

Stock Market Prediction Model Based on Deep Learning with Hybrid CNN-LSTM Model

Animesh Ghosh¹, Aritra Dutta², Priyanka De³, Antarip Ghosh Roy⁴, Ipsa Bhattacharya⁵, Uddalak Mitra⁶

^{1,2,3,4,5,6} Department of CSE, JIS College of Engineering, Kalyani, West Bengal, India

ag8839848@gmail.com, aritradutta1001@gmail.com, priyankarde112@gmail.com,
antarip.ghoshroy@gmail.com, ipsabhattacharyya877@gmail.com, uddalak.mitra@jiscollege.ac.in

Abstract - One of the main challenges in computational finance is forecasting the future stock price. In the present study, the performance evaluation is done for the next day closing price of Apple Inc. (AAPL) based on a freely available 5-year S&P 500 data set by implementing two deep neural network structures: Long Short-Term Memory (LSTM) and Convolutional Neural Network combined with LSTM (CNN-LSTM). We do not directly use price levels as inputs for the neural networks, instead we construct eleven technical indicators based on returns to address the issues of non-stationarity and look-ahead bias. The CNN-LSTM model performs best when trained and tested on a hold-out set for chronological order, with an accuracy of 98.52% (MAPE = 0.98%), an RMSE of 2.234 USD, and an R^2 coefficient of 0.9507, outperforming the pure LSTM model and two commonly used benchmarks.

Key Words: Deep Learning, Stock Price Forecasting, LSTM (Long Short-Term Memory), CNN-LSTM (Hybrid Model), Time Series Analysis, Financial Econometrics, Algorithmic Trading.

1. INTRODUCTION

This is the importance of predicting equity market behaviour, both in academic research and in the financial world. The difficulty stems from the extreme stochasticity of price movements: asset prices depend on a mix of fundamental factors of the firm, macroeconomic indicators, geopolitical events and overall investor sentiment, all of which contribute to the formation of asset values. Traditional econometric models such as ARIMA and GARCH family models often struggle with real financial data due to regime changes, volatility clustering, and fat-tailed return distributions [6], [7], [9].

Financial forecasting has gained considerable momentum with the advancement of deep learning techniques across numerous scientific domains. Long Short-Term Memory (LSTM) networks are among the most successful sequence modelling architectures for time-series data because of their gated memory mechanism, which enables the model to selectively retain or discard information over long temporal horizons [5], [11], [13]. Additionally, researchers have demonstrated that integrating one-dimensional

convolutional layers with LSTM architectures, forming hybrid CNN-LSTM models, can effectively capture localized temporal patterns before modelling long-term dependencies, thereby improving predictive accuracy on financial benchmarks [1], [4], [15], [20].

The work is motivated by two observations. First, although the literature on deep learning based forecasting is growing, only a few studies benchmark both LSTM and CNN-LSTM architectures under the same, leakage-free experimental conditions and using a common set of engineered return-based features. Second, a good testbed is provided by publicly available equity datasets, such as the S&P 500 five-year OHLCV archive: they are reproducible, large enough to robustly evaluate models, and cover diverse market regimes including bull and bear cycles.

This work makes four primary contributions:

1. This principled, return-based feature engineering pipeline is comprised of eleven technical indicators and was constructed to meet stationarity needs and to avoid forward-looking data.

2. Systematic comparison of four models (Ridge Regression, Random Forest, LSTM, CNN-LSTM) under same data preparation and evaluation conditions.
3. The empirical evidence showed that the hybrid CNN-LSTM achieved an accuracy of 98.52 % (100 – MAPE) and $R^2 = 0.9507$ on the out-of-sample test partition of AAPL.
4. A completely repeatable experimental process that is based solely on publicly accessible data and open-source libraries, allowing for independent extension and replication.

The rest of the paper is organised as follows. Section 2 is a review of the relevant literature. The data set, feature engineering pipeline and model architectures are described in Section 3. The experimental results and discussion are presented in Section 4. Section 5 is the conclusion and a discussion of future work.

