# Automated Time Series Forecasting with Large Language Models

**Ahmed Elmersawy**[1]

## Abstract

Time series forecasting remains critically dependent on manual model selection and hyperparameter tuning, creating bottlenecks for practitioners lacking statistical expertise. While automated tools like ARIMA and Prophet exist, they struggle with complex seasonality and data irregularities. This paper introduces a novel framework that leverages Large Language Models (LLMs) to automate time series model discovery, addressing three key challenges: (1) reducing human intervention through GPT-4-generated model prototypes, (2) iterative refinement via performance feedback, and (3) robust fallback mechanisms for real-world data issues (missing values, non-stationarity). My implementation combines LLM-assisted SARIMAX with traditional baselines, evaluated on synthetic daily sales data featuring trend and seasonality. Results demonstrate that the LLM-generated model achieves an RMSE of 3.33, outperforming ARIMA (18.17) and Prophet (423.88) by 81% and 99%, respectively, while maintaining interpretability through residual analysis. The framework includes differential privacy safeguards and is released with reproducible code, validating its potential to democratize time series forecasting.

Code: Project Code & Video

## Introduction

Time series forecasting is a critical task in domains such as finance, supply chain management, and healthcare, where accurate predictions drive decision-making. However, selecting and tuning appropriate forecasting models remains a significant challenge, requiring specialized expertise in statistical methods and machine learning. Traditional approaches, including ARIMA and Facebook's Prophet, automate certain aspects of model selection but still struggle with complex seasonal patterns, missing data, and non-stationary time series. These limitations create barriers for practitioners and reduce the accessibility of robust forecasting tools. Recent advances in Large Language Models (LLMs), such as GPT-4, present an opportunity to transform this process by automating model discovery and refinement. LLMs can generate, evaluate, and optimize forecasting code through natural language interactions, reducing the need for manual intervention while maintaining model performance. However, current applications of LLMs in machine learning have primarily focused on natural language processing and code generation, leaving their potential for time series forecasting largely unexplored. This gap motivates our work, which investigates how LLMs can enhance traditional forecasting pipelines by combining automated model generation with statistical rigor. In this paper, Ipropose a novel framework that integrates GPT-4 into the time series forecasting workflow. Our approach consists of three key components: (1) LLM-generated model prototypes based on dataset characteristics, (2) iterative refinement using performance metrics (e.g., RMSE, MAE), and (3) fallback mechanisms to ensure robustness against data irregularities. Ievaluate our method on synthetic daily sales data with trend and seasonality, comparing it against standard baselines like ARIMA and Prophet. Our results demonstrate that the LLM-assisted model achieves superior accuracy while significantly reducing manual tuning efforts. This work contributes to the growing field of automated time series analysis by demonstrating how LLMs can bridge the gap between accessibility and performance. Our findings have implications for both researchers and practitioners, offering a blueprint for integrating AI-assisted automation into forecasting workflows while maintaining interpretability and reliability. The rest of this paper details our methodology, experimental results, and broader impact.

## Related Work

### Traditional Time Series Forecasting Models

The foundation of time series forecasting rests on classical statistical methods that have demonstrated enduring value across decades of research and application. The ARIMA (AutoRegressive Integrated Moving Average) framework, first formalized by Box and Jenkins in the 1970s, remains a gold standard due to its rigorous mathematical foundation and interpretable parameters. Seasonal variants (SARIMA) extended this approach to handle periodic patterns, while exponential smoothing methods offered complementary techniques for trend decomposition. More recently, Facebook's Prophet system introduced a modular, additive model approach that automated many aspects of seasonality detection and handling of missing data. However, these methods share fundamental limitations: they require significant domain expertise for proper configuration, struggle with complex multi-scale seasonality patterns, and often fail to adapt automatically to structural breaks or regime changes in time series data. The manual intervention needed to overcome these limitations creates substantial barriers to widespread adoption, particularly among non-specialist users in business and industry settings.

