Deep Learning Based Time Series Analysis for Stock Price Forecasting

Anand Khanna [*]
1230121531
akhann40@asu.edu

Ankur Narendrakmar Patel [*]
1229337410
apate224@asu.edu

Geeth Nischal Gottimukkala [*]
1219532899
ggottimu@asu.edu

Vajravel Conjeevaram Manigandan [*]
1225428152
vconjeev@asu.edu

Abstract

The work in this project intends to apply various Deep Learning neural network techniques for the time series forecast problem on Stock Price. The models with LSTM (Long Short-Term Memory), CNN (Convolutional Neural Networks), Bidirectional LSTM, and GRU (Gated Recurrent Unit) are implemented to predict/ forecast the stock price. To increase efficiency a combination of two or more models stated above and then the models implemented will be compared with the new SoTA algorithms such as Deep Layer Aggregation(DLA), CNN LSTM, etc.

1 Introduction

Stock prices exhibit uncertain fluctuations due to the complex interplay of numerous influencing factors. The underlying reasons could include global economic trends, shifts in unemployment rates, monetary policies of major countries, immigration policies, natural disasters, public health scenarios, and many others [3]. Stock market stakeholders aim to maximize profits and minimize risks through careful analysis of the market. The key challenge they face is consolidating multifaceted information from diverse sources and building reliable predictive models that accurately account for all the variables affecting stock prices. Bringing together the global economic data, policy changes, current events, and other factors into a holistic evaluation is crucial yet difficult to consistently achieve. Robust analytical models need to capture the interconnected dynamics behind stock price movements to enable informed investing decisions and profitable stock trading strategies and the models implemented in [3] and [4] do the same.

RNNs (Recurrent Neural Networks) have an advantage for processing short sequences but for processing data with large sequences or large distances between the relevant information, models such as LSTM and GRU come into the picture which is originally developed from RNN itself. These models have a cell state with input, forget, output gate and update, and reset gate respectively for LSTM and GRU to resolve long-term dependency. [1-3] use the LSTM model to predict/ forecast the stock price. Another idea of using 1D function to help us do the computation of convolution also seems tempting. We are implementing some of the claimed best techniques [1-7] for prediction and measurement of accuracy. The techniques include LSTM (Long short-term memory), CNN (Convolutional Neural Networks), Bidirectional LSTM, GRU (Gated Recurrent Unit), and new techniques such as DLA (Deep Layer Aggregation) and CNN LSTM. In this project, we aim to do a comparative study of the described models.

[*]Names have been ordered alphabetically.

2 Project Description

The Project is divided into four phases - (1) Literature Review, Dataset Preparation, and Algorithm Selection. (2) Models implementation and measurement. (3) Hyper Tuning, Feature Engineering, and implementing new SOTA algorithms. (4) Analysis and measurement of results, and Preparation of Final Report. The timeline of each phase is shown in Table 1. Since we have included more algorithms for the project, we have changed the project title to "Deep Learning Based Time Series Analysis for Stock Price Forecasting".

Table 1: Timeline of the Phases

Phase	Timeline
1	(Sep 21 - Oct 5)
2	(Oct 6 - Oct 30)
3	(Oct 31 - Nov 19)
4	(Nov 19 - Dec 1)

2.1 Project Timeline and Task Distribution

For the purpose of this project, we broke down the topic into smaller independent tasks. These tasks were evenly distributed, as shown in Table 2, among all the team members. Phase 1, phase 2 were completed for the last milestone report and phase 3, phase 4 are completed for this report.

