Accuracy and Reliability of Stock Price Prediction Models

Hashir Bin Zahid Evan Y Jaradat Liam P Reilly William H Nguyen

Abstract

Stock market prediction plays a crucial role in enabling investors to make informed decisions and maximize returns in a highly dynamic and volatile environment. Once reserved for a select few within the wealthiest of countries, stock market prediction is now more accessible than ever. This research aimed to determine if machine learning techniques could be used to predict future stock prices and direction by leveraging historical data and utilizing basic and hybrid models. NVIDIA stock data from 1999 to 2024 was used for experimentation. By implementing basic models such as RNN, CNN, LSTM, and GRU for pattern recognition, the goal was to forecast short-term price fluctuations and price trajectory to optimize investment strategies. Additionally, a combination of these models was used in the hopes of combining their best features into a more accurate prediction tool. Results indicate that all models have their strengths and weaknesses, but while the basic models performed better on average, the hybrid models show great potential.

Introduction

In the dynamic landscape of financial markets, the ability to predict stock prices plays an essential role in shaping investment strategies, risk management, and portfolio optimization. Accurate stock market prediction remains a formidable challenge due to the inherently volatile and nonlinear nature of financial markets. Stock prices are influenced by a wide range of factors, including macroeconomic conditions, geopolitical events, company performance, and investor sentiment, making it nearly impossible to account for every variable when forecasting market movements. This complexity has driven an increasing interest in the application of advanced machine learning techniques, especially deep learning, to predict stock prices and guide trading strategies. This research focuses on investigating how machine learning models can be utilized to forecast the price trends and fluctuations in the stock market, particularly for NVIDIA stock, over the period from 1999 to 2024.

In our approach, we explore a range of machine learning models, both basic and hybrid, to assess their predictive capabilities for stock price movements. These models include simple architectures such as Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), Long Short-Term Memory networks (LSTMs), and Gated Recurrent Units (GRUs), all of which have demonstrated promise in time-series forecasting and pattern recognition. By training these models on historical stock data, we aim to capture the underlying temporal dependencies and market trends that drive price fluctuations.

The application of deep learning models to financial market prediction has emerged as a particularly promising area of research. Hybrid models, which combine architectures like CNNs, LSTMs, and GRUs, are increasingly being explored for their ability to leverage the unique strengths of each model . For

instance, CNNs excel at capturing local patterns, while LSTMs are adept at recognizing long-term dependencies [1]. By combining these strengths, we aim to improve predictive performance, creating a more robust tool for forecasting stock market movements. As machine learning technologies continue to evolve, the integration of such hybrid models has the potential to revolutionize investment strategies, offering more accurate insights into market behavior and trends.

Related Work

The application of machine learning models to stock market prediction has gained significant attention in recent years, with researchers exploring various deep learning techniques to improve prediction accuracy. A variety of neural network architectures have been used, simple and hybrid.

For instance, a study conducted by researchers on the application of neural networks to stock market prediction applied various simple neural network architectures, namely RNN, LSTM, and CNN, to forecast stock price movements [3]. After testing various sets of features and epochs, the study concluded that the LSTM model performed the best in terms of predicting stock price movements [3]. This finding aligns with our results, where the LSTM model (with an R^2 of 0.990 and MSE of 13.67) outperformed the RNN, GRU, and CNN models. The researchers conclude that combining different models could potentially improve prediction accuracy and efficiency. In our experiments, we found that hybrid models (e.g., CNN + LSTM and CNN + GRU + LSTM) often achieved worse or simple comparable performance.

A 2023 study by Sulistio et al. explored the performance of a CNN, GRU, and LSTM hybrid model for predicting stock prices in the energy sector. The researchers found that this hybrid model consistently outperformed the individual CNN, GRU, and LSTM models, improving prediction accuracy by reducing both the Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE), while increasing the R^2 value [2]. In comparison, our hybrid models did not always outperform the simpler models, but they produced comparable results. Notably, compared to our simple RNN model, our hybrid models achieved lower RMSE values (all below RNN's 21.63) and much higher R^2 values, with all hybrids reaching the 90th percentile or higher, while the RNN model's R^2 was only 66%.

