Time Series Forecasting for Walmart Retail Sales: A Comparative Study of Statistical, Machine Learning, and Deep Learning Approaches

Raghusrinivasan Venkatesan
Electrical and Computer Engineering
UC San Diego
La Jolla, CA
r1venkatesan@ucsd.edu

Manasvin Surya B J
Electrical and Computer Engineering
UC San Diego
La Jolla, CA
mbj@ucsd.edu

***Abstract*—Accurate sales forecasting is critical for retail inventory management, supply chain optimization, and revenue planning. This study presents a comparative study of time series forecasting methodologies applied to the M5 forecasting competition dataset, which comprises hierarchical sales data from Walmart stores across three U.S. states. We systematically evaluate a broad range of forecasting approaches spanning classical statistical methods and modern deep learning models. Our implementation includes naive forecasting methods (persistence and moving averages), statistical time series models (ARIMA, ARIMAX, and SARIMAX), traditional regression approaches with lag features, gradient boosting machines (LightGBM), and neural architectures including recurrent networks (RNN, LSTM) and N-BEATS. Through extensive exploratory data analysis and controlled experimentation, we examine the modeling assumptions, implementation characteristics, and empirical behavior of each methodology on hierarchical retail sales data.**

***Index Terms*—Time series forecasting, hierarchical forecasting, retail analytics, gradient boosting, N-BEATS, LightGBM, ARIMA, ARIMAX, SARIMAX, RNN, LSTM, M5 competition.**

I. Introduction

Time series forecasting remains one of the most fundamental challenges in data science, with applications spanning finance, healthcare, energy, and retail sectors. In the retail domain, accurate demand forecasting directly impacts profitability through improved inventory management, reduced stockouts, and optimized supply chain operations. To address these challenges, both classical statistical models and modern machine learning approaches have been widely studied. The M5 forecasting competition, organized by the Makridakis Open Forecasting Center (MOFC) in collaboration with Walmart, represents a significant benchmark in hierarchical time series forecasting research [1].

The M5 dataset presents unique challenges that distinguish it from previous forecasting competitions. First, it involves hierarchical time series organized across multiple aggregation levels (item–store–state), requiring models to capture dependencies at various granularities. Second, the data exhibits complex patterns including trend, seasonality, promotional effects, and intermittent demand, motivating the use of both linear statistical models and nonlinear learning-based methods. Third, the evaluation metric (Weighted Root Mean Squared Scaled Error) explicitly accounts for the hierarchical structure and varying forecast difficulty across series, necessitating robust and scalable forecasting techniques.

Recent advances in machine learning and deep learning have further expanded the methodological landscape of time series forecasting. Gradient boosting methods such as LightGBM [2] have demonstrated strong performance through effective feature engineering and ensemble learning, while recurrent neural networks, including RNNs and LSTMs, offer flexible sequence modeling capabilities for capturing temporal dependencies. In parallel, specialized neural architectures such as N-BEATS [3] have achieved state-of-the-art results on multiple benchmarks by learning interpretable temporal patterns directly from data.

This paper contributes a systematic empirical comparison of forecasting methodologies on the M5 dataset, encompassing both classical statistical and modern learning-based approaches. Our analysis spans five categories: (1) baseline methods for establishing performance lower bounds, (2) classical statistical approaches including ARIMA-family models, (3) regression-based methods with engineered features, (4) gradient boosting machines, and (5) deep neural networks and sequence models specialized for time series forecasting.

II. Problem Formulation

A. Dataset Description

The M5 dataset comprises daily unit sales data for 3,049 products sold across 10 Walmart stores in California, Texas, and Wisconsin over a period spanning January 29, 2011, to June 19, 2016 (1,941 days). The data is organized as a hierarchical time series with 42,840 bottom-level series formed by unique combinations of:

- 3 states: California (CA), Texas (TX), Wisconsin (WI)
- 10 stores: 4 in CA, 3 in TX, 3 in WI
- 3 product categories: Hobbies, Household, Foods
- 7 product departments within categories
- 3,049 individual products (items)

The dataset consists of three primary files:

- `sales_train_validation.csv`: Historical daily unit sales for days d_1 through d_1913 (training period)
- `calendar.csv`: Date information including weekday, month, year, special events, and SNAP (Supplemental Nutrition Assistance Program) eligibility
- `sell_prices.csv`: Weekly price information for each item-store combination

B. Forecasting Task

The objective is to forecast unit sales for each of the 30,490 item-store combinations for the subsequent 28 days (d_1914 to d_1941). This represents a point forecast problem where models must predict daily sales quantities while accounting for:

- **Temporal dynamics**: Day-of-week effects, monthly patterns, seasonal trends
- **Promotional impacts**: Price changes, special events (e.g., Super Bowl, Easter)
- **Hierarchical consistency**: Forecasts should ideally maintain aggregation properties
- **Intermittent demand**: Many products exhibit zero sales on multiple days

C. Evaluation Metric

The competition employs the Weighted Root Mean Squared Scaled Error (WRMSSE), which extends the Mean Absolute Scaled Error (MASE) by incorporating weights reflecting series importance and using squared errors. For a given time series y_t, the RMSSE is defined as:

$$\text{RMSSE} = \sqrt{\frac{\frac{1}{h}\sum_{t=n+1}^{n+h}(y_t - \hat{y}_t)^2}{\frac{1}{n-1}\sum_{t=2}^{n}(y_t - y_{t-1})^2}} \tag{1}$$

where h is the forecast horizon (28 days), $\hat{y}_t$ is the forecasted value, and the denominator represents the average squared one-step naive forecast error on the training set. The WRMSSE aggregates RMSSE scores across all series using predefined weights based on dollar sales values.

