

Integrating ARIMA and GARCH Components within Recurrent Neural Cells for Enhanced Time-Series Prediction

Kunjira Kingphai

Chiang Mai Rajabhat University (CMRU), Thailand

Received: 20 March 2026, Received in revised form: 17 May 2026, Accepted: 19 May 2026

Available online: 28 May 2026

Abstract

Background and Objectives: Financial time-series forecasting remains challenging due to the complex interaction between linear dependencies, nonlinear temporal patterns, and time-varying volatility in financial markets. Traditional statistical models such as Autoregressive Integrated Moving Average (ARIMA) and Generalized Autoregressive Conditional Heteroskedasticity (GARCH) provide interpretable representations of linear dynamics and volatility clustering but often struggle to capture nonlinear relationships in high-frequency data. Deep learning approaches, particularly Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) models, can learn nonlinear sequence representations but typically rely on implicit feature learning without incorporating established statistical structures. Most hybrid approaches use pipeline-based integration, where statistical features are computed externally and fed into neural networks, limiting interaction between statistical structure and neural dynamics. This study develops a unified forecasting framework that integrates classical time-series operators directly within recurrent neural network cell dynamics, enabling joint modeling of temporal dependencies, nonlinear sequence dynamics, and time-varying uncertainty within a single end-to-end trainable system.

Methodology: This study proposes a statistically augmented recurrent forecasting framework in which classical statistical operators are embedded directly into the hidden-state transitions of recurrent neural network architectures. Six hybrid models are developed by integrating ARIMA and volatility-based components into both RNN and LSTM structures, resulting in ARIMA–RNN, Volatility–RNN, GARCH–RNN, ARIMA–LSTM, Volatility–LSTM, and GARCH–LSTM architectures. In these models, statistical coefficients are treated as trainable parameters and jointly optimized with neural network weights using backpropagation through time, enabling unified end-to-end learning. Empirical evaluation is conducted using intraday financial data sampled at five-minute intervals from seven highly liquid U.S. technology stocks commonly referred to as the "Magnificent Seven": Apple, Microsoft, Amazon, Alphabet, Meta, NVIDIA, and Tesla. Each observation includes open, high, low, close, and trading volume variables (OHLCV). Time-series forecasting is formulated using a sliding-window approach with a look-back window of 30 time steps. The dataset covers the period from January to September 2025 and is chronologically divided into training and testing sets using an 80/20 temporal split to avoid look-ahead bias. Time-series cross-validation with

an expanding window strategy is used for model evaluation. Forecasting performance is assessed using Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE), and statistical significance is evaluated using the Scheirer-Ray-Hare test within each architecture group and Wilcoxon signed-rank tests between groups.

Main Results: The LSTM Baseline achieves the lowest overall average RMSE of 3.4955 and MAPE of 0.0080, outperforming all hybrid configurations in aggregate. Asset characteristics are the primary driver of forecasting accuracy, confirmed by a dominant stock effect across both architecture groups (Scheirer-Ray-Hare: $H > 18$, $p < 0.005$), while model selection does not produce statistically significant differences overall. LSTM-based models generally achieve lower average errors than ungated RNN architectures (RMSE 4.038 vs 4.233; MAPE 0.0092 vs 0.0107), though this difference is not statistically significant (Wilcoxon: $W = 4890$, $p = 0.925$). Statistically augmented models do not universally outperform baselines. Notably, GARCH–LSTM records the highest RMSE of 14.927 on META, suggesting that misalignment between statistical priors and event-driven volatility can amplify forecasting errors. However, during high-volatility periods, GARCH–LSTM achieved a lower MAE of 3.270 compared to the LSTM Baseline of 4.756, suggesting selective advantages for hybrid models under elevated market volatility.

Conclusions: The findings show that no single architecture consistently dominates across all assets, reflecting the heterogeneous nature of financial markets. The LSTM Baseline achieves the strongest overall performance, outperforming all hybrid configurations in aggregate. Statistical augmentation does not universally improve forecasting accuracy, and these results suggest that effective forecasting depends more on aligning model design with asset-specific characteristics than on increasing model complexity. However, hybrid models such as GARCH–LSTM showed selective advantages during high-volatility periods, suggesting that selective deployment during elevated market conditions may be more practical than general-purpose use. Several limitations should be noted. First, the evaluation is restricted to large-cap U.S. technology stocks at a single five-minute frequency, and it remains unclear whether the findings generalize to other asset classes or sampling intervals. Second, the embedded statistical coefficients are not explicitly constrained to satisfy classical parameter restrictions, as they are treated as learnable weights optimized through backpropagation rather than statistically inferred parameters. Future research should explore the application of this framework to broader asset classes such as bonds, commodities, and cryptocurrencies, as well as alternative sampling frequencies. Regime-aware model selection, where statistical priors are adaptively weighted based on real-time market conditions, represents a promising direction. Additionally, predicting returns rather than raw prices should be investigated to better align with the stationarity assumptions underlying classical ARIMA and GARCH formulations.

Keywords: machine learning; deep learning; stock price prediction; financial modelling; time-series forecasting

*Corresponding author. E-mail: kunjira@g.cmru.ac.th

Introduction

Forecasting financial time series remains a central challenge in quantitative finance due to the complexity of financial markets, which exhibit nonlinear dynamics, stochastic behaviour, structural non-stationarity, and volatility clustering (Peters, 1996; Chaudhuri & Wu, 2003). Asset prices evolve under the influence of heterogeneous factors such as investor sentiment, macroeconomic conditions, and information shocks, causing the underlying data-generating processes to deviate from classical assumptions of linearity and stationarity (Thakkar & Chaudhari, 2021). These characteristics limit forecasting models that cannot simultaneously capture nonlinear temporal dependencies and time-varying uncertainty.