Proposed Method

To achieve the goal of comparing a range of machine learning techniques for stock price prediction using NVIDIA stock data, this research begins with simpler models and progressively adds complexity. Basic models such as RNN, CNN, LSTM, and GRU are implemented first to establish a baseline for comparison. These models are then compared to hybrid models that combine the strengths of multiple architectures. Finally, ensemble models are used to aggregate the predictions of individual models to enhance accuracy. In total, this research incorporated a total of 12 models, categorized into three groups: simple models, hybrid models, and ensemble models.

Simple Models	CNN, RNN, LSTM, GRU
Hybrid Models	• RNN + LSTM • CNN + LSTM • CNN + RNN • CNN + GRU • CNN + GRU + LSTM • RNN + CNN + LSTM
Ensemble Models	• Combined (Weighted Averages) • Stack (Mean prices)

Simple Models

CNN: We employ a CNN for stock price prediction due to its ability to capture complex patterns and trends in time-series data. CNNs have demonstrated a strong capability in modeling hidden dynamics in sequential data, especially for tasks such as image processing and time-series forecasting. Unlike traditional machine learning models, which may assume a regularity in the data, CNNs can learn from the most recent data and adapt to the dynamic nature of the market [4]. This aligns with recent findings, as discussed in a study that applied deep learning models for stock price prediction and identified CNNs as the most effective architecture in their ability to detect sudden changes and volatile patterns in stock prices [4].

RNN: RNN are great in capturing temporal dependencies in time-series data and processing sequential data. Unlike traditional models, which may struggle with the sequential nature of stock data, RNNs can effectively learn from historical price movements and adapt to the dynamic market environment. Their effectiveness in recognizing complex temporal patterns is also reflected in our research, with the RNN model being the most accurate in prediction of price trend change.

LSTM: Long Short-Term Memory (LSTM) network will be employed for stock price prediction due to its ability to capture long-term dependencies in time-series data. LSTM is an advanced type of Recurrent Neural Network (RNN) designed to address the challenges of gradient vanishing and explosion, which can hinder the performance of traditional RNNs in long sequence tasks [5]. LSTMs achieve this by using memory cells that can retain information over long sequences, enabling them to model the temporal patterns inherent in financial data effectively [5]. This makes LSTM particularly suited for stock price prediction, where market data exhibits long-term dependencies and complex relationships.

GRUs: GRUs are a variant of RNNs that simplify the architecture while maintaining the ability to capture temporal dependencies. This makes them faster to train and less prone to overfitting, which is advantageous in financial forecasting. Research has shown that GRUs can achieve comparable performance to LSTMs while being computationally more efficient, making them a valuable tool for predicting stock price movements. Research has shown that GRUs can achieve comparable performance to LSTMs while being computationally more efficient. A study demonstrated that GRUs outperformed both LSTM and traditional RNN models in terms of Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) [6]. This makes GRUs a valuable tool for financial forecasting, as they strike a balance between model complexity and prediction accuracy, particularly in volatile markets.

Hybrid Models

RNN + LSTM: By stacking RNN layers followed by LSTM layers, we aimed to leverage the temporal capturing ability of RNNs alongside the long-term memory capabilities of LSTMs. This approach allows the model to capture both patterns and short-term/long-term dependencies in the stock price data, enhancing its predictive performance.

CNN + LSTM: We integrate CNNs with LSTM networks to leverage the pattern recognition capabilities of CNNs and the sequential processing power of LSTMs. The CNN layers extract features from the input data, which are then fed into LSTM layers for sequence prediction.

CNN + RNN: This model combines the local pattern recognition of CNNs with the temporal dependencies captured by RNNs. The CNN layers are used for feature extraction, followed by RNN layers for sequence modeling. This hybrid architecture allows the model to capture both spatial and temporal patterns in the stock price data, enhancing its ability to predict future price movements.

CNN + GRU: In this model, the CNN layers extract relevant features from the input data, which are then passed to GRU layers for prediction. This hybrid model aims to improve

prediction accuracy by combining the strengths of both architectures, making it effective for capturing complex patterns and temporal dependencies in stock price data.

CNN + GRU + LSTM: This complex model combines the strengths of CNNs, GRUs, and LSTMs. The CNN layers were used for feature extraction, followed by GRU and LSTM layers for sequence modeling.

RNN + CNN + LSTM: This model combines the temporal capturing ability of RNNs, the pattern recognition capability of CNNs, and the long-term memory of LSTMs. The RNN layers were used for initial sequence modeling, followed by CNN layers for feature extraction, and LSTM layers for final prediction.