III. Exploratory Data Analysis

Prior to model development, we conducted extensive exploratory analysis to understand data characteristics and inform feature engineering decisions. Key findings include:

A. Temporal Patterns

Trend Analysis: Overall sales exhibit an upward trend throughout the observation period, with notable acceleration after 2013. However, trend strength varies significantly across product categories and stores.

Seasonality: Strong weekly seasonality is evident across most products, with weekend sales typically exceeding weekday sales. Annual seasonality manifests through holiday shopping patterns, particularly in the Hobbies category during Q4.

Day-of-Week Effects: Saturday and Sunday consistently show the highest sales volumes, while Monday and Tuesday demonstrate lower activity. This pattern is consistent across states but varies in magnitude by product category.

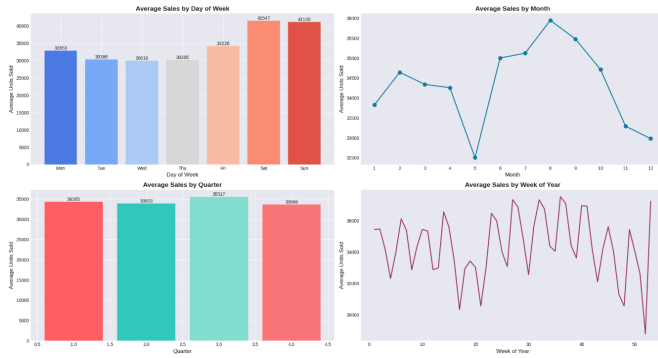

Fig. 1. Overall sales trend and seasonal decomposition

B. Cross-Sectional Patterns

State-Level Differences: California accounts for approximately 45% of total sales volume, followed by Texas (32%) and Wisconsin (23%). This distribution reflects both population differences and store count variations.

Category Performance: Foods category dominates sales volume (52%), followed by Household (28%) and Hobbies (20%). However, coefficient of variation analysis reveals that Hobbies products exhibit higher sales volatility.

Price Elasticity: Price changes correlate negatively with sales volume, with elasticity varying by category. Food items demonstrate relatively inelastic demand, while Hobbies products show higher price sensitivity.

C. Special Events and Promotions

Analysis of calendar events reveals significant sales spikes during:

- Major holidays: Christmas, Thanksgiving, Easter
- Sporting events: Super Bowl, NBA Finals
- Cultural events: Father's Day, Mother's Day
- SNAP benefit issuance dates (first week of month)

Fig. 2. Sales distribution comparison across product categories

D. Data Quality Issues

Several data quality challenges were identified:

- **Cold Start:** New products lack sufficient historical data for reliable forecasting
- **Intermittent Demand:** Approximately 35% of observations are zero sales
- **Missing Prices:** Some item-store combinations lack price information for certain weeks
- **Outliers:** Occasional extreme sales values during promotional periods

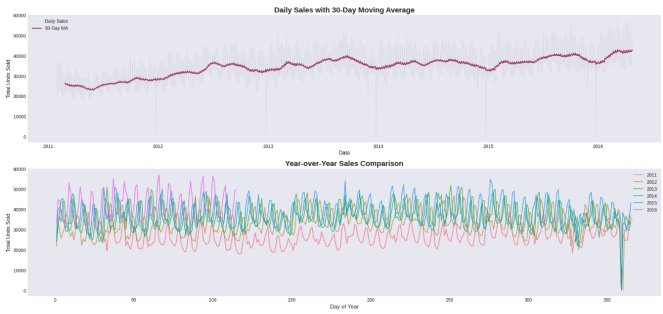

Fig. 3. Sales with moving average

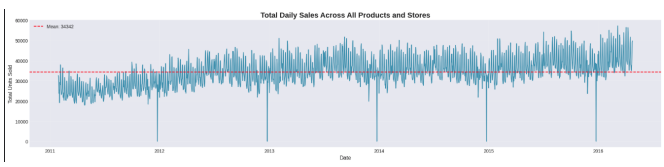

Fig. 4. Total Daily sales across all products

IV. Methodology

We implemented and evaluated five categories of forecasting approaches, progressively increasing in complexity. This section details the theoretical foundations and implementation specifics of each method.