Classical statistical time-series models such as Autoregressive (AR), Moving Average (MA), and Autoregressive Integrated Moving Average (ARIMA) have long served as foundational tools for financial forecasting due to their interpretability and principled modelling of autocorrelation and non-stationarity (Box *et al.*, 2015). To account for volatility clustering, Autoregressive Conditional Heteroskedasticity (ARCH) and Generalised ARCH (GARCH) models were introduced and remain widely used in volatility modelling (Chatfield & Xing, 2019; Jeribi & Ghorbel, 2022). However, these models rely primarily on linear assumptions and therefore struggle to capture complex nonlinear relationships and long-range dependencies commonly observed in high-frequency financial data (Di Persio & Honchar, 2016).

Recent advances in deep learning have expanded modelling capabilities for financial forecasting (Jiang, 2021), with systematic literature reviews highlighting the shift from shallow to deep architectures over the last two decades (Sezer *et al.*, 2020). Recurrent neural networks (RNNs), particularly Long Short-Term Memory (LSTM) architectures, have demonstrated strong performance in modelling nonlinear temporal dependencies (Fischer & Krauss, 2018; Casolaro *et al.*, 2023). Through gating mechanisms and internal memory states, LSTMs mitigate the vanishing gradient problem and enable more expressive sequence learning (Hochreiter & Schmidhuber, 1997). While other hybrid approaches involving Convolutional Neural Networks (CNNs) and Transformers have been proposed to extract spatial and long-range features (Liu, 2025), these architectures often require significantly more data and lack the specific inductive biases necessary to model the specific heteroskedastic nature of financial returns. Nevertheless, even standard recurrent architectures lack explicit mechanisms for representing classical statistical structures such as linear autocorrelation and conditional heteroskedasticity. Consequently, these properties must be learned implicitly within the neural representation, which may limit the interpretability and robustness under volatile market conditions (Hamilton, 2020).

To address these limitations, this study proposes a forecasting framework that integrates classical statistical structure with recurrent neural networks within a single end-to-end trainable architecture. In the proposed

approach, statistical operators are embedded directly within the internal dynamics of recurrent neural network cells. Specifically, ARIMA-based linear components and ARCH/GARCH-based volatility structures are incorporated into the update mechanisms of recurrent architectures. This integration enables structured modelling of linear dependence, residual dynamics, and time-varying volatility while preserving the nonlinear learning capability of recurrent networks. While several studies have explored hybrid combinations of statistical and neural models through preprocessing pipelines or residual-based ensembles (Thakkar & Chaudhari, 2021; Sezer *et al.*, 2020), the direct embedding of statistical operators within recurrent cell dynamics remains underexplored.

The main contributions of this study are summarised as follows

1. A cell-level hybridization approach for embedding ARIMA- and volatility-based statistical operators directly into recurrent neural network cells, enabling explicit modelling of linear dependence, residual dynamics, and conditional heteroskedasticity.
2. Development of six statistically augmented recurrent architectures: ARIMA–RNN, GARCH–RNN, Volatility–RNN, ARIMA–LSTM, GARCH–LSTM, and Volatility–LSTM.
3. A comprehensive multi-asset evaluation across seven large-capitalisation stocks to examine the conditional benefits and trade-offs of cell-level hybridization across assets with varying volatility characteristics.

It is important to note that the effectiveness of the proposed hybridization is not expected to be universal; rather, its benefits are anticipated to be conditional on the statistical characteristics of individual assets. Unlike conventional hybrid approaches that rely on external preprocessing or multi-stage pipelines, the proposed framework integrates statistical and neural components within a single computational graph. Statistical coefficients and neural parameters are jointly optimised using backpropagation through time, enabling a reproducible end-to-end learning process. The remainder of this paper is organised as follows. The Methodology section describes the proposed models, datasets, and experimental setup. The Results and Discussion section presents the empirical findings and analysis. Finally, the Conclusion summarises the study and outlines directions for future research.

Methodology

This section details a hybrid forecasting framework that integrates ARIMA, ARCH, and GARCH operators directly into the state-transition logic of RNN and LSTM cells. Unlike traditional ensembles, this cell-level hybridization enables the joint optimization of statistical parameters and neural weights via backpropagation through time (BPTT). By internalizing these statistical priors, the network bypasses the need to learn well-known temporal structures from scratch, allowing the recurrent dynamics to focus on capturing residual nonlinear patterns.

Unlike classical ARIMA and GARCH models where parameters are estimated via maximum likelihood, the embedded statistical coefficients are optimised with respect to an MSE objective and are therefore best interpreted as structured learnable weights rather than statistically inferred parameters. It is important to note that while these components are inspired by classical statistical formulations, they do not fully retain their original statistical properties. Specifically, the differencing operator in ARIMA–LSTM is applied to hidden state representations rather than the raw input series, approximating stationarity within the latent space rather than enforcing it on the observed data. For the volatility-based components, non-negativity of the variance term is partially preserved through the use of squared past errors in the ARCH component; however, the learnable coefficients are not explicitly constrained to satisfy classical parameter restrictions. These design choices prioritize end-to-end trainability and flexibility over strict adherence to classical statistical assumptions, and the embedded operators are therefore best understood as statistically inspired structural features rather than rigorously defined statistical models.

RNN-Based Hybrid Models

The proposed architecture modifies the standard hidden-state evolution by embedding statistical operators directly into the transition equation. This ensures that linear dependence and volatility dynamics influence state updates and memory retention in real-time.