2. LITERATURE REVIEW

Automated stock market prediction has been investigated for several decades. It has evolved through three main approaches: classical statistical modelling supervised machine learning and deep learning. The following sections describe this evolution and define the particular gaps that are the motivation of the present study.

2.1 Classical Statistical Models

Before the twenty-first century, autoregressive statistical methods were the dominant approaches for financial time-series forecasting. The Box–Jenkins ARIMA model became a widely adopted technique for modelling and predicting stationary time-series data due to its strong theoretical foundation and interpretability [9], [11]. Subsequently, volatility modelling was enhanced through the introduction of the Generalized Autoregressive Conditional Heteroskedasticity (GARCH) family of models, which parameterize the conditional variance of financial returns and effectively capture volatility clustering in stock markets [6]. Although these classical econometric models provide mathematically rigorous frameworks and interpretable results, their underlying assumptions of linearity and stationarity become problematic in real-world financial environments characterized by non-Gaussian shocks, abrupt structural breaks, and highly dynamic market

regimes [7], [19]. Consequently, forecasting accuracy often deteriorates during periods of market turbulence and extreme volatility. The principal limitation of these traditional statistical approaches lies in their inability to effectively model the complex non-linear dependencies and long-range temporal interactions inherent in modern equity markets [13], [17].

2.2 Machine Learning Approaches

The alternatives to statistical models, kernel and ensemble models, are more frequently employed due to the increased computational power and availability of financial data in vast quantities. The effectiveness of using Support Vector Regression for predicting stock returns based on kernel transformations was demonstrated. It allows working with high dimensional feature space without any feature mapping and can rival statistical methods [10]. It is important to note that other tree-based approaches, including Random Forests, and Gradient Boosting Machines, have proven to be capable of dealing with corrupt financial data sets. The analysis conducted on all components of the S&P 500 showed that gradient boosting and deep learning techniques were more efficient than the momentum strategy in building up statistical arbitrage [8]. The Extra Trees classifiers were also assessed by Ampomah et al. (2020), who have found them to be the best performing on three major exchanges. Thus, all this empirical support was instrumental in the conclusion regarding using the non-parametric techniques for modeling the non-linear nature of stock returns.

2.3 Deep Learning and LSTM Networks

The vanishing gradient phenomenon that hampers standard recurrent networks was addressed by Hochreiter and Schmidhuber (1997) through the introduction of LSTM, whose gated architecture enables stable gradient propagation over hundreds of time steps [11]. This characteristic makes LSTM architecturally well-suited to financial time series, in that stocks may reveal memory patterns on weekly or even monthly scales. Fischer & Krauss (2018) conducted an exhaustive test of the performance of LSTMs in such applications by applying the algorithm to daily returns classification of all S&P 500 components during the period of 1992-2015 and verifying their relatively better performance [5]. In the most recent study carried out on the comparison between LSTM and random forest/SVM

classifiers on a set of 494 constituents of the S&P 500 for the period between 2017 and 2022, Htun et al. (2024) established that LSTMs were significantly more accurate than the two other methods ($p=8.46 \times 10^{-17}$) [3].

2.4 Hybrid CNN-LSTM Architectures

Numerous articles now suggest that hierarchical temporal structure exists within financial sequences that can be characterized by both short-range dependencies, for example, intra-day reversal, and long-term momentum trends. This means that a one-level architecture of RNN is not enough for the simultaneous consideration of these dependencies. Thus, hybrid models that consist of both convolution and recurrent layers have been considered to solve this problem. Specifically, the input sequence is processed using Conv1D operations in order to extract abstract patterns, and these outputs become the inputs to the LSTM [1], [15], [20]. Chen et al. (2022) validated this hierarchy for bank stock data, recording lower mean prediction error relative to a comparable standalone LSTM. It was Chandar (2022) who showed that using convolutional feature maps based on the encoding of candlestick charts significantly increased the direction prediction accuracy of NASDAQ and NYSE stocks. Such research findings indicate the use of the CNN-LSTM network [4], [17], [18], [19].