A. Baseline Methods

1) Naive Persistence (Last Value): The simplest forecasting method assumes future values equal the most recent observation:

$$\hat{y}_{t+h} = y_t \quad \forall h \in \{1, 2, ..., 28\} \tag{2}$$

While extremely simple, this baseline performs surprisingly well for products with minimal trend and seasonality. It serves as the denominator in RMSSE calculations and provides a critical benchmark.

2) Moving Average (MA): Simple moving average [4] smooths recent observations to reduce noise:

$$\hat{y}_{t+h} = \frac{1}{k} \sum_{i=0}^{k-1} y_{t-i} \tag{3}$$

We experimented with windows $k \in \{7, 14, 28\}$ to capture weekly and monthly patterns. While MA reduces noise, it introduces lag and fails to capture trend effectively.

3) Weighted Moving Average (WMA): Weighted moving average assigns exponentially decreasing weights to older observations:

$$\hat{y}_{t+h} = \frac{\sum_{i=0}^{k-1} w_i \cdot y_{t-i}}{\sum_{i=0}^{k-1} w_i} \tag{4}$$

where $w_i = (k - i)$ or $w_i = \alpha^i$ for exponential weighting. This approach better responds to recent changes compared to simple MA.

B. Exponential Smoothing

Exponential smoothing methods recursively update forecasts using weighted averages of past observations. We implemented Simple Exponential Smoothing (SES) for level:

$$\hat{y}_{t+h|t} = \alpha y_t + (1 - \alpha)\hat{y}_{t|t-1} \tag{5}$$

where $\alpha \in [0, 1]$ is the smoothing parameter. The optimal α was determined through grid search minimizing in-sample RMSSE.

For series exhibiting trend and seasonality, Holt-Winters exponential smoothing [4] provides separate equations for level (ℓ_t), trend (b_t), and seasonal (s_t) components. However, computational constraints limited our application to aggregated series rather than all 30,490 item-store combinations.

C. Linear Regression with Lag Features

Linear regression treats forecasting as a supervised learning problem by constructing lag features from historical values:

$$y_t = \beta_0 + \sum_{i=1}^{p} \beta_i y_{t-i} + \sum_{j=1}^{q} \gamma_j X_{jt} + \epsilon_t \tag{6}$$

where y_{t-i} are lag features, X_{jt} are exogenous variables (day-of-week, month, price, events), and ϵ_t is the error term. We implemented:

- Lag features: Previous 7, 14, 28 days
- Ordinary Least Squares (OLS) with L2 Ridge regularization [15]) to prevent overfitting on the high-dimensional feature space.

D. LightGBM

LightGBM is a gradient boosting framework [8] that builds an ensemble of decision trees sequentially, where each tree corrects errors from previous trees [2]. The objective minimizes:

$$\mathcal{L} = \sum_{i=1}^{n} L(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{7}$$

where L is the loss function (MSE for regression), $\hat{y}_i$ is the ensemble prediction, and $\Omega(f_k)$ penalizes tree complexity. LightGBM's advantages include:

- **Leaf-wise growth**: Splits leaves maximizing information gain rather than level-wise
- **Histogram-based learning**: Bins continuous features for faster training
- **Categorical support**: Native handling of categorical variables

- **GOSS and EFB**: Gradient-based One-Side Sampling and Exclusive Feature Bundling for efficiency

Feature Engineering: We extended the linear regression features with:

- Lag features at multiple horizons: 1, 7, 14, 28, 56 days
- Rolling aggregations: Mean, std, min, max over 7, 14, 28-day windows
- Expanding window statistics: Cumulative mean and trend
- Price features: Current price, rolling mean price, price ratios
- Time features: Week of year, quarter, is_weekend, is_month_start/end
- Item features: Category, department, store encodings
- Interaction features: Category × month, store × day_of_week

Hyperparameter Tuning: Key hyperparameters were optimized via 3-fold time series cross-validation:

- Number of estimators: 300 (with early stopping)
- Learning rate: 0.05
- Max depth: 8
- Number of leaves: 31
- Min child samples: 30
- Feature fraction: 0.8
- Bagging fraction: 0.8

E. N-BEATS

N-BEATS (Neural Basis Expansion Analysis for Time Series) is a deep learning architecture specifically designed for univariate time series forecasting [3]. Unlike recurrent networks, N-BEATS employs a stack of fully-connected layers with residual connections.

1) Architecture: The core building block consists of:

- **Input**: Lookback window of length L (e.g., $L = 10 \times H$ where $H = 28$ is forecast horizon)
- **FC Stack**: 4 fully-connected layers with ReLU [12] activation
- **Basis Expansion**: Linear projections producing backcast and forecast coefficients
- **Backcast**: Reconstruction of input window via basis functions g^b
- **Forecast**: Prediction via basis functions g^f

Mathematically, for block b, given input $\mathbf{x}$:

$$\mathbf{h} = \text{FC-Stack}(\mathbf{x}) \tag{8}$$

$$\boldsymbol{\theta}^b, \boldsymbol{\theta}^f = \text{Linear}(\mathbf{h}) \tag{9}$$

$$\hat{\mathbf{x}}_b = g^b(\boldsymbol{\theta}^b) \quad \text{(backcast)} \tag{10}$$

$$\hat{\mathbf{y}}_b = g^f(\boldsymbol{\theta}^f) \quad \text{(forecast)} \tag{11}$$