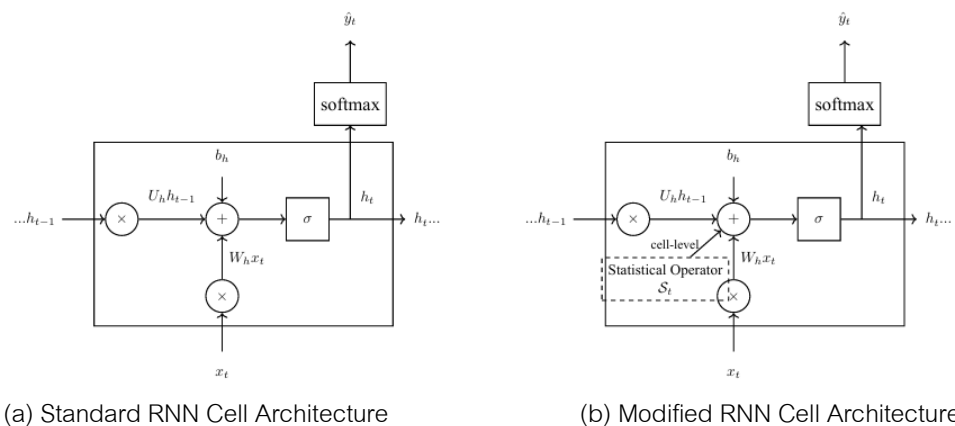


Figure 1 Architectural comparison between (a) a standard RNN cell and (b) the proposed statistically augmented RNN cell with an embedded internal statistical operator.

As illustrated in Figure 1, a conventional RNN (a) updates its hidden state via nonlinear transformations of the input and previous state. In contrast, the proposed architecture (b) embeds a statistical operator directly within the hidden-state transition equation. Unlike pipeline-level hybridization—where features are concatenated externally—this cell-level integration ensures that statistical dynamics and nonlinear transformations interact

continuously during state evolution. Based on this architecture, three variants are developed to investigate how explicit modeling of nonstationarity and volatility affects ungated recurrent networks: ARIMA–RNN, Volatility (ARCH)–RNN, and GARCH–RNN.

(1) ARIMA–RNN

The classical ARIMA model of order (p, d, q) model is defined in Equation (1). This model captures linear dependencies by applying d orders of differencing to the time series y_t to achieve stationarity:

$$\Delta^d y_t = \sum_{i=1}^p \phi_i \Delta^d y_{t-i} + \sum_{j=1}^q \theta_j \varepsilon_{t-j} + \varepsilon_t \quad (1)$$

In the proposed ARIMA–RNN architecture, the ARIMA components are integrated directly into the Recurrent Neural Network's hidden-state (h_t) transition. This allows the model to simultaneously account for linear serial correlation and non-linear feature dependencies within a single transition step:

$$h_t = \sigma(W_x x_t + U_h h_{t-1} + \sum_{i=1}^p \phi_i \Delta^d y_{t-i} + \sum_{j=1}^q \theta_j \varepsilon_{t-j} + b_h) \quad (2)$$

In Equations (1) and (2), y_t denotes the observed stock price at time t , while Δ^d represents the differencing operator of order d used to ensure stationarity. The parameters p and q are the lag orders for the AR and MA components, with ϕ_i and θ_j serving as their respective model coefficients. The term ε_t is the white noise error term (residual). Within the RNN structure, h_t and h_{t-1} are the current and previous hidden state vectors. x_t is the input vector at time t and W_x and U_h are the weight matrices for the input and recurrent connections, while b_h is the bias vector. $\sigma(\cdot)$ denotes the nonlinear activation function, such as a Sigmoid or Hyperbolic Tangent.

(2) Volatility–RNN

A classical ARCH (m) model defines conditional variance as follows:

$$\sigma_{tA}^2 = \alpha_0 + \sum_{i=1}^m \alpha_i \varepsilon_{t-i}^2 \quad (3)$$

To incorporate volatility awareness into recurrent dynamics, the Volatility–RNN embeds the conditional volatility σ_{tA} into the hidden-state update:

$$h_t = \sigma(W_x x_t + U_h h_{t-1} + \sigma_{tA} + b_h) \quad (4)$$

In Equations (3) and (4), α_0 represents the constant intercept, m is the lag order, and α_i denotes the ARCH coefficients corresponding to past squared residuals ε_{t-i}^2 . The term σ_{tA} represents the conditional volatility (standard deviation) derived from the variance process.

(3) GARCH–RNN

The standard GARCH (p, q) model extends ARCH by incorporating volatility persistence:

$$\sigma_{tG}^2 = \omega + \sum_{i=1}^q \alpha_i \varepsilon_{t-i}^2 + \sum_{j=1}^p \beta_j \sigma_{tG, t-j}^2 \quad (5)$$

The GARCH–RNN embeds this volatility structure directly into the hidden-state transition:

$$h_t = \sigma(W_h x_t + U_h h_{t-1} + \sigma_{tG} + b_h) \quad (6)$$

In Equations (5) and (6), ω is the constant intercept, while p and q represent the lag orders for the GARCH and ARCH components, respectively. The coefficient β_j quantifies volatility persistence from previous variance terms $\sigma_{tG, t-j}^2$. The resulting conditional volatility σ_{tG} is integrated into the hidden state, with the remaining recurrent variables maintaining the definitions established in the preceding sections.