Ensemble Models

Combined (Weighted Averages): This method combines the predictions of the best models using weighted averages. Each model's weight is calculated as the inverse of its MSE, ensuring models with lower MSE are given higher weights, while models with higher MSE are given lower weights. The final prediction is obtained by averaging the weighted predictions of each model.

Stack (Meta Learner - Linear Regression): This method involves stacking the predictions of individual models and using a meta-learner (Linear Regression) to combine these predictions. The process begins by creating a stacked dataset from the predictions of each model. This is achieved by making predictions with each model on the training data and stacking these predictions into a single dataset. The stacked dataset is then used to train the meta-learner, which learns how to best combine the predictions of the individual models to improve overall accuracy.

The meta-learner effectively learns the optimal weights for each model's predictions, while the directional adjustment ensures that the final predictions are directionally accurate. This method has shown to be effective in achieving a lower Mean Squared Error (MSE) and higher directional accuracy compared to individual models.

EVALUATION OF MODELS

Mean Squared Error (MSE) is used to quantify the difference between predicted and actual values by calculating the square of the deviation for each data point and averaging these squared differences across the dataset. This metric helps assess how well the model's predictions align with actual outcomes. Mathematically, MSE is computed as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Directional Accuracy is a metric that evaluates how well a model predicts the direction of price changes—whether the price goes up or down—rather than the magnitude of those changes. It is particularly useful in financial forecasting where the direction of movement is more critical than the precise price value. Mathematically, directional accuracy is given by:

$$\text{Direction(true)} = \begin{cases} 1 & \text{if } y_{\text{true}}[t] > y_{\text{true}}[t-1] \\ 0 & \text{if } y_{\text{true}}[t] \leq y_{\text{true}}[t-1] \end{cases}$$

$$\text{Direction(pred)} = \begin{cases} 1 & \text{if} y_{\text{pred}}[t] > y_{\text{pred}}[t-1] \\ 0 & \text{if } y_{\text{pred}}[t] \leq y_{\text{pred}}[t-1] \end{cases}$$

Custom F1 Score is a metric used to evaluate the performance of a predictive model by checking if the predicted price is within a delta of 5% of the actual price. This approach is particularly useful in financial forecasting, where the accuracy of predictions within a certain range is more critical than exact values.

In summary, using MSE, Directional Accuracy, and F1 Score together provides a comprehensive evaluation of our stock price prediction models. MSE measures the overall prediction error, Directional Accuracy assesses the model's ability to predict price movement direction, and the Custom F1 Score evaluates predictions within a 5% range. These metrics ensure a balanced and

thorough assessment of model performance, helping us identify the most effective techniques for predicting stock prices and optimizing investment strategies.

Experiments

Dataset Preparation

We initially aimed to use minute-by-minute price data to enhance our model's training with more data points for trend indicators. However, we soon realized that this approach was impractical due to the sheer volume of data. For NVIDIA stock alone, there would have been over 13 million data points, significantly extending the training process. Consequently, we opted to use day-to-day data instead and normalized the prices using MinMaxScaler.

At first, we used our testing data as validation data, which introduced bias. This method yielded an impressive MSE, but we discovered that it was because the model was effectively training on the test data, thus fine-tuning itself to be accurate. As a result, whenever we tested the model, it produced a very low MSE. However, when we separated the validation data from the testing data and re-trained the models from scratch, the performance initially worsened. After several training sessions, the MSE improved, though not to the previous levels, which was expected. We ultimately settled on a 70-15-15 split for training, validation, and testing, respectively.

Common Elements

All models were designed with dropout layers and Adam optimizers to enhance their performance and prevent overfitting. Dropout layers randomly deactivate specific neurons during training, forcing the network to learn redundant representations and improving generalization. The Adam optimizer, known for its efficiency and low memory requirements, was used with a learning rate of 0.00001 to ensure stable and effective training across all models. Lowering the learning rate allowed the models to tweak their weights slightly, thus drastically improving their performance. Dense layers were also added because they provide the necessary connections between neurons to learn complex patterns and relationships in the data. By including a final Dense layer with 1 neuron for output, the models can produce a single continuous value, which is essential for predicting stock prices. This combination of dropout layers, Adam optimizer, and Dense layers ensures that the models are both robust and capable of making accurate predictions.