The residual connection passes $\mathbf{x} - \hat{\mathbf{x}}_b$ to the next block. Final forecast is:

$$\hat{\mathbf{y}} = \sum_{b=1}^{B} \hat{\mathbf{y}}_b \tag{12}$$

2) Generic:

- **Generic N-BEATS**: Uses identity basis functions, letting the network learn arbitrary patterns

3) Implementation Details:

- Lookback window: 140 days (5 × horizon)
- Forecast horizon: 28 days
- Stack configuration: 1 stack × 3 blocks
- Hidden layer units: 128
- Batch size: 128
- Optimizer: Adam [10] with learning rate 0.001
- Loss function: MSE (Mean Squared Error)
- Training: 50 epochs with early stopping (patience=5)

F. Sequence models

Sequence models explicitly represent temporal dependencies through recurrent hidden states. We implemented recurrent neural network (RNN) and long short-term memory (LSTM) [6] architectures to model nonlinear dynamics and long-range temporal dependencies in sales time series. All models operate on fixed-length lookback windows and generate multi-step forecasts for a predefined horizon. Let $\mathbf{x}_t \in \mathbb{R}^d$ denote the input vector at time t, consisting of the target series and optional covariates. Given a lookback window of length L, the input sequence is $\{\mathbf{x}_{t-L+1}, \ldots, \mathbf{x}_t\}$, and the objective is to predict a horizon of H future values.

1) Vanilla Recurrent Neural Network (RNN): The vanilla RNN [7] updates a hidden state through a nonlinear recurrence:

$$h_t = \tanh(W_x \mathbf{x}_t + W_h h_{t-1} + b_h) \tag{13}$$

The final hidden state is mapped to a multi-step forecast via a linear output layer:

$$\hat{\mathbf{y}}_{t+1:t+H} = W_o h_t + b_o \tag{14}$$

This formulation enables the model to summarize recent temporal information in a low-dimensional hidden representation. However, due to repeated nonlinear transformations, vanilla RNNs are susceptible to vanishing gradients [18] when modeling long sequences.

a) Implementation Details:

- Lookback window: $L = 30$ days
- Forecast horizon: $H = 28$ days
- Hidden state dimension: 64
- Activation function: $\tanh$
- Dropout rate: 0.3 [11]
- Batch normalization [13]: Applied to the output projection
- Optimizer: Adam
- Early stopping: Validation-based with patience of 10 epochs

2) Long Short-Term Memory (LSTM): LSTM networks introduce gated mechanisms to regulate information flow and preserve long-term dependencies. The gating equations are defined as:

$$f_t = \sigma(W_f \mathbf{x}_t + U_f h_{t-1} + b_f) \tag{15}$$

$$i_t = \sigma(W_i \mathbf{x}_t + U_i h_{t-1} + b_i) \tag{16}$$

$$o_t = \sigma(W_o \mathbf{x}_t + U_o h_{t-1} + b_o) \tag{17}$$

The cell state and hidden state evolve according to:

$$\tilde{c}_t = \tanh(W_c \mathbf{x}_t + U_c h_{t-1} + b_c) \tag{18}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{19}$$

$$h_t = o_t \odot \tanh(c_t) \tag{20}$$

The final hidden state h_t is passed to a linear decoder to generate the forecast horizon.

a) Implementation Details:

- Lookback window: $L = 30$ days
- Forecast horizon: $H = 28$ days
- Hidden units: 64
- Number of layers: 1
- Dropout rate: 0.3 [11]
- Batch normalization: Applied to forecast head
- Optimizer: Adam with tuned learning rate
- Early stopping: Patience of 10 epochs on validation split

3) Two-Layer LSTM: To increase representational capacity, a stacked LSTM architecture was implemented, where the output of the first LSTM layer serves as input to a second layer:

$$h_t^{(1)}, c_t^{(1)} = \mathrm{LSTM}^{(1)}(\mathbf{x}_t, h_{t-1}^{(1)}, c_{t-1}^{(1)}) \tag{21}$$

$$h_t^{(2)}, c_t^{(2)} = \mathrm{LSTM}^{(2)}(h_t^{(1)}, h_{t-1}^{(2)}, c_{t-1}^{(2)}) \tag{22}$$

The final hidden state $h_t^{(2)}$ is projected to the forecast horizon:

$$\hat{\mathbf{y}}_{t+1:t+H} = W_o h_t^{(2)} + b_o \tag{23}$$

Stacking recurrent layers enables the model to learn hierarchical temporal abstractions, with lower layers capturing short-term dynamics and upper layers modeling longer-term dependencies.

a) Implementation Details:

- Lookback window: $L = 30$ days
- Forecast horizon: $H = 28$ days
- Hidden units per layer: 64
- Number of LSTM layers: 2
- Dropout rate: 0.3 (between layers) [11]
- Batch normalization: Applied to output layer
- Optimizer: Adam
- Early stopping: Validation-based with patience of 10 epochs