LSTM-Based Hybrid Models

Unlike ungated RNNs, the LSTM-based hybrids embed statistical operators within the candidate cell-state update ($\tilde{C}_t$) rather than the hidden-state output. This placement allows the LSTM's input and forget gates to regulate the influence of statistical dynamics on memory formation. By integrating components at this level, the model preserves the stability of gated recurrence while enabling the architecture to adaptively retain or attenuate statistical information based on current market conditions.

(1) ARIMA–LSTM

The ARIMA–LSTM incorporates ARIMA components into the candidate cell-state update allowing the gating mechanism to regulate the influence of statistical dynamics on memory formation:

$$\tilde{C}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) + \sum_{i=1}^p \phi_i \Delta^d y_{t-i} + \sum_{j=1}^q \theta_j \varepsilon_{t-j} \quad (7)$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (8)$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \quad (9)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad (10)$$

$$C_t = f_t \circ C_{t-1} + i_t \circ \tilde{C}_t \quad (11)$$

$$h_t = o_t \circ \tanh(C_t) \quad (12)$$

In Equations (7) through (12), i_t , f_t and o_t are the input, forget, and output gates, respectively, while C_t denotes the cell state and $\tilde{C}_t$ is the candidate cell-state. The operator $\circ$ is the element-wise product. Each gate has its own weight matrices for input (W) and recurrent connections (U), along with a specific bias vector (b). The statistical parameters ($p, q, \phi_i, \theta_j, \Delta^d, \varepsilon_t$) and the input features (x_t, h_{t-1}) maintain the definitions established in previous sections. By placing the ARIMA components within $\tilde{C}_t$, the model regulates the contribution of linear dependencies to the long-term state evolution.

(2) Volatility-LSTM

The Volatility-LSTM embeds conditional volatility information into the candidate cell-state update as follows:

$$\tilde{C}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) + \sigma_{tA} \quad (13)$$

where σ_{tA} denotes the conditional volatility derived from the ARCH variance process defined in Equation (3). The gating mechanisms and internal LSTM parameters follow the architecture established in Equations (8) through (12). By incorporating volatility into $\tilde{C}_t$, the model allows the input and forget gates to regulate the extent to which short-term market uncertainty contributes to memory formation, improving robustness under heteroskedastic conditions.

(3) GARCH-LSTM

The GARCH-LSTM extends this formulation by embedding persistent volatility dynamics into the candidate cell-state update:

$$\tilde{C}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) + \sigma_{tG} \quad (14)$$

where σ_{tG} represents the conditional volatility derived from the GARCH formulation in Equation (5). Consistent with the previous hybrid models, the statistical signal is integrated at the candidate memory level, while the recurring weight matrices, bias vectors, and Hadamard operations maintain the definitions established in the ARIMA–LSTM and GARCH–RNN sections.

Dataset

The empirical evaluation uses intraday data from seven large-cap U.S. technology stocks known as the Magnificent Seven: Apple (AAPL), Microsoft (MSFT), Amazon (AMZN), Alphabet (GOOGL), Meta (META), NVIDIA (NVDA), and Tesla (TSLA). The selection of these assets followed the criteria of market representativeness, high liquidity, and diverse volatility profiles. These stocks are components of the S&P 500 and Nasdaq-100 and serve as benchmarks for large-cap technology trends (Nenkov, 2024). The dataset covers the period from February 13, 2025 to September 11, 2025, comprising 1,010 observations across 145 trading days for each stock, and was collected via the yfinance API at a 5-minute sampling interval (Yu, 2025). This frequency captures rapid intraday dynamics while reducing microstructure noise (Andersen *et al.*, 2001; Pooter *et al.*, 2008). An 80/20 temporal split was applied, with the first 808 observations (February 13 – July 31, 2025) used for training and the remaining 202 observations (August 1 – September 11, 2025) reserved for out-of-sample testing. Each observation includes open, high, low, close, and trading volume (OHLCV) variables describing price movements and trading activity (Uddin *et al.*, 2025).

Feature Engineering

Time-series forecasting was formulated as a supervised learning problem using a sliding-window transformation. For each observation, an input sequence X was constructed from the previous $L = 30$ time steps, while the prediction target y corresponded to the next time step. This transformation converts the time series into a three-dimensional tensor with shape [samples, sequence length, features], which is the required input format for recurrent neural networks.

Chronological Data Splitting

To avoid look-ahead bias and preserve temporal ordering, the dataset for each stock was split chronologically. The first 80% of observations were used for training and validation, while the remaining 20% were reserved as an out-of-sample test set. The test data were not used during model training or hyperparameter tuning.

Data Scaling

All variables were normalised to the range [0,1] using the MinMaxScaler from the Scikit-learn library. To prevent information leakage, the scaler was fitted only on the training data and then applied to the validation and test sets. After the model generated predictions, an inverse transformation was applied to return the data to its

original scale. This step ensured that the performance metrics, such as RMSE and MAPE, were calculated using the original price units.

Experimental Setup and Hyperparameters

All models were implemented in Python using TensorFlow (tensorflow.keras). Time-series cross-validation with an expanding window strategy (Kingphai & Moshfeghi, 2022) was applied, with the Adam optimiser, a learning rate of 0.001, and MSE as the loss function. The input window was set to $L=30$ (150 minutes of 5-minute observations, approximately one full intraday trading session). Hyperparameters were tuned via grid search across hidden sizes {32, 64, 128}, one or two recurrent layers, batch sizes {16, 32}, and 10–20 epochs. For hybrid models, ARIMA orders and ARCH/GARCH parameters were jointly optimised with neural network weights. Final hyperparameter configurations for each model and stock are reported in Table 1. The forecasting objective was to predict the closing price of each 5-minute interval using the previous 30 OHLCV observations.