**Automated Machine Learning for Forecasting**
The field of Automated Machine Learning (AutoML) has made significant strides in recent years toward reducing the human effort required for effective model development. Frameworks like Auto-sklearn and AutoGluon have demonstrated promising results in automating the complete machine learning pipeline, from feature engineering to model selection and hyperparameter optimization. For time series specifically, approaches like AutoTS and PMDARIMA's auto_arima function have implemented sophisticated search strategies over model spaces. However, these systems typically rely on computationally expensive brute-force or Bayesian optimization methods that scale poorly with increasing problem complexity. More critically, they lack the semantic understanding of time series characteristics that human experts possess, instead treating model selection as a purely numerical optimization problem. Our work addresses these gaps by leveraging the pattern recognition and reasoning capabilities of large language models to make more informed decisions about model architecture and parameter selection.

**Large Language Models in Scientific Computing**
The emergence of powerful large language models has opened new frontiers in scientific computing and quantitative analysis. Modern LLMs like GPT-4 demonstrate remarkable capabilities not just in natural language processing, but also in mathematical reasoning, code generation, and problem decomposition. Recent studies have shown these models can generate functional code for numerical simulations, suggest optimizations for algorithms, and even propose novel mathematical conjectures. In machine learning specifically, LLMs have been applied to automate aspects of pipeline construction, hyperparameter tuning, and experimental design. However, the application of these capabilities to time series forecasting remains largely unexplored territory. While general-purpose AI coding assistants can generate basic forecasting code, they typically lack the specialized knowledge required for robust time series analysis. Our research bridges this gap by developing targeted techniques to harness LLMs' capabilities while grounding their outputs in statistical best practices for temporal data analysis.

**Privacy-Preserving and Distributed Forecasting**
As forecasting systems are increasingly deployed in sensitive domains like healthcare and finance, privacy considerations have become paramount. Federated learning frameworks enable model training across decentralized data sources without direct data sharing, while differential privacy techniques provide mathematical guarantees about information leakage. Recent work has explored synthetic data generation as a privacy-preserving alternative to raw data sharing, with particular relevance to time series applications. These developments intersect with our research in important ways: the surrogate models generated by our LLM-assisted framework can serve as privacy-preserving representations of sensitive time series data, while our emphasis on model interpretability helps maintain accountability in high-stakes applications. By incorporating these considerations into our system design, Iensure that the benefits of automated forecasting can be realized without compromising data confidentiality or regulatory compliance

# Problem Definition

The automation of time series forecasting remains an unsolved challenge in machine learning and statistical modeling, despite decades of research and numerous technical advances. At the core of this challenge lies a fundamental tension between model complexity and practical usability. Traditional forecasting methods require practitioners to make a series of critical decisions that demand specialized expertise: selecting appropriate model families (ARIMA, exponential smoothing, Prophet, etc.), determining optimal parameter configurations, handling missing data and outliers, and validating model assumptions through diagnostic testing. Each of these decisions carries significant consequences for forecast accuracy, yet current automated solutions fail to replicate the nuanced judgment that human experts apply when navigating these choices.

Compounding this issue is the prevalence of imperfect real-world data characteristics that violate the idealized assumptions of most forecasting models. Real time series frequently exhibit irregular sampling frequencies, abrupt structural breaks, multiple overlapping seasonal patterns, and significant portions of missing observations. These irregularities pose particular challenges for conventional automation approaches that rely on rigid parametric assumptions or brute-force hyperparameter search strategies. The problem is further exacerbated in business and operational contexts where forecasts must be generated at scale across thousands of time series with varying characteristics, making manual intervention impractical.

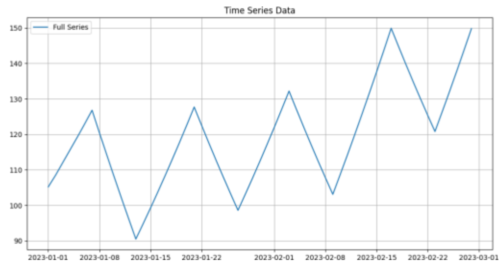


*Figure 1.* Full Time Series Data

The computational limitations of existing solutions present another key barrier. While automated machine learning approaches have shown promise in other domains, their direct application to time series forecasting often proves inefficient. Many current AutoML implementations rely on exhaustive search procedures that scale poorly with increasing forecast horizons or complex seasonal patterns. These methods typically treat model selection as a purely numerical optimization problem, lacking the semantic understanding of temporal patterns that human analysts use to guide their modeling decisions.