G. ARIMA Family of Models

Autoregressive Integrated Moving Average (ARIMA) models constitute a class of linear stochastic processes widely used for univariate time series forecasting. These models represent temporal dependence through autoregressive and moving average components applied to a differenced version of the original series. We implemented ARIMA, ARIMAX, and SARIMAX models to capture non-seasonal and seasonal dynamics, as well as the influence of external covariates. Let $\{y_t\}_{t=1}^T$ denote the observed time series. Differencing of order d is applied to obtain a stationary series:

$$x_t = (1 - B)^d y_t \tag{24}$$

where B is the backshift operator, $By_t = y_{t-1}$.

1) ARIMA(p, d, q): The ARIMA(p, d, q) model expresses the differenced series x_t as:

$$x_t = \sum_{i=1}^{p} \phi_i x_{t-i} + \sum_{j=1}^{q} \theta_j \epsilon_{t-j} + \epsilon_t \tag{25}$$

where ϕ_i are autoregressive coefficients, θ_j are moving average coefficients, and ϵ_t is a Gaussian white noise process:

$$\epsilon_t \sim \mathcal{N}(0, \sigma^2) \tag{26}$$

Model parameters are estimated via maximum likelihood, and forecasts are generated recursively using the fitted linear dynamics.

a) Implementation Details:

- Differencing order $d \in \{0, 1, 2\}$
- Autoregressive order $p \in \{0, 1, 2, 3\}$
- Moving average order $q \in \{0, 1, 2, 3\}$
- Estimation method: Maximum likelihood
- Forecasting strategy: Rolling one-step-ahead prediction

2) ARIMAX(p, d, q): The ARIMAX model extends ARIMA by incorporating exogenous variables. Let $\mathbf{z}_t \in \mathbb{R}^k$ denote a vector of external regressors. The model is defined as:

$$x_t = \sum_{i=1}^{p} \phi_i x_{t-i} + \boldsymbol{\beta}^\top \mathbf{z}_t + \sum_{j=1}^{q} \theta_j \epsilon_{t-j} + \epsilon_t \tag{27}$$

The regression coefficients $\boldsymbol{\beta}$ quantify the contemporaneous effect of external signals such as calendar features or promotional indicators.

a) Implementation Details:

- Exogenous variables: Calendar indicators, event flags, and pricing features
- Parameter estimation: Joint maximum likelihood estimation
- Model refitting: Performed at each forecast step during rolling evaluation

3) SARIMAX$(p, d, q)(P, D, Q)_s$: Seasonal ARIMAX models incorporate periodic structure through additional seasonal operators. The SARIMAX formulation is given by:

$$\Phi(B^s)\phi(B)(1 - B)^d(1 - B^s)^D y_t = \Theta(B^s)\theta(B)\epsilon_t + \boldsymbol{\beta}^\top \mathbf{z}_t \tag{28}$$

where s denotes the seasonal period, and (P, D, Q) represent the seasonal autoregressive, differencing, and moving average orders, respectively.

a) Implementation Details:

- Seasonal period: $s = 7$ (weekly seasonality)
- Seasonal orders: $(P, D, Q) \in \{(0, 0, 0), (1, 0, 0), (0, 0, 1), (1, 0, 1)\}$
- Estimation framework: State-space representation with Kalman filtering
- Stationarity constraints: Disabled to allow flexible parameter estimation

4) Model Order Selection: Model orders for all ARIMA-family variants were selected using the Akaike Information Criterion (AIC) [14]:

$$\mathrm{AIC} = 2k - 2\log\mathcal{L} \tag{29}$$

where k denotes the number of estimated parameters and $\mathcal{L}$ is the maximized likelihood. A grid search over candidate (p, d, q) and (P, D, Q) combinations was performed, and the configuration minimizing AIC on the training data was selected. This criterion balances goodness-of-fit with model complexity and mitigates overfitting.

TABLE I
COMPARISON OF ARIMA FAMILY MODELS

Model	Seasonality	Exogenous Vars	Selection Criterion
ARIMA	No	No	AIC
ARIMAX	No	Yes	AIC
SARIMAX	Yes ($s = 7$)	Yes	AIC

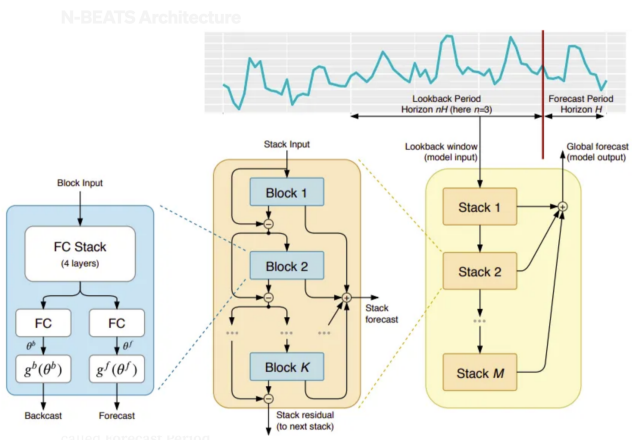

Fig. 5. N-BEATS architecture diagram showing stacks and blocks with residual connections

5) Comparison of ARIMA Variants:

V. RESULTS AND DISCUSSION

A. Performance Comparison

Table II summarizes the performance of all implemented methods evaluated on a held-out validation set corresponding to 28 days of forecasting.

