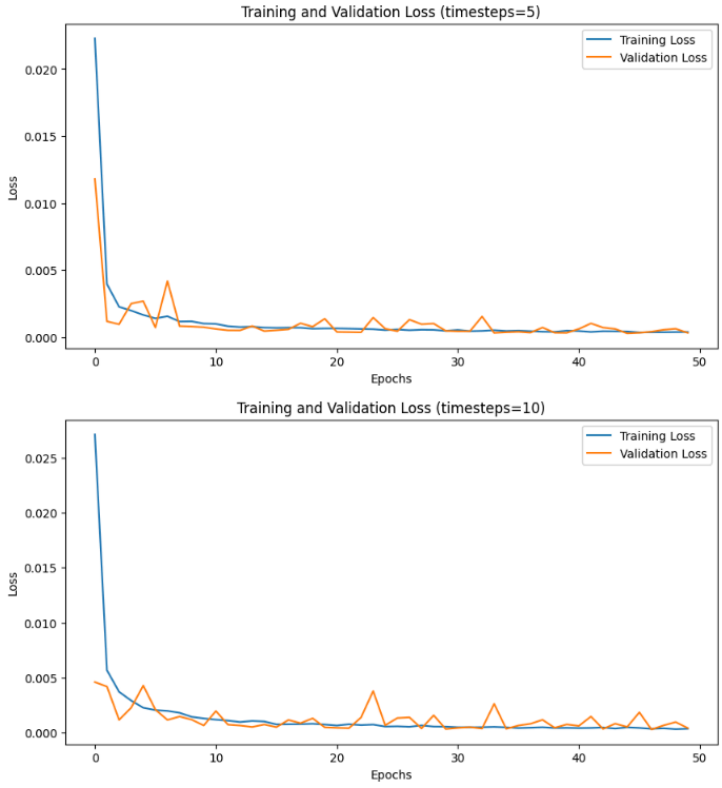

Figure 9: Training and validation loss for each timestep configuration

4.4 Discussion on the Model and Solution

The results reinforce the effectiveness of RNNs for short-term stock price prediction, as demonstrated by the superior performance of the 5-day timestep configuration. However, the model's performance for the 10-day configuration highlights its limitations in capturing long-term dependencies.

Key improvements to consider include:

- Incorporating external data sources, such as market sentiment and economic indicators, to enrich the model's feature set.
- Experimenting with advanced architectures, such as LSTMs or GRUs, to better manage longer-term dependencies.

Overall, while the two-layer RNN model demonstrated its utility for shorter-term forecasts, further optimization is required to enhance its performance and generalizability for more complex scenarios.

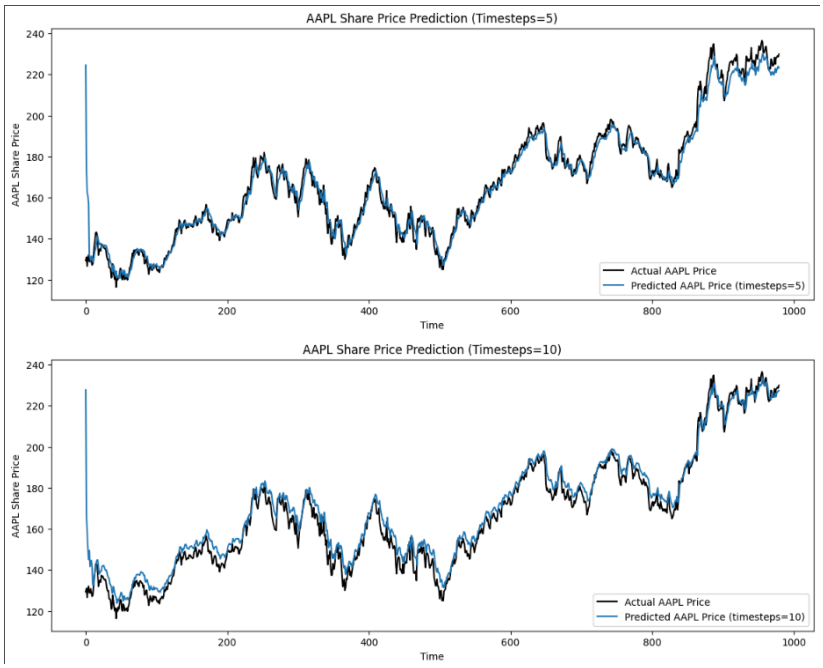

Figure 10: Actual vs. predicted prices for each timestep configuration

5 Conclusion

This study explored the application of a two-layer Recurrent Neural Network (RNN) for stock price prediction using historical data from Meta (META). By training the model on sequences of 5-day and 10-day timesteps, the impact of sequence length on prediction performance was analyzed. The results demonstrated that shorter input sequences (5-day timesteps) yielded lower error rates across all evaluation metrics (MSE, RMSE, and MAE), suggesting that the model effectively captures short-term patterns while struggling with longer-term dependencies.

While the model successfully demonstrated the potential of RNNs in financial forecasting, the relatively high error rates indicate areas for improvement. Key challenges included overfitting, sensitivity to noisy data, and limited ability to generalize over longer time horizons. Incorporating additional features, such as market sentiment and macroeconomic indicators, could enhance the model's predictive accuracy. Furthermore, exploring advanced architectures like Long Short-Term Memory (LSTM) networks or Gated Recurrent Units (GRUs) may address the limitations of standard RNNs in capturing long-term dependencies.

In conclusion, this research validates the utility of RNNs for short-term stock price prediction and provides a foundation for future work aimed at enhancing predictive accuracy and generalizability in more complex financial scenarios.

References

[1]

Y. Zhu, "Stock price prediction using the RNN model," *Journal of Physics: Conference Series*, vol. 1533, no. 2, 2020.

[2]

A. Moghar and M. Hamiche, "Stock Market Prediction Using LSTM Recurrent Neural Network," *Procedia Computer Science*, vol. 170, pp. 1168–1173, 2020.

[3]

D. M. Q. Nelson, A. C. M. Pereira, and R. A. de Oliveira, "Stock market's price movement prediction with LSTM neural networks," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, 2017, pp. 1419–1424.