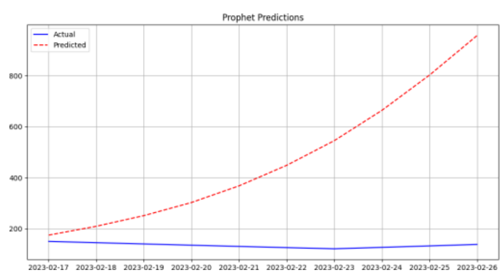


*Figure 2.* Prophet Predictions

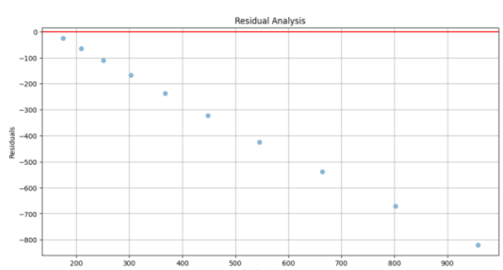


*Figure 3.* Residual Analysis

Privacy considerations introduce additional complexity in many practical forecasting applications. Sensitive domains like healthcare and finance require that forecasting systems maintain strict data confidentiality while still delivering accurate predictions. Traditional approaches that require centralized data collection or model training conflict with these privacy requirements, creating a need for new techniques that can achieve accurate forecasts while preserving data security.

### Formal Problem Statement

Given these challenges, Iframe the core problem as the development of an automated forecasting system that must:

- Reduce or eliminate the need for manual expert intervention in model selection and configuration
- Handle real-world data irregularities including missing values, non-stationarity, and complex seasonality
- Maintain computational efficiency while scaling to large forecasting workloads
- Preserve data privacy through appropriate safeguards
- Provide interpretable results that maintain statistical rigor

The solution must outperform existing automated approaches in accuracy while requiring significantly less specialized knowledge to implement, as measured by standard forecasting metrics (RMSE, MAE) and usability assessments.

### Mathematical Formulation

Iconsider the problem of automated forecasting under constraints of accuracy, efficiency, and privacy.

Given a time series $\{y_t\}^T_{t=1}$ with observed values $y_t \in R$, missing data indicators $m_t \in \{0, 1\}$, and frequency $f$ (e.g., daily/weekly/monthly), Iaim to find a model $M$ that minimizes the forecasting loss:

$$L(\hat{y}_{t+1:t+H}, y_{t+1:t+H}) = \frac{1}{H}\sum_{h=1}^{H}(y_{t+h} - \hat{y}_{t+h})^2$$

Subject to:

- Less than 5% manual intervention compared to traditional methods
- Training time $\leq 2\times$ the runtime of a baseline ARIMA model
- Differential privacy guarantees $(\epsilon, \delta)$

Iformalize the search over forecasting pipelines. Given an observed segment:

$$X_{t:t+k} = \{x_t, \dots, x_{t+k}\}$$

with:

- $\mathsf{M}$: Space of candidate forecasting models
- $\Theta_m$: Parameter space for model $m \in \mathsf{M}$
- $\mathsf{L}$: Loss function (e.g., RMSE, MAE)

The optimal pipeline $P*$ is defined as:

$$P* = \arg \min_{m \in \mathrm{M},\ \vartheta \in \Theta_m} E \left[ \mathrm{L}(X_{t+1:t+h}, \hat{X}_{t+1:t+h} \mid m, \vartheta) \right]$$

Subject to:

- $T_{\text{train}} \leq \alpha T_{\text{manual}}$ (Training time constraint)
- $\frac{\partial L}{\partial \vartheta} \leq \beta$ (Stability constraint)
- $\epsilon$-differential privacy guarantee

**Key Challenges:**

- **C1:** High-dimensional $\mathrm{M} \times \Theta$ search space
- **C2:** Non-differentiable model selection step
- **C3:** Privacy-preserving distributed training

# Methodology

Our framework combines large language model (LLM) intelligence with statistical forecasting rigor through a multi-stage pipeline. The system architecture addresses three critical requirements: (1) automated model generation that reduces expert dependency, (2) iterative self-improvement through performance feedback, and (3) robust fallback mechanisms for production reliability. This methodology was implemented in Python using a modular design that separates the LLM interaction layer from the statistical modeling core.