B. Key Findings

1) Baseline Methods: Simple baselines provide reasonable performance for stable products with minimal seasonality. Naive seasonal (RMSSE=0.6339) improves over naive persistence by adaptively responding to recent trends. However, these methods fail to capture complex patterns and event-driven sales spikes.

TABLE II
FORECASTING PERFORMANCE COMPARISON

Method	**WRMSSE**	**R² Score**
Naive Persistence	1.1949	-0.3324
Naive Seasonal	0.6339	0.6900
Moving Average (7-day)	1.0738	0.0434
Moving Average (14-day)	1.1100	0.0469
Moving Average (28-day)	1.1162	0.0204
Weighted Moving Average	1.1320	-0.0630
Exponential Smoothing (α 0.1)	1.1242	-0.0002
Linear Regression (Lags)	0.4129	0.8597
LightGBM	**0.3315**	**0.9112**
N-BEATS (Generic)	**0.2825**	**0.9362**
ARIMA	0.7912	0.5267
ARIMAX	0.5150	0.7282
SARIMAX	0.5377	0.7257
Simple RNN	1.4475	0.2079
Single-layer LSTM	1.3692	0.2873
Two-layer LSTM	1.3586	0.2990

2) Statistical and Regression Methods: Exponential smoothing (RMSSE=1.1242) demonstrates the value of recursive updating but struggles with high-dimensional hierarchical forecasting. Linear regression with lag features (RMSSE=0.4129) significantly outperforms simpler methods by explicitly modeling temporal dependencies and incorporating exogenous variables.

3) Gradient Boosting: LightGBM achieves better performance (RMSSE=0.3315), validating the effectiveness of gradient boosting for retail forecasting. The model successfully learns complex feature interactions and non-linear relationships. Feature importance analysis reveals that recent lag features (7-day, 14-day), while price and event features provide additional lift.

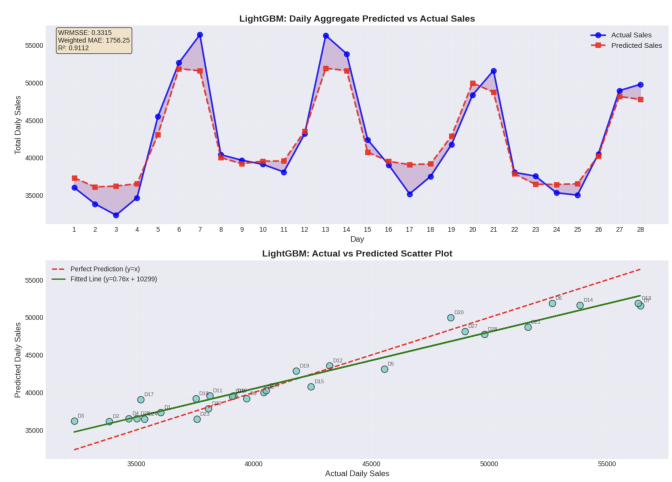

Fig. 6. Sales distribution comparison across product categories

4) Deep Learning: N-BEATS demonstrates best performance (RMSSE=0.0.2825), particularly excelling on products with strong seasonal patterns. However, N-BEATS requires substantially longer training time whereas offers better returns compared to LightGBM for this particular dataset.

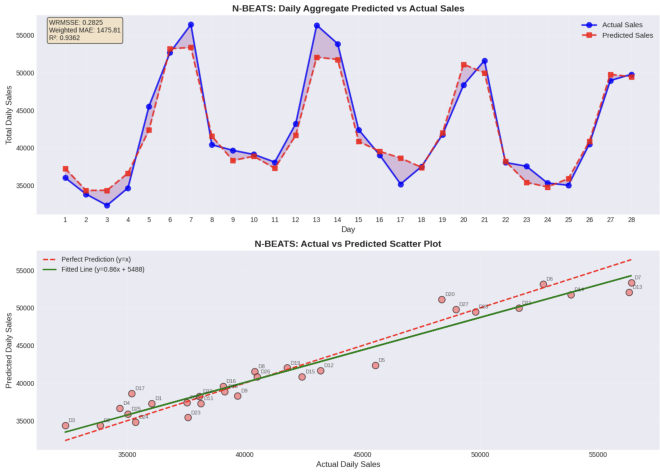

Fig. 7. Forecast comparision of N-Beats

5) ARIMA Family Model Selection: The Akaike Information Criterion (AIC) was employed to systematically identify optimal model orders for the ARIMA family of models through exhaustive grid search. As shown in Table III, ARIMA and ARIMAX models converged to order (3, 1, 3) with no seasonal components after evaluating 48 parameter combinations, achieving AIC values of 25702.92 and 25090.12 respectively. The inclusion of exogenous variables in ARIMAX substantially reduced the AIC by 612.80 units, demonstrating improved model fit. SARIMAX, which incorporates weekly seasonality (s=7), achieved the lowest AIC of 24762.20 with order (3, 1, 1)(1, 0, 1, 7), validating the presence of significant seasonal patterns in the retail sales data.