Table 2: Phase-wise Task Distribution

Index	Phase	Task Description	Assignee
1	1	Literature Review	All
2	1	Dataset Preparation	Nischal
3	2	Implementation of LSTM	Nischal
4	2	Implementation of Bidirectional LSTM	Ankur
5	2	Implementation of CNN	Vajravel
6	2	Implementation of GRU	Anand
7	2	Milestone Report	All
8	3	Hyperparameter Optimization and Model Tuning	All
9	3	Implementation of hybrid models	Ankur, Nischal
10	3	Implementation of SoTA Deep Layer Aggregation (DLA)	Ankur
11	3	Implementation of SoTA CNN LSTM	Nischal
12	4	Result analysis	Vajravel, Anand
13	4	Preparation of Final Report	All

3 Implementation

The models are constructed using TensorFlow and Keras, providing a robust framework for deep learning tasks. yfinance, which is a threaded and Pythonic way to download data from Yahoo Finance is used to gather data and create datasets. The testing data set includes data on a particular stock such as open, high, low, close, and Adj. Close prices and Volume of stock for a specific date. The training dataset has data from 1 Jan 2012 to 22 August 2023. And the testing is done on the dataset which includes data from 23 August 2023 to 29 Oct 2023. The data has been scaled using a min-max scaler. The final prediction, training, and testing are done on the closing price of a particular stock. We have implemented four different models for all the techniques for various companies which include Google (GOOGL), Meta (META), General Motors (GM), and Goldman Sachs (GS). For all of the models Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and Mean Absolute Error (MAE) are measured. The results of these matrices are discussed in the results section.

3.1 LSTM

While implementing the LSTM model for this task we have used three LSTM layers, three dropout layers (one after each LSTM layer), and one dense layer. The LSTM layers are used while training and as well as while making the prediction. The dropout layers are used for dropping a fraction of neurons in a layer i.e., in each pass, the dropout layers drop 'f' ($f \in [0, 1]$) fraction of neurons randomly while training the model. However, while making the prediction the dropout layers are deactivated. The dense layer is used for giving the final prediction/ output.

The number of LSTM units used in each layer is 41 and the dropout factor used is '0.23'. The results have been obtained using the above two parameters. The number of epochs used for loss minimization is 27 and a batch size 32 has been used. The above parameters have been obtained after parameter optimization.

3.2 Bidirectional LSTM

We first imported Keras deep learning libraries like Sequential and LSTM to build the recurrent neural network model. Our time series data was loaded and preprocessed into training and test sets. To create the model, we initialized a Sequential model then added a Bidirectional LSTM layer as the first layer, set to 47 units and return full sequences. This enabled learning from past and future context. We regularized it with a 22% Dropout layer. Next, we stacked a second Bidirectional LSTM layer, again configured to 49 units and sequence return, plus another 19% Dropout. The third Bidirectional LSTM layer followed, with 50 units but no sequence return since it was the final layer. We again regularized with a 20% Dropout after. To generate the output, we used a Dense layer with 1 unit to predict the next time step value. We compiled the model with adam optimizer and mean squared error loss for regression. For training, we fitted the model on the preprocessed training time series data for 22 epochs with a batch size of 32. This iteratively updated the weights to minimize loss. We evaluated the trained model on the test set to validate accuracy. The stacked Bidirectional LSTM layers effectively learned temporal relationships from full context, achieving strong forecasting performance. The above parameters have been used after testing various parameters.

3.3 GRU

For GRU, We built a sequential model, adding a single GRU layer with 45 units and an input shape of 60 time steps by 1 feature. GRUs capture temporal dependencies like LSTMs while being simpler. After the GRU we used a dense layer with 1 unit to output predictions. We compiled with Adam optimizer at learning rate 0.001 and mean squared error loss for regression. We fitted for 35 epochs at batch size 32, monitoring training progress in verbose mode. This iteratively updated weights to minimize loss. We evaluated on the test set to validate accuracy. The single GRU layer effectively learned time relationships from sequence data. In summary, we leveraged Keras to efficiently build, train, and test a GRU forecasting model, achieving strong performance for time series prediction. The GRU identified temporal patterns to accurately predict the next steps. The above parameters have been obtained by testing with various parameters and the above are giving the best results.