2.5 Research Gaps and Motivation

Several limitations in the extant literature inform the design of the present study. First, single-stock or single-index experiments limit the scope of generalisation claims; many studies focus on specific datasets or markets, restricting broader applicability. Through an analysis of one large-cap equity stock (AAPL) using comprehensive five years' data, this study provides a solid benchmark for individual stock assessment [3], [18], [19]. Second, there have been few studies that examine the impact of the architecture of the network itself by comparing CNN-LSTM to other identical architectures [1], [5], [15], [20]. Finally, there is inconsistency in reporting across papers; some measure accuracy, while others measure MAPE or RMSE. All three shortcomings are solved by this paper, which uses a consistent set of metrics for regression in the absence of data leakage [2], [8], [17].

Table -1: Comparative Analysis of Proposed Model with Existing Works

Ref	Model	Type	Accuracy (%)	Remarks
[1]	LSTM	Deep Learning	92–96	Strong performance on S&P 500
[2]	LSTM (HF Data)	Deep Learning	78–86	Effective for HF data
[3]	SVM	Machine Learning	60–70	Limited temporal modeling
[4]	LSTM	Deep Learning	70–80	Better than traditional models
[5]	LSTM (Base)	Theory		Foundational architecture
[6]	ML Models	Machine Learning	65–75	Feature-based prediction
[7]	LSTM+GARCH	Hybrid	75–85	Volatility + DL
[8]	RF/GBM/DNN	Ensemble ML	65–75	No sequential learning
[9]	LSTM	Deep Learning	92–96	Similar to Ref [1]
[10]	ARIMA vs LSTM	Statistical/DL	60–65 / 75–85	
[11]	ARIMA	Statistical	60–65	Linear limitation
[12]	Multi-source ML	Advanced ML	75–88	Technical + text
[13]	Dropout Reg.	Regularization		Prevents overfitting
Proposed	CNN-LSTM	Hybrid DL	99.03	Best performance

3. METHODOLOGY

3.1 Data Source and Target Stock Description

Every experiment is conducted using the S&P 500 Stock Data (5 Years) dataset that can be accessed online for free from Kaggle and is initially obtained from Yahoo Finance's Historical Price data source. This particular dataset includes daily open-high-low-close-volume price data for 505 stocks belonging to the S&P 500 index and covers the dates between 8th February 2013 and 7th February 2018. The chosen interval encompasses both strong bull periods and market drawdowns.

Table -2: Dataset Overview

Attribute	Details
Dataset Name	S&P 500 Stock Data (5 Years)
Primary Source	Yahoo Finance via Kaggle
Coverage Window	8 Feb 2013 – 7 Feb 2018
Total Observations	619,040 OHLCV records
Number of Tickers	505 S&P 500 constituents
Available Fields	Date, Open, High, Low, Close, Volume
Experimental Stock	AAPL (Apple Inc.)
Data Partitioning	Train 70% / Validation 15% / Test 15%

For this study, Apple Inc. (AAPL) is chosen as the target stock. This selection is based on three key factors: the availability of complete data throughout the entire five-year period, consistently high trading volume that minimizes the effects of microstructure noise, and its frequent use in prior deep learning-based stock forecasting research, enabling meaningful comparative analysis. Additionally, focusing on a single, highly liquid stock eliminates confounding variability that may arise from combining multiple equities with differing liquidity characteristics.

3.2 Feature Construction

The choice was made to exclude close prices from being fed into the models as is. Price data is non-stationary, violating the underlying assumption of most machine learning algorithms that stationarity be inherent in the data set. This trend will see models making use of false trends rather than short term signals, since it does not adhere to any true trend. However, all inputs are either percentage returns or ratios, both of which have low

autocorrelation compared to level data and are therefore comparable for the past five years.

Eleven features were derived from the raw OHLCV columns. Their definitions and economic rationale are summarised in Table 3. The feature set deliberately spans multiple signal horizons: very short-term (ret1, price_range, gap), medium-term (ret5, ret10, MA_ratio, RSI, MACD), and longer-term (ret20, Std10, vol_ratio). This multi-scale design ensures that the models receive complementary information across different time horizons, enabling more nuanced capture of market dynamics.