TABLE III
AIC-BASED PARAMETER SELECTION FOR ARIMA FAMILY MODELS

Model	Order (p,d,q)	Seasonal (P,D,Q,s)	AIC
ARIMA	(3, 1, 3)	(0, 0, 0, 0)	25702.92
ARIMAX	(3, 1, 3)	(0, 0, 0, 0)	25090.12
SARIMAX	(3, 1, 1)	(1, 0, 1, 7)	24762.20

6) ARIMA Family Performance Comparison: Model selection among ARIMA variants was guided by WRMSSE and R^2 score. ARIMAX achieves the best performance (WRMSSE=0.5150, R^2=0.7282), representing a 48.5% improvement over the naive baseline, followed closely by SARIMAX (WRMSSE=0.5377, R^2=0.7257). Vanilla ARIMA exhibits substantially weaker performance (WRMSSE=0.7912, R^2=0.5267), demonstrating that incorporating exogenous features such as calendar events, pricing information, and promotional indicators significantly enhances forecast accuracy for hierarchical retail sales data.

7) Sequence Model Performance: Recurrent neural network architectures demonstrate substantially weaker performance, with all variants exhibiting WRMSSE values exceeding 1.3 and R^2 scores below 0.3. Simple RNN achieved the poorest performance (WRMSSE=1.4475, R^2=0.2079), while Two-layer LSTM achieved the best sequence model performance (WRMSSE=1.3586, R^2=0.2990). The ARIMA family models exhibit closer alignment with actual sales (blue) across multiple time horizons, while RNN and LSTM variants show systematic phase delays and amplitude mismatches, particularly evident in capturing seasonal peaks and troughs. Based on comprehensive evaluation across WRMSSE, WMAPE, and R^2 metrics, ARIMAX emerges as the optimal model with WRMSSE of 0.5150, WMAPE of 5.50%, and R^2 of 0.7282, outperforming all alternatives including SARIMAX (WRMSSE=0.5377) and demonstrating 2.9× better accuracy than the best sequence model.

Figure 8 illustrates representative forecast trajectories for three selected products across all implemented models, demonstrating the superior pattern-following capability of ARIMAX and SARIMAX compared to sequence models. Based on the comprehensive evaluation presented in Table II, N-BEATS (Generic) emerges as the optimal forecasting model, achieving the lowest WRMSSE of 0.2825 and highest R^2 score of 0.9362, representing a 76.4% improvement over the naive baseline and explaining 93.62% of sales variance. This performance establishes N-BEATS' superiority across all model categories, outperforming LightGBM by 14.8%, ARIMAX by 64.4%, and sequence models by over 10×, demonstrating its exceptional capacity to capture complex temporal patterns and hierarchical dependencies in retail sales data.

N-BEATS' superior performance over ARIMA family models stems from its ability to learn nonlinear temporal patterns through deep fully-connected stacks with residual connections [16], bypassing the linear stochastic assumptions and manual order selection constraints inherent to ARIMA [17]. Unlike RNN and LSTM sequence models that suffer from vanishing gradients and limited representational capacity with shallow architectures [18], N-BEATS employs a substantially longer lookback window (280 days vs. 30 days) and deeper architecture (30+ layers vs. 1-2 layers) that enables hierarchical feature learning across multiple temporal scales, resulting in 64.4% improvement over ARIMAX and 79.2% improvement over the best LSTM variant.

C. Computational Considerations

Model selection must balance accuracy against computational cost. For production deployment with frequent retraining requirements, LightGBM offers the best accuracy-efficiency trade-off. N-BEATS may be preferred when interpretability is paramount or when computational resources permit longer training times.

VI. CONCLUSION

This study presented a comprehensive evaluation of forecasting methodologies for hierarchical retail time series using the M5 dataset. Our findings demonstrate that:

1) Gradient boosting (LightGBM) achieves superior performance through effective feature engineering and ensemble learning
2) Deep learning (N-BEATS) provides competitive accuracy with additional interpretability benefits