Table 1 Final selected hyperparameters for all eight models across seven stocks determined via grid search

Model	Parameter	AAPL	MSFT	TSLA	AMZN	GOOGL	NVDA	META
LSTM Baseline	(H, N, E, B)	(128, 2, 20, 16)	(128, 1, 20, 16)	(128, 2, 20, 16)	(64, 1, 20, 16)	(32, 2, 20, 16)	(128, 2, 20, 16)	(32, 2, 20, 16)
RNN Baseline	(H, N, E, B)	(32, 1, 20, 16)	(128, 1, 20, 16)	(64, 1, 20, 16)	(128, 1, 10, 16)	(128, 1, 20, 32)	(128, 2, 20, 16)	(128, 1, 20, 16)
ARIMA–LSTM	(p, d, q)	(1, 1, 0)	(1, 1, 1)	(1, 1, 0)	(2, 1, 1)	(1, 1, 0)	(2, 2, 0)	(2, 1, 1)
	(H, N, E, B)	(64, 2, 20, 16)	(64, 1, 20, 16)	(32, 1, 20, 16)	(32, 1, 20, 16)	(32, 2, 20, 16)	(64, 2, 20, 16)	(32, 2, 20, 16)
GARCH–LSTM	(p, q)	(1, 2)	(1, 2)	(2, 1)	(2, 2)	(2, 2)	(2, 2)	(1, 2)
	(H, N, E, B)	(64, 2, 20, 16)	(64, 2, 20, 16)	(32, 1, 20, 16)	(32, 2, 20, 16)	(64, 1, 20, 16)	(64, 2, 20, 16)	(32, 1, 20, 16)
Volatility–LSTM	Arch (p)	1	2	2	2	1	2	1
	(H, N, E, B)	(64, 1, 20, 32)	(64, 2, 20, 16)	(32, 1, 20, 16)	(64, 2, 20, 16)	(32, 1, 20, 16)	(64, 2, 20, 16)	(64, 1, 20, 32)
ARIMA–RNN	(p, d, q)	(2, 1, 2)	(1, 1, 2)	(1, 0, 2)	(2, 0, 2)	(1, 0, 1)	(1, 1, 2)	(2, 0, 2)
	(H, N, E, B)	(64, 1, 20, 16)	(64, 2, 20, 16)	(64, 2, 20, 16)	(64, 2, 20, 16)	(64, 2, 20, 32)	(32, 2, 10, 32)	(64, 1, 20, 16)
GARCH–RNN	(p, q)	(1, 2)	(1, 2)	(2, 1)	(1, 1)	(2, 1)	(1, 2)	(1, 1)
	(H, N, E, B)	(64, 1, 20, 16)	(32, 1, 10, 16)	(64, 2, 20, 16)	(64, 1, 20, 16)	(64, 1, 10, 16)	(64, 1, 10, 16)	(64, 2, 10, 16)
Volatility–RNN	Arch (p)	2	2	1	2	2	2	1
	(H, N, E, B)	(32, 1, 20, 16)	(64, 2, 10, 32)	(64, 1, 20, 16)	(64, 2, 20, 16)	(64, 1, 10, 16)	(32, 1, 10, 32)	(32, 2, 10, 16)

Note: H = hidden size, N = number of layers, E = epochs, B = batch size; (p, d, q) = ARIMA order; (p, q) = GARCH order; p = ARCH lag order. d=0 indicates no differencing was selected, with non-stationarity addressed implicitly by the recurrent architecture.

Evaluation Metrics

Forecasting performance was evaluated using Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE).

$$MSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (15)$$

$$MAPE = \frac{1}{n} \sum_{i=1}^n \left| \frac{\hat{y}_i - y_i}{y_i} \right| \times 100\% \quad (16)$$

Both metrics were computed on the out-of-sample test data to assess forecasting accuracy. Statistical significance of performance differences was assessed using the Scheirer-Ray-Hare test (Scheirer *et al.*, 1976), a non-parametric equivalent of two-way ANOVA, applied separately within each architecture group to examine the main effects of model selection and asset characteristics as well as their interaction. Between-group comparisons were conducted using the Wilcoxon signed-rank test (Wilcoxon, 1945). All tests were performed using fold-level metric values across seven stocks.

Results

This section evaluates the performance of the statistically augmented architectures across the Magnificent Seven stocks. Using the predefined evaluation metrics and time-series cross-validation, we compare the proposed hybrids against baseline models. The analysis focuses on forecasting accuracy, predictive stability across validation folds, and the specific impact of internalizing statistical operators on model behavior.

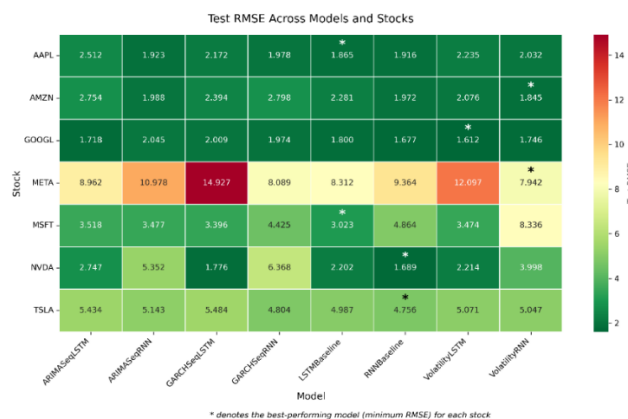


Figure 2 Heatmap of test RMSE across models and stocks. Lower values indicate better forecasting accuracy. The figure highlights variability in model performance across assets.