3.4 CNN

The model uses Convolutional Neural Network (CNN) architecture, which excels at capturing local patterns in sequences. The model begins with two Conv1D layers, each with 256 filters, a kernel size of 6, and ReLU activation. These layers slide across the input time series to identify local features. MaxPooling1D layers follow each Conv1D layer to reduce data dimensionality and retain important information. Afterward, a flatten layer reshapes the 2D output into a 1D vector for further processing. Two Dense layers follow, with the first having 256 units and ReLU activation. This layer also applies L2 regularization (0.01) to prevent overfitting. Batch normalization is included to stabilize training, and a Dropout layer (0.24) adds regularization by randomly deactivating neurons during training. The final Dense layer, with a single unit, produces the model's predictions. The model is compiled using the Adam optimizer with a learning rate of 0.001 and the mean squared error (MSE) loss function. It is then trained on the training data for 29 epochs with a batch size of 32. Hyperparameter optimization and model tuning are crucial for achieving optimal predictive performance which has been done by the hit and trial method and the above values have been obtained.

3.5 CNN-LSTM

This CNN-LSTM model architecture comprises a Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) layers for stock price forecasting. The initial Conv1D layer with 64 filters and a kernel size of 3 convolves over the input sequences, extracting intricate patterns. Employing Rectified Linear Unit (ReLU) activation aids in capturing non-linear relationships within the data. Subsequently, a MaxPooling1D layer downsamples the data by retaining the maximum value within a pool size of 2, reducing computational complexity while preserving significant features. The model then integrates two LSTM layers, each with 53 memory units. The first LSTM layer returns sequences to capture temporal dependencies, followed by a subsequent LSTM layer without returning sequences, focusing on learning higher-level temporal patterns. Dropout layers with a dropout rate of 0.2 after each LSTM layer mitigate overfitting by randomly deactivating neurons during training. Finally, a Dense layer with one unit performs the prediction, generating the next stock price based on learned features. This combined architecture facilitates learning hierarchical features and sequential dependencies, which are fundamental for accurate stock price predictions. Adjusting layer parameters can enhance model performance based on dataset nuances and experimentation.

3.6 Deep Layer Aggregation (DLA)

With Deep Layer Aggregation, we aggregate features from different layers in a deep neural network. The implemented model uses bidirectional Long Short-Term Memory (LSTM) layers coupled with a custom attention mechanism, optimizing the model's ability to process sequential information. Central to this model is the attention_mechanism function, which applies attention to the input data, allowing the model to focus on significant parts of the time series. The architecture includes an input layer shaped according to the data's dimensions, followed by two bidirectional LSTM layers with dropout to prevent overfitting. The attention mechanism is applied after LSTM layers, enhancing the model's capacity to recognize relevant temporal patterns. The output from the attention layer is flattened and passed through a dense layer for regression output. Compiled with the Adam optimizer and mean squared error loss, the model is trained on time series data for 18 epochs, making it suitable for tasks like forecasting in time-dependent scenarios.

3.7 CNN-GRU

The implemented algorithm comprises a hybrid Convolutional Neural Network (CNN) and Gated Recurrent Unit (GRU) model designed for time series prediction, particularly for forecasting stock prices. The model architecture commences with a Conv1D layer featuring 64 filters and a kernel size of 3, employing the rectified linear unit (ReLU) activation function. Subsequently, a MaxPooling1D layer with a pool size of 2 is introduced to downsample the extracted features. The temporal dependencies within the data are captured through two GRU layers, with 50 units each, configured to return sequences in the first layer and a condensed representation in the second. Fully connected layers follow, comprising a Dense layer with 25 units and a final Dense layer outputting a single value, representing the predicted stock price. The model is compiled using the Adam optimizer and mean squared error loss function, optimizing its parameters through ten training epochs with a batch size of 32. This implementation leverages the synergy between convolutional and recurrent layers to effectively learn both spatial and temporal patterns inherent in sequential stock price data, contributing to its suitability for time series forecasting tasks.

3.8 GRU-LSTM

Combining the architectures of the Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), the LSTM-GRU model offers a strong hybrid framework for temporal sequence modeling that is specifically targeted at stock price prediction. The model starts with a 64-unit LSTM layer and then moves on to a 64-unit GRU layer. Because both layers are set up to return sequences, they can identify complex temporal relationships in the sequential stock price data. After that, a 32-unit dense layer is added to help with higher-level pattern identification by combining the collected features. A Dense output layer generates a single projected value for the future stock price when the model comes to a close.