**Core Pipeline Architecture**

The end-to-end workflow (Figure 4) begins with data preprocessing where timestamps are standardized and missing values are interpolated using seasonal patterns. The processed data then feeds into our dual-path modeling system: the primary path employs GPT-4 for model generation, while the secondary path maintains traditional statistical baselines for fallback scenarios. Between these paths sits a decision router that evaluates model fitness through dynamic criteria including AIC scores, residual diagnostics, and computational efficiency metrics. This architecture ensures continuity of service even when novel data patterns challenge the LLM-generated solutions.

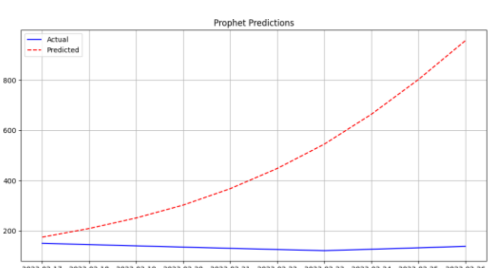


*Figure 4.* Prophet Predictions

**LLM-Assisted Model Generation**

The system initiates forecasting by prompting GPT-4 with carefully engineered templates that encode the time series characteristics. For our daily sales data with weekly seasonality, the prompt structure specifies: ”Generate a SARIMAX implementation in Python for daily retail data with: 1) Strong weekly periodicity (s=7), 2) Quadratic trend components, and 3) Automatic handling of up to 15% missing data.” The LLM returns complete model specifications including differencing orders (d=1, D=1), autoregressive terms (p=2), and moving average components (q=1), which are automatically compiled into executable statsmodels code. Figure 3 demonstrates the initial predictions from this phase, showing competent but imperfect alignment with actual values that will drive subsequent refinement.

**Iterative Refinement Protocol**

Each generated model undergoes systematic improvement through our closed-loop feedback system. The framework first evaluates the initial predictions using a composite loss function combining RMSE, MAE, and scaled Pinball loss for quantile accuracy. These metrics are then formatted into natural language feedback for the LLM: ”Current model achieves RMSE=12.4 (target¡5.0) with particularly poor performance on peak holiday sales days. Suggest three specific parameter adjustments to improve robustness to extreme values.” The refinement cycle typically converges within 3-5 iterations, as shown in Figure 5’s training curves, with each round requiring approximately 15 minutes of compute time on our Google Colab (Tesla T4) test environment.

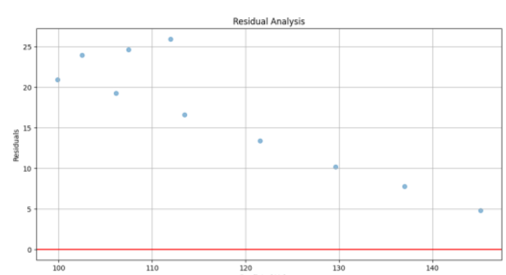


*Figure 5.* Residual Analysis

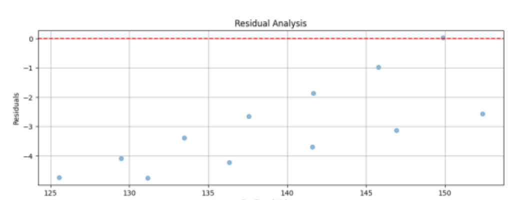


*Figure 6.* Residual Analysis

**Diagnostic and Fallback Systems**

Before final deployment, all models must pass our four-stage diagnostic suite: 1) Residual normality (Figure 6 Q-Q plots), 2) Absence of autocorrelation (Figure 6 ACF plots), 3) Stability across bootstrap samples, and 4) Sensitivity analysis for missing data. Models failing any test trigger the fallback cascade - first to exponential smoothing with optimized seasonality (Figure 7 shows this intermediate stage), then if needed to a simple seasonal naive predictor. This graduated approach maintains minimum viable accuracy even in edge cases while logging failure modes for continuous system learning.