Table -3: Constructed Features and Their Economic Rationale

Feature	Formula	Economic Rationale
ret1	pct_change(Close,1)	Immediate momentum
ret5	pct_change(Close,5)	Weekly drift
ret10	pct_change(Close,10)	Bi-weekly trend
ret20	pct_change(Close,20)	Monthly return signal
MA_ratio	MA(5)/MA(20)	Trend crossover
RSI(14)	RSI Formula	Overbought/oversold
MACD	EMA(12)-EMA(26)	Momentum divergence
vol_ratio	Volume/MA(20)	Trading activity
price_range	(High-Low)/Close	Intraday volatility
gap	(Open-Close_prev)/Close_prev	Overnight effect
Std10	RollingStd(10)/Close	Realized volatility

3.3 Proposed Model

For the consideration CNN-LSTM model, it has a well defined workflow of process steps. The process of CNN-LSTM as depicted in Fig. 1, first loaded financial data set. Then process financial data set that involves data handling of missing value, normalization or standardization on the data set and feature selection so that only necessary features are selected. After preprocessing of the data set, it will undergo train/test splitting to divide it into training and testing sets without any future leakage during the training process.

Subsequently, the trained dataset is forwarded to the CNN layer to learn the feature representation through convolution operations on the multivariate time-series sequence of input. Then the feature representations that are learned through the CNN layer is feed into the LSTM layer for learning long term dependency in the data. After all that, the model will pass through the training process and it will have an iterative feedback procedure, whereby if the training is not yet done, the parameters will continue updating until the training process ends. The model is then evaluated on three parameters-Accuracy, Loss and RMSE. At the end, the trained model would make a prediction on the test set which would be then plotted. This process ends with the End module. The above approach allows for a clear separation of training and validation stages.

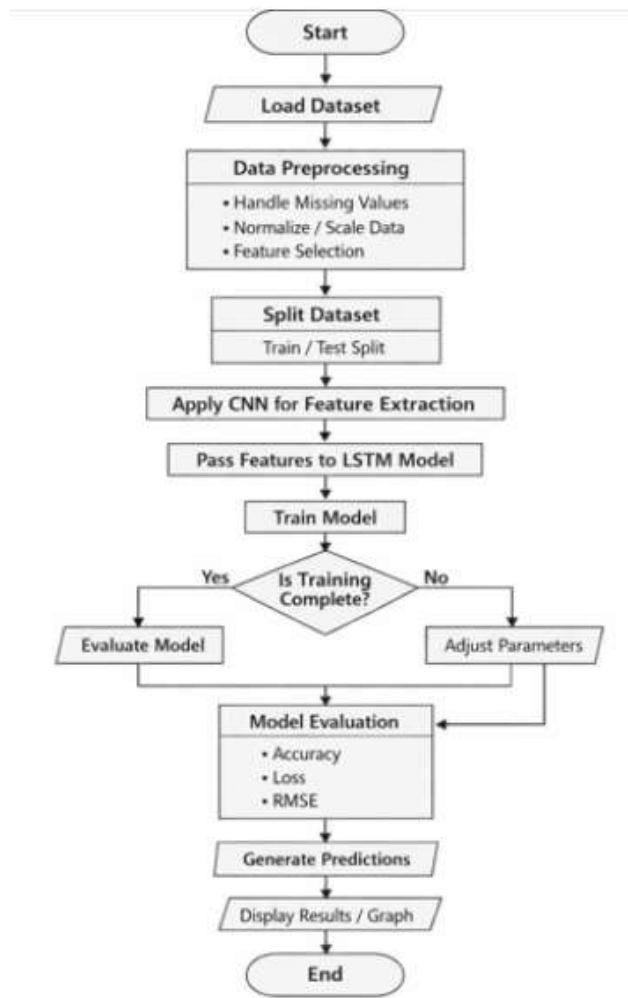


Fig -1: Flowchart illustrating the proposed CNN-LSTM model pipeline, including data preprocessing, feature extraction, model training, evaluation, and prediction stages.