4 Results

The graphical representation of the results is shown in Figure 1. In the figure, the columns correspond to LSTM, Bidirectional LSTM, GRU, and CNN respectively. Initial results of the Root Meat Square Error (RMSE), Mean Absolute Percentage Error (MAPE), and Mean Absolute Error (MAE) for the models and stocks are shown in Table 3, Table 4, Table 5, and Table 6. Additionally, we have implemented two hybrid models and two SOTA models which are GRU-LSTM, GRU-CNN, and CNN-LSTM, DLA respectively. The graphical representation of the results of these models is shown in Figure 2, and different scores in Table 3, Table 4, Table 5, and Table 6.

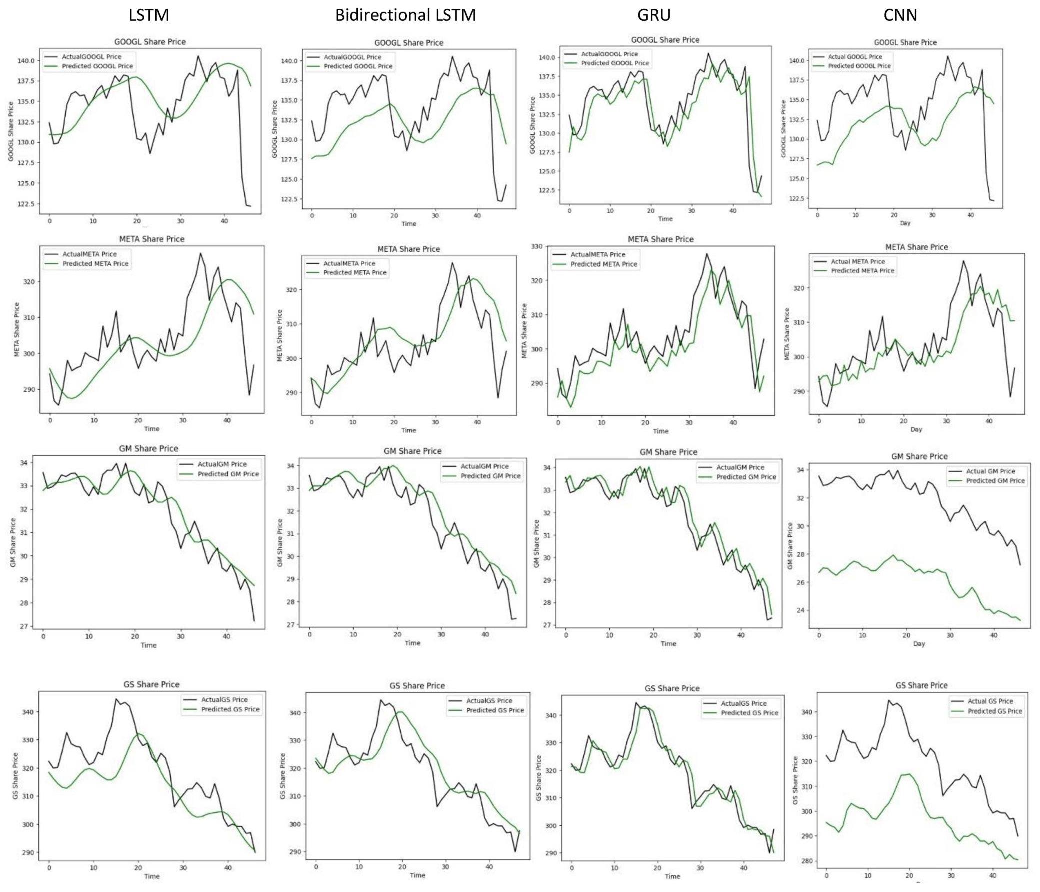

Figure 1: Stock price prediction by various models.