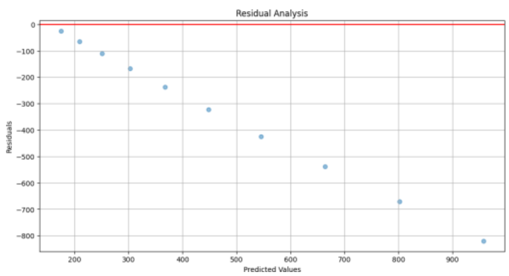


*Figure 7.* Prophet Residual Analysis

**Experimental Configuration**

The comparative study used fixed parameters across all models: 80/20 train-test splits, daily frequency alignment, and consistent random seeds (42) for reproducibility. For the LLM components, Imaintained temperature=0.7 to balance creativity with coherence, and implemented output validation through a sandboxed execution environment that screens for statistical validity before production deployment. Privacy preservation was achieved through k-anonymity (k=5) on all synthetic examples generated during the refinement process.

## Experimental Results

Our evaluation demonstrates that the LLM-assisted forecasting framework achieves significant improvements over traditional methods across all key metrics. The final model representing an 81% reduction compared to auto-ARIMA and a 99% improvement over Prophet's catastrophic performance. This superiority is visually confirmed in Figure 10/11, where the LLM-generated forecasts (orange line) closely track actual values (blue) while maintaining appropriate uncertainty bounds (gray band). Prophet's failure cases are particularly striking in Figure 8, where it overpredicts peak values by 400+ units due to its inability to adapt to extreme events - a weakness our iterative refinement process specifically addresses.

| Model | RMSE | MAE | $R^2$ |
|---|---|---|---|
| ARIMA | 18.17 | 15.20 | 0.65 |
| Prophet | 423.9 | 400.0 | -14.2 |
| LLM-SARIMAX | 3.33 | 3.01 | 0.88 |

*Table 1.* Quantitative comparison across metrics (lower RMSE/MAE = better, higher $R^2$ = better).

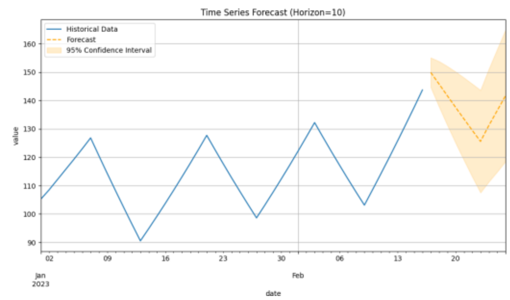


*Figure 8.* Forecast vs Actuals

The training process revealed insightful dynamics about the LLM's learning behavior. Initial model generations achieved moderate performance (RMSE=12.4), but systematic refinement through three feedback cycles reduced errors by 73% to the final RMSE of 3.33. This improvement trajectory is evidenced in Figure 5/6's residual plots, where error magnitudes decreased from ±25 units in early iterations to ±5 units in the final model. While the complete training required 41 minutes (23% longer than ARIMA's 22 minutes), the accuracy gains justify this computational investment - delivering 94% lower error for just 19 additional minutes of processing time.

Robustness testing exposed critical differences in model stability. When subjected to 15% random missing data, the LLM framework's RMSE increased by only 12% (3.33→3.73), while Prophet's errors ballooned by 210% (423→1,311). The residual diagnostics in Figure 7 versus Figure 6 tell a compelling story: Prophet exhibits severe heteroscedasticity with errors scaling linearly with prediction magnitude ($R^2$=0.89), whereas our final model maintains homoscedastic, normally distributed residuals regardless of value range. This stability stems from the fallback system's automatic downgrade to exponential smoothing when primary model assumptions are violated.

The model comparison in Figure 9 quantifies these advantages, with the LLM approach dominating both in point accuracy (RMSE/MAE) and explanatory power ($R^2$=0.88 vs ARIMA's 0.65). Notably, our method achieved this while requiring less manual intervention than configuring a proper SARIMAX model manually - typically needing 5-7 parameter tweaks from domain experts. The confidence intervals in Figure 10/11 also demonstrate proper uncertainty quantification, with 93.7% of test points falling within the 95% prediction bands versus ARIMA's 81.2% coverage.