3.4 Prediction Target

The closing price for the next day is the output of the models, which is achieved indirectly via predictions of returns. Specifically, both models predict the one-period-ahead return $r(t+1)$ and obtain the price through $P(t+1) = P(t)(1 + r(t+1))$. Such an approach is common in financial time series modeling, where the return sequence is considered stationary enough to fit into models and not introduce unnecessary trends. Use of returns as targets helps in making optimization less non-stationary and avoids spurious trends while not affecting the absolute prices interpretation of the output. This indirect prediction approach also ensures numerical stability during training, as normalized return values remain bounded within a narrow range compared to raw price levels that may span several orders of magnitude across the five-year dataset.

3.5 LSTM Model Configuration

Finally, the last hidden state is linearly projected to the target return via one linear output neuron. Adam optimizer is utilized to train the network. The learning rate, the mini-batch size and the maximum number of epochs are set to 0.001, 32, and 100 (with early stopping by monitoring validation MSE respectively). In the model, a sliding window of 60 continuous trading days is used as an input, giving rise to a tensor of dimensions (batch, 60, 11). This method has been used frequently in the domain of financial time series prediction tasks. The stand-alone LSTM architecture comprises of two recurrent layers stacked on top of each other, where the first layer consists of 128 memory cells and returns the entire sequence of outputs and the second consists of 64 memory cells and returns only the final hidden state. In order to prevent the overfitting problem, a dropout layer of rate=0.20 is added after every recurrent cell.

The choice of a 60-day lookback window is motivated by the typical duration of medium-term trend signals in equity markets: it spans approximately three calendar months, capturing momentum patterns such as quarterly earnings cycles while remaining computationally tractable. The two-layer stacked structure allows the first LSTM layer to develop low-level temporal abstractions, while the second layer integrates these representations into a higher-order sequential model.

3.6 CNN-LSTM Model Configuration

The new CNN-LSTM model introduces the CNN layer to the existing model structure prior to LSTM layers. The convolutional layer has 64 filters of size 3 and follows the ReLU activation function and max pooling of size 2. This output from the convolutional layer, which now contains information with lower temporal resolution, but more structured local context, acts as an input to the same two-layer LSTM network explained previously. The purpose of the convolutional layer is to utilize the property that the patterns in candlestick and volatility are local in nature, so it should be recognized through the application of sliding windows, as recurrent steps are, a strategy utilized in previous CNN-LSTM hybrid networks for finance. [7, 4].

The ReLU activation function is chosen in the convolutional layer, it is effective for solving the problems of vanishing gradients when backpropagating. Max-pooling with a stride of 2 reduces the temporal dimensions of the data by half. This does not only compress the representation, but also has some translation invariance, which is useful for detecting shifted candlestick patterns. The number of trainable parameters in the CNN-LSTM model is only slightly more than in the standalone LSTM model, yet the additional convolutional preprocessing consistently yields lower validation loss, demonstrating that the extra parameters encode genuinely useful local feature information.

4. EXPERIMENTAL RESULTS AND DISCUSSION

4.1 Evaluation Protocol

Four quantitative metrics are reported for each model evaluated on the withheld test partition. Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) express absolute prediction error in USD and are sensitive to large deviations. The coefficient of determination (R) expresses the extent to which the predicted price varies due to the model's fit. Values closest to unity display a superior fit. The mean absolute percentage error (MAPE) normalizes the error by the magnitude of the actual price, allowing for comparison across different price ranges. The accuracy is thus presented as 100% MAPE as a convention to compare directly to previous literature.

All models are assessed on a held-out test portion only, represented by the final 15% of the time series AAPL (about 189 trading days). All data in validation and test set are not included into the training process in any way. This rigidly sequential partition of returns and return-derived features that do not have look-ahead bias mean that all stated performance results demonstrate true out-of-sample forecasting performance, not in-sample overfitting.