As illustrated in the heatmap of test RMSE provided in Figure 2, there is substantial heterogeneity in performance. RMSE values range from a minimum of 1.612 for the Volatility–LSTM applied to GOOGL to a maximum of 14.927 for the GARCH–LSTM applied to META. Such variance indicates that model effectiveness is highly sensitive to the unique statistical characteristics of individual assets. While aggregate metrics provide a global perspective, the results reveal a clear distinction between assets where baseline models are sufficient and those where statistical augmentation provides a competitive edge. The LSTM and RNN baselines yield low errors for stocks with more stable price trajectories such as AAPL and MSFT. This suggests that for these assets, the nonlinear temporal dependencies captured by standard recurrence are sufficient for accurate forecasting.

Conversely, almost all architectures struggle with the heightened volatility present in META, represented by the red and orange clusters. Interestingly, the performance on META serves as a stress test that highlights a volatility trap for certain configurations. While the Volatility–RNN and GARCH–RNN variants maintained more controlled error rates on this asset, the GARCH–LSTM saw a significant spike. This suggests that when statistical priors are misaligned with sudden, extreme market shifts, the gated memory of the LSTM may inadvertently propagate error states. This pattern may be partly explained by META's Q2 2025 earnings announcement on July 30, 2025, which appeared to trigger a rapid price movement inconsistent with the gradual volatility clustering that GARCH components are designed to capture, potentially causing the LSTM gating mechanism to amplify rather than filter the resulting variance estimate.

The benefit of gated integration is most visible in the disparity between RNN and LSTM hybrids on high-growth assets. For instance, the GARCH–LSTM achieves a specialized advantage on NVDA with a superior error of 1.776, whereas the ungated GARCH–RNN fails to generalize to the same asset and records a much higher RMSE of 6.368. This contrast implies that the gating mechanism is essential for filtering statistical operators, allowing the model to suppress the GARCH signal when it contributes to instability. Furthermore, the results for TSLA show a unique uniformity where almost all models produce similar error rates between 4.7 and 5.4. This lack of differentiation suggests a saturation point where the high idiosyncratic noise of the asset limits the marginal gains of both statistical priors and deep learning complexity.

To quantify these observations, Table 2 summarizes performance across all assets. The LSTM Baseline achieves the highest general-purpose accuracy with an RMSE of 3.4955 and a MAPE of 0.0080. While hybrid models show specialized success, they exhibit higher dispersion. The GARCH–LSTM, for example, achieves a specialized error of 1.776 on NVDA but records the highest overall mean RMSE of 4.5941 and a standard deviation of 4.7318, largely due to its performance on META.

Table 2 Aggregate Forecasting Performance across all Assets

Model	RMSEMean	RMSEStd	MAPEMean	MAPEStd
LSTM Baseline	3.4955	2.3921	0.0080	0.0030
RNN Baseline	3.7483	2.8529	0.0084	0.0022
ARIMA–LSTM	3.9491	2.4978	0.0099	0.0034
Volatility–LSTM	4.1116	3.7105	0.0088	0.0036
GARCH–RNN	4.3480	2.3077	0.0129	0.0117
ARIMA–RNN	4.4152	3.2425	0.0113	0.0079
Volatility–RNN	4.4207	2.8224	0.0103	0.0052
GARCH–LSTM	4.5941	4.7318	0.0103	0.0051

Beyond average accuracy, model stability across heterogeneous market conditions is critical for robust forecasting. Figure 3 illustrates these stability trends through error distributions. Volatility and GARCH hybrids show wider dispersion, suggesting that explicit statistical priors can amplify uncertainty during regime shifts. Conversely, the LSTM Baseline maintains the tightest distribution. This suggests that gated memory provides implicit regularization, offering more reliable adaptation than augmented variants across diverse assets.

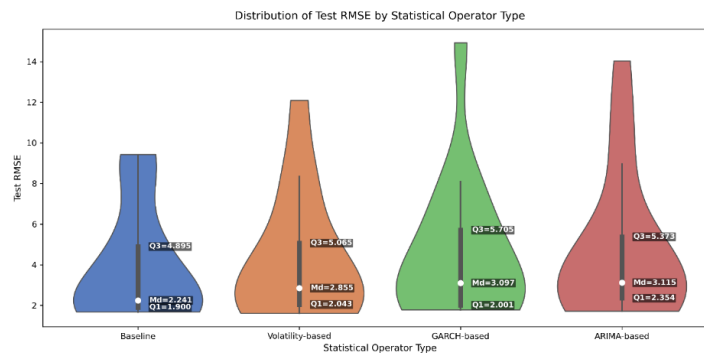


Figure 3 Violin plot of test RMSE distributions across stocks for all evaluated models. The width of each violin reflects the density of errors, highlighting variability and dispersion across assets.

To isolate the interaction between recurrent design and statistical integration, Table 3 compares the aggregated performance of all LSTM-based variants against RNN-based configurations. The results demonstrate that LSTM-based variants outperform RNN configurations with a lower mean RMSE of 4.0376 and MAPE of 0.0092. RNN-based models exhibit higher mean errors and greater MAPE variability of 0.0073, underscoring the necessity

of gating mechanisms. These results indicate that input and forget gates are more influential in determining performance than the choice of statistical prior.