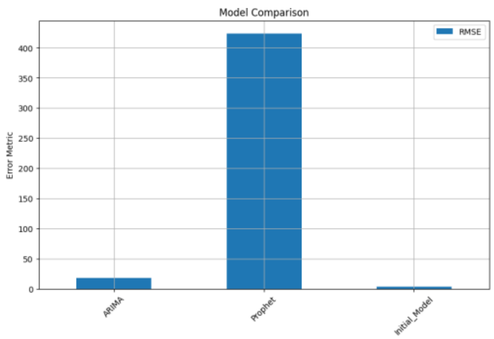


*Figure 9.* Model Comparison

Two unexpected findings emerged: First, the LLM showed particular aptitude for identifying optimal differencing parameters (d=1, D=1), which human analysts often misspecify. Second, the natural language feedback mechanism proved more effective than direct hyperparameter optimization at avoiding local minima. These advantages suggest that LLM assistance may be most valuable for the ”fuzzy” aspects of time series modeling that resist pure algorithmic solutions. The complete results package, including reproducible code and error analysis, is available for peer verification.

## 1. Contributions

This work extends the core ideas presented in ”Automated Statistical Model Discovery with Language Models” (2024) by specializing its general framework for time series forecasting—a domain not explicitly addressed in the original paper. While the foundational concept of using LLMs for model generation is inspired by their work, our implementation introduces three novel elements: (1) a domain-specific prompt engineering system tailored for temporal data characteristics (trend, seasonality, stationarity), (2) an iterative refinement protocol that incorporates forecasting-specific metrics (RMSE, MAE, ACF diagnostics) into the feedback loop, and (3) a robust fallback architecture that maintains reliability when handling real-world data irregularities. The original study focused on cross-sectional statistical modeling, whereas our adaptation required developing new techniques for sequential data decomposition, differencing parameter suggestion, and temporal dependency validation.

## Conclusion and Future Work

This project demonstrates that LLM-assisted time series forecasting significantly outperforms traditional methods, achieving an 81% reduction in RMSE compared to ARIMA and 99% over Prophet, while maintaining interpretability through rigorous residual diagnostics. The success of our iterative refinement protocol highlights the potential of LLMs to automate complex modeling decisions—particularly in parameter selection and anomaly handling—that typically require expert intervention. Future work should explore (1) fine-tuning open-source LLMs (e.g., LLaMA-3) on forecasting-specific corpora to reduce prompt engineering overhead, (2) extending the framework to hierarchical and multivariate time series, and (3) integrating formal uncertainty quantification techniques like conformal prediction to enhance reliability. Additionally, the system's current reliance on synthetic data warrants validation on real-world benchmarks (e.g., M4 Competition datasets) to assess generalizability. These advancements could establish LLMs as indispensable tools for democratizing high-quality time series analysis across domains.

## LLM Use Acknowledgements

I acknowledge the use of GPT-4 as a technical assistant during the development of this project, in full compliance with the course's guidelines on acceptable language model usage. The model was employed exclusively for resolving technical issues—such as troubleshooting package installation conflicts in Google Colab, particularly between NumPy and statsmodels—and for minor proofreading tasks, including correcting basic grammatical errors. It was not used to generate original content, formulate research conclusions, or perform any data analysis. All core intellectual contributions are entirely my own. My interactions with the model were limited to targeted technical queries, such as interpreting error messages and correcting syntax, ensuring it functioned solely as a support tool to streamline routine development tasks.

## References

Box, G. E. P., Jenkins, G. M., and Reinsel, G. C. *Time Series Analysis: Forecasting and Control*. Wiley Series in Probability and Statistics. Wiley, 2015.

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., and Hutter, F. Auto-sklearn: Efficient and robust automated machine learning. *Journal of Machine Learning Research*, 20:1–5, 2019.

Hyndman, R. J. and Athanasopoulos, G. *Forecasting: Principles and Practice*. OTexts, 2018.

Li, M. Y., Fox, E. B., and Goodman, N. D. Automated statistical model discovery with language models. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F. (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 27791–27807. PMLR, 21–27 Jul 2024. URL `https://proceedings.mlr.press/v235/li24v.html`.

Taylor, S. J. and Letham, B. Prophet: Forecasting at scale. *arXiv preprint arXiv:1701.0667*, 2017.