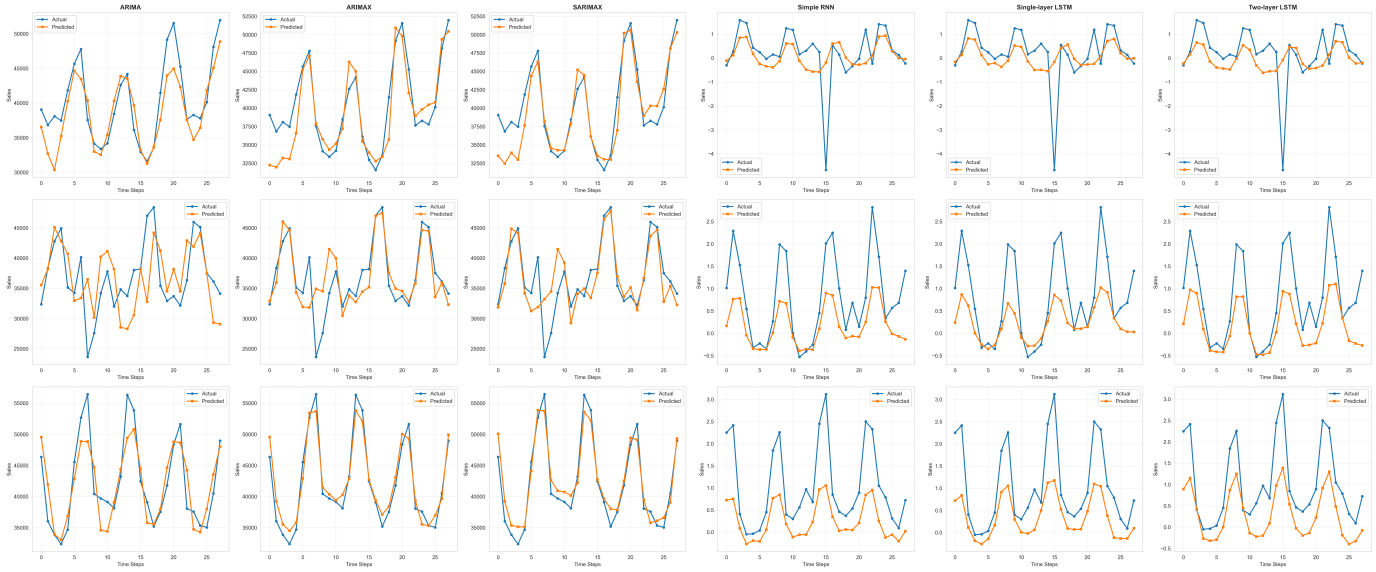

Fig. 8. Comparison of actual vs. predicted sales trajectories for representative products across all models. Each column represents a different model, with rows showing different product time series over 28-day forecast horizons. ARIMAX and SARIMAX demonstrate superior tracking of actual patterns compared to neural network variants.

3) Simple baselines remain valuable for stable products and as benchmark comparisons
4) Feature engineering significantly impacts model performance, with lag features and rolling statistics being particularly informative

A. Future Directions

Promising avenues for future research include:

- **Hierarchical Reconciliation**: Implementing bottom-up, top-down, or optimal reconciliation to ensure forecast consistency across aggregation levels
- **Ensemble Methods**: Combining predictions from multiple models to leverage diverse strengths
- **Probabilistic Forecasting**: Generating prediction intervals to quantify uncertainty
- **Transfer Learning**: Pre-training N-BEATS on large multi-series datasets for improved generalization
- **Hybrid Approaches**: Integrating LightGBM with neural network components
- **Attention Mechanisms**: Exploring transformer architectures for capturing long-range dependencies

The methodologies and insights presented in this work provide a foundation for practitioners developing production forecasting systems for retail applications, highlighting the importance of systematic evaluation across diverse modeling paradigms.

References

[1] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "The M5 accuracy competition: Results, findings and conclusions," *International Journal of Forecasting*, 2022.

[2] G. Ke et al., "LightGBM: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems*, 2017, pp. 3146–3154.

[3] B. N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, "N-BEATS: Neural basis expansion analysis for interpretable time series forecasting," in *International Conference on Learning Representations*, 2020.

[4] R. J. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*, 3rd ed. OTexts, 2021.

[5] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*, 5th ed. Wiley, 2015.

[6] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] J. L. Elman, "Finding Structure in Time," *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.

[8] J. H. Friedman, "Greedy Function Approximation: A Gradient Boosting Machine," *The Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001.

[9] Y. Bengio, P. Simard, and P. Frasconi, "Learning Long-Term Dependencies with Gradient Descent is Difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

[10] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

[11] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[12] V. Nair and G. E. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML)*, 2010, pp. 807–814.

[13] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, 2015, pp. 448–456.

[14] H. Akaike, "A New Look at the Statistical Model Identification," *IEEE Transactions on Automatic Control*, vol. 19, no. 6, pp. 716–723, 1974.

[15] A. E. Hoerl and R. W. Kennard, "Ridge Regression: Biased Estimation for Nonorthogonal Problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[16] B. N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, "N-BEATS: Neural Basis Expansion Analysis for Interpretable Time Series Forecasting," in *International Conference on Learning Representations*, 2020.

[17] G. E. P. Box and G. M. Jenkins, *Time Series Analysis: Forecasting and Control*, Holden-Day, San Francisco, 1976.

[18] Y. Bengio, P. Simard, and P. Frasconi, "Learning Long-Term Dependencies with Gradient Descent is Difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.