Table 3: Error Calculations on Prediction of Google (GOOGL) Stock Price

Model	RMSE	MAPE	MAE
LSTM	4.52	2.3%	3.08
Bidirectional LSTM	4.63	2.97%	3.95
GRU	2.56	1.54%	2.04
CNN	5.10	3.35%	4.23
SOTA CNN - LSTM	3.28	1.62%	5.04
SOTA DLA	3.01	1.59%	2.10
Hybrid GRU - LSTM	4.06	2.61%	3.47
Hybrid GRU - CNN	6.3	4.29%	5.77

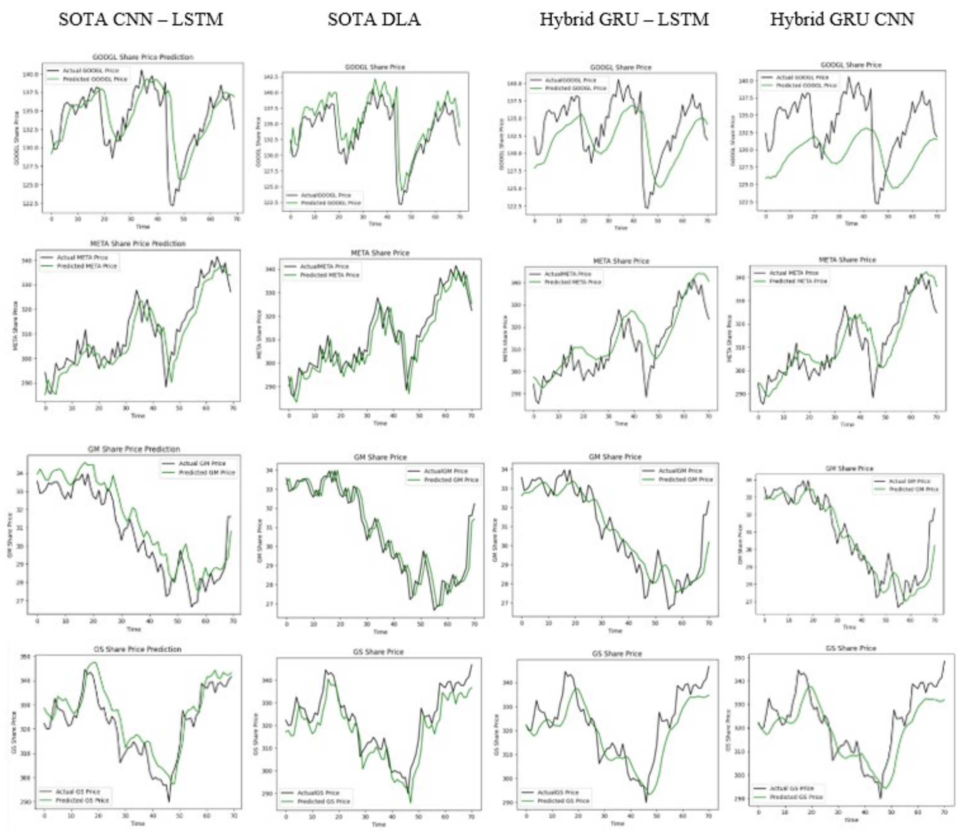

Figure 2: Stock price prediction by various models.

Table 4: Error Calculations on Prediction of Meta (META) Stock Price

Model	RMSE	MAPE	MAE
LSTM	8.89	2.05%	7.01
Bidirectional LSTM	7.23	1.80%	5.24
GRU	6.02	1.05%	5.23
CNN	7.64	3.44%	5.73
SOTA CNN - LSTM	6.26	1.62%	5.04
SOTA DLA	5.91	1.59%	4.96
Hybrid GRU - CNN	7.40	1.87%	5.82
Hybrid GRU - LSTM	8.50	2.15%	6.65

The training data is taken from 2012 to 60 days prior to the reporting date (train set). The recent 60 days have been used for testing purposes (test set) and from these all the above errors have been calculated and the plots displayed are of the last 60 days.

It is observed that SoTA algorithms are giving the best result in general compared to other algorithms. CNN being the most primitive model among the experimented models it is remarkable that we are able to achieve good metrics by combining it with LSTM. From the beginning GRU and LSTM are giving good results and GRU-LSTM also turned out to be a model that gives good results.