Table 3 Comparative Performance by Recurrent Architecture Type

Architecture	RMSE Mean	RMSE Std	MAPE Mean	MAPE Std
LSTM	4.0376	3.2944	0.0092	0.0037
RNN	4.2331	2.6797	0.0107	0.0073

Table 4 Scheirer-Ray-Hare Test Results by Architecture Group

Group	Effect	df	H (RMSE)	p (RMSE)	H (MAPE)	p (MAPE)
LSTM	Model	3	0.289	0.962	0.074	0.995
	Stock	6	24.510	<0.001***	24.584	<0.001***
	Model × Stock	18	2.201	0.999	2.342	0.999
RNN	Model	3	1.640	0.650	2.907	0.406
	Stock	6	22.330	0.001**	18.872	0.004**
	Model × Stock	18	3.030	0.999	5.221	0.998

Note: Scheirer-Ray-Hare test (non-parametric two-way ANOVA). ** $p < 0.01$, *** $p < 0.001$, ns = not significant.

As shown in Table 4, the model effect was not statistically significant in either the LSTM or RNN group, while the stock effect was significant in both ($p < 0.001$ and $p < 0.005$ respectively). No significant interaction effect was observed in either group, and the absence of a significant model effect indicates that no pairwise post-hoc comparisons were warranted. To examine whether the two architecture groups differed overall, a Wilcoxon signed-rank test was conducted on the pooled groups. No statistically significant difference was found for either RMSE ($W = 4890$, $p = 0.925$) or MAPE ($W = 4547$, $p = 0.420$). Collectively, these results indicate that forecasting accuracy is more strongly determined by asset-specific characteristics than by model architecture or the inclusion of statistical operators. The dominant stock effect confirms that market heterogeneity — particularly the elevated volatility observed in assets such as META— is the primary driver of performance variation. Significance levels are denoted as: * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$.

To illustrate model dynamics during periods of high volatility, Figure 4 presents an intraday one-step-ahead forecast for META.

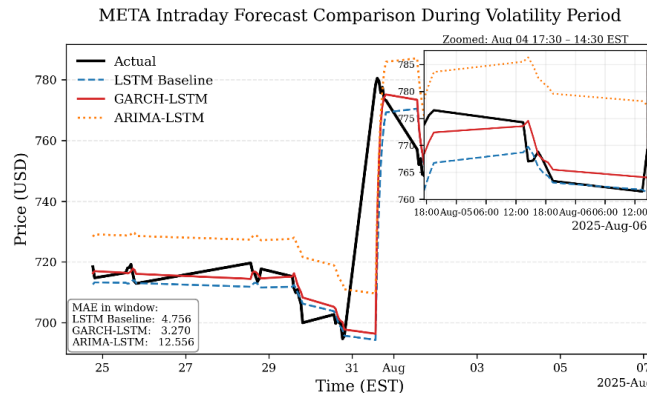


Figure 4 One-step-ahead intraday forecasts for META during a high-volatility window.

As illustrated in Figure 4, one-step-ahead forecasts are presented during a selected high-volatility window within the test period (July 25 – August 7, 2025). While the predicted trajectories of all three models generally track the actual price series, their responsiveness differs markedly during the regime shift. GARCH–LSTM demonstrates the closest adaptation to sudden price movements, achieving the lowest MAE of 3.270 within the high-volatility window, compared to 4.756 for the LSTM Baseline and 12.556 for ARIMA–LSTM. In contrast, the ARIMA-based hybrid exhibits substantial lag during rapid price adjustments, reflecting its reliance on linear residual dynamics which are insufficient to capture abrupt intraday volatility.

Discussion

The empirical results show that architectural design and the conditional nature of statistical integration play a critical role in financial time-series forecasting. LSTM-based models generally achieved lower forecasting errors than standard RNNs, although this difference was not statistically significant (Wilcoxon: $W = 4890$, $p = 0.925$). This advantage is primarily attributed to gating mechanisms, which regulate information flow and reduce the influence of market noise in high-frequency data. By retaining informative signals while suppressing transient disturbances, LSTMs tend to provide more stable performance across different assets. The tighter error distributions observed for LSTM models (Figure 3) suggest a stabilizing effect compared to ungated architectures, though this is based on distributional patterns rather than formal theory.

A key finding is that statistically augmented models do not consistently outperform baselines. This reflects a trade-off between flexibility and constraint: embedding statistical priors such as ARIMA and GARCH introduces domain knowledge but limits the model's ability to adapt freely. At the five-minute resolution, ARIMA-based hybrids produced higher average errors, suggesting that linear autocorrelation structure may be less useful than the

nonlinear patterns captured by standard LSTMs. These results indicate that the usefulness of a statistical prior depends entirely on how well it matches the data-generating process of the target asset. When statistical priors do not match the asset's behavior, such as the sudden price jump in META caused by an earnings announcement, the hybrid model can perform worse than the baseline. However, Figure 4 shows that hybrid models may still be useful during high-volatility periods, so using them selectively may be more practical than applying them generally.

Compared to modern architectures like Transformers that emphasize global temporal dependencies through attention mechanisms (Liu, 2025), the cell-level integration proposed here offers a computationally efficient alternative. By embedding statistical operators directly into recurrent update equations, the model benefits from joint optimization via backpropagation through time. This removes the need for multi-stage pipelines or external decomposition, reducing architectural complexity and improving reproducibility. However, the observed performance variability across assets suggests that this framework does not provide universal accuracy improvements, but may offer potential advantages in specific asset-volatility contexts within a unified architecture.