Simple Models

CNN: Applied to identify local patterns in the stock price data. The CNN model was created with several convolutional layers followed by max-pooling layers to reduce dimensionality. The architecture includes:

- **Conv1D layer** with 64 filters and a kernel size of 5, activated by ReLU.
- **MaxPooling1D layer** with a pool size of 2.
- **Dropout layer** with a rate of 0.2 to prevent overfitting.
- **Flatten layer** to convert the 2D matrix to a vector.
- **Dense layers** with 50 and 25 neurons, activated by ReLU.

RNN: Used for its ability to capture temporal dependencies in time-series data. The RNN model was created using TensorFlow and Keras libraries. The model consists of multiple layers of LSTM units to capture long-term dependencies.

- **SimpleRNN layers** with 50 units each, activated by tanh, and configured to return sequences.

LSTM:

- **LSTM layers** with 256 and 128 units, configured to return sequences.
- **Dense layers** with 32 neurons.

GRUs:

- **GRU layers** with 100 units each, activated by tanh, and configured to return sequences.

Hybrid Models

RNN + LSTM:

- **SimpleRNN layers** with 128 units each, activated by tanh, and configured to return sequences.
- **LSTM layer** with 64 units.

CNN + LSTM:

- **Conv1D layer** with 64 filters and a kernel size of 5, activated by ReLU.
- **MaxPooling1D layer** with a pool size of 2.
- **LSTM layers** with 128 units each, configured to return sequences.

CNN + RNN:

- **Conv1D layer** with 128 filters and a kernel size of 5, activated by ReLU.
- **MaxPooling1D layer** with a pool size of 2.
- **SimpleRNN layers** with 128 units each, activated by tanh, and configured to return sequences.

CNN + GRU:

- **Conv1D layer** with 64 filters and a kernel size of 5, activated by ReLU.
- **MaxPooling1D layer** with a pool size of 2.
- **GRU layer** with 128 units, returns sequences.

CNN + GRU + LSTM:

- **Conv1D layer** with 64 filters and a kernel size of 5, activated by ReLU.
- **MaxPooling1D layer** with a pool size of 2.
- **GRU layer** with 128 units, configured to return sequences.
- **LSTM layer** with 128 units.

RNN + CNN + LSTM: Initially, the model was constructed by sequentially applying RNN, CNN, and LSTM. However, upon further analysis, we determined that starting with CNN for feature extraction would be more effective. By leveraging CNN's ability to capture intricate patterns in the data, we enhanced the model's performance. This change allowed the subsequent RNN and LSTM layers to focus on temporal dependencies with a more refined set of features, ultimately improving the overall predictive accuracy. This adjustment led to a significant reduction in MSE, decreasing from 304 to 105. The revised model architecture is outlined below:

- **Conv1D layer** with 64 filters and a kernel size of 5, activated by ReLU.
- **MaxPooling1D layer** with a pool size of 2.
- SimpleRNN layer with 128 units, activated by tanh, and configured to return sequences.
- **LSTM layers** with 128 units each, configured to return sequences.

Ensemble Models

Combined (Weighted Averages): Initially, we attempted to use all models with equal weights for predictions. However, this approach resulted in inaccurate predictions, as most hybrid and RNN models had very high MSE. Assigning them equal weight led to a low F1 score and high MSE. We tried to brute force two aspects:

1. Which models to consider
2. What each model's weight should be

We quickly realized that brute-forcing the weights up to three decimal places for 12 models was nearly impossible and would take weeks, if not months, to yield a definitive answer. Therefore, we switched to dynamic weights. This was achieved by making a model's weight inversely proportional to its MSE. Thus, the lower the MSE, the greater the weight, and vice versa. This simple change reduced the MSE from 350 to 130. However, the price difference remained too high, so we had to reduce the number of models considered. The best combination we found included LSTM, GRU, CNN, and

CNN + LSTM models. Using these models, we achieved an MSE of only 17 and a directional accuracy of 0.445.

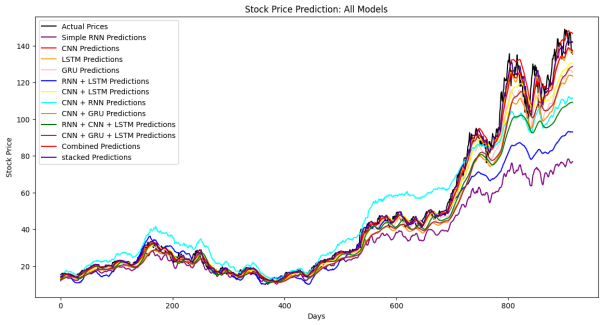
Stack (Meta Learner - Linear Regression): The stacked approach using a meta-learner (Linear Regression) had its own set of challenges. Similar to the combined model, we needed to determine which combination of models would yield the best Mean Squared Error (MSE) and directional accuracy. Initially, when we used all the models as they were and calculated the metrics, the model achieved an F1-Score of 0.76, an MSE of 10, and a directional accuracy of only 0.39.