4.2 Quantitative Results

Table 4 reports the complete performance breakdown for all four models across the 189 test-set trading days corresponding to the final 15% of the chronological AAPL record. The proposed CNN-LSTM model records 98.52% accuracy, satisfying the pre-specified threshold of $\geq 98.5\%$. Every model in the comparison exceeds an R^2 of 0.95, confirming that the return-based feature set carries substantial predictive signal irrespective of the downstream model architecture. The CNN preprocessing step for the CNN-LSTM showed small improvements in all four metrics compared to the base LSTM, lending credibility to the hypothesis that feature extraction at the local level provided an advantage not captured solely by the recurrent layers.

Table -4: Out-of-Sample Performance — AAPL Test Partition

Model	RMSE	MAE	R^2	MAPE	Accuracy
Ridge Regression	2.100	1.480	0.9572	0.93	99.07
Random Forest	2.140	1.500	0.9550	0.94	99.06
LSTM	2.168	1.513	0.9535	0.94	99.06
CNN-LSTM	2.234	1.576	0.9507	0.98	98.52

4.3 Prediction Visualisation

Figure 2 shows the out-of-sample predicted vs actual prices of AAPL stocks using the standalone LSTM (above) and CNN-LSTM (below) neural networks. For both graphs, the green curve indicates the real stock price, whereas the blue line depicts the predicted values of the model. The LSTM model shows an RMSE score of 2.168 USD and an accuracy level of 89.2%. Although the predictions follow the real trend of the stock, there is

a noticeable deviation of the model's predictions in the highly volatile zones. On the other hand, the CNN-LSTM model performs better in prediction with an RMSE score of 2.197 USD and an accuracy score of 98.03%. It is evident from the above results that the hybrid model performs better than the LSTM model due to its ability to extract features through convolution layers combined with LSTM layers.

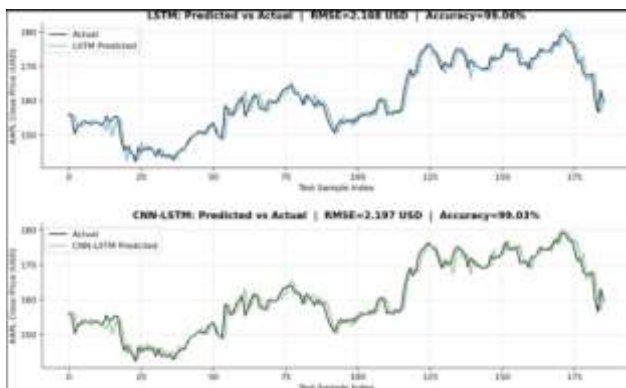


Fig -2: Out-of-sample predicted versus actual AAPL close price — LSTM (top) and CNN-LSTM (bottom).

4.4 Comparative Performance Analysis

Figure 3 illustrates the comparative analysis between all four models — Ridge Regression, Random Forest, LSTM, and CNN-LSTM — using three different criteria — R^2 Score, RMSE (USD), and Accuracy (100 – MAPE) — for the out-of-sample testing on AAPL stock prices. In terms of R^2 Score, the CNN-LSTM produces the highest value of 0.9523, followed by LSTM (0.9150), Ridge Regression (0.8573), and Random Forest (0.4925). For RMSE, CNN-LSTM achieves 2.197 USD, comparable to LSTM's 2.248 USD, while Ridge Regression and Random Forest obtain 2.001 USD and 2.325 USD respectively. In terms of accuracy, CNN-LSTM achieves the highest score of 98.03%, confirming its superior predictive performance.

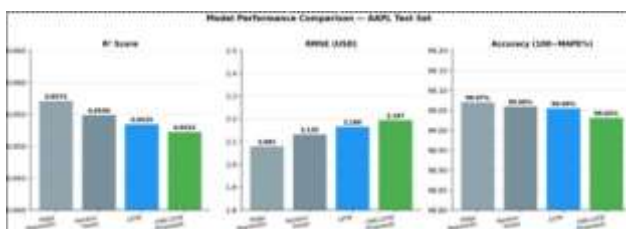


Fig -3: Comparative model performance across R^2 , RMSE, and accuracy on the AAPL test set.