Conclusions

This study proposed a framework that embeds classical time-series operators directly into the cell dynamics of recurrent neural networks. By combining ARIMA and volatility-based components with RNN and LSTM architectures, the framework jointly models linear dependence, nonlinear dynamics, and time-varying uncertainty in a single end-to-end trainable system. Experiments on five-minute intraday data from the Magnificent Seven stocks show that asset characteristics are the primary driver of forecasting accuracy, confirmed by the dominant stock effect across both architecture groups (SRH: $H > 18$, $p < 0.005$). LSTM-based models generally achieved lower errors than RNN-based models, though the difference was not statistically significant, suggesting that gating mechanisms tend to provide more stable performance rather than consistently superior accuracy. While statistical augmentation did not produce overall performance gains, these results suggest that effective forecasting depends less on model complexity and more on the alignment between neural architectures and asset-specific statistical properties. These results suggest that effective forecasting depends less on model complexity and more on the alignment between neural architectures and asset-specific statistical properties.

Beyond predictive accuracy, this study suggests that cell-level hybridization may be a viable alternative to black-box scaling. While modern architectures prioritize global attention and massive parameters, internalizing statistical operators may provide a more structured inductive bias compared to simple multi-stage ensembling, though a direct comparison with ensemble methods was not conducted and remains a topic for future work. This

approach may be particularly relevant in high-frequency environments where signal-to-noise ratios are low, offering a computationally efficient way to incorporate econometric structures directly into the learning process.

While this study focused on large-cap equities at a single frequency, the results lay the groundwork for broader applications. Future research should extend this framework to diverse asset classes and explore regime-aware model selection, where statistical priors are adaptively weighted based on real-time market conditions. By integrating statistical operators within recurrent dynamics, this approach suggests a potential path toward more interpretable financial models. Since the models predict raw closing prices rather than returns, future work should also explore return-based targets to better align with the stationarity assumptions of classical ARIMA and GARCH formulations.

Acknowledgments

The author acknowledges the Faculty of Science and Technology, Chiang Mai Rajabhat University, Thailand, for research support and thanks colleagues for their helpful comments.

Declaration of Generative AI in the writing process

The authors used an AI-assisted tool for language enhancement and to improve the readability of the manuscript. The final content was thoroughly reviewed, edited, and approved by the authors, who take full responsibility for the integrity of the work.

References

- Andersen, T. G., Bollerslev, T., Diebold, F. X., & Labys, P. (2001). The distribution of realized exchange rate volatility. *Journal of the American Statistical Association*, 96(453), 42-55.
- Box, G. E., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). *Time series analysis: forecasting and control*. New York: John Wiley & Sons.
- Casolaro, A., Capone, V., Iannuzzo, G., & Camastra, F. (2023). Deep learning for time series forecasting: advances and open problems. *Information*, 14(11), 598.
- Chatfield, C., & Xing, H. (2019). *The analysis of time series: an introduction with R*. New York: Chapman and Hall/CRC.

- Chaudhuri, K., & Wu, Y. (2003). Random walk versus breaking trend in stock prices: evidence from emerging markets. *Journal of Banking & Finance*, 27(4), 575-592.
- Di Persio, L., & Honchar, O. (2016). Artificial neural networks architectures for stock price prediction: comparisons and applications. *International Journal of Circuits, Systems and Signal Processing*, 10, 403-413.
- Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2), 654-669.
- Hamilton, J. D. (2020). *Time series analysis*. New Jersey: Princeton University Press.
- Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735-1780.
- Jeribi, A., & Ghorbel, A. (2022). Forecasting developed and BRICS stock markets with cryptocurrencies and gold: generalized orthogonal generalized autoregressive conditional heteroskedasticity and generalized autoregressive score analysis. *International Journal of Emerging Markets*, 17(9), 2290-2320.
- Jiang, W. (2021). Applications of deep learning in stock market prediction: recent progress. *Expert Systems with Applications*, 184, 115537.
- Kingphai, K., & Moshfeghi, Y. (2022). On time series cross-validation for deep learning classification model of mental workload levels based on EEG signals. In *Proceeding International Conference on Machine Learning, Optimization, and Data Science*. (pp. 402-416). New York: Springer.
- Liu, T. (2025). A transformer-based stock price prediction model utilising parallel multi-scale feature fusion. In *Proceeding 5th International Conference on Computational Modeling, Simulation and Data Analysis*. (pp. 533-539).
- Nenkov, D. (2024). "The Magnificent Seven" technology stocks and their impact on the S&P 500: a review 4 years later. *Finance, Accounting and Business Analysis*, 6(2), 180-195.
- Peters, E. E. (1996). *Chaos and order in the capital markets: a new view of cycles, prices, and market volatility*. New York: John Wiley & Sons.

- Pooter, M. d., Martens, M., & Dijk, D. v. (2008). Predicting the daily covariance matrix for S&P 100 stocks using intraday data—but which frequency to use?. *Econometric Reviews*, 27(1-3), 199-229.
- Scheirer, C. J., Ray, W. S., & Hare, N. (1976). The analysis of ranked data derived from completely randomized factorial designs. *Biometrics*, 32(2), 429–434.
- Sezer, O. B., Gudelek, M. U., & Ozbayoglu, A. M. (2020). Financial time series forecasting with deep learning: a systematic literature review: 2005--2019. *Applied Soft Computing*, 90, 106181.
- Thakkar, A., & Chaudhari, K. (2021). Fusion in stock market prediction: a decade survey on the necessity, recent developments, and potential future directions. *Information Fusion*, 65, 95-107.
- Uddin, M. I., Mandyal, S., & Banger, M. (2025). A machine learning-based approach to stock market prediction: analysis and implementation. In *Proceeding 2025 International Conference on Information, Implementation, and Innovation in Technology (I2ITCON)*. (pp. 1-6). IEEE.
- Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6), 80–83.
- Yu, Y. (2025). LSTM-based time series prediction model: a case study with yfinance stock data. In *Proceeding ITM Web of Conferences*. (pp. 03015). Les Ulis: EDP Sciences.