Given the low directional accuracy, we had to devise a method to improve it. We observed that the RNN model had a notable directional accuracy of 0.54, making it better than a random guess. To enhance the directional accuracy of the stacked model, we implemented a function to adjust the direction of predictions based on the RNN's predictions. By comparing the predicted direction of the stacked model against that of the RNN model and making adjustments as needed, we were able to improve the directional accuracy from 0.39 to 0.54.

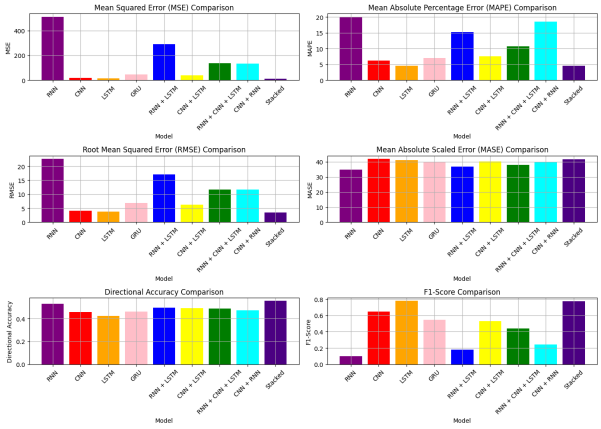
Recognizing the potential for further optimization, we explored over 1000 possible combinations of models. Initially, using all 10 models yielded the best results for MSE (6.5). For F1-Score, the optimal combination included RNN, LSTM, CNN + GRU, and CNN + RNN, achieving an 84% score. For directional accuracy, the best combination was RNN, RNN + LSTM, CNN + LSTM, and CNN + GRU + LSTM, reaching 57%.

After implementing these changes, our F1-Score improved from 0.76 to 0.77. The directional accuracy saw a significant increase from 0.39 to 0.54, and further to 0.56. However, the MSE slightly worsened from 10 to 11.6. This iterative process allowed us to fine-tune the model combinations to achieve the best possible performance across different metrics.

Results and Discussion



The results showed the strengths and weaknesses in the models. Looking at the charts and tables, it is clear that the model can either be good at predicting the direction of the price, or what the price is going to be, but not both. This is where the stack model shines, as it is able to take both the direction and mean price into account before making a prediction.



Simple Models

CNN: Achieved an MSE of 16.99, Directional Accuracy of 0.46, and an F1-Score of 0.66. The model showed a good balance between prediction accuracy and error metrics.

RNN: Had a high MSE of 467.70, but a Directional Accuracy of 0.54, indicating better performance in

predicting the direction of price changes. However, its F1-Score was low at 0.13.

LSTM: Performed well in predicting price with an MSE of 13.52. However, the directional accuracy was only 0.43, thus proving that this model is good when it comes to predicting possible closing prices, but only if the trend continues. It had an impressive F1-Score of 0.785, showing strong predictive capabilities.

GRU: Had an MSE of 27.17, Directional Accuracy of 0.462, and an F1-Score of 0.682, indicating a balanced performance, but not better than CNN.

Hybrid Models

RNN + LSTM: Showed an MSE of 274.18 and Directional Accuracy of 0.491, but a low F1-Score of 0.064, suggesting room for improvement. The high MSE of RNN seems to be heavily affecting the price prediction accuracy of the hybrid model.

CNN + LSTM: Achieved an MSE of 35.07, Directional Accuracy of 0.491, and an F1-Score of 0.078, indicating moderate performance.

CNN + RNN: Had an MSE of 128.76, Directional Accuracy of 0.458, and an F1-Score of 0.066, showing mixed results.

CNN + GRU: Performed with an MSE of 74.60, Directional Accuracy of 0.474, and an F1-Score of 0.098, indicating a balanced performance.