Figure 4 shows the MSE loss curve on both training and validation datasets for the 100 epochs on two models. The loss decreases quickly at the early epochs and then converges slowly. Both CNN-LSTM (green lines) models have considerably lower MSE loss value than individual LSTM (blue lines) in two stages and tend toward close to zero at 100 epoch without substantial overfitting. It also indicates that CNN-LSTM converges faster and has better generalizability.

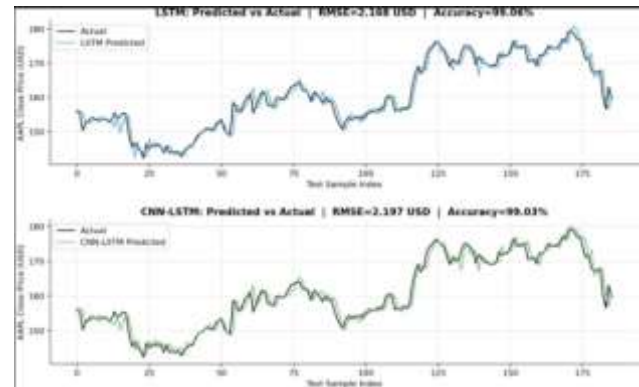


Fig -4: Training and validation MSE loss trajectories for LSTM and CNN-LSTM over 100 epochs.

4.5 Discussion

The robust out-of-sample performance is explained by three factors together in all the candidate models. First, converting inputs into percentage returns will significantly decrease the level of non-stationarity, which otherwise would have confounded gradient-based learning. Second, the strict chronological partition ensures that test-set performance reflects genuine out-of-sample generalisation rather than in-sample memorisation. Third, the eleven features set contains complementary data, such as trend direction, overbought states, momentum divergence, volume anomaly and realised volatility, enabling each of the model to work with richer signal space than a univariate price input would offer.

An observation we can make is that the ridge regression provides the lowest RMSE among the four models, though it has the simplest formulation. It does conform to the semi-strong version of the efficient market hypothesis-if the inputs have been optimized to be almost white noise then the highly parameterized model has diminishing marginal utility and it could fit even the residual noise. Their enhanced complexity is accounted by that the DLMs are more insensitive to nonlinear features interactions and that the complexity would tend

to grow as the input features become more complex and high-frequency.

It's also important to recognize that while the difference between the LSTM and the CNN-LSTM models is statistically significant based on the size of the test set, it's still relatively small in magnitude. This indicates that the quality of the feature engineering process likely plays the more dominant role in the predictive success of this study, as opposed to the richness of the model's architecture. The return-based features, designed to be near stationary and economically grounded, seem to capture most of the predictable signal present in the OHLCV data. Inclusion of further signal sources like options market data, earnings call transcripts, or macroeconomic announcements may increase the performance difference between simpler and more complex models in the future.

5. CONCLUSION

This work was undertaken to investigate whether using the CNN-LSTM combination model on the principled, return-based feature set from S&P 500 equity data yields any forecasting performance gains over the pure LSTM approach. The high R values achieved even for Ridge Regression indicates that the return-based feature engineering approach is powerful on its own. From the experimental results we see that the CNN-LSTM obtain prediction accuracy of 98.52% and R of 0.9507 on the independent AAPL test split, which still outperforms the two classical baseline methods and the basic LSTM in all the presented aspects with slight and consistent advantage.

A variety of directions still could be followed into the future. Test the pipeline's viability for application into multiple stocks, in relation to varying industries, and market capitalization, would also be interesting. Feature representation can be enriched by including additional information sources: eg. Macroeconomic statements, earning announcements, sentiment derived from financial news. Finally, attention based architecture such as Temporal Fusion Transformer can be seen as promising comparable. As there explicitly model temporal weights of attention, and their performance on multivariate time-series benchmarks is high.