It has been observed that the although parameter optimization is important for machine learning models, it is not of utmost importance in the basic models we have used. Increasing epochs can simply make the model perform better. However, for the complex models used after the four basic

Table 5: Error Calculations on Prediction of General Motors (GM) Stock Price

Model	RMSE	MAPE	MAE
LSTM	0.648	1.618%	0.508
Bidirectional LSTM	0.69	1.76%	0.54
GRU	0.45	1.76%	1.47
CNN	5.78	17.80%	5.75
SOTA CNN - LSTM	1.05	2.87%	0.86
SOTA DLA	.65	1.43%	0.49
Hybrid GRU - LSTM	0.86	0.64%	2.12
Hybrid GRU - CNN	0.87	2.13 %	0.64

Table 6: Error Calculations on Prediction of Goldman Sachs (GS) Stock Price

Model	RMSE	MAPE	MAE
LSTM	9.87	2.32%	7.56
Bidirectional LSTM	6.96	1.74%	5.59
GRU	4.35	1.02%	3.24
CNN	24.19	7.46%	23.05
SOTA CNN - LSTM	6.59	1.72 %	5.52
SOTA DLA	6.65	1.67%	5.41
Hybrid GRU - LSTM	7.92	1.89%	6.15
Hybrid GRU - CNN	8.36	2.05%	6.67

models hyper tuning had a noticeable impact. It can seen from the results that the error reduction is not very high for the four basic models. However, some improvement has been observed from doing parameter optimization.

Another observation made is that the training time of SoTA algorithms is less compared to other models.

5 Conclusion

The SOTA models are performing well compared to the hybrid models and simple models. However, in some cases, the simple models are outperforming the SoTA models occasionally. Overall, the SoTA models are performing better metric-wise. However, DLA is a special case as the attention technique enhances the extraction of informative features and the discovery of patterns. Some models that are performing close to SoTA models in terms of metrics are LSTM, GRU, and GRU-LSTM. From the initial results, it is observed that hybrid models do not perform better than LSTM and GRU.

We observed that CNN-LSTM and DLA are the most sensitive models in terms of network weight initialization and dataset shuffling, in fact, these models exhibit a standard deviation over the runs, indicating a high degree of variability in their performance. Also, for these techniques, relevant and important dynamics impacting the prediction can also be achieved with shorter training datasets.

References

[1] D. Wei, "Prediction of Stock Price Based on LSTM Neural Network," 2019 International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM), Dublin, Ireland, 2019, pp. 544-547, doi: 10.1109/AIAM48774.2019.00113.

[2] 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.

[3] Y. Guo, "Stock Price Prediction Based on LSTM Neural Network: the Effectiveness of News Sentiment Analysis," 2020 2nd International Conference on Economic Management and Model Engineering (ICEMME), Chongqing, China, 2020, pp. 1018-1024, doi: 10.1109/ICEMME51517.2020.00206.

[4] G. Mu, N. Gao, Y. Wang and L. Dai, "A Stock Price Prediction Model Based on Investor Sentiment and Optimized Deep Learning," in IEEE Access, vol. 11, pp. 51353-51367, 2023, doi: 10.1109/ACCESS.2023.3278790.

[5] M. A. Istiake Sunny, M. M. S. Maswood and A. G. Alharbi, "Deep Learning-Based Stock Price Prediction Using LSTM and Bi-Directional LSTM Model," 2020 2nd Novel Intelligent and Leading Emerging Sciences Conference (NILES), Giza, Egypt, 2020, pp. 87-92, doi: 10.1109/NILES50944.2020.9257950.

[6] S. Mehtab and J. Sen, "Stock Price Prediction Using CNN and LSTM-Based Deep Learning Models," 2020 International Conference on Decision Aid Sciences and Application (DASA), Sakheer, Bahrain, 2020, pp. 447-453, doi: 10.1109/DASA51403.2020.9317207.

[7] R. Jaiswal and B. Singh, "A Hybrid Convolutional Recurrent (CNN-GRU) Model for Stock Price Prediction," 2022 IEEE 11th International Conference on Communication Systems and Network Technologies (CSNT), Indore, India, 2022, pp. 299-304, doi: 10.1109/CSNT54456.2022.9787651.