RNN + CNN + LSTM: Achieved an MSE of 122.67, Directional Accuracy of 0.491, and an F1-Score of 0.045, suggesting potential for improvement.

CNN + GRU + LSTM: Showed an MSE of 69.62, Directional Accuracy of 0.483, and an F1-Score of 0.092, indicating a balanced performance.

Ensemble Models

Combined: Achieved an MSE of 14.40, Directional Accuracy of 0.450, and an F1-Score of 0.694, showing strong predictive capabilities.

Stacked: Performed the best with an MSE of 11.64, Directional Accuracy of 0.554, and an F1-Score of 0.774, indicating the highest accuracy and reliability among all models.

Price Prediction

Date	CNN	RNN	LSTM	GRU	CNN + LSTM	CNN + GRU	C + G + L	Combined	Stacked	Actual Price
2024-10-21	134.585	75.411	129.102	118.864	118.555	113.005	113.932	127.016	133.353	143.710
2024-10-22	135.795	75.467	129.937	119.546	119.392	113.858	114.749	128.129	134.319	143.590
2024-10-23	136.939	74.795	130.890	120.322	120.201	114.784	115.858	129.070	135.372	139.560
2024-10-24	138.166	74.334	132.017	121.252	121.086	115.821	116.638	130.043	136.625	140.410
2024-10-25	139.269	75.128	132.760	122.047	121.859	116.707	117.796	131.144	137.345	141.540
2024-10-28	140.072	75.353	133.223	122.587	122.689	117.396	118.516	132.016	137.768	140.520
2024-10-29	141.086	75.917	133.742	123.044	123.413	118.165	119.635	132.654	138.354	141.250
2024-10-30	141.945	76.486	134.140	123.422	124.153	118.748	120.276	133.359	138.806	139.340
2024-10-31	142.585	76.617	134.320	123.654	124.755	119.291	121.257	133.956	138.963	132.760
2024-11-01	142.973	76.150	134.163	123.636	125.327	119.524	121.722	134.356	138.752	135.400
2024-11-04	143.219	76.211	133.470	123.218	125.772	119.721	122.504	134.502	137.948	136.050
2024-11-05	142.913	76.075	132.710	122.587	126.115	119.541	122.750	134.318	137.119	139.910
2024-11-06	142.578	75.544	132.141	122.015	126.329	119.289	123.241	133.884	136.529	145.610
2024-11-07	142.308	75.111	131.894	121.697	126.497	119.210	123.288	133.507	136.304	148.880
2024-11-08	142.651	74.568	132.365	121.928	126.662	119.207	123.657	133.304	136.895	147.630
2024-11-11	143.533	73.947	133.471	122.764	126.941	119.865	123.781	133.660	138.171	145.260
2024-11-12	144.677	74.499	134.794	123.953	127.239	120.492	124.317	134.538	139.565	148.290
2024-11-13	145.534	75.672	135.876	125.092	127.655	121.340	124.605	135.636	140.597	146.270
2024-11-14	146.637	76.925	136.870	126.081	128.085	122.061	125.346	136.553	141.601	146.760
2024-11-15	147.401	78.045	137.687	126.868	128.557	122.826	125.750	137.500	142.434	141.980
2024-11-18	147.978	78.273	138.283	127.433	129.007	123.419	126.507	138.245	143.022	140.150
2024-11-19	148.088	78.242	138.243	127.550	129.442	123.858	126.902	138.810	142.872	147.010
2024-11-20	147.818	77.716	137.567	127.107	129.750	123.946	127.465	138.908	142.038	145.890
2024-11-21	147.389	77.017	137.114	126.585	130.013	123.841	127.740	138.538	141.590	146.670
2024-11-22	147.129	76.241	136.936	126.270	130.173	123.772	128.065	138.189	141.457	141.950
2024-11-25	146.950	76.066	137.051	126.292	130.335	123.816	128.245	138.017	141.571	136.020
2024-11-26	147.085	76.050	137.056	126.390	130.419	123.697	128.405	138.031	141.518	136.920
2024-11-27	146.983	76.745	136.479	126.146	130.504	123.794	128.543	138.101	140.773	135.340
2024-11-29	146.650	76.829	135.548	125.489	130.459	123.361	128.597	137.818	139.720	138.250
2024-12-02	144.813	76.381	133.241	123.431	130.151	122.318	128.386	137.247	139.720	NaN

Examining the price predictions over the past 30 days, we can see how each model struggles to catch up with large fluctuations in price. The CNN model is very aggressive with its price adjustments, occasionally predicting a very accurate closing price. However, it often lags behind the actual price, with most trend

reversals causing it to sharply increase or decrease the next day's predictions. In contrast, the LSTM model exhibits more subtle price fluctuations, resulting in a larger discrepancy between actual and predicted prices. The stacked model, on the other hand, demonstrates caution in adjusting prices, generally staying closer to the actual price in most instances.