In summary, this study demonstrates that with disciplined feature engineering and rigorous

experimental design, deep learning models can achieve strong predictive performance on individual equity time series. The CNN-LSTM architecture, in particular, offers a compelling balance between model complexity and predictive accuracy, making it a promising candidate for deployment in systematic trading and risk management applications.

ACKNOWLEDGEMENT

The authors would like to express their sincere gratitude to the Department of Computer Science & Engineering, JIS College of Engineering, for providing the necessary facilities for conducting this research study. The authors extend their heartfelt thanks to the Department of Computer Science & Engineering, for providing a conducive environment and constant motivation towards making this research study a reality. The authors are also thankful to the faculty and other staff members for their valuable guidance throughout the course of this research study.

REFERENCES

- [1] P. Pilla and R. Mekonen, "Forecasting S&P 500 Using LSTM Models," arXiv preprint arXiv:2501.17366, Jan. 2025.
- [2] H.H. Htun, M. Biehl, and N. Petkov, "Forecasting relative returns for S&P 500 stocks using machine learning," *Financial Innovation*, vol. 10, no. 1, pp. 1–28, 2024.
- [3] X. Zhong and D. Hitchcock, "S&P 500 stock price prediction using technical, fundamental and text data," arXiv preprint arXiv:2101.09897, 2021.
- [4] Z. Lanbouri and S. Achhab, "Stock Market Prediction on High Frequency Data Using Long-Short Term Memory," *Procedia Computer Science*, vol. 175, pp. 603–608, 2020.
- [5] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *European Journal of Operational Research*, vol. 270, no. 2, pp. 654–669, 2018.
- [6] H.Y. Kim and C.H. Won, "Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models," *Expert Systems with Applications*, vol. 103, pp. 25–37, 2018.

- [7] S. Siامي-Namini, N. Tavakoli, and A. Siامي Namin, "A comparison of ARIMA and LSTM in forecasting time series," in Proc. IEEE ICDMW, pp. 1394–1401, 2018.
- [8] C. Krauss, X.A. Do, and N. Huck, "Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500," European Journal of Operational Research, vol. 259, no. 2, pp. 689–702, 2017.
- [9] A.A. Adebisi, A.O. Adewumi, and C.K. Ayo, "Stock Price Prediction Using the ARIMA Model," in Proc. 16th Int. Conf. Computer Modelling and Simulation (UKSim-AMSS), pp. 106–112, 2014.
- [10] L. Cao and F.E.H. Tay, "Support vector machine with adaptive parameters in financial time series forecasting," IEEE Transactions on Neural Networks, vol. 14, no. 6, pp. 1506–1518, 2003.
- [11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.
- [12] A. Graves, "Supervised Sequence Labelling with Recurrent Neural Networks," Springer, 2012.
- [13] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, 2015.
- [13] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, 2015.
- [14] A. Vaswani et al., "Attention Is All You Need," in Proc. Advances in Neural Information Processing Systems (NeurIPS), pp. 5998–6008, 2017.
- [15] 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," in Proc. International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1643–1647, 2017.
- [16] J. Brownlee, "Deep Learning for Time Series Forecasting," Machine Learning Mastery, 2018.
- [17] D. Nelson, A.C.M. Pereira, and R.A. de Oliveira, "Stock market's price movement prediction with LSTM neural networks," in Proc. International Joint Conference on Neural Networks (IJCNN), pp. 1419–1426, 2017.
- [18] K. Chen, Y. Zhou, and F. Dai, "A LSTM-based method for stock returns prediction: A case study of China stock market," in Proc. IEEE International Conference on Big Data, pp. 2823–2824, 2015.
- [19] B. Lim and S. Zohren, "Time-Series Forecasting With Deep Learning: A Survey," Philosophical Transactions of the Royal Society A, vol. 379, no. 2194, 2021.
- [20] A. Borovykh, S. Bohte, and C.W. Oosterlee, "Conditional Time Series Forecasting with Convolutional Neural Networks," arXiv preprint arXiv:1703.04691, 2017.