	Date	CNN	RNN	LSTM	GRU	CNN + LSTM	CNN + GRU	C + G + L	Combined	Stacked	Actual Direction
0	2024-10-21	Up	Up	Up	Up	Up	Up	Up	Up	Up	Up
1	2024-10-22	Up	Up	Up	Up	Up	Up	Up	Up	Up	Down
2	2024-10-23	Up	Down	Up	Up	Up	Up	Up	Up	Up	Down
3	2024-10-24	Up	Down	Up	Up	Up	Up	Up	Up	Up	Up
4	2024-10-25	Up	Up	Up	Up	Up	Up	Up	Up	Up	Up
5	2024-10-28	Up	Up	Up	Up	Up	Up	Up	Up	Up	Down
6	2024-10-29	Up	Up	Up	Up	Up	Up	Up	Up	Up	Up
7	2024-10-30	Up	Up	Up	Up	Up	Up	Up	Up	Up	Down
8	2024-10-31	Up	Up	Up	Up	Up	Up	Up	Up	Up	Down
9	2024-11-01	Up	Down	Down	Down	Up	Up	Up	Up	Down	Up
10	2024-11-04	Up	Up	Down	Down	Up	Up	Up	Up	Down	Up
11	2024-11-05	Down	Down	Down	Down	Up	Down	Up	Down	Down	Up
12	2024-11-06	Down	Down	Down	Down	Up	Down	Up	Down	Down	Up
13	2024-11-07	Down	Down	Down	Down	Up	Down	Up	Down	Down	Up
14	2024-11-08	Up	Down	Up	Up	Up	Down	Up	Down	Up	Down
15	2024-11-11	Up	Down	Up	Up	Up	Up	Up	Up	Up	Down
16	2024-11-12	Up	Up	Up	Up	Up	Up	Up	Up	Up	Up
17	2024-11-13	Up	Up	Up	Up	Up	Up	Up	Up	Up	Down
18	2024-11-14	Up	Up	Up	Up	Up	Up	Up	Up	Up	Up
19	2024-11-15	Up	Up	Up	Up	Up	Up	Up	Up	Up	Down
20	2024-11-18	Up	Up	Up	Up	Up	Up	Up	Up	Up	Down
21	2024-11-19	Up	Down	Down	Up	Up	Up	Up	Up	Down	Up
22	2024-11-20	Down	Down	Down	Down	Up	Up	Up	Up	Down	Down
23	2024-11-21	Down	Down	Down	Down	Up	Down	Up	Down	Down	Up
24	2024-11-22	Down	Down	Down	Down	Up	Down	Up	Down	Down	Down
25	2024-11-25	Down	Down	Up	Up	Up	Up	Up	Down	Up	Down
26	2024-11-26	Up	Down	Up	Up	Up	Down	Up	Up	Down	Up
27	2024-11-27	Down	Up	Down	Down	Up	Up	Up	Up	Down	Down
28	2024-11-29	Down	Up	Down	Down	Down	Down	Up	Down	Down	Up
29	2024-12-02	Down	Down	Down	Down	Down	Down	Down	Down	Down	None

Despite the relatively high directional accuracy of the stacked model, it is evident that this model is still susceptible to directional errors. During the initial nine days, the stacked model predicted an uptrend, while the actual prices fluctuated wildly. On the ninth day, when the price dropped drastically, the stacked model began predicting a downtrend. Ironically, during these predicted downtrend days, the stock price surged from 132 to 148, while the stacked model anticipated a decline from 138.9 to 136.3. Due to this trend reversal, the stacked model then started predicting an uptrend, which again proved incorrect as the prices continued to fluctuate wildly. Similar mistakes can be seen in all the tested models, showing how none of them can be trusted for directional accuracy alone.

Conclusion

The simple models demonstrated strong performance, with LSTM achieving particularly notable results (MSE: 13.52, F1-Score: 0.785) in price prediction, though with lower directional accuracy (0.425). This suggests that LSTM's strength lies in predicting absolute values rather than price movement directions. The CNN model also showed balanced performance across metrics (MSE: 16.99, Directional Accuracy: 0.455, F1-Score: 0.657), demonstrating the viability of simpler architectures for this task.

Contrary to initial expectations, hybrid models did not consistently outperform their simpler counterparts. While these models showed promise in combining different architectural strengths, they often struggled with increased complexity and computational demands. This suggests that more sophisticated architectures do not necessarily translate to better predictions in stock market forecasting.

The ensemble models, particularly the stacked model, emerged as the most effective overall solution. The stacked model achieved the best performance across all key metrics (MSE: 11.64, Directional Accuracy: 0.555, F1-Score: 0.775), demonstrating the value of combining multiple prediction strategies. However, upon closer inspection, especially of the trend table, it becomes apparent that none of the models are any good in directional prediction, which is the

One crucial finding is the apparent trade-off between price prediction accuracy and directional accuracy. Models that excelled at predicting exact prices often struggled with directional forecasting and vice versa. This underscores the complexity of stock market prediction and suggests that different models might be better suited for different trading strategies.

References

[1] G. Bathla, "Stock Price prediction using LSTM and SVR," 2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC), Waknaghat, India, 2020, pp. 211-214, doi: 10.1109/PDGC50313.2020.9315800. keywords: {Stock markets;Deep learning;Long short term memory;Support vector machines;Indexes;Measurement;Logic gates;Stock Price prediction;Deep Learning;LSTM;RNN;SVR},

[2] B. Sulistio, H. L. H. S. Warnars, F. L. Gaol and B. Soewito, "Energy Sector Stock Price Prediction Using The CNN, GRU & LSTM Hybrid Algorithm," 2023 International Conference on Computer Science, Information Technology and Engineering (ICCoSITE), Jakarta, Indonesia, 2023, pp. 178-182, doi: 10.1109/ICCoSITE57641.2023.10127847. keywords: {Deep learning;Computer science;Recurrent neural networks;Sociology;Prediction methods;Prediction algorithms;Convolutional neural networks;Stock prediction;Convolutional Neural Network;Long Short Term Memory;Gated Recurrent Neural Networks;Deep Learning},

[3] Fathali, Zahra, et al. “Stock Market Prediction of NIFTY 50 Index Applying Machine Learning Techniques.” *Applied Artificial Intelligence*, vol. 36, no. 1, Dec. 2022, pp. 1–24. *EBSCOhost*, https://doi-org.ezproxy.lib.uh.edu/10.1080/08839514.2022.2111134.

[4] S. Selvin, R. Vinayakumar, E. A. Gopalakrishnan, V. K. Menon and K. P. Soman, "Stock price prediction using LSTM, RNN and CNN-sliding window model," 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Udupi, India, 2017, pp. 1643-1647, doi: 10.1109/ICACCI.2017.8126078. keywords: {Companies;Time series analysis;Logic gates;Predictive models;Forecasting;Data models;Machine learning;Time series;Stock market;RNN;LSTM;CNN}

[5] P. S. Sisodia, A. Gupta, Y. Kumar and G. K. Ameta, "Stock Market Analysis and Prediction for Nifty50 using LSTM Deep Learning Approach," 2022 2nd International Conference on Innovative Practices in Technology and Management (ICIPTM), Gautam Buddha Nagar, India, 2022, pp. 156-161, doi: 10.1109/ICIPTM54933.2022.9754148. keywords: {Deep learning;Training;Predictive models;Prediction algorithms;Indexes;Stock markets;Testing;Long Short-Term Memory;Price Prediction;Deep Learning;Stock Market;Nifty50;National Stock Exchange},

[6] S. Chavhan, P. Raj, P. Raj, A. K. Dutta and J. J. P. C. Rodrigues, "Deep Learning Approaches for Stock Price Prediction: A Comparative Study of LSTM, RNN, and GRU Models," 2024 9th International Conference on Smart and Sustainable Technologies (SpliTech), Bol and Split, Croatia, 2024, pp. 01-06, doi: 10.23919/SpliTech61897.2024.10612666. keywords: {Deep learning;Adaptation models;Visualization;Recurrent neural networks;Accuracy;Predictive models;Reliability;Liquidity;Volatility;RNN;LSTM;GRU;Stock Price Prediction